CO24: Chaos

T.P. Galligan
Brasenose College
Trinity Term 2017

1 Abstract

We investigate solutions to the Lorenz equations by means of the Runge-Kutta method, and illustrate the generation of chaos. We test the stability of these equations, and demonstrate the *butterfly effect* phenomenon.

2 Introduction

Chaos theory examines dynamical systems that are highly sensitive to their initial conditions; small differences in these can yield wildly diverging outcomes for such systems - a response often referred to as the *butterfly effect*. This renders the long-term prediction of the behaviour of these systems impossible. The theory of chaos is of active interest in modern physics. For example, the turbulent fluctuations of fusion plasmas are of great interest in magnetohydrodynamics, and an understanding of this chaotic behaviour is vital if mankind is to successfully operate a fusion reactor[2]. In this report, we demonstrate the behaviour of a simple chaotic system described by the Lorenz equations.

3 Theory

As detailed in [3], the Lorenz equations for the time dependent variables y_1, y_2, and y_3 are

$$\frac{\mathrm{d}y_1}{\mathrm{d}t} = a(y_2 - y_1) \tag{1}$$
$$\frac{\mathrm{d}y_2}{\mathrm{d}t} = ry_1 - y_2 - y_1y_3 \tag{2}$$
$$\frac{\mathrm{d}y_3}{\mathrm{d}t} = y_1y_2 - by_3 \tag{3}$$

Looking for stationary solutions (i.e. $(\dot{y_1}, \dot{y_2}, \dot{y_3}) = \vec{0}$), we find three solutions:

$$y_1 = y_2 = \pm\sqrt{b(r-1)} \tag{4}$$
$$y_3 = r - 1 \tag{5}$$
$$y_1 = y_2 = y_3 = 0 \tag{6}$$

However, the way in which the system approaches these solutions is dependent on r:

1. if $r < 1$ the only solution is $\vec{y} = (y_1, y_2, y_3) = \vec{0}$
2. if $1 < r < \sim 24$, the system will approach one of the other two solutions, but will never reach it exactly.
3. if $r > \sim 24$, then the system will converge on neither of the two other solutions, but instead will oscillate seemingly randomly between their neighbourhoods. This behaviour generates a *strange attractor*.

The strange attractor arises in a simplified system of equations describing the two dimensional flow of a fluid with similar characteristics to a simplified atmospheric system. In the 1960s, Lorenz accidentally discovered the chaotic behaviour of this system under the conditions detailed above.

4 Method

We use a Runge-Kutta method for solving Eq. (1)-(3). The Runge-Kutta methods are a family of iterative methods. We use the most common of these methods, known as the RK4 method[4]. This approach calculates a weighted average of the slopes in a small interval in order to approximate the value of the function at a later point. This approach is iterated over a large timestep interval in order to estimate the solution of the system of equations. With each iteration, the *local* truncation error is $\mathcal{O}(\delta t^5)$, while the *global* error is $\mathcal{O}(\delta t^4)$[4]. This results in a very high degree of accuracy, compared to lower order methods such as the popular Euler's method.
In principle, RK4 can be applied to an arbitrarily large system of non-linear differential equations ,provided we have the right boundary conditions. However in practice, this would greatly increase the computing time. In this situation, other methods such as matrix diagonalisation[5].

5 Results

We implement the RK4 algorithm for a variety of r-regimes, with $n = 10^4$, $\delta t = 0.01$, and $\vec{y} = (1, 2, 3)$:

r = 0.5

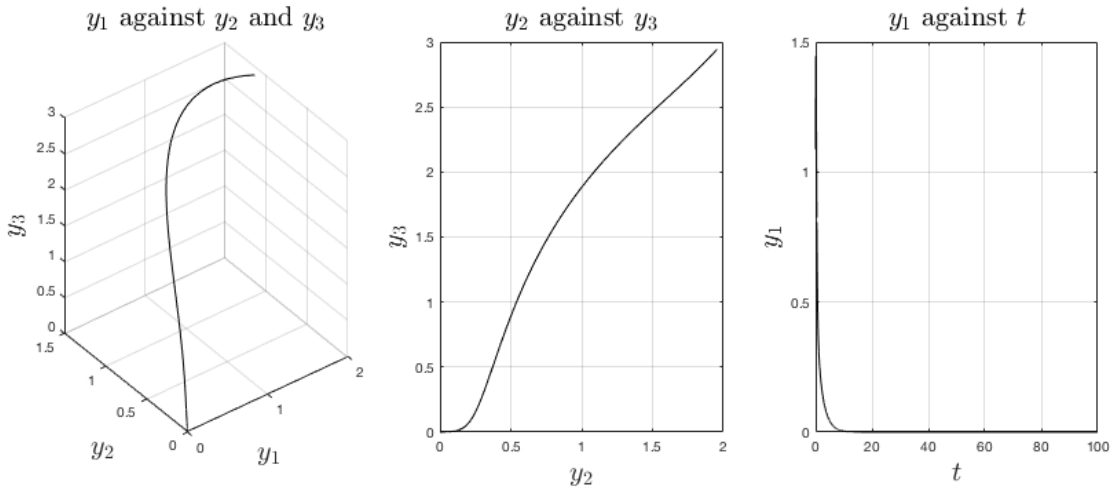

Figure 1: Results from r=0.5

When $r < 1$, the only real solution lies at (0,0,0).

r = 5

In this regime, as expected we see the system approach a solution, but never absolutely reach it; indeed it will spiral inwards towards the point indefinitely.

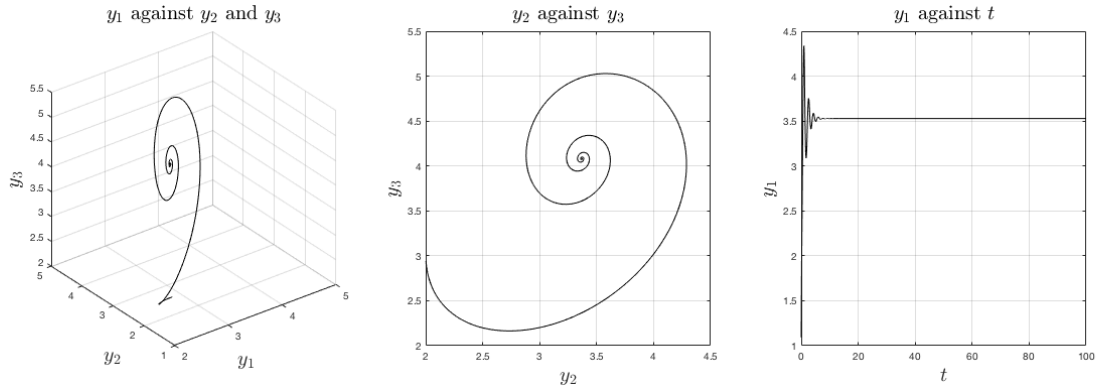

Figure 2: Results from r=5

r = 30

When r is large, the system's behaviour becomes chaotic, and the point wanders between the two non-zero solutions.

