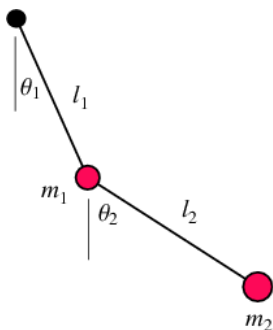


Project 3 - Double Pendulum and Chaos



1 Derivation of Equations of Motion

The following derivation is from <http://scienceworld.wolfram.com/physics/DoublePendulum.html>

The equations of motion for the double pendulum are given by

$$\begin{aligned} & (m_1 + m_2)L_1 \frac{d^2\theta_1}{dt^2} + m_2L_2 \frac{d^2\theta_2}{dt^2} \cos(\theta_1 - \theta_2) \\ & + m_2L_2 \left(\frac{d\theta_2}{dt} \right)^2 \sin(\theta_1 - \theta_2) + g(m_1 + m_2) \sin(\theta_1) = 0 \\ & m_2L_2 \frac{d^2\theta_2}{dt^2} + m_2L_1 \frac{d^2\theta_1}{dt^2} \cos(\theta_1 - \theta_2) \\ & - m_2L_1 \left(\frac{d\theta_1}{dt} \right)^2 \sin(\theta_1 - \theta_2) + m_2g \sin(\theta_2) = 0 \end{aligned}$$

If we use the variables

$$\begin{aligned} u_1 &= \theta_1(t) \\ u_2 &= \theta_1'(t) \\ v_1 &= \theta_2(t) \\ v_2 &= \theta_2'(t) \end{aligned}$$

we get the system

$$\begin{aligned}
 (m_1 + m_2)L_1 \frac{du_2}{dt} + m_2L_2 \frac{dv_2}{dt} \cos(u_1 - v_1) \\
 + m_2L_2 (v_2)^2 \sin(u_1 - v_1) + g(m_1 + m_2) \sin(u_1) &= 0 \\
 m_2L_2 \frac{dv_2}{dt} + m_2L_1 \frac{du_2}{dt} \cos(u_1 - v_1) \\
 - m_2L_1 (u_2)^2 \sin(u_1 - v_1) + m_2g \sin(v_1) &= 0 \\
 \frac{du_1}{dt} &= u_2(t) \\
 \frac{dv_1}{dt} &= v_2(t)
 \end{aligned}$$

Using the substitutions $a = (m_1 + m_2)L_1$, $b = m_2L_2 \cos(u_1 - v_1)$, $c = m_2L_1 \cos(u_1 - v_1)$, $d = m_2L_2$, $e = -m_2L_2 (v_2)^2 \sin(u_1 - v_1) - g(m_1 + m_2) \sin(u_1)$, and $f = m_2L_1 (u_2)^2 \sin(u_1 - v_1) - m_2g \sin(v_1)$ we can write this system as

$$a \frac{du_2}{dt} + b \frac{dv_2}{dt} = e \quad (1)$$

$$c \frac{du_2}{dt} + d \frac{dv_2}{dt} = f \quad (2)$$

$$\frac{du_1}{dt} = u_2(t) \quad (3)$$

$$\frac{dv_1}{dt} = v_2(t) \quad (4)$$

Equations (1) and (2) can be solved for du_2/dt and dv_2/dt by

$$\frac{du_2}{dt} = \frac{ed - bf}{ad - cb} \quad (5)$$

$$\frac{dv_2}{dt} = \frac{af - ce}{ad - cb} \quad (6)$$

Equations (3)-(6) are now in the form that Matlab can use.

