Derek Frank

dmfrank@ucsc.edu

AMS 147

3/15/10

Course Project

Problem #1: Three Body System

This project implements many methods to solve a three body system of moving objects, given an initial position and velocity of each body. The problem is considered in a two-dimensional plane.

Initial Conditions:

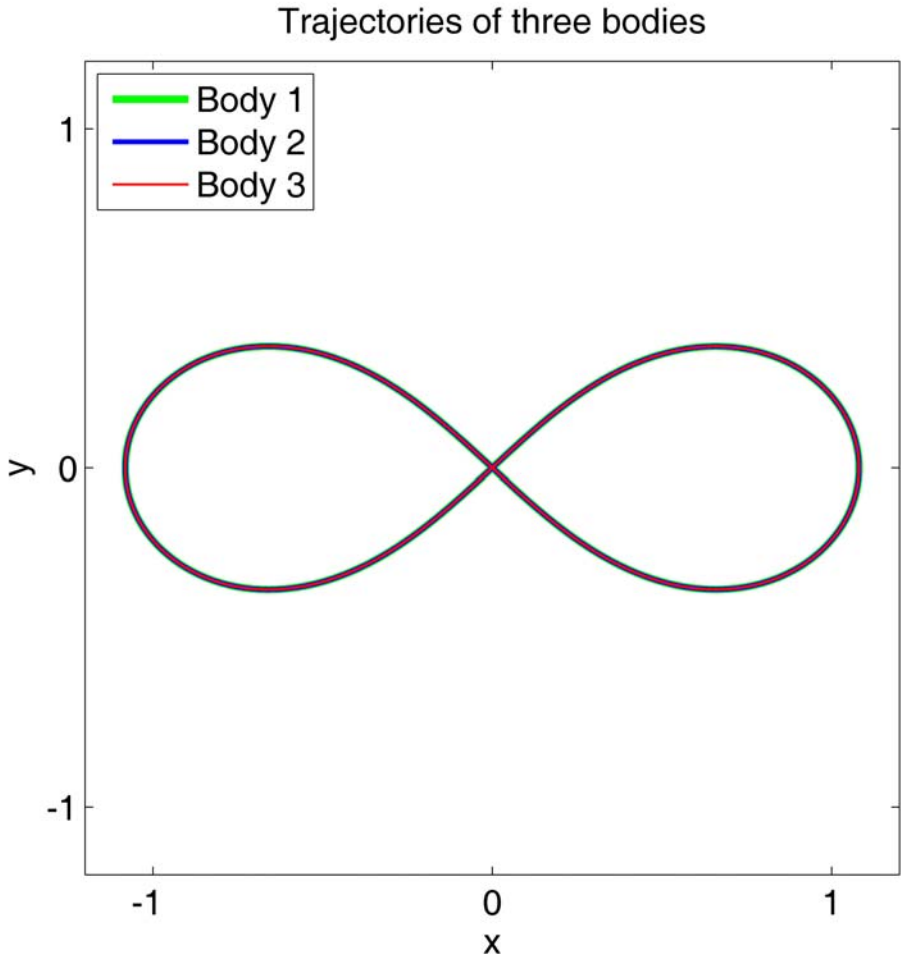
$$m = 1, c = 1$$
$$t = 0 \implies (x(t), y(t))$$
$$x_1(0), y_1(0) = (0.97000436, -0.24308753)$$
$$x_2(0), y_2(0) = (-0.97000436, 0.24308753)$$
$$x_3(0), y_3(0) = (0, 0)$$
$$u_1(0), v_1(0) = (0.93240737\ 2, 0.86473146\ 2)$$
$$u_2(0), v_2(0) = (0.93240737\ 2, 0.86473146\ 2)$$
$$u_3(0), v_3(0) = (-0.93240737, -0.86473146)$$

1.

a. In the first problem, I am to plot the trajectories of the three bodies with the given initial conditions. Using a time step of $h = {}^{1}/_{128}$, I will determine the plot for $t\epsilon[0,40]$.

b. Using Matlab, I will implement the Runge-Kutta method to use the initial conditions to find the positions and velocities of each body at times for $t\epsilon[0,40]$. As well, the forces (gravitational in this case) acting on each body for the same time span. The forces acting on each particular body is the sum of the forces acting on it. For example, the force on body 1 consists of both the gravitational pull from body 2 and body 3, so the sum of both forces is the total force acting on body 1. The same applies on the other two bodies.

c.