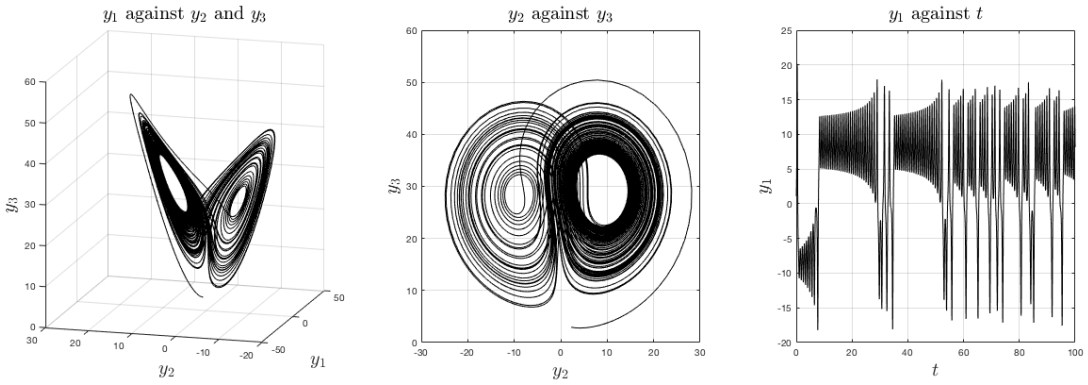

Figure 3: Results from r=30

Butterfly Effect

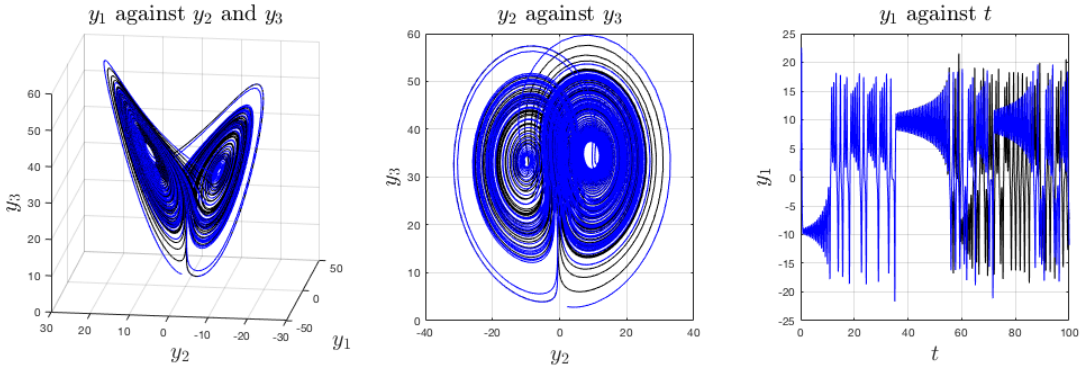

Figure 4: Deviations due to a change in 1 part in 10^{14} in y_1

Changing y_1 by 1 part in 10^{14}, we can see that although the motion is intitially similar for the two cases, at $t \approx 55s$, the systems begin to diverge until very quickly their behaviour is completely uncorrelated.

6 Conclusions

We have demonstrated the chaotic behaviour of the Lorenz equations by use of the fourth-order Runge-Kutta (RK4) method for solving coupled ordinary differential equations. While RK4 is simple to employ and has generally smaller local and global errors than many other iterative numerical methods, it suffers from long computing times for more complex systems. The so-called 'butterfly effect' was demonstrated from a very small offset in intial conditions; a change of 1 in 10^{14} comparable to changing the diameter of the solar system by the length of a mobile phone. Within only 60 'seconds', the two systems had completely diverged, demonstrating the unpredictable, but nonetheless deterministic, nature of chaotic systems. The RK4 method could be applied to a variety of important problems in physics, including the hydrostatic equilibrium equations that govern the behaviour of white dwarves.

References

[1] National Institute of Standards and Technology (NIST)
http://physics.nist.gov/cgi-bin/cuu/Value?ryd

[2] Gyrokinetic simulations of turbulent transport: size scaling and chaotic behaviour, *Villard* et al. *Plasma Physics and Controlled Fusion, IOP Science*, 2010.

[3] C024 Lab Script, Department of Physics, University of Oxford https://www-teaching.physics.ox.ac.uk/practical_course/scripts/srv/local/rscripts/trunk/Computing/C024/C024.pdf

[4] *Wikipedia* article on Runge-Kutta Methods. https://en.wikipedia.org/wiki/Runge%E2%80%93Kutta_methods

[5] http://www.cds.caltech.edu/~murray/courses/cds101/fa02/precourse/delvecchio-26sep02.pdf
http://www-teaching.physics.ox.ac.uk/