Initial Conditions The initial conditions given in the reference are (angles given in terms of radians)

$$u_1(0) = 1.5$$

$$u_2(0) = 0.0$$

$$v_1(0) = 3.0$$

$$v_2(0) = 0.0$$

The physical parameters are given by

$$L_1 = 1$$

$$L_2 = 2$$

$$m_1 = 2$$

$$m_2 = 1$$

$$g = 9.8$$

2 Matlab Project 3.

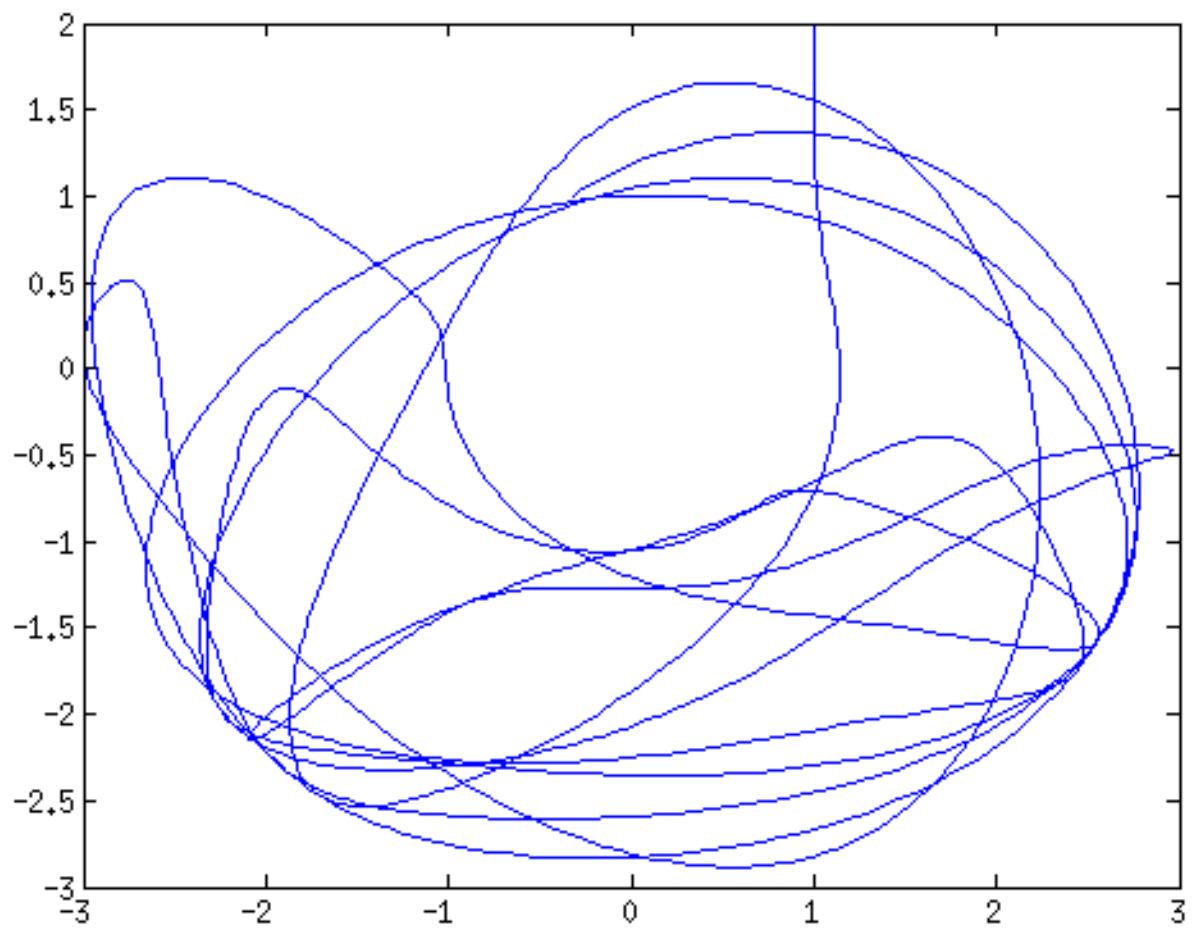
Using the system of equations given above, and the parameters given above, reproduce the animated graphic contained in the reference.

Find the component plots $\theta_1(t) = u_1(t)$ vs t and $\theta_2(t) = v_1(t)$ vs t .

