0

A Tutorial Introduction to MATLAB and the Symbolic Math Toolbox

Aims and Objectives

- To provide a tutorial guide to MATLAB.
- To give practical experience in using the package.
- To promote self-help using the online help facilities.
- To provide a concise reference source for experienced users.

On completion of this chapter the reader should be able to

- use MATLAB and the Symbolic Math Toolbox;
- produce simple MATLAB programs;
- access some MATLAB commands and programs over the Web.

It is assumed that the reader is familiar with either the Windows or UNIX platform. This book was prepared using MATLAB (Version 6[1]) but most programs should work under earlier and later versions of the package.

The commands and programs listed in this chapter have been chosen to allow the reader to become familiar with MATLAB and the Symbolic Math Toolbox

[1] As this book was in production, The MathWorks, Inc. was preparing to release MATLAB 7. Please note that the program files presented in this book will work with the updated version of MATLAB.

within a few hours. They provide a concise summary of the type of commands that will be used throughout the text. New users should be able to start on their own problems after completing the chapter, and experienced users should find this chapter an excellent source of reference. Of course, there are many MATLAB textbooks on the market for those who require further applications or more detail.

If you experience any problems there are several options. There is an excellent index within MATLAB, and MATLAB program files (or M-files) can be downloaded from the MATLAB Central File Exchange at

http://www.mathworks.com/matlabcentral/.

The M-files can be found at the link "Companion Software for Books." Download the zipped M-files and **Extract** the relevant M-files from the archive onto your computer.

You can either run the tutorials interactively in the Command Window or save the commands onto an M-file. When in the Command Window, command lines may be returned to and edited using the up and down arrow keys. M-files will be discussed in Section 0.3.

0.1 Tutorial One: The Basics and the Symbolic Math Toolbox (One Hour)

There is no need to copy the comments; they are there to help you. Click on the MATLAB icon and copy the command after the >> prompt. Press **Return** at the end of a line to see the answer or use a semicolon to suppress the output. You can interrupt a computation at any time by holding **CTRL+C**. This tutorial should be run in the Command Window.

MATLAB Commands	**Comments**
`>> % This is a comment.`	`% Helps when writing % programs.`
`>> 3^2*4-3*2^5*(4-2)`	`% Simple arithmetic.`
`>> sqrt(16)`	`% The square root.`
`>> % Vectors and Matrices.`	
`>> u=1:2:9`	`% A vector u=(1,3,5,7,9).`
`>> v=u.^2`	`% A vector v=(1,9,25,49,81).`
`>> A=[1,2;3,4]`	`% A is a 2 by 2 matrix.`
`>> A'`	`% The transpose of matrix A.`
`>> det(A)`	`% The determinant of a matrix.`
`>> B=[0,3,1;0.3,0,0;0,0.5,0]`	`% B is a 3 by 3 matrix.`

```
>> eig(B)                          % The eigenvalues of B.
>> % List the eigenvectors and eigenvalues of B.
>> [Vects,Vals]=eig(B)
>> C=[100;200;300]                 % A 3 by 1 matrix.
>> D=B*C                           % Matrix multiplication.
>> E=B^4                           % Powers of a matrix.
>> % Complex numbers.
>> % The semicolons suppress the output.
>> z1=1+i;z2=1-i;z3=2+i;
>> z4=2*z1-z2*z3                   % Simple arithmetic.
>> abs(z1)                         % The modulus.
>> real(z1)                        % The real part.
>> imag(z1)                        % The imaginary part.
>> exp(i*z1)                       % Returns a complex number.
>> % The Symbolic Math Toolbox.
>> sym(1/2)+sym(3/4)               % Symbolic evaluation.
>> % Double precision floating point arithmetic.
>> 1/2+3/4
>> % Variable precision arithmetic (50 d.p's).
>> vpa('pi',50)
>> syms x y z                      % Symbolic objects.
>> z=x^3-y^3
>> factor(z)                       % Factorization.
>> % Symbolic expansion of the last answer.
>> expand(ans)
>> simplify(z/(x-y))               % Symbolic simplification.
>> % Symbolic solution of algebraic equations.
>> [x,y]=solve('tau*x*y','tau*x*y-y')
>> f='mu*x*(1-x)'                  % Define a function.
>> subs(f,x,1/2)                   % Substitute for x, gives
                                   % f(1/2).
```

```
>> fof=subs(f,x,f)              % A function of a function.
>> syms x
>> limit(x/sin(x),x,0)          % Limits.
>> diff(f,x)                    % Differentiation.
>> % Partial differentiation (w.r.t. y twice).
>> diff('x^2+3*x*y-2*y^2','y',2)
>> int('sin(x)*cos(x)',x,0,pi/2)
                                % Integration.
>> int('1/x',x,0,inf)           % Improper integration.
>> syms n
>> s1=symsum(1/n^2,1,inf)       % Symbolic summation.
>> syms x
>> g=exp(x)
>> taylor(g,10)                 % Taylor series expansion.
>> syms x s
>> laplace(x^3)                 % Laplace transform.
>> ilaplace(6/s^4)              % Inverse Laplace transform.
>> syms x w
>> fourier(-2*pi*abs(x))        % Fourier transform.
>> ifourier(4*pi/w^2)           % Inverse Fourier transform.
>> quit                         % Exit MATLAB.
>> % End of Tutorial 1.
```