7 Appendix

7.1 Testing r regimes

```
%C024: Chaos
%Tom Galligan
%University of Oxford - HT17

%enter input values
y1= input('Enter y1(0): ');
y2 = input('Enter y2(0): ');
y3 = input('Enter y3(0): ');
dt = 0.01;
r = input('Enter r: ');
n = input('Enter n: ');
a = 10;
b = 8/3;
y_0 = [y1,y2,y3];%initial conditions
t = zeros(n,1);

f_0 = [a*(y_0(2)-y_0(1)),r*y_0(1)-y_0(2)-y_0(1)*y_0(3),y_0(1)*y_0(2)-b*
   y_0(3)]; %find RHS of system of equations as a vector (using initial
   conditions)
Y_1 = zeros(n,1);
Y_2 = zeros(n,1);
Y_3 = zeros(n,1);



 for i=1:n %run algorithm using a for loop
     y_1 = y_0 + 0.5*f_0*dt;
     f_1 = [a*(y_1(2)-y_1(1)),r*y_1(1)-y_1(2)-y_1(1)*y_1(3),y_1(1)*y_1
        (2)-b*y_1(3)];
     y_2 = y_1 + 0.5*f_1*dt;
     f_2 = [a*(y_2(2)-y_2(1)),r*y_2(1)-y_2(2)-y_2(1)*y_2(3),y_2(1)*y_2
        (2)-b*y_2(3)];
     y_3 = y_2 + f_2*dt;
     f_3 = [a*(y_3(2)-y_3(1)),r*y_3(1)-y_3(2)-y_3(1)*y_3(3),y_3(1)*y_3
        (2)-b*y_3(3)];
```

```
      y_4 = y_0 + (1/6)*(f_0 + 2*f_1 + 2*f_2 + f_3)*dt;


      Y_1(i) = y_4(1);
      Y_2(i) = y_4(2);
      Y_3(i) = y_4(3);
      y_0 = y_4; %restart the for loop by setting the first value of the
         sequence equal to the last.
      t(i+1)= t(i)+dt;


 end
t(n+1)=[]; %remove last entry of t


figure %plot the graphs
subplot(1,3,1)

    plot3(Y_2,Y_1,Y_3, 'Color','k') %plot the variables against
       eachother
    grid on %add a grid to the plot for easy visual analysis
    xlabel('$y_1$', 'Interpreter','latex','fontsize',20)
    ylabel('$y_2$','Interpreter','latex','fontsize',20)
    zlabel('$y_3$','Interpreter','latex','fontsize',20)
    title('$y_1$ against $y_2$ and $y_3$','Interpreter','latex','
       fontsize',20)
subplot(1,3,2)

    plot(Y_2,Y_3,'Color','k')
    grid on %add a grid to the plot for easy visual analysis
    xlabel('$y_2$', 'Interpreter','latex','fontsize',20)
    ylabel('$y_3$','Interpreter','latex','fontsize',20)
    title('$y_2$ against $y_3$','Interpreter','latex','fontsize',20)

subplot(1,3,3)
    plot(t,Y_1,'Color','k')
    grid on %add a grid to the plot for easy visual analysis
    xlabel('$t$', 'Interpreter','latex','fontsize',20)
    ylabel('$y_1$','Interpreter','latex','fontsize',20)
    title('$y_1$ against $t$','Interpreter','latex','fontsize',20)
```