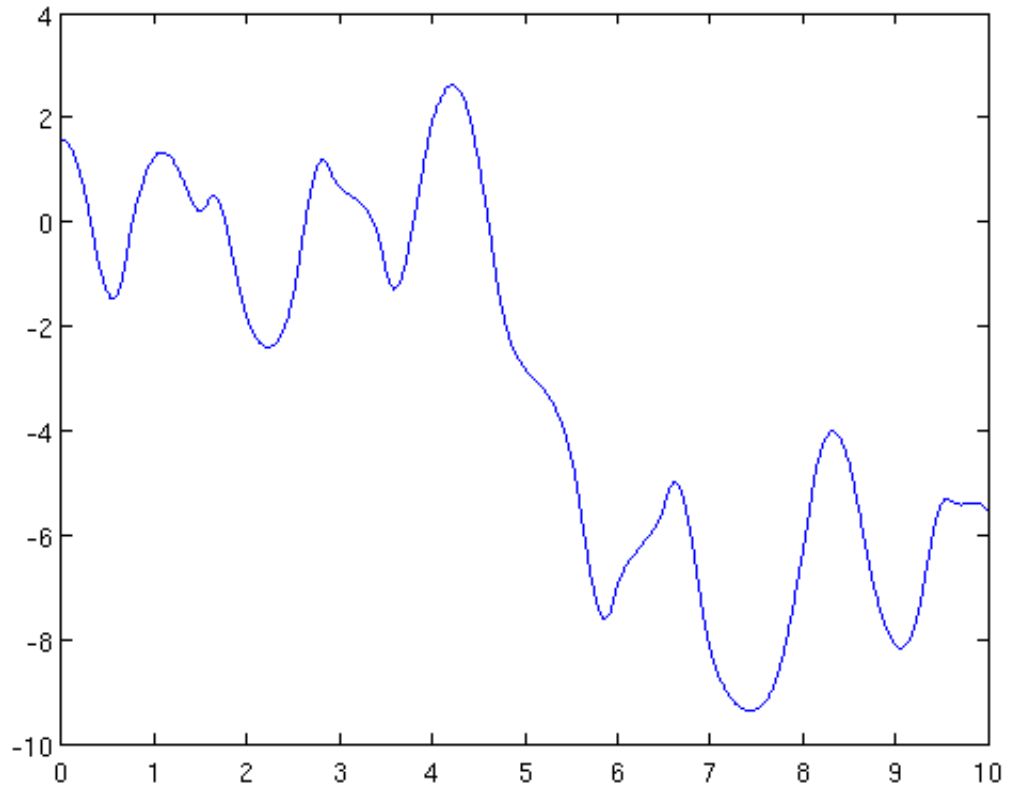
Examine several other initial conditions, and check for chaotic orbits.