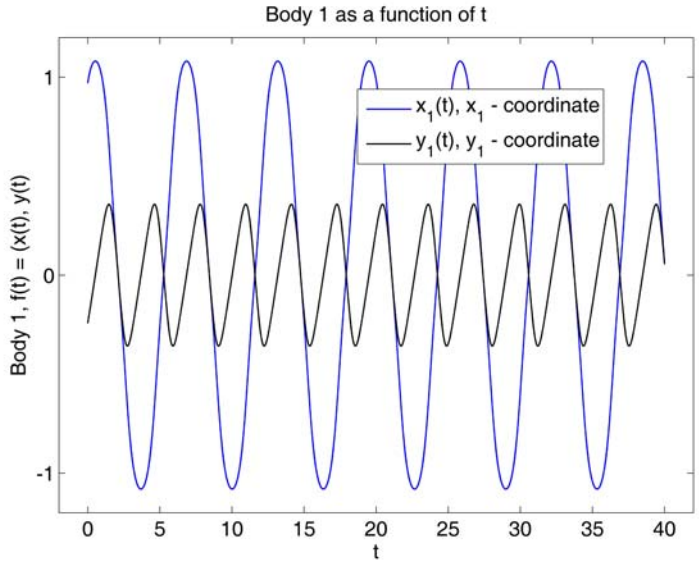
d. The results plotted above show the trajectory of this three body system. The trajectory, on the time interval $t\epsilon[0,40]$, shows the three bodies continually tracing on a figure eight shape.

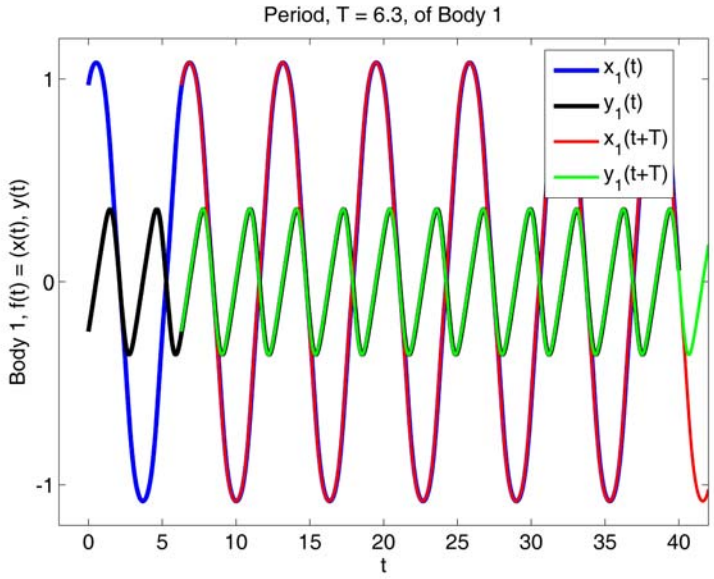
2.

a. In the second problem, I am to plot the coordinates of body 1 as a function of *t*. If it appears that body 1 is periodic, then I am then to approximate a value for the period, *T*, and plot the coordinates of body 1 as a function of time plus period, $x_1(t+T)$ and $y_1(t+T)$.

b. After plotting body 1 as a function of time, I then test plotting time plus a period, which I guess based on the plot, against the regular function of time. If the curves overlap, then the period I guessed is a close approximation for the period.

c.

