2007 Summer College on Plasma Physics

30 July - 24 August, 2007

Numerical methods and simulations. Lecture 2: Simulation of ordinary differential equations.

B. Eliasson

Institut fur Theoretische Physik IV
Ruhr-Universitaet Bochum
Bochum, Germany

Summer College on Plasma Physics
ICTP
Trieste, Italy
30 July – 24 August 2007

Course: **Numerical methods and simulations**

Tutor: Bengt Eliasson
E-mail: bengt@tp4.rub.de
Internet address: `www.tp4.rub.de/∼bengt`

Lecture 2: Simulation of ordinary differential equations

- Will continue to write programs in Matlab

- Learn how to write loops and functions

- Simulate ordinary differential equations

   - Lotka-Volterra equation: A model for predator-prey systems (for example foxes and rabits).

   Assignment: Modify the program to solve the Lorentz system, which is used to study chaos (the Lorentz attractor).

To solve a differential equation numerically in time, we discretize the time so that the solution is only defined at discrete points in time, separated by the timestep Δt, Then we find the solution at later times by using the solution at earlier times.

The Runge-Kutta algorithm:

   To solve the time-dependent equation

$$\frac{d\mathbf{x}(t)}{dt} = \mathbf{F}(\mathbf{x}(t), t), \tag{1}$$

   with the initial condition $\mathbf{x} = \mathbf{x}_0$, where $\mathbf{F}$ is a function (which may be vector-valued) of $\mathbf{x}$ and t, we use the Runge-Kutta algorithm

(1) $\mathbf{F}_1 \leftarrow \mathbf{F}(\mathbf{x}, t)$

(2) $\mathbf{F}_2 \leftarrow \mathbf{F}(\mathbf{x} + \Delta t \mathbf{R}_1/2, t + \Delta t/2)$

(3) $\mathbf{F}_3 \leftarrow \mathbf{F}(\mathbf{x} + \Delta t \mathbf{R}_2/2, t + \Delta t/2)$

(4) $\mathbf{F}_4 \leftarrow \mathbf{F}(\mathbf{x} + \Delta t \mathbf{R}_3, t + \Delta t)$

(5) $\mathbf{x} \leftarrow \mathbf{x} + (\Delta t/6)(\mathbf{R}_1 + 2\mathbf{R}_2 + 2\mathbf{R}_3 + \mathbf{R}_4)$

This gives the solution $\mathbf{x}$ at time $t + \Delta t$. The steps (1)–(5) are repeated with the new values of $\mathbf{x}$ until we have reached the end of the simulation.

Example: Write a program that solves the Lotka-Volterra equation

$$\frac{dx}{dt} = ax - bxy \tag{2}$$

$$\frac{dy}{dt} = -cy + dxy \tag{3}$$

with the initial conditions $x(0) = y(0) = 2$, and where the constants are $a = 1.5$, $b = 1$, $c = 3$ and $d = 1$. Plot the result. Here

$$\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix} \tag{4}$$

and

$$\mathbf{F}(\mathbf{x}, t) = \begin{bmatrix} ax - bxy \\ -cy + dxy \end{bmatrix} \tag{5}$$

We now write the main program "main.m" for our simulation. Select from the Matlab menu *File / New / M-file*, then from the menu of the new window, choose *File / Save as*, and Filename: `main.m`

We now write the program in the file:
===

```
% Main program main.m
clear %Clear all variables

Nt=1000; % The number of time steps

x=2; %Initial values
y=2; z=2;

t=0; % Start at time t=0
```

```
dt=0.02; % The time step


% Create vectors to store results
xvec=zeros(1,Nt);
yvec=zeros(1,Nt);
tvec=zeros(1,Nt);

% Store initial valuse
tvec(1)=0;
xvec(1)=x;
yvec(1)=y;

for j=1:Nt
  % Call the Runge-Kutta routine.
  [x,y]=RungeKutta(x,y,t,dt);
  t=t+dt;

  %Store the results
  tvec(j+1)=t;
  xvec(j+1)=x;
  yvec(j+1)=y;
end

%Plot the results
subplot(2,1,1) plot(tvec,xvec)
subplot(2,1,2) plot(tvec,yvec)
```

==

Then choose from the menu *File / Save*.