3 Writeup:

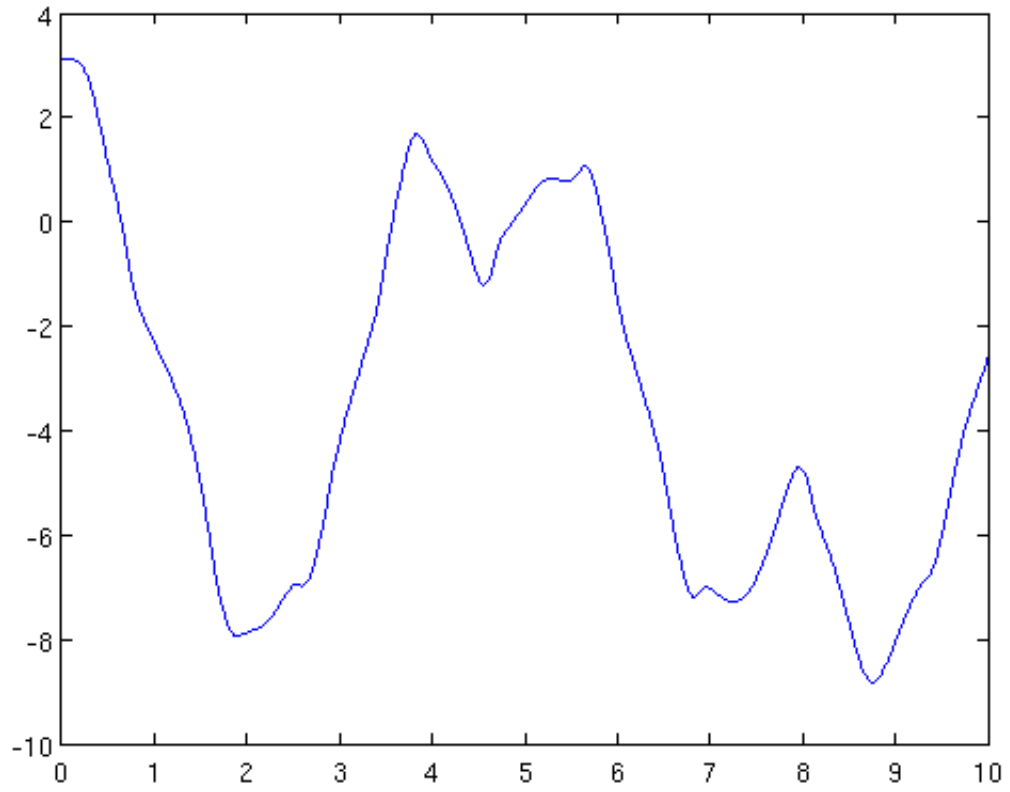
With the initial conditions given above, we have the following parametric plot, $(x_2(t), y_2(t))$



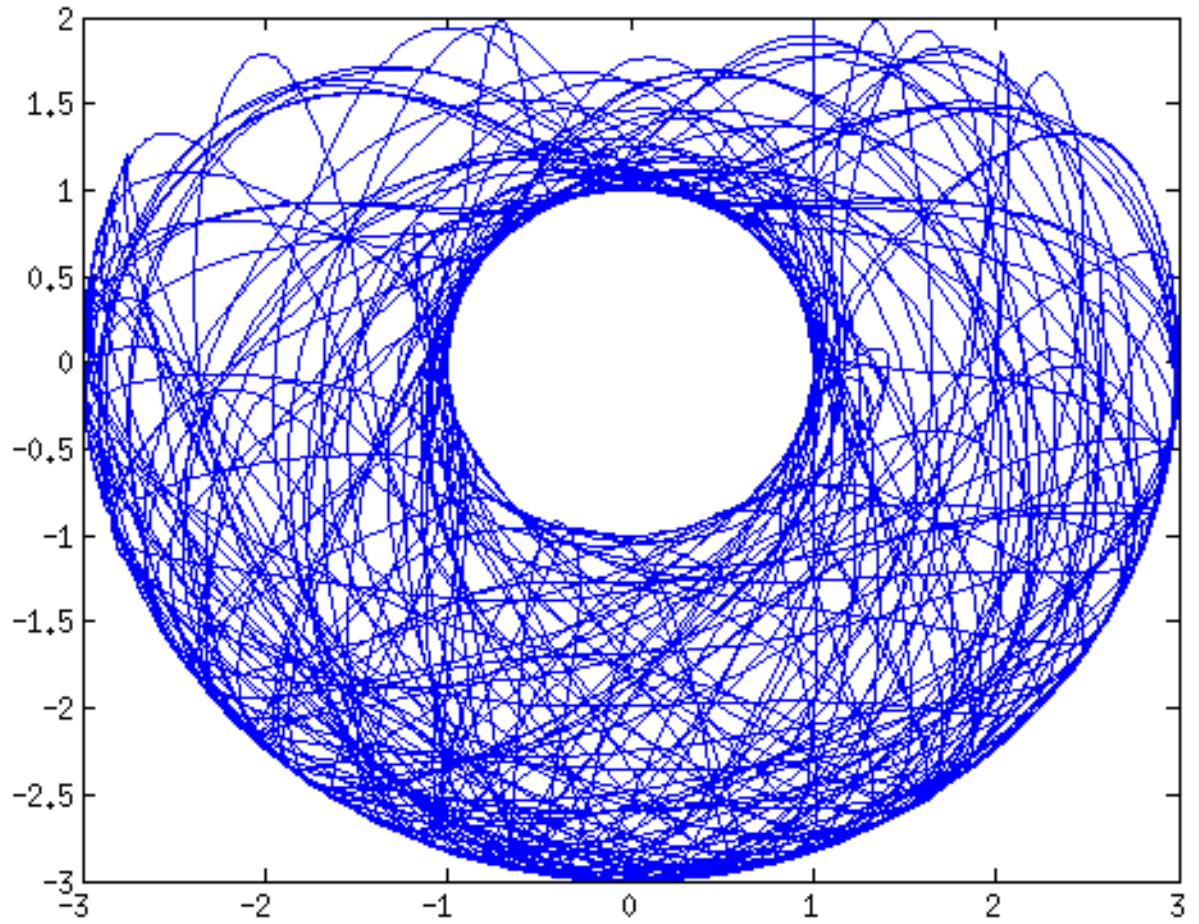
and the component plots $\theta_1(t)$ vs t