0.2 Tutorial Two: Plots and Differential Equations (One Hour)

More involved plots will be discussed in Section 0.3. You will need the Symbolic Math Toolbox for most of the commands in this section.

MATLAB Commands

```
>> % Plotting graphs.

>> % Set up the domain and plot a simple function.
>> x=-2:.01:2;
>> plot(x,x.^2)
```

```
>> % Plot two curves on one graph and label them.
>> t=0:.1:100;
>> y1=exp(-.1.*t).*cos(t);y2=cos(t);
>> plot(t,y1,t,y2),legend('y1','y2')

>> % Plotting with the Symbolic Math Toolbox.
>> ezplot('x^2',[-2,2])

>> % Plot with labels and title.
>> ezplot('exp(-t)*sin(t)'),xlabel('time'),ylabel('current'),
   title('decay')

>> % Contour plot on a grid of 50 by 50.
>> ezcontour('y^2/2-x^2/2+x^4/4',[-2,2],50)

>> % Surface plot. Rotate the plot using the mouse.
>> ezsurf('y^2/2-x^2/2+x^4/4',[-2,2],50)

>> % Combined contour and surface plot.
>> ezsurfc('y^2/2-x^2/2+x^4/4',[-2,2])

>> % A parametric plot.
>> ezplot('t^3-4*t','t^2',[-3,3])

>> % 3D parametric curve.
>> ezplot3('sin(t)','cos(t)','t',[-10,10])

>> % Implicit plot.
>> ezplot('2*x^2+3*y^2-12')
```

Symbolic solutions to ordinary differential equations (ODEs). If an analytic solution cannot be found the message "Warning: explicit solution could not be found" is shown.

```
>> % The D denotes differentiation w.r.t. t.
>> dsolve('Dx=-x/t')

>> % An initial value problem.
>> dsolve('Dx=-t/x','x(0)=1')

>> % A second-order ODE
>> dsolve('D2I+5*DI+6*I=10*sin(t)','I(0)=0','DI(0)=0')

>> % A linear system of ODEs.
>> [x,y]=dsolve('Dx=3*x+4*y','Dy=-4*x+3*y')

>> % A system with initial conditions.
```

Table 0.1: ODE function summary. There are other solvers, but these should suffice for the material covered in this book.

Solver	ODE	Method	Summary
ode45	Nonstiff	Runge–Kutta	Try this first
ode23	Nonstiff	Runge–Kutta	Crude tolerances
ode113	Nonstiff	Adams	Multistep solver
ode15s	Stiff	Gears	If you suspect stiffness
ode23s	Stiff	Rosenbrock	One-step solver

```
>> [x,y]=dsolve('Dx=x^2','Dy=y^2','x(0)=1,y(0)=1')

>> % A 3D linear system.
>> [x,y,z]=dsolve('Dx=x','Dy=y','Dz=-z')
```