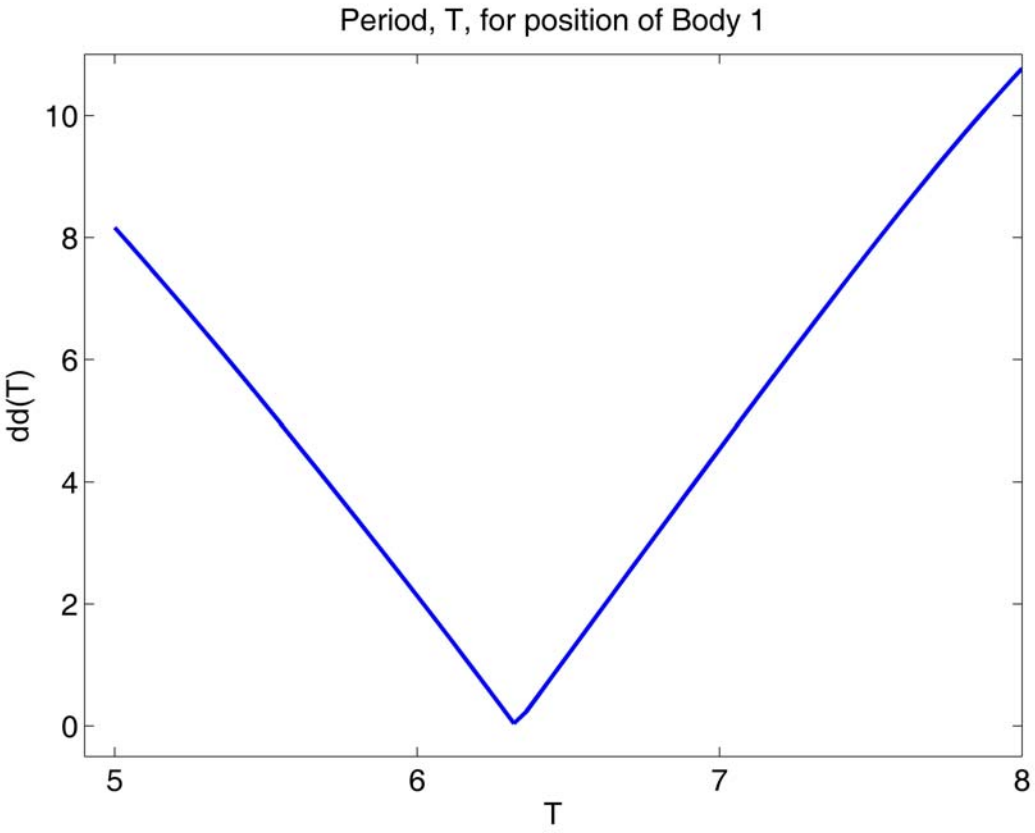




d. The results here show that the position as a function of time for body 1 is periodic. The period I approximated was *T=6.3*, which gives one full period then begins to overlap the original functions.

3.

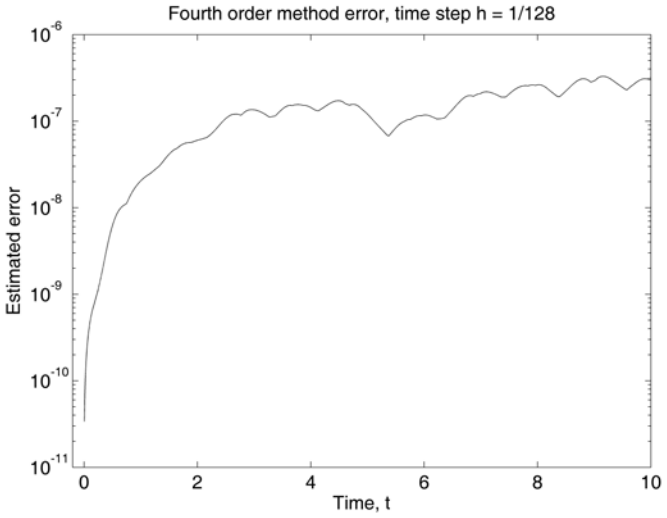
a. The third problem states to determine the period, *T*, for the position of body 1 as a function of time, using the golden search method and the cubic spline.
b. The cubic spline is implemented in the distance function, which determines the difference between two positional values as a function of time. In this case, the first values is a position at time *t* and the second value is a position at time *t+T*. The cubic spline takes the values of position determined in the first problem, using the Runge-Kutta method, and creates a function of values so that we can plug values of time in. The golden search method continually plugs values for the period into the distance formula until the distance between the two position is minimized.
c.