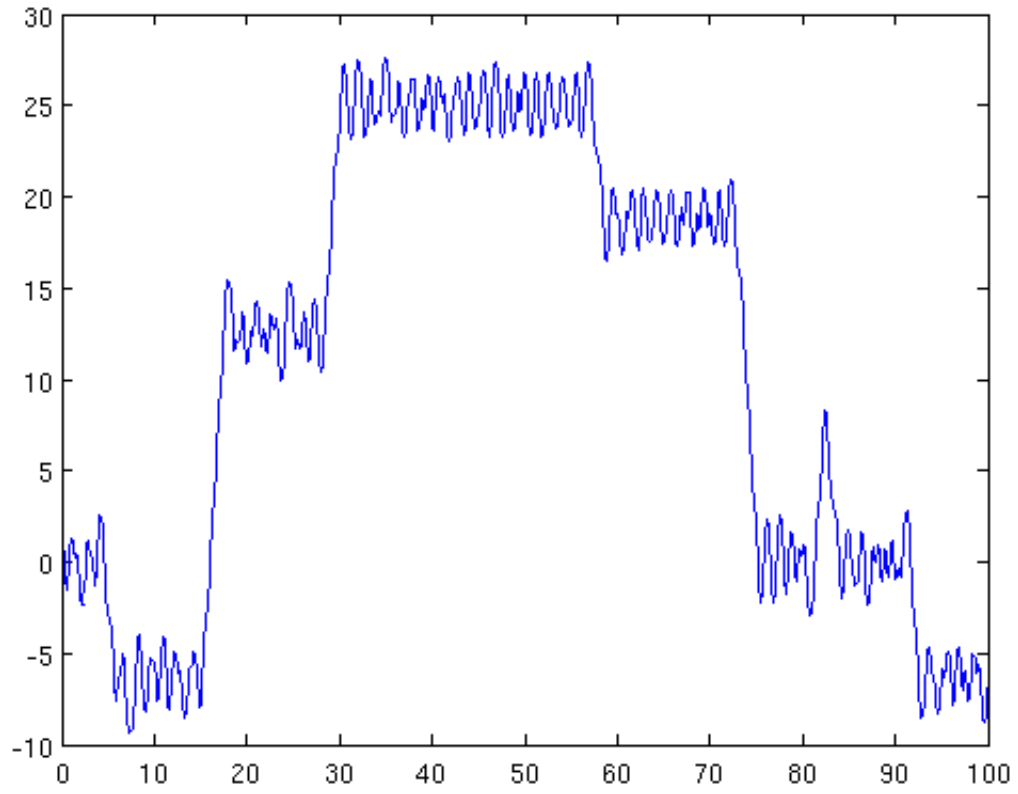
and the component plots $\theta_2(t)$ vs t



