Computing with MATLAB

R. Turner

UW Math Dept

Madison, WI

Matlab

Matlab is a very useful piece of software with extensive capabilities for numerical computation and graphing.

In MATLAB the basic structure is a matrix.

lab login

1) To log on type
`student`

2) As a password use

`b1oh7`.

(Note that the ’oh’ are letters)
and wait for it to come up in the Windows mode.

calculator

I. It is easy to use MATLAB as a calculator. The symbols +, $-$, and $/$ have their usual meaning; * denotes multiplication and $\hat{}$ exponentiation. E.g. type

$$((5.76\hat{}2) * 3.01)/8.8$$

and press ENTER; the screen should read

$$\text{ans } = 11.3482$$

variables

II. If you type

$$x = 2;$$

and then

$$y = x * (x + 2)$$

the screen should read

$$\text{ans } = 8$$

(If you don’t type the semicolon after the 2, you will get some additional output on the screen). The computer will also remember that $x = 2$ if you use x again unless you reset the variable. To reset x, type `clear` x; to reset the whole workspace, type `clear`

functions

III. MATLAB has a large number of built in functions. E.g.

$$\exp(x) \quad \text{is} \quad e^x$$

$$\sin(x) \quad \text{is} \quad \sin x$$

$$\log(x) \quad \text{is the natural log of} \quad x$$

Other functions have common abbreviations: cos, acos (for inverse cosine, tan, atan, tanh, etc. To get help type

help

For help on a particular topic type

help topic

graph

IV. MATLAB can graph functions.
. To plot a function over $n+1$ equally spaced points of the interval $[a, b]$, calculate $h = (b-a)/n$ and type:

$$t = a : h : b; \quad OR \quad t = a : (b-a)/n : b;$$

MATLAB generates an $n+1$ component vector. If you don't type the semicolon, the vector will be printed out. Typing

$$t = 0 : .2 : 1;$$

gives a 6 component vector while

$$r = 0 : .3 : 1;$$

gives the vector with components $0 \;\; .3 \;\; .6 \;\; .9$.

plot sine

To graph e.g. $\sin(5t)$ over $[a, b]$, type

$$y = \sin(5 * t);$$

$$\text{plot } (t, y)$$

You get the plotted points connected by straight line segments. Do this for $a = 0, b = 1$ and $n = 20$ (i.e. $h = .05$)

plot

To plot with separated points use

$$\text{plot } (t, y, \text{''} * \text{''})$$

Type

help plot

to get much more information about plot.

graph labels

V. To add a title or labels to the graph follow the original plot command above by e.g.

$$\text{title }(' \text{graph } 1')$$

$$\text{xlabel }(' \text{t} = \text{time }')$$

$$\text{ylabel }(' \text{y} = \text{distance }')$$

.

either before or after the ’gset’ commands. Then type replot to get a postscript file.

array multiplication

VI. For the simple example of $\sin(5t)$ above, to graph it involves calculating the function at the points in the given interval. For a more complicated expression like $z = e^t \sin(5t)$, you have to calculate e^t and $\sin(5t)$ at each point and multiply them. This is called ARRAY MULTIPLICATION. To do this for z on e.g. $[-1, 1]$ and $n = 80$, type

$$t = -1 : 0.025 : 1;$$

$$z = \exp(t). * \sin(5 * t)$$

The $.*$ indicates array multiplication. If you omit the period before the $*$, you will get an error message when you try to use the function.
Plot a labelled graph for z on the interval above.

plot two functions

VI. To plot 2 functions, say $w = te^t$ and $u = t^3 \log(1+|t|)$ on the same graph with e.g. the first a solid curve and the second with 'stars', type

$$\begin{aligned} t &= -1 : 0.1 : 1; \\ z &= t. * exp(t) \\ u &= t.\hat{}3. * log(1 + abs(t)) \end{aligned} \tag{1}$$

and then

$$\text{plot}(t, z, t, u, " * ").$$

M-files and Solving IVP’s

- 1. The goal of this section is to introduce you to M-files and allow you to try out some Runge-Kutta solvers.