Now we write the Runge-Kutta subroutine: From the Matlab menu, choose *File / New / M-file*, then choose from the menu of the new file *File / Save as*, and Filename: `RungeKutta.m`

In the new file, we write

==

```
% Runge-Kutta alogrithm, RungeKutta.m
function [x,y]=RungeKutta(x,y,t,dt)

  [F1x,F1y]=F(x,y,t);
```

```
[F2x,F2y]=F(x+0.5*dt*R1x,y+0.5*dt*R1y,t+0.5*dt);
[F3x,F3y]=F(x+0.5*dt*R2x,y+0.5*dt*R2y,t+0.5*dt);
[F4x,F4y]=F(x+dt*R3x,y+dt*R3y,t+dt);

x=x+dt/6*(F1x+2*F2x+2*F3x+F4x);
y=y+dt/6*(F1y+2*F2y+2*F3y+F4y);
```

==

Choose *File / Save.*

Finally, we write the function that defines the right-hand side of the differential equation. From the Matlab menu, choose *File / New / M-file*, and then *File / Save as* and Filename: `F.m`

In the new file, we write

==

```
% The right-hand side of the differential equation, F.m
function [Fx,Fy]=F(x,y,t)

%Lotka-Volterra equation

a=1.5;
b=1;
c=3;
d=1;

Fx=a*x-b*x*y;
Fy=-c*y+d*x*y;
```
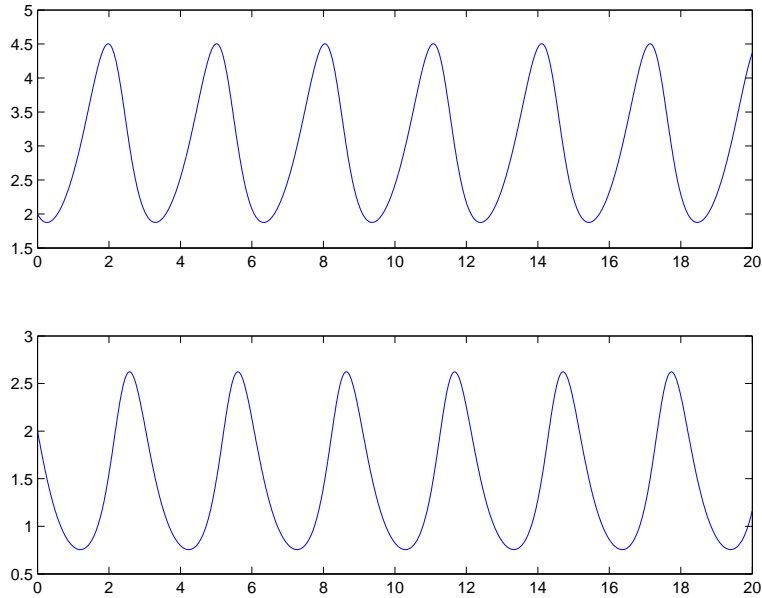
==

Choose *File / Save.*

We now run the program. In the Matlab window, type in `main` (Enter).

The result is

The upper plot (x) shows the prey (cheeps or rabbits), and the lower plot (y) shows the predators (wolves or foxes) that catch the prey. When there are many rabbits, the foxes also increase, and when the number of rabbits decrease the number of foxes also decrease ...

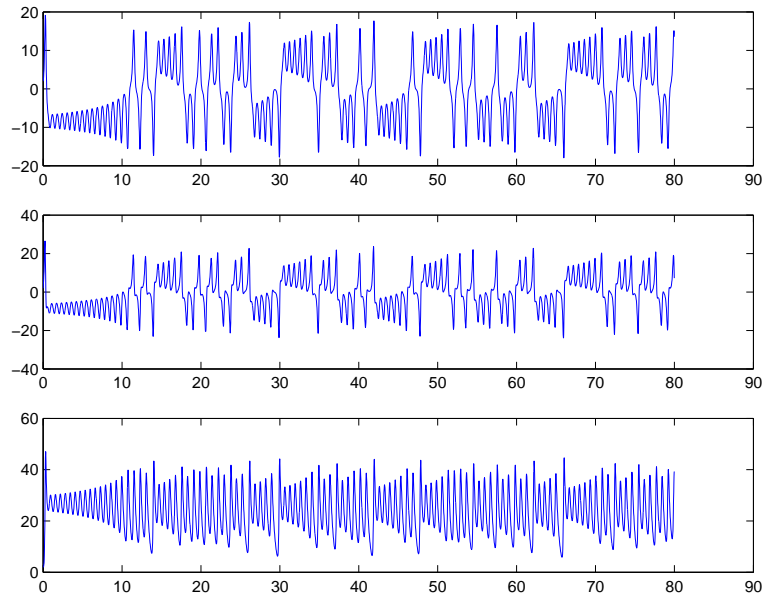**Assignment:** Modify the program so that it solves the Lorentz system of equations:

$$\frac{dx}{dt} = 10(y - x) \tag{6}$$

$$\frac{dy}{dt} = 28x - y - xz \tag{7}$$

$$\frac{dz}{dt} = xy - \frac{8}{3}z \tag{8}$$

with the initial conditions $x = y = z = 2$. Solve the system from $t = 0$ to $t = 80$ and plot x, y and z in three subpanels of the same figure.

The result is

We see that the solution exhibits a very irregular and chaotic behavior. The Lorentz system has been studied extensively both numerically and theoretically. The solution is attracted to a fractal set in x, y, z space, called a "strange attractor".