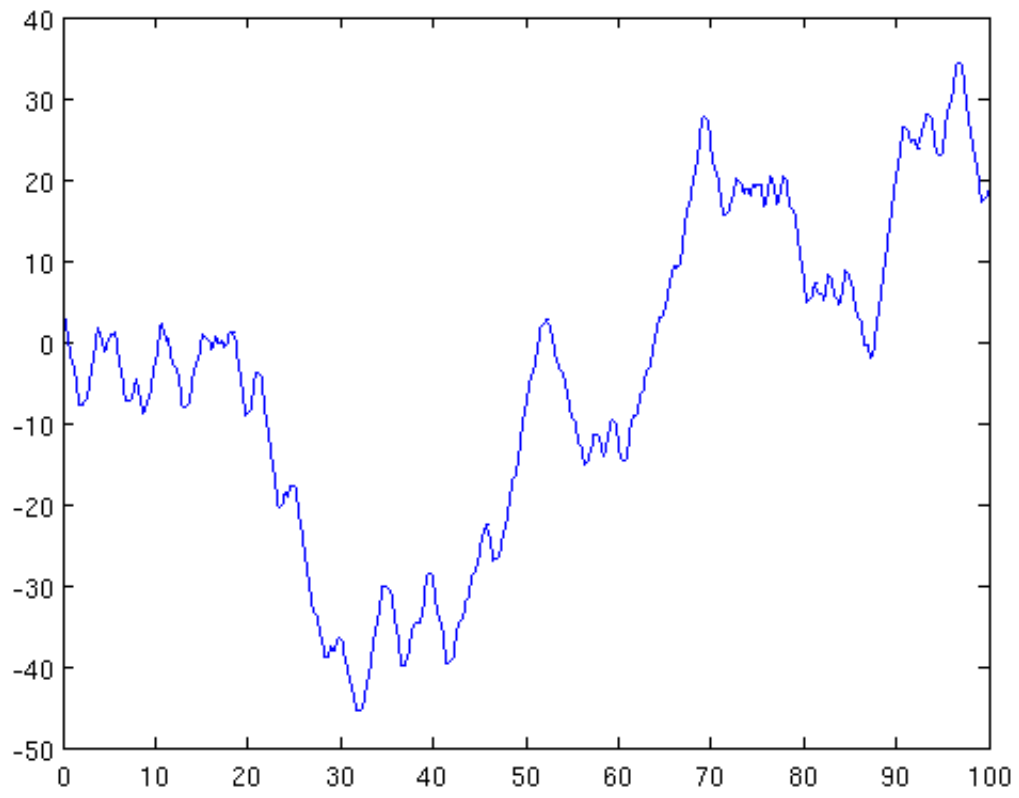
Integrating further, till $T = 100$, we obtain the parametric plot



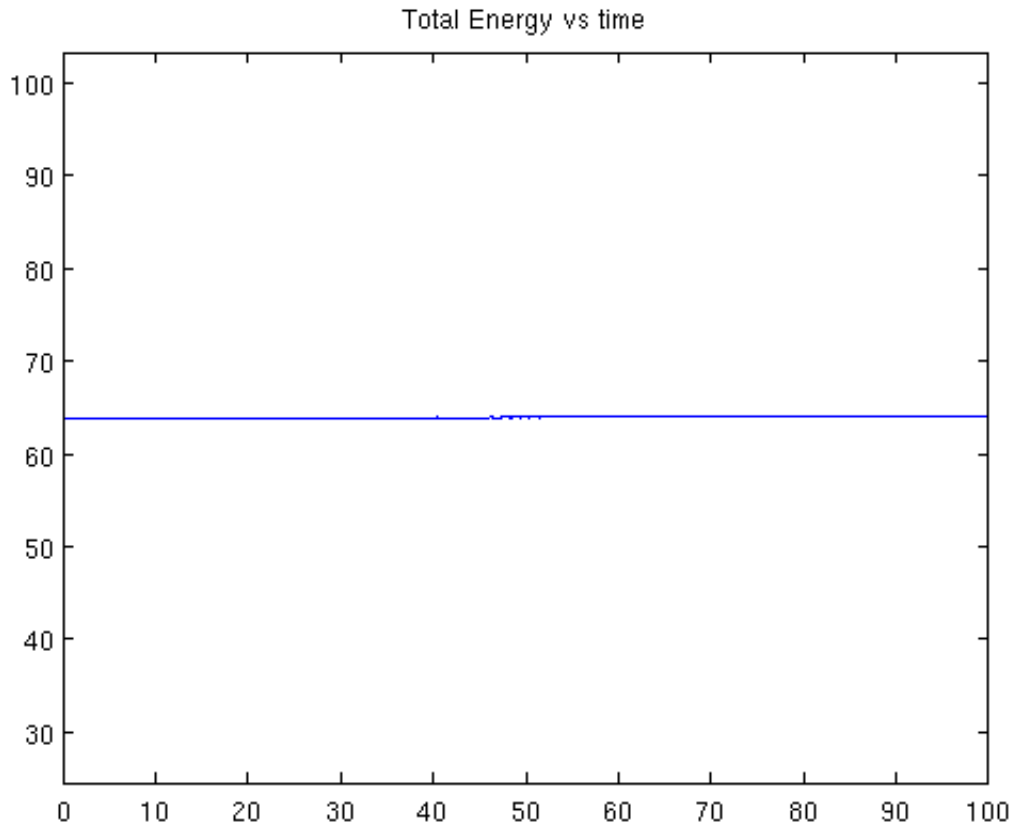
and the component plots:



and



As a check on the accuracy of the numerical integration, we plot the total energy $E = T + V$ against time



4 Summary and Conclusions

Although it is clear that the motion is not very periodic, it is difficult to say that it is chaotic without a strict definition of “chaos.” In fact, if the initial angular displacements are small, the motion is not chaotic. But as the energy of the system increases, the motion of the end of the pendulum becomes more and more complex (chaotic).

Appendix - Code Listings

```
function [t,x2,y2] = double_pendulum_demo(time)

% setup up parameters
m1=2; m2=1; L1=1; L2=2;g=32.0;
% redefine relative tolerance
options=odeset('RelTol',1.0e-6);
% call runge kutta solver
[t,y]=ode45('double_pendulum',[0 time],[1.57;0.0;3.14;0.0],options);
% calculate potential energy
V = -(m1+m2)*g*L1*cos(y(:,1)) - m2*g*L2*cos(y(:,3));
% calculate kinetic energy
T = 0.5*m1*L1*L1*y(:,2).^2+0.5*m2*(L1*L1*y(:,2).^2+L2*L2*y(:,4).^2+2*L1*L2*y(:,2)*y(:,4));
% calculate total energy
E=T+V;
% total energy should be conserved (ie constant)
% plot(t,E);

% location of end point of double pendulum
x2 = L1*sin(y(:,1))+L2*sin(y(:,3));
y2 = -L1*cos(y(:,1)) - L2*cos(y(:,3));
% parametric plot
plot(x2,y2);
pause;
plot(t,y(:,1));
pause;
plot(t,y(:,3));
pause;
plot(t,T);
pause;
plot(t,V);
pause;
```

```
plot(t,E);
min(E)
max(E)
(max(E)-min(E))/max(E)
```

```
function [ yprime ] = double_pendulum(t,y)
%UNTITLED Summary of this function goes here
% Detailed explanation goes here
```

```
yprime = zeros(4,1);
```

```
m1=2;
m2=1;
L1=1;
L2=2;
g=32.0;
a = (m1+m2)*L1;
b = m2*L2*cos(y(1)-y(3));
c = m2*L1*cos(y(1)-y(3));
d = m2*L2;
e = -m2*L2*y(4)*y(4)*sin(y(1)-y(3)) - g*(m1+m2)*sin(y(1));
f = m2*L1*y(2)*y(2)*sin(y(1)-y(3)) - m2*g*sin(y(3));
```

```
yprime(1) = y(2);
yprime(3) = y(4);
yprime(2) = (e*d-b*f)/(a*d-c*b);
yprime(4) = (a*f-c*e)/(a*d-c*b);
```

```
end
```