M-files and Solving IVP’s

- 1. The goal of this section is to introduce you to M-files and allow you to try out some Runge-Kutta solvers.
- 2. When a function is used repeatedly, instead of retyping it each time, it is convenient to create a file, called an M-file for the function

New function

Suppose one wants to make a file containing the function $5t^3e^{-t}$ and to call it `fun.m`. The name of an M-file must begin with a letter and consist entirely of letters, numbers, and underscores; it must end with ' .m ' (e.g. file.m)

Edit

To edit a new M-file in the Matlab workspace, click on FILE, down to NEW, right to M-file to get a blank page. On this type:

```
        function y=fun(t)
        % my function
        y = 5.∗(t.∧3).∗exp(-t);
```

click on FILE, Save-As, give it the name ’fun.m’, and exit the editor

Plot 'fun'

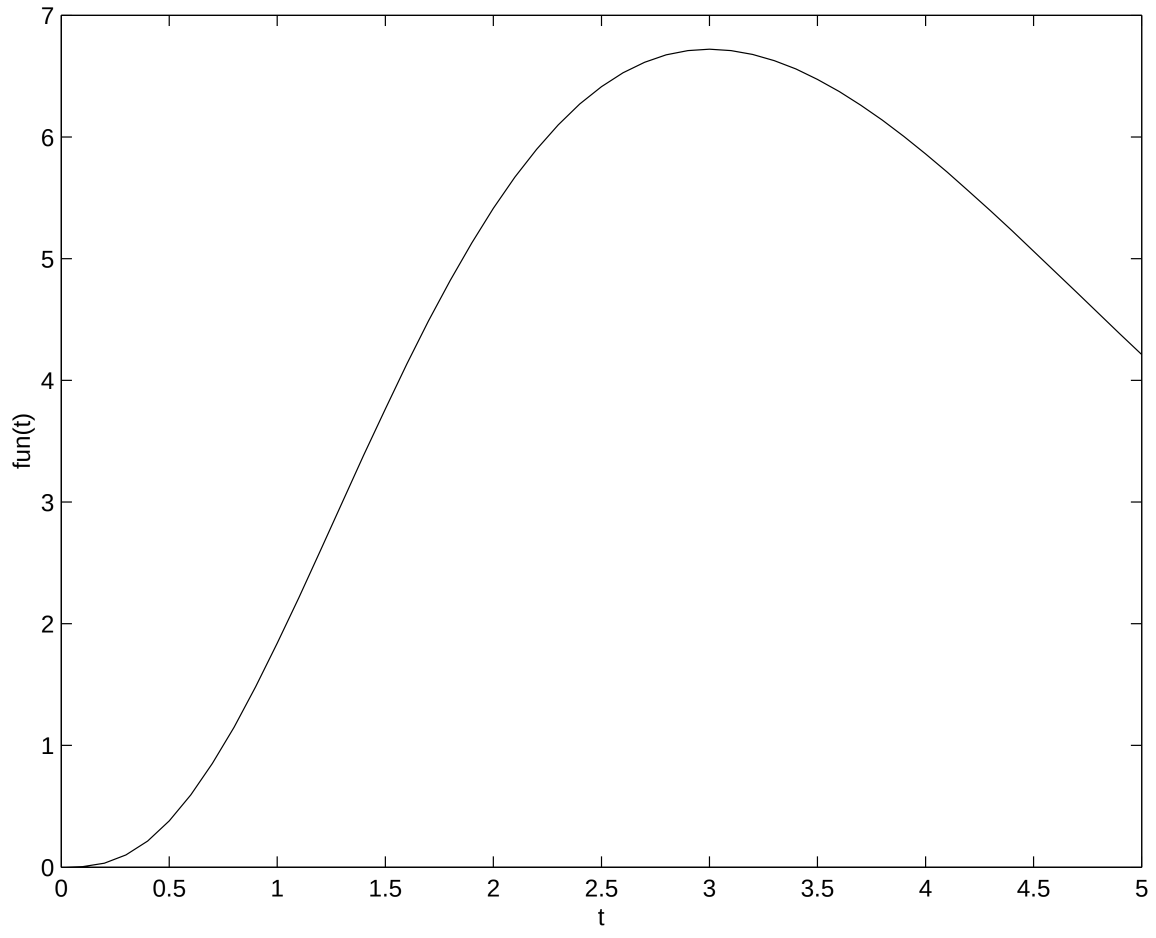
Let's plot the function 'fun' from $t = 0$ to $t = 5$, type

```
t= 0:.1:5;
```

