

# MATH1052 Matlab Module 6: Runge Kutta method for ODEs and Systems

## 1. How the labs run

**Come in and sit down:** at a computer in the lab 67-542.

**Write down:** in this workbook any tips, tricks and hints about using Matlab.

**Work through:** the exercises in this booklet. If you get stuck, there are Hints and Solutions on the webpage.

**Use Matlab to solve:** the problems on your weekly tutorial sheet.

**Work on:** the assignment questions using the Matlab skills that you have learned

## 2. Runge-Kutta method

In the previous module numerical solutions to differential equations were achieved by a relatively simple algorithm: Euler's method. Much more sophistication is possible and the Runge-Kutta method is one scheme which produces a far more accurate estimate of the solution to an ODE.

Euler's method is termed a first order approximation (because the approximation essentially involves a first order Taylor expansion). The Runge-Kutta method is halfway between a 4<sup>th</sup> order and 5<sup>th</sup> order method. Consequently, the inbuilt Matlab program to run the method is called **ODE45**.

The syntax to solve an initial value problem with the ODE45 routine in Matlab is:

- Mathematical formulation:  $\frac{dy}{dt} = f(t, y), \quad y(0) = y_0 \quad 0 \leq t \leq T$
- Matlab formulation: `>> ode45('func', [0,T], y0)`

where **func** is a Matlab M-file function which accepts variables  $(t,y)$  and returns  $f(t,y)$ .

H1

**Example:** Use ODE45 to find a numerical solution to  $\frac{du}{dt} = ku(1-u)$ , over the time interval  $[0,100]$  where  $k = 0.2$  and  $u(0) = 0.001$ . This equation is a common modeling tool for epidemiology, growth of marketshare and population dynamics.

To solve the problem:

*Firstly* construct a function representing the right-hand-side of the equation. For example, if we call it **RHS.m**, then it will be something like:

```
function output = RHS(t,u)
% Evaluates the rhs of the ode
k = 0.2;
output = k*u*(1-u);
```

*Secondly*, call the ODE45 procedure from the *Command Window*. You could alternatively store the commands in an M-file and run them that way.

```
>> ode45('RHS' , [0,100] , 0.001)
```

Thirdly, to visualize the solution, you can produce a tabulated data or a graph. The procedure ODE45 should produce a graph for you automatically. Alternatively, run the program again with the command:

```
>> [t,u] = ode45('RHS' , [0,100] , 0.001);
```

The semicolon will suppress output. The solution will go into the vectors  $t$  and  $u$ . Now, to view the solution as data or graphically call, at the *Command Window*,

```
>> [t,u]
```

or

```
>> plot(t,u)
```

---



---

S1 **Exercise:** Find the initial condition  $y_0$ , so that the initial value problem:

$$\frac{du}{dt} = y(t - y), \quad y(0) = y_0,$$

satisfies  $y(1) = 0.5$ . You might use trial-and-error or a more sophisticated method.

---



---



---

### 3. Runge-Kutta method for systems of ODEs

The numerical schemes have so far solved first order differential equations. However, it is possible to solve second order ODEs, for example

$$\frac{d^2 y}{dt^2} + \sin(y) = 0, \quad y(0) = 1, \quad dy/dt(0) = 0 \quad (1)$$

and coupled systems of ODEs, for example

$$\begin{aligned} \frac{dx}{dt} &= x + y \\ \frac{dy}{dt} &= y - 3x \end{aligned} \quad x(0) = 1, \quad y(0) = -1 \quad (2)$$

#### (a) Systems of ODEs

We will start with *systems* of equations. The syntax remains very similar to the scalar case.

H2 **Example:** Solve the system (2) with ODE45 over  $0 \leq t \leq 10$ .

This time, the function for the right hand side must accept variables  $t$  and  $(x,y)$ . The form of ODE45 requires that the  $(x,y)$  variables arrive as a single vector, and we call it  $v$  here. The function should also return a (column) vector with two components, as the right hand side has two parts.

```

function output = systemRHS(t,v)
% Evaluates the rhs of the coupled ode system
x = v(1);
y = v(2);
output1 = x + y;
output2 = y - 3*x;
output=[output1; output2];

```

Then at the *Command Window*, again call ODE45:

```
>> [t,v] = ode45('systemRHS' , [0,10] , [1, -1]);
```

This time, the initial conditions  $x(0)=1$  and  $y(0)=-1$  are described by a *vector*. The output will be a vector of timesteps,  $t$ , as well as vector  $v$  which describes  $x$  and  $y$ .

If you want to see  $x$  against time, then call

```
>> plot(t, v(:,1))
```

If you want to see  $y$  against time, then call

```
>> plot(t, v(:,2))
```

If you want to see  $x$  against  $y$  (called a phase plot) then call

```
>> plot(v(:,1), v(:,2))
```

---



---



---

**S2** **Exercise:** Find a numerical solution for the following ODE over  $0 \leq t \leq 20$ .

$$\begin{aligned} \frac{dx}{dt} &= (1 - 0.1y)x \\ \frac{dy}{dt} &= (-1 + 0.2x)y \quad x(0) = 20, y(0) = 20 \end{aligned}$$

---



---



---

### (b) Second order ODEs

Higher order ODEs must be converted into a system of ODEs first. They can then be approached with the ODE45 routine.

**H3** **Example:** Solve the second order differential equation:

$$\ddot{x} + 2\dot{x} + x = 0, \quad x(0) = 1, \dot{x}(0) = 3$$

To decompose this into two first order equations, we introduce a new function  $y$ :

$$\dot{x} = y$$

and then

$$\dot{y} = \ddot{x} = -2\dot{x} - x = -2y - x$$

so the system of equations, with associated initial conditions, is:

$$\begin{aligned} \dot{x} &= y \\ \dot{y} &= -2y - x, \quad x(0) = 1, \quad y(0) = 3 \end{aligned}$$

This system can then be solved the same way as the previous section. The function for the right hand side in this case would appear as:

```
function output = secondorderRHS(t,v)
% Evaluates the rhs of the second order ode system
x = v(1);
y = v(2);
output1 = y;
output2 = -2*y - x;
output=[output1; output2];
```

and the command line would be similar to earlier:

H4

```
>> [t,v] = ode45('secondorderRHS', [0,10], [1, 3]);
```

To view the resultant solution, we are interested in  $x(t)$  for each  $t$ :

```
>> plot(t,v(:,1))
```

---

---

---

S3

**Exercise:** Find a numerical solution to the forced second order ODE:

$$\ddot{x} + \sin(x) = \cos(2t), \quad x(0)=0, \quad \dot{x}(0) = 1, \quad 0 \leq t \leq 100.$$

---

---

---

## 4. Index

In this Lab you learned:

- the ODE45 Runge-Kutta solution routine
- solving systems of equations with ODE45
- decomposing higher order ODEs into systems, and solving them with ODE45.

Copyright © Elliot Tonkes 2003