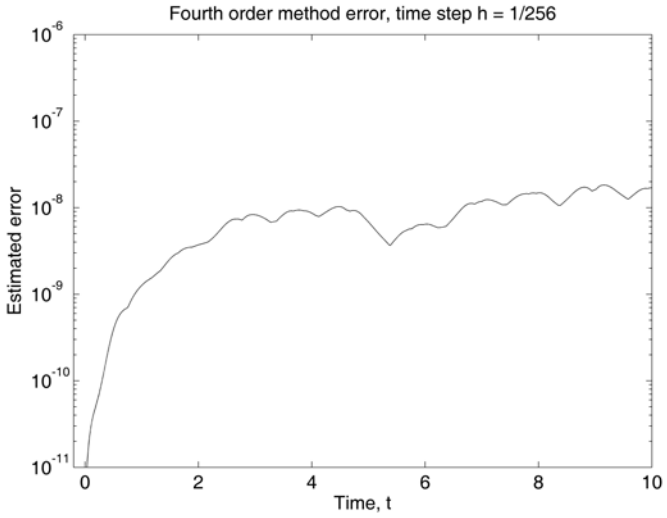


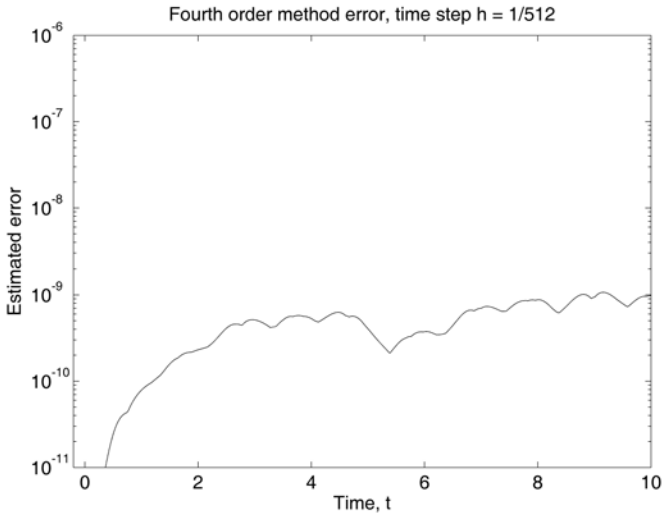
d. The results, using the golden search method and the cubic spline, approximate the period to be *T=6.325900*.

4.

a. In the final problem, I am to calculate and plot the estimated errors as a function of time. The error comes from the use of the fourth order method and depends on the step size, *h*.
b. Using matlab, I will implement the estimation of error for the fourth order method. I will also use different step sizes to demonstrate the difference in error. I will use $h=1/128$, $h=1/256$, and $h=1/512$.
c.







d. The results show the error estimations using three different step sizes. Each of the chosen step sizes were chosen because they produce error that is not greater than $0.5*10^{-5}$. Each curve looks very similar except with a shift in the error, in particular, the minimums and maximums appear to take place in the same time instant for each time step. The smaller the time step, however, the smaller the error it is associated with.