To get a string of t values, spaced at intervals of $.1$. Now type

```
        plot(t,fun(t));
```

Plot



IVP

consider the Initial Value Problem

$$x' = 1-x$$

$$x(0) = 0.1$$

Since the equation is linear and first order, it can be solved explicitly:

$$x(t) = 1-.9e^{-t}$$

Numerics

Usually an explicit solution is inaccessible and a numerical ’integration’ is needed. For that reason we next introduce a numerical solver to be used to find an approximate solution. Again, one can type commands in the workspace (command window) at the MATLAB prompt, but it is convenient to have M-files which can be used repeatedly. For solving an initial value problem we create TWO files:

FIRST file

Create an M-file as in (2) for the right-hand side of the differential equation. Create a file called `dy.m` containing:

```
function y = dy(t,x)
y = 1-x;
```

What is in FIRST file?

The file dy.m gives the 'dynamics' for the differential equation; that is, gives the RIGHT HAND SIDE of the differential equation. It is important that the same symbol be used just after the word 'function' and at the beginning of the second line, defining the function explicitly. Remember that if you do not type a semi-colon at the end of a line, all the output of that line will be printed to the screen. The inclusion of both t and x in dy(t,x) is not necessary if there is no t in the right hand side of the differential equation. However, it is probably a good habit to write the two variable for an eventual case of the occurrence of t. The order (t, x) is also important

SECOND file

Open a second M-file called
`dyr.m` containing:

```
        tspan= [0 15];
        x0 =[0.1];
        [t,x]=ode45('dy',tspan,x0);
        plot(t,x)
```

What is in SECOND file?

Exit the file dyr.m (this file is used to ’run’ the ode solver and could be call dyrun.m or other name, but we have kept the names short).
The first line gives the set of values for the independent variable at which the solution will be given; the second gives the initial data; and the third ’calls’ the solver, which looks in the file dy.m to see what the right hand side of the differential equation is. The last line is the command to plot the output.