7.2 Demonstrating the butterfly effect

```
%CO24: Chaos - butterfly effect demo
%Tom Galligan
%University of Oxford - HT17



%%Initial version
%enter input values
y1= 1;
y2 = 2;
y3 = 3;
dt = 0.01;
r = 35;
```

```
n = 1e4;
a = 10;
b = 8/3;
y_0 = [y1,y2,y3];%initial conditions
t = zeros(n,1);

f_0 = [a*(y_0(2)-y_0(1)),r*y_0(1)-y_0(2)-y_0(1)*y_0(3),y_0(1)*y_0(2)-b*
   y_0(3)]; %find RHS of system of equations as a vector (using initial
   conditions)
Y_1 = zeros(n,1);
Y_2 = zeros(n,1);
Y_3 = zeros(n,1);



 for i=1:n %run algorithm using a for loop
     y_1 = y_0 + 0.5*f_0*dt;
     f_1 = [a*(y_1(2)-y_1(1)),r*y_1(1)-y_1(2)-y_1(1)*y_1(3),y_1(1)*y_1
        (2)-b*y_1(3)];
     y_2 = y_1 + 0.5*f_1*dt;
     f_2 = [a*(y_2(2)-y_2(1)),r*y_2(1)-y_2(2)-y_2(1)*y_2(3),y_2(1)*y_2
        (2)-b*y_2(3)];
     y_3 = y_2 + f_2*dt;
     f_3 = [a*(y_3(2)-y_3(1)),r*y_3(1)-y_3(2)-y_3(1)*y_3(3),y_3(1)*y_3
        (2)-b*y_3(3)];
     y_4 = y_0 + (1/6)*(f_0 + 2*f_1 + 2*f_2 + f_3)*dt;


     Y_1(i) = y_4(1);
     Y_2(i) = y_4(2);
     Y_3(i) = y_4(3);
     y_0 = y_4; %restart the for loop by setting the first value of the
        sequence equal to the last.
     t(i+1)= t(i)+dt;


 end
t(n+1)=[];

%%Modified version

%enter input values
z1= 1+1e-14;
z2 = 2;
z3 = 3;
dt = 0.01;
r = 35;
n = 1e4;
a = 10;
b = 8/3;
z_0 = [z1,z2,z3];%initial conditions
t = zeros(n,1);

f_0 = [a*(z_0(2)-z_0(1)),r*z_0(1)-z_0(2)-z_0(1)*z_0(3),z_0(1)*z_0(2)-b*
   z_0(3)]; %find RHS of system of equations as a vector (using initial
```

```
   conditions)
Z_1 = zeros(n,1);
Z_2 = zeros(n,1);
Z_3 = zeros(n,1);



 for i=1:n %run algorithm using a for loop
     z_1 = z_0 + 0.5*f_0*dt;
     f_1 = [a*(z_1(2)-z_1(1)),r*z_1(1)-z_1(2)-z_1(1)*z_1(3),z_1(1)*z_1
        (2)-b*z_1(3)];
     z_2 = z_1 + 0.5*f_1*dt;
     f_2 = [a*(z_2(2)-z_2(1)),r*z_2(1)-z_2(2)-z_2(1)*z_2(3),z_2(1)*z_2
        (2)-b*z_2(3)];
     z_3 = z_2 + f_2*dt;
     f_3 = [a*(z_3(2)-z_3(1)),r*z_3(1)-z_3(2)-z_3(1)*z_3(3),z_3(1)*z_3
        (2)-b*z_3(3)];
     z_4 = z_0 + (1/6)*(f_0 + 2*f_1 + 2*f_2 + f_3)*dt;


     Z_1(i) = z_4(1);
     Z_2(i) = z_4(2);
     Z_3(i) = z_4(3);
     z_0 = z_4; %restart the for loop by setting the first value of the
        sequence equal to the last.
     t(i+1)= t(i)+dt;
     end

t(n+1)=[];
figure %plot the graphs
subplot(1,3,1)

    plot3(Y_2,Y_1,Y_3, 'Color','k') %plot the variables against
       eachother
    grid on %add a grid to the plot for easy visual analysis
    xlabel('$y_1$', 'Interpreter','latex','fontsize',20)
    ylabel('$y_2$','Interpreter','latex','fontsize',20)
    zlabel('$y_3$','Interpreter','latex','fontsize',20)
    title('$y_1$ against $y_2$ and $y_3$','Interpreter','latex','
       fontsize',20)
    hold on
    plot3(Z_2,Z_1,Z_3, 'Color','b') %plot the variables against
       eachother
subplot(1,3,2)

    plot(Y_2,Y_3,'Color','k')
    grid on %add a grid to the plot for easy visual analysis
    xlabel('$y_2$', 'Interpreter','latex','fontsize',20)
    ylabel('$y_3$','Interpreter','latex','fontsize',20)
    title('$y_2$ against $y_3$','Interpreter','latex','fontsize',20)
    hold on
    plot(Z_2,Z_3,'Color','b') %plot perturbed variables against
       eachother
subplot(1,3,3)
    plot(t,Y_1,'Color','k')
```

```
    grid on %add a grid to the plot for easy visual analysis
    xlabel('$t$', 'Interpreter','latex','fontsize',20)
    ylabel('$y_1$','Interpreter','latex','fontsize',20)
    title('$y_1$ against $t$','Interpreter','latex','fontsize',20)
    hold on
    plot(t,Z_1,'Color','b')
    hold off

figure
plot(Y_1,Z_1,'Color','k')
xlabel('$y_1$', 'Interpreter','latex','fontsize',20)
    ylabel('$z_1$','Interpreter','latex','fontsize',20)
    title('$z_1$ against $y_1$','Interpreter','latex','fontsize',20)
```