Appendix:

1. "f_sys.m"

```
function [z]=f_sys(w,t)
%
% This function calculates f_sys(w,t)
%
X1=[w(1), w(2)];    % position of body #1
X2=[w(3), w(4)];    % position of body #2
X3=[w(5), w(6)];    % position of body #3
%
z=zeros(1,12);
%
z(1)=w(7);
z(2)=w(8);
z(3)=w(9);
z(4)=w(10);
z(5)=w(11);
z(6)=w(12);
%
dX21=X2-X1;     % vector X2-X1
dX31=X3-X1;     % vecotr X3-X1
dX32=X3-X2;     % vecotr X3-X2
%
G21=dX21/norm(dX21)^3;  % gravitational force on body #1 from
body #2
G12=-G21;       % gravitational force on body #2 from body #1
G31=dX31/norm(dX31)^3;  % gravitational force on body #1 from
body 3
G13=-G31;       % gravitational force on body #3 from body #1
G32=dX32/norm(dX32)^3;  % gravitational force on body #2 from
body #3
G23=-G32;       % gravitational force on body #3 from body #2
% Sum of forces acting on each body
P1=G21+G31;     % Forces acting on body 1
P2=G12+G32;     % Forces acting on body 2
P3=G13+G23;     % Forces acting on body 3
%
z(7)=P1(1);
z(8)=P1(2);
z(9)=P2(1);
z(10)=P2(2);
z(11)=P3(1);
z(12)=P3(2);
```

"calc_3body.m"

```
% This code implements the classical four stage fourth order
% Runge Kutta method to solve the two body problem. After
% the calculation, it saves the workspace to a data file.
%
clear
%
m=12;
w0=[0.97000436, -0.24308753, -0.97000436, 0.24308753, 0, 0,...
    0.93240737/2, 0.86473146/2, 0.93240737/2, 0.86473146/2,...
    -0.93240737, -0.86473146];
cf=2^7;
h=1/cf;
nstep=40*cf;
```

```
%
w=zeros(nstep+1,m);
t=zeros(nstep+1,1);
t(1)=0;
w(1,1:m)=w0;
%
p=4;
d=[0,   1/2, 1/2, 1  ];
c=[0,   0,   0,   0  ;
   1/2, 0,   0,   0  ;
   0,   1/2, 0,   0  ;
   0,   0,   1,   0  ];
b=[1/6, 1/3, 1/3, 1/6];
k=zeros(p,m);
%
for j=1:nstep,
  for i=1:p,
    k(i,1:m)=h*feval('f_sys', w(j,1:m)+c(i,1:i-1)*k(1:i-1,1:m),
t(j)+d(i)*h);
  end
  w(j+1,1:m)=w(j,1:m)+b*k;
  t(j+1)=t(j)+h;
end
%
save data_3body
```

“plot_traject.m”

```
% This code reads in the data file generated by calc_3body.m.
% Then it plots the trajectories of three bodies.
%
clear
figure(1);
clf reset
%
load data_3body
%
set(gcf,'position',[100,100,500,500])
axes('position',[0.15,0.12,0.75,0.75])
plot(w(:,1),w(:,2),'g-','linewidth',3.0)
hold on
plot(w(:,3),w(:,4),'b-','linewidth',2.0)
hold on
plot(w(:,5),w(:,6),'r-','linewidth',1.0)
%
set(gca,'fontsize',14)
axis equal
axis([-1.2,1.2,-1.2,1.2])
set(gca,'xtick',[-2:1:2])
set(gca,'ytick',[-2:1:2])
xlabel('x')
ylabel('y')
title('Trajectories of three bodies')
legend('Body 1','Body 2','Body 3',2)
```