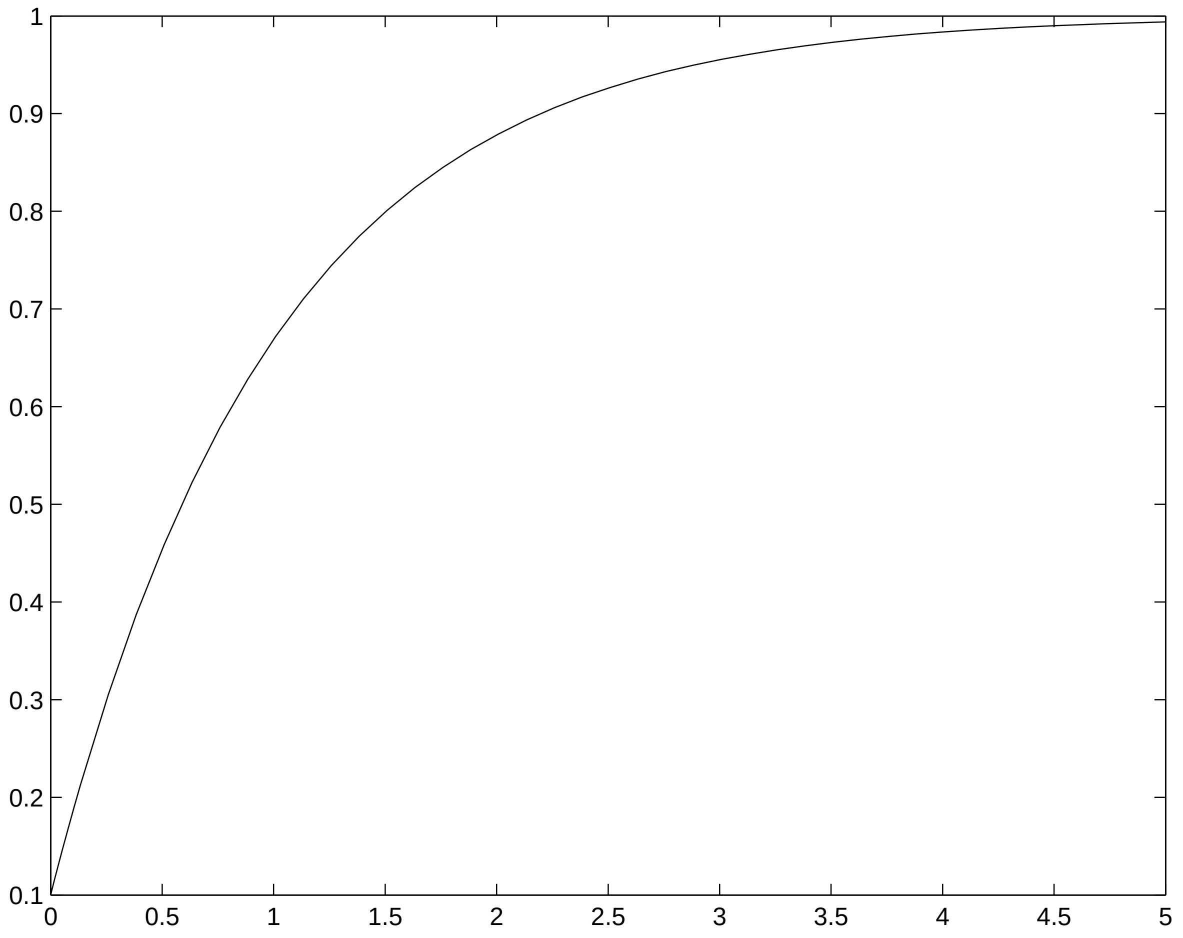
Run programs

To run the program, at the MATLAB prompt, type

```
dyr
```

(one can also type dyr.m).

Graphics



Flexible

The syntax is quite flexible in terms of the choice of letters used. One can, for example, use the letter x in dy.m and let the dependent variable be z in dyr.m. However, one MUST have corresponding names in the file call in the first slot of ’ode45’ and the name of the .m file which describes the right hand side of the differential equation. Note that the lines in the file dyr.m could be typed one at a time in the MATLAB workspace, but then for a rerun with a slight change in the initial value, e.g., one would have to retype it all as opposed to changing one number in the file dyr.m.

global- change FIRST file

It is useful to be able to have a parameter in a differential equation that one can allow to take different values. To do this edit the program dy.m, inserting a line and altering the file so that it reads:

```
function y = dy(t,x)
global a
y = a-x;
```

new SECOND file

After exiting dy.m open dyr.m and change it to read:

```
hold on
global a
t= 0:.1:15;
x0 =[0.1];
for a = 1:4
[t,x]=ode45('dy',tspan,x0);
plot(t,x)
end
hold off
```

loop

In this case we have included a loop

```
for a = 1:4
*COMMANDS*
end
```

hold and off

The ’for’ statement begins a loop in which a takes the values 1,2,3,4, in turn, the solution is recomputed and plotted (that is the intervening commands are executed). The ’hold on’ prevents erasure of the the solution plot for $a = 1$ when the $a = 2$ case is done, etc. When the loop is complete the ’hold off’ allows erasure (for the next use of the plot command). The ’global’ statement tells the two program dy.m and dyr.m to SHARE the values of a.

clear

Note: if one is using two windows and the one with the MATLAB workspace has not been closed, then to run the new version of dyr.m one must type

```
        clear
```

and then type

```
        dyr
```

to see the output of the new file. The 'global' statement is extremely useful in that it allows one to vary a parameter in a differential equation and see the quantitative and qualitative changes in the output.

system

Minor changes in the files dy.m and dyr.m allow one to solve systems of first order differential equations. For example, suppose one wants to solve the (Initial Value Problem)

$$\begin{aligned} \frac{dx_1}{dt} &= -0.2x_1 + 2x_2 \\ \frac{dx_2}{dt} &= -3x_1 - 0.5x_2 \qquad (S) \end{aligned}$$

with initial data:

$$x_1(0) = 5, \quad x_2(0) = 6.$$

dynamics for two

Create a file `two.m` containing:

function y= two(t,x)

```
        r1 = -.2*x(1) + 2*x(2);
        r2 = -3*x(1) -0.5*x(2);
        y = [r1;r2];
```

where $r1$ and $r2$, respectively, are the first line and second line of the right hand side of the system S. The syntax with the square brackets is important here in that MATLAB must read the right side of the differential equation as a ONE by TWO matrix.

second file

Corresponding to the original dyr.m we create a file ’rtwo.m’ to run the solver:

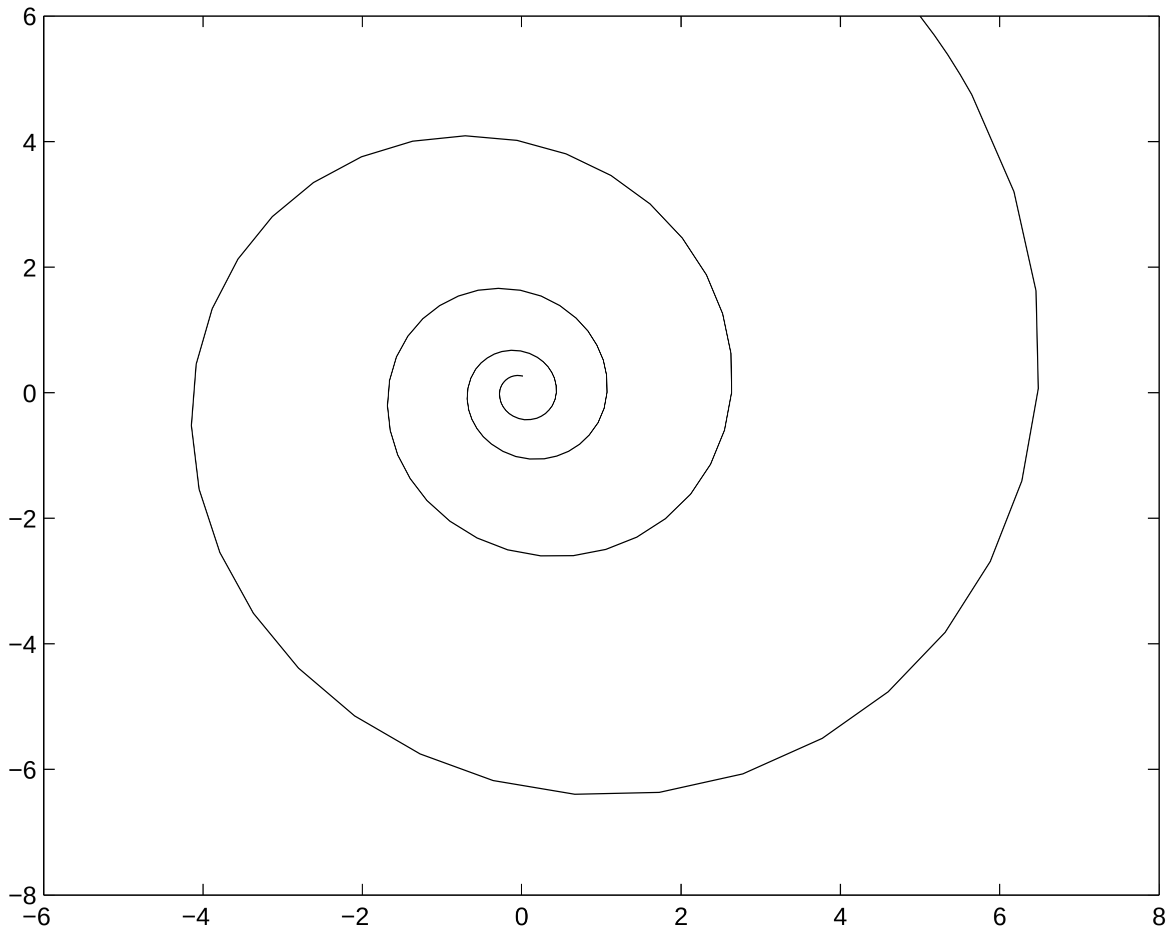
```
tspan = [0 10];
x0 =[5.0 6.0];
x=ode45(’two’,tspan,x0);
plot(x(:,1),x(:,2))
```

Note the agreement of the first argument of ode45 and the name of the first file.

run

After running, the programs plots the TWO arguments in the same plane.

graph



plot

If one wants to plot t versus the second component, one can replace the plot command by `plot(t,x(:,2))` The extension to systems containing three variables $x(1), x(2), x(3)$ follows a similar pattern.

clear

To run the altered program one must again type

```
clear
```

hint

Use the up arrow on the keyboard to go back to the line where one previously typed ’clear’ and then press ’enter’.