Teaching ODEs using the Symbolic Math Toolbox in MATLAB® v 5.3.1

Chris Shoostarian
University of Luton

chris.shoostarian@luton.ac.uk

I have been teaching the applications of ordinary differential equations in modelling electronic circuits to electronic engineering students over the past year. This year I have decided to strengthen my portfolio of mathematics teaching tools [1] to include MATLAB's Symbolic Math Toolbox [2].

The Symbolic Math Toolbox provides a link between MATLAB and the symbolic algebra program known as Maple. Unlike the rest of MATLAB, which provides powerful numerical routines, the Symbolic Math Toolbox deals with symbols, formulas, and equations. In dealing with differential equations, it leads to explicit formulas for the solutions, provided such formulas exist at all. The Symbolic Math Toolbox is included in Version 5.3.1; however, it is distributed as an additional toolbox. Therefore it may not be available on your platform. If you are unsure whether or not it is available, execute the command *help symbolic* at the prompt. If the Symbolic Math Toolbox is available, you will get a list of all the commands available in the toolbox. If it is not, you will get an error message, saying *symbolic not found*.

For a general introduction to the Symbolic Math Toolbox, find a copy of the *Symbolic Math Toolbox User's Guide*. This provides a very good tutorial on the use of the Symbolic Toolbox. In this article, I put my emphasis on using the Symbolic Math Toolbox to solve differential equations. This means that MATLAB will try to tell you the exact, analytic solution to an equation, or even to an initial value problem. MATLAB can be successful, of course, only if such a solution exists.

Method of isoclines using DFIELD5

A first order ordinary differential equation has the form

$$x' = f(t,x)$$

To solve this equation we must find a function $x(t)$ such that

$$x' = f(t,x(t)), \quad \text{for all } t$$

This means that at every point $(t,x(t))$ on the graph of x, the graph must have slope equal to $f(t,x(t))$. We can turn this interpretation around to give a geometric view of what a differential equation is, and what it means to solve the equation. At each point (t,x), the number $f(t,x)$ represents the slope of a solution curve through this point. This collection of lines is called a direction field, and it provides the geometric interpretation of a differential equation. To find a solution we must find a curve in the plane which is tangent at each point to the direction line at that point.

Admittedly, it is difficult to visualise such a direction field. This is where the MATLAB routine dfield5 demonstrates its value. Given a differential equation, it will plot the direction lines at a large number of points-enough so that the entire direction line filed can be visualised by mentally interpolating between the field elements. This enables the user to get some geometric insight into the solutions of the equation.

Starting DFIELD5

The MATLAB function dfield5 is not distributed with MATLAB. The best source for the software is the web page http://math.rice.edu/dfield. There you will find

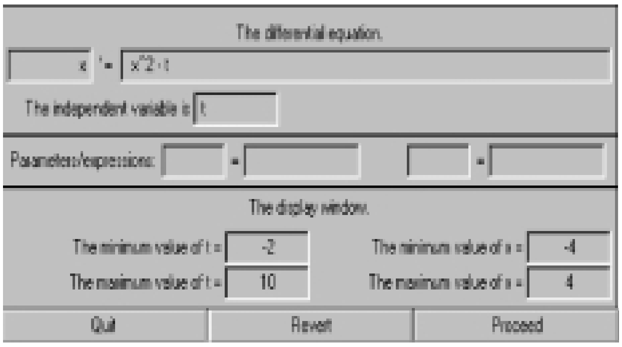
Supplier's contact details

The MathWorks Ltd
Matrix House
Cowley Park
Cambridge, CB4 0HH

Tel: 01223 423 200
Fax: 01223 423 289/255
info@uk.mathworks.com
http://www.mathworks.co.uk

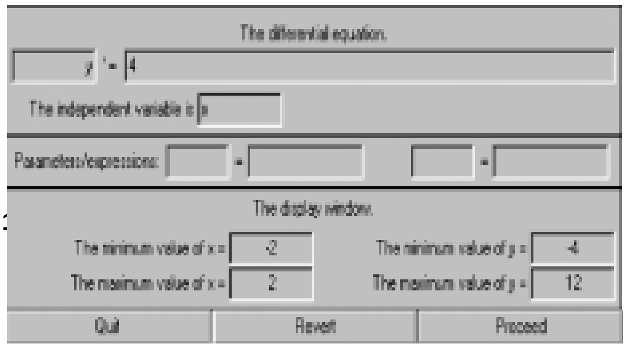
versions, which are compatible with all versions of MATLAB going back to 3.5. This is also the location where the most up to date versions of dfield will be found in the future. To describe the method we use the first order differential equation dy/dx=4. In order to get the function "DFIELD5", type at MATLAB prompt, >>dfield5. MATLAB will show the default *window* as shown in Figure 1.