Numerical Solutions to ODEs. These methods are usually used when an analytic solution cannot be found—even though one may exist! Table 0.1 lists some initial value problem solvers in MATLAB. See the online MATLAB index for more details or type >> help *topic*.

```
>> % Set up a 1D system with initial condition 50,
>> % and run for 0<t<100.
>> deq1=inline('x(1)*(.1-.01*x(1))','t','x');
>> [t,xa]=ode45(deq1,[0 100],50);
>> plot(t,xa(:,1))

>> % A 2D system with initial conditions [.01,0],
>> % and run for 0<t<50.
>> deq2=inline('[.1*x(1)+x(2);-x(1)+.1*x(2)]','t','x');
>> [t,xb]=ode45(deq2,[0 50],[.01,0]);
>> plot(xb(:,1),xb(:,2))

>> % A 3D system with initial conditions.
>> deq3=inline('[x(3)-x(1);-x(2);x(3)-17*x(1)+16]','t','x');
>> [t,xc]=ode45(deq3,[0 20],[.8,.8,.8]);
>> plot3(xc(:,1),xc(:,2),xc(:,3))

>> % A phase plane portrait of a stiff Van der Pol system.
>> deq4=inline('[x(2);1000*(1-(x(1))^2)*x(2)-x(1)]','t','x');
>> [t,xd]=ode23s(deq4,[0 3000],[.01,0]);
>> plot(xd(:,1),xd(:,2))

>> % Displacement against time.
>> plot(t,xd(:,1))

>> % End of Tutorial 2.
```

0.3 MATLAB Program Files, or M-Files

Sections 0.1 and 0.2 illustrate the use of the Command Window and its interactive nature. More involved tasks will require more lines of code and it is better to use MATLAB program files, or M-files. Each program is displayed between horizontal lines and kept short to aid in understanding; the output is also included.

Function M-files. The following M-file defines the logistic function.

```
% Program 1 - The logistic function.
% Save the M-file as fmu.m.
function y=fmu(mu,x)
y=mu*x*(1-x);
% End of Program 1.

>> fmu(2,.1)

ans = 0.1800
```

The for... end loop. For repetitive tasks.

```
% Program 2 - The Fibonacci sequence.
% Save the M-file as Fibonacci.m.
Nmax=input('Input the number of terms of the Fibonacci
            sequence:')
F=zeros(1,Nmax);
F(1)=1;F(2)=1;
for i=3:Nmax
    F(i)=F(i-1)+F(i-2);
end
F
% End of Program 2.

>>Fibonacci
Input the number of terms of the Fibonacci sequence:10

Nmax=10

F = 1  1  2  3  5  8  13  21  34  55
```

Conditional statements. If... then... elseif... else... end, etc.

```
% Program 3 - The use of logical expressions.
% Save this M-file as test1.m.
integer=input('Which integer do you wish to test?')
if integer == 0
    disp(['The integer is zero.'])
elseif integer < 0
```

```
    disp(['The integer is negative.'])
else
    disp(['The integer is positive.'])
end
% End of Program 3.

>> test1
Which integer do you wish to test?-100

integer=-100

The integer is negative.
```

Graphic programs. Multiple plots with text.

```
% Program 4 - Two curves on one graph (Figure 0.1).
% Save the M-file as plot1.m.
% Tex symbols can be plotted within a figure window.
clear all
fsize=15;
x=0:.1:7;
figure;
plot(x,2*x./5,x,-x.^2/6+x+7/6)
text(1,2,'\leftarrow curve 1','Fontsize',fsize)
text(2.6,1,'\leftarrow line 1','Fontsize',fsize)
xlabel('x','Fontsize',fsize);
ylabel('y','Fontsize',fsize)
title('Two curves on one plot','Fontsize',fsize)
% End of Program 4.

>> plot1
```

Labeling solution curves. Solutions to the pendulum problem.

```
% Program 5 - Solution curves to different ODEs (Figure 0.2).
% Save M-file as deplot1.m.
clear;
figure;
hold on
deqn=inline('[x(2);-25*x(1)]','t','x');
[t,xa]=ode45(deqn,[0 10],[1 0]);
plot(t,xa(:,1),'r');
clear
deqn=inline('[x(2);-2*x(2)-25*x(1)]','u','x');
[t,xb]=ode45(deqn,[0 10],[1 0]);
plot(t,xb(:,1),'b');
legend('harmonic motion','damped motion')
```

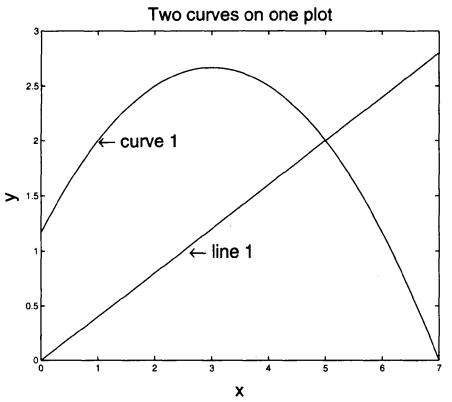

Figure 0.1: A multiple plot with text.