“visualize_1.m”

```
% This code reads in the data file generated by calc_3body.m.
% Then it visualizes the trajectories of three bodies.
```

```
%
clear
figure(2);
clf reset
%
set(gcf,'position',[100,100,500,500])
set(gcf, 'Backingstore','off')
axes('position',[0.15,0.12,0.75,0.75])
set(gca,'color',[0.2,0.2,0.2])
%
head_1=line('color','g','marker','o','markersize',10,...
    'markerfacecolor','g','erase','background',...
    'xdata',[],'ydata',[]);
head_2=line('color','b','marker','o','markersize',10,...
    'markerfacecolor','b','erase','background',...
    'xdata',[],'ydata',[]);
head_3=line('color','r','marker','o','markersize',10,...
    'markerfacecolor','r','erase','background',...
    'xdata',[],'ydata',[]);
%
body_1=line('color','g','linestyle','-','linewidth',2.5,...
    'erase','background','xdata',[],'ydata',[]);
body_2=line('color','b','linestyle','-','linewidth',2.5,...
    'erase','background','xdata',[],'ydata',[]);
body_3=line('color','r','linestyle','-','linewidth',2.5,...
    'erase','background','xdata',[],'ydata',[]);
%
tail_1=line('color','w','linestyle','-','linewidth',1.0,...
    'erase','none','xdata',[],'ydata',[]);
tail_2=line('color','w','linestyle','-','linewidth',1.0,...
    'erase','none','xdata',[],'ydata',[]);
tail_1=line('color','w','linestyle','-','linewidth',1.0,...
    'erase','none','xdata',[],'ydata',[]);
tail_3=line('color','w','linestyle','-','linewidth',1.0,...
    'erase','none','xdata',[],'ydata',[]);
%
axis equal
axis([-1.2,1.2,-1.2,1.2])
set(gca,'fontsize',14)
set(gca,'xtick',[-2:1:2])
set(gca,'ytick',[-2:1:2])
xlabel('x')
ylabel('y')
title('Trajectories of three bodies')
%
load data_3body
n=size(w,1);
dn=4;
x1=w(1:dn:n,1);
y1=w(1:dn:n,2);
x2=w(1:dn:n,3);
y2=w(1:dn:n,4);
x3=w(1:dn:n,5);
y3=w(1:dn:n,6);
%
nx=size(x1,1);
dib=16/dn;
dit=8/dn;
for i=1:nx,
```

```
    set(head_1,'xdata',x1(i),'ydata',y1(i));
    set(head_2,'xdata',x2(i),'ydata',y2(i));
    set(head_3,'xdata',x3(i),'ydata',y3(i));
    ib=max(1,i-dib);
    it=max(1,ib-dit);
    set(body_1,'xdata',x1(ib:i),'ydata',y1(ib:i));
    set(body_2,'xdata',x2(ib:i),'ydata',y2(ib:i));
    set(body_3,'xdata',x3(ib:i),'ydata',y3(ib:i));
    set(tail_1,'xdata',x1(it:ib),'ydata',y1(it:ib));
    set(tail_2,'xdata',x2(it:ib),'ydata',y2(it:ib));
    set(tail_3,'xdata',x3(it:ib),'ydata',y3(it:ib));
    drawnow
    pause(0.01)
  end
```

2. “body1_t.m”

```
% This code reads in the data file generated by calc_3body.m.
% Then it plots the trajectories of three bodies.
%
clear
figure(3);
clf reset
%
load data_3body
%
set(gcf,'position',[100,100,500,500])
axes('position',[0.15,0.12,0.75,0.75])
plot(t,w(:,1),'b-','linewidth',1.0)
hold on
plot(t,w(:,2),'r-','linewidth',1.0)
%
set(gca,'fontsize',14)
axis([-2,42,-1.2,1.2])
set(gca,'xtick',[0:5:40])
set(gca,'ytick',[-2:1:2])
xlabel('t')
ylabel('Body 1, f(t) = (x(t), y(t)')
title('Body 1 as a function of t')
legend('x_1(t), x_1 - coordinate','y_1(t), y_1 - coordinate',2)
```

“body1_periodT.m”

```
% This code reads in the data file generated by calc_3body.m.
% Then it plots the trajectories of three bodies.
%
clear
figure(4);
clf reset
%
load data_3body
T=6.3;
tT=t(:,1)+T;
%
set(gcf,'position',[100,100,500,500])
axes('position',[0.15,0.12,0.75,0.75])
plot(t,w(:,1),'b-','linewidth',3.0)
hold on
plot(t,w(:,2),'k-','linewidth',3.0)
hold on
```

```
plot(tT,w(:,1),'r-','linewidth',2.0)
hold on
plot(tT,w(:,2),'g-','linewidth',2.0)
%
set(gca,'fontsize',14)
axis([-2,42,-1.2,1.2])
set(gca,'xtick',[0:5:40])
set(gca,'ytick',[-2:1:2])
xlabel('t')
ylabel('Body 1, f(t) = (x(t), y(t)')
title('Period, T = 6.3, of Body 1')
legend('x_1(t)','y_1(t)','x_1(t+T)','y_1(t+T)',2)
```