Figure 1: The DFIELD5 default window



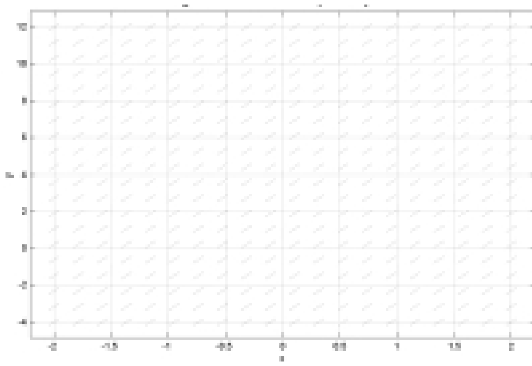
The window divides into two parts: the differential equation part and below it the display window part. The differential equation part dialogue boxes can be changed to suit our example. To start, replace the *x* in the top left hand corner box with *y*. Moving to the next box, which is located immediately to the right of the equal sign, replace the current expression *x^2-t* with 4. The next box, which appears immediately below, contains the independent variable *t* and we can change this to *x*. Since there are no parameters values the following two boxes are left blank. We can now concentrate on the display window part by inserting the minimum and maximum values of *x* in the two boxes on the bottom left hand corner of the window respectively. The minimum and maximum values of *y* can also be in the two boxes on the bottom right hand corner of the window respectively. The resulting window appears in Figure 2.

Figure 2: The modified DFIELD5 window for y'=4



By clicking the box on the bottom right hand corner labelled **Proceed**, the direction field plots for the given differential equation will appear as shown in Figure 3.

Figure 3: The direction field plots for y' = 4



Features of DFIELD5 Display window

- i DFIELD5 Display window is a rectangular grid labelled with the differential equation *dy/dx*=4 or in shortened version *y'* = 4 on top.
- ii The independent variable *x* on the bottom.
- iii The dependent variable *y* on the left.
- iv Inside this rectangle is a grid of points, 20 in each direction, for a total of 400 points.
- v At each point with coordinates (*x*, *y*) there is shown a small line segment centered at (*x*, *y*) with slope = 4.

Figure 4: Several solutions of the ODE y' = 4

A *solution curve* of a differential equation *y'* = 4 is the graph of a function *y'* = 4*x*+*c*, which solves the differential equation.

Computing and plotting a solution curve is very easy using dfield5. Choose an initial point for the solution, move the mouse to that point, and click the mouse button. The computer will compute and plot the solution through the selected point. After computing and plotting several solutions, the display should look something like that shown in Figure 4. We are able to choose just one line from the family of lines. We do this by specifying a particular point through which the line must pass.

Solving Ordinary Differential Equations using the function "dsolve"

The routine *dsolve* is probably the most useful differential equations tool in the Symbolic Toolbox. Perhaps the best way to explain the syntax of the *dsolve* command is to start with an example [3]. In a simple electrical circuit consisting of inductance *L* and resistance *R* connected to a constant voltage source *E*, the current *i* is given by the modified differential equation

$\frac{di}{dt} = \frac{E - Ri}{L}$, where *L* and *R* are constants.

The function *dsolve* computes symbolic solutions and a general solution to this differential equation is easily obtained. By default, the independent variable is '*t*'. The independent variable may be changed from '*t*' to some other symbolic variable by including that variable as the last input argument. For our problem we have:

```
>>y=dsolve('Dy=(E-R*y)/L','t')      [The dsolve command to solve]

y =
E/R+exp(-R/L*t)*C1                  [MATLAB solution]

>> pretty(y)                        [The command pretty for a clearer display]
                R t
   E/R + exp(-----) C1
                 L
               [For given boundary conditions of E = 5V, R = 100Ω, L = 100 mH at t = 0, i= 0.01A]
>>y=dsolve('Dy=(5-100*y)/100','y(0)=0.01','t')

y =
1/20-1/25*exp(-t)                   [The MATLAB solution output]

>> ezplot(1000*y,[0,4])             [The plot command for generating Figure 5]
>> grid
```

Figure 5: The graph of the equation 50-40exp(-t)

Conclusion

The *Symbolic Math Toolbox* in MATLAB is an effective and accessible tool in a balanced portfolio of mathematics teaching tools for solving particular types of differential equations where there is an exact, analytic solution to be found. With its accompanying powerful plotting functions it provides additional support for presenting the concepts and theorems in an appetising and concrete manner, such as to make applications seem natural and encouraging the students to an explorative method of learning.

References

[1] Clements R R, Teaching the Numerical Solution of ODEs Using Spreadsheets, *Maths&Stats* v6, n2, May 1995, also at http://w3.bham.ac.uk/ctimath/reviews/may95/odes.pdf

[2] Polking J C & Arnold D, *Ordinary Differential Equations using MATLAB* (1997), (2nd ed), Prentice Hall

[3] Bishop O, *Understanding Electrical & Electronics Maths* (1993), Newnes

[4] Lindfield and Penny J, *Numerical Methods Using MATLAB* (1995), Ellis Horwood