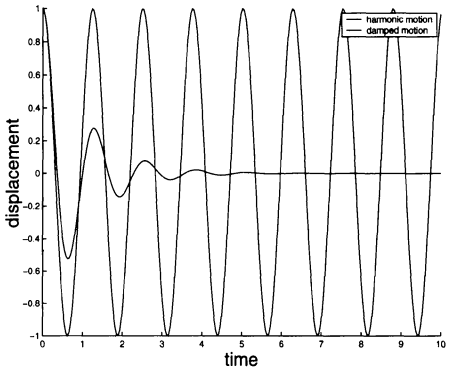

Figure 0.2: Harmonic and damped motion of a pendulum.

```
xlabel('time','Fontsize',15);
ylabel('displacement','Fontsize',15);
hold off
% End of Program 5.

>> deplot1
```

Echo commands. Useful when debugging a program.

```
% Program 6 - Use echo to display command lines.
% Save the M-file as poincare.m.
echo on
r(1)=1;
for n=2:5
```

```
    r(n)=r(n-1)/(1+2*pi*r(n-1))
end
echo off
% End of Program 6.

>> poincare
r(n)=r(n-1)/(1+2*pi*r(n-1))

r = 1.0000    0.1373    0.0737    0.0504    0.0383
```

0.4 Hints for Programming

The MATLAB language contains very powerful commands, which means that some complex programs may contain only a few lines of code. Of course, the only way to learn programming is to sit down and try it. This section has been included to point out common errors and give advice on how to troubleshoot. Remember to check the help and index pages in MATLAB and the Web if the following does not help with your particular problem.

Common Typing Errors. The author strongly advises new users to run Tutorials 0.1 and 0.2 in the Command Window environment, which should reduce typing errors.

- Be careful to include all mathematics operators.
- When dealing with vectors put a full stop (.) in front of the mathematical operator.
- Make sure brackets, parentheses, etc., match up in correct pairs.
- Remember MATLAB is case sensitive.
- Check the syntax, enter **help** followed by the name of the command.

Programming Tips. The reader should use the M-files in Section 0.3 to practice simple programming techniques.

- It is best to clear values at the start of a large program.
- Use the **figure** command when creating a new figure.
- Preallocate arrays to prevent MATLAB from having to continually resize arrays—for example, use the zeros command. An example is given in Program 2 above.
- Vectorize where possible. Convert for and while loops to equivalent vector or matrix operations.

- If a program involves a large number of iterations, for example 50,000, then run it for three iterations first and use the echo command.
- If the computer is not responding hold **CTRL+C** and try reducing the size of the problem.
- Click on **Debug**, **Save and Run** in the M-file window; MATLAB will list any error messages.
- Find a similar M-file in a book or on the Web, and edit it to meet your needs.
- Make sure you have the correct packages, for example, the symbolic commands will not work if you do not have the Symbolic Math Toolbox.
- Check which version of MATLAB you are using.

0.5 MATLAB Exercises

1. Evaluate the following:
   (a) $4+5-6$;
   (b) 3^{12};
   (c) $\sin(0.1\pi)$;
   (d) $(2-(3-4(3+7(1-(2(3-5))))))$;
   (e) $\frac{2}{5}-\frac{3}{4}\times\frac{2}{3}$.
2. Given that
$$A=\begin{pmatrix}1&2&-1\\0&1&0\\3&-1&2\end{pmatrix},\quad B=\begin{pmatrix}1&2&3\\1&1&2\\0&1&2\end{pmatrix},\quad C=\begin{pmatrix}2&1&1\\0&1&-1\\4&2&2\end{pmatrix},$$
   determine the following:
   (a) $A+4BC$;
   (b) the inverse of each matrix if it exists;
   (c) A^3;
   (d) the determinant of C;
   (e) the eigenvalues and eigenvectors of B.
3. Given that $z_1=1+i$, $z_2=-2+i$, and $z_3=-i$, evaluate the following:
   (a) $z_1+z_2-z_3$;
   (b) $\frac{z_1z_2}{z_3}$;

(c) e^{z_1};

(d) $\ln(z_1)$;

(e) $\sin(z_3)$.