3. "dd.m"

```
function [dist]=dd(T,td,yd)
%
t1=[0:0.1:8];
y1=spline(td,yd,t1);
t2=T+t1;
y2=spline(td,yd,t2);
dist=norm(y1-y2);
```

"golden_search.m"

```
clear
%
load data_3body
td=t(:,1);
yd=w(:,1);
%
a=5;
b=7;
tol=1.0e-10;
n=0;
%
g=(sqrt(5)-1)/2;
r1=a+(b-a)*(1-g);
f1=dd(r1,td,yd);
r2=a+(b-a)*g;
f2=dd(r2,td,yd);
%
while (b-a) > tol,
  n=n+1;
  if f1 < f2,
    b=r2;
    r2=r1;
    f2=f1;
    r1=a+(b-a)*(1-g);
    f1=dd(r1,td,yd);
  else
    a=r1;
    r1=r2;
    f1=f2;
    r2=a+(b-a)*g;
    f2=dd(r2,td,yd);
  end
end
T=(a+b)/2
```

"calc_d_T.m"

```
clear
%
load data_3body
td=t(:,1);
yd=w(:,1);
%
Ta=[5:0.04:8];
n=size(Ta,2);
da=zeros(1,n);
%
for k=1:n,
  da(k)=dd(Ta(k),td,yd);
end
%
save data_d_T
```

"plot_d_T.m"

```
clear
figure(5)
clf reset
axes('position',[0.15,0.13,0.75,0.75])
%
load data_d_T
%
plot(Ta,da,'linewidth',2.0)
axis([4.9,8,-.5,11])
set(gca,'fontsize',14)
set(gca,'xtick',[5:8])
set(gca,'ytick',[0:2:12])
xlabel('T')
ylabel('dd(T)')
title('Period, T, for position of Body 1')
```

4. "calc_3body_h2.m"

```
% This code implements the classical four stage fourth order
% Runge Kutta method to solve the two body problem. After
% the calculation, it saves the workspace to a data file.
%
clear
%
load data_3body
% h/2
cf2=2*cf;
h2=1/cf2;
nstep2=10*cf2;
%
w2=zeros(nstep+1,m);
t2=zeros(nstep+1,1);
t2(1)=0;
w2(1,1:m)=w0;
%
k2=zeros(p,m);
%
for j=1:nstep2,
  for i=1:p,
    k2(i,1:m)=h2*feval('f_sys', w2(j,1:m)+c(i,1:i-1)*k2(1:i-
1,1:m), t2(j)+d(i)*h2);
```

```
  end
  w2(j+1,1:m)=w2(j,1:m)+b*k2;
  t2(j+1)=t2(j)+h2;
end
%
save data_3body_h2
```

“error_3body.m”

```
clear
figure(6)
clf reset
axes('position',[0.15,0.13,0.75,0.75])
%
load data_3body_h2
%
diff=zeros(nstep+1,m);
for j=1:m,
    for i=0:nstep,
        diff(i+1,j)=abs(w(i+1,j)-w2(2*i+1,j));
    end
end
err=zeros(nstep+1);
for i=1:nstep+1,
    err(i)=(16/15)*norm(diff(i,1)+diff(i,2)+diff(i,3)...
        +diff(i,4)+diff(i,5)+diff(i,6)+diff(i,7)+diff(i,8)...
        +diff(i,9)+diff(i,10)+diff(i,11)+diff(i,12));
end
%
semilogy(t, err,'k-')
%hold on
%loglog(t, err,'bs','Markerfacecolor','b')
%
axis([-.2,10,1e-11,1e-6])
set(gca,'fontsize',14)
set(gca,'xtick',[0:2:10])
set(gca,'ytick',10.^[-12:1:-2])
xlabel('Time, t')
ylabel('Estimated error')
title('Fourth order method error')
```