4. Evaluate the following limits if they exist:

(a) $\lim_{x\to 0} \frac{\sin x}{x}$;

(b) $\lim_{x\to\infty} \frac{x^3+3x^2-5}{2x^3-7x}$;

(c) $\lim_{x\to\pi} \frac{\cos x+1}{x-\pi}$;

(d) $\lim_{x\to 0^+} \frac{1}{x}$;

(e) $\lim_{x\to 0} \frac{2\sinh x-2\sin x}{\cosh x-1}$.

5. Find the derivatives of the following functions:

(a) $y = 3x^3 + 2x^2 - 5$;

(b) $y = \sqrt{1+x^4}$;

(c) $y = e^x \sin x \cos x$;

(d) $y = \tanh x$;

(e) $y = x^{\ln x}$.

6. Evaluate the following definite integrals:

(a) $\int_{x=0}^{1} 3x^3 + 2x^2 - 5dx$;

(b) $\int_{x=1}^{\infty} \frac{1}{x^2}dx$;

(c) $\int_{-\infty}^{\infty} e^{-x^2}dx$;

(d) $\int_0^1 \frac{1}{\sqrt{x}}dx$;

(e) $\int_0^{\frac{2}{\pi}} \frac{\sin(1/t)}{t^2}dt$.

7. Graph the following:

(a) $y = 3x^3 + 2x^2 - 5$;

(b) $y = e^{-x^2}$ for $-5 \le x \le 5$;

(c) $x^2 - 2xy - y^2 = 1$;

(d) $z = 4x^2e^y - 2x^4 - e^{4y}$ for $-3 \le x \le 3$ and $-1 \le y \le 1$;

(e) $x = t^2 - 3t$, $y = t^3 - 9t$ for $-4 \le t \le 4$.

8. Solve the following differential equations:

   (a) $\frac{dy}{dx} = \frac{x}{2y}$, given that $y(1) = 1$;

   (b) $\frac{dy}{dx} = \frac{-y}{x}$, given that $y(2) = 3$;

   (c) $\frac{dy}{dx} = \frac{x^2}{y^3}$, given that $y(0) = 1$;

   (d) $\frac{d^2x}{dt^2} + 5\frac{dx}{dt} + 6x = 0$, given that $x(0) = 1$ and $\dot{x}(0) = 0$;

   (e) $\frac{d^2x}{dt^2} + 5\frac{dx}{dt} + 6x = \sin(t)$, given that $x(0) = 1$ and $\dot{x}(0) = 0$.

9. Carry out 100 iterations on the recurrence relation

$$x_{n+1} = 4x_n(1 - x_n),$$

   given that (a) $x_0 = 0.2$ and (b) $x_0 = 0.2001$. List the final 10 iterates in each case.

10. Type "help while" after the >> prompt to read the help page on the while command. Use a while loop to program Euclid's algorithm for finding the greatest common divisor of two integers. Use your program to find the greatest common divisor of 12,348 and 14,238.

Recommended Textbooks

[1] W. Palm III, *Introduction to MATLAB 7 for Engineers*, McGraw–Hill, New York, 2004.

[2] B. Daku, *Learn MATLAB Fast: The M-Tutor Interactive Tutorial*, John Wiley, New York, 2002.

[3] B. R. Hunt, R. L. Lipsman, J. Rosenberg, and K. R. Coombes, *A Guide to MATLAB*, Cambridge University Press, Cambridge, UK, 2001.

[4] R. Pratab, *Getting Started with MATLAB: Version 6: A Quick Introduction for Scientists and Engineers*, Oxford University Press, London, 2001.

[5] D. Hanselman and B. R. Littlefield, *Mastering MATLAB* 6, Prentice–Hall, Upper Saddle River, NJ, 2000.