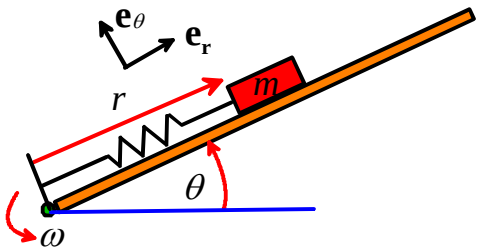
School of Engineering
Brown University

EN40: Dynamics and Vibrations

Homework 3: Solving equations of motion for particles
Solutions

1. The figure shows a simple idealization of a type of MEMS gyroscope (way over-simplified, actually…). It consists of a mass that slides up and down a guide, which spins at constant angular speed ω. The mass is attached to a spring with stiffness k and unstretched length L_0. The goal of this problem is to find, and solve, the equation governing the radial distance *r(t)* of the mass from the pivot. Friction and gravity should be neglected.



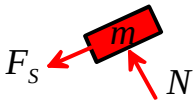
1.1 Write down the acceleration vector, in polar coordinates, in terms of time derivatives of r and ω

From notes

$$\mathbf{a} = \left[\frac{d^2r}{dt^2} - \omega^2 r\right]\mathbf{e}_r + \left[2\frac{dr}{dt}\omega\right]\mathbf{e}_\theta$$

[2 POINTS]

1.2 Draw a free body diagram showing the forces acting on the mass.



[2 POINTS]

1.3 Use Newton's law **F**=m**a** (and the spring force law) to show that *r(t)* satisfies the differential equation

$$\frac{d^2r}{dt^2} + \Omega^2 r = \frac{k}{m}L_0$$

where $\Omega^2 = \frac{k}{m} - \omega^2$ is a constant (introduced to simplify the solution)

$$\mathbf{F} = m\mathbf{a} \Rightarrow m\left[\frac{d^2r}{dt^2} - \omega^2 r\right]\mathbf{e}_r + m\left[2\frac{dr}{dt}\omega\right]\mathbf{e}_\theta = -k(r - L_0)\mathbf{e}_r + N\mathbf{e}_\theta$$

and hence

$$m\left[\frac{d^2r}{dt^2} - \omega^2 r\right] = -k(r - L_0) \Rightarrow \frac{d^2r}{dt^2} + \left(\frac{k}{m} - \omega^2\right)r = \frac{k}{m}L_0$$

[2 POINTS]

1.4 Assume that at time t=0, the mass is not moving radially (so $dr/dt = 0$) and the length of the spring is equal to its unstretched length (so $r = L_0$). Use Mupad to solve the differential equation, following the procedure described in class. To stop Mupad giving complex exponentials in the solution, you can use

```
assume(`&Omega;`>0)
```

(Here, `` `&Omega;` `` is the Mupad for the Greek symbol Ω - if you use some other variable to denote Ω in your calculation you will have to change the `` `&Omega;` `` to be your variable)

Here's the mupad

```
reset()
assume(`&Omega;`>0)
diffeq := r''(t)+`&Omega;`^2*r(t)=k/m*L0
```

$$r''(t) + r(t)\,\Omega^2 = \frac{L0\,k}{m}$$

```
IC := r(0)=L0, r'(0)=0
```

$$r(0) = L0,\ r'(0) = 0$$

```
solve(ode({diffeq,IC}, r(t)),IgnoreSpecialCases)
```

$$\left\{\cos(\Omega\,t)\left(L0 - \frac{L0\,k}{\Omega^2\,m}\right) + \frac{L0\,k}{\Omega^2\,m}\right\}$$

[2 POINTS]

1.5 Hence, write down a formula for the frequency of vibration of the mass. Does the frequency increase or decrease with the angular speed ω? What happens if the bar spins very rapidly, so that $\omega^2 > k/m$?

We see that the solution is predicting simple harmonic motion – the angular frequency of vibration is simply Ω. As ω increases, the frequency gets smaller (from the definition of Ω). For $\omega^2 > k/m$ we can re-solve the ODE and make Ω negative – in this case

```
reset()
assume(`&Omega;`>0)
diffeq := r''(t)-`&Omega;`^2*r(t)=k/m*L0
```

$$r''(t) - \Omega^2\,r(t) = \frac{L0\,k}{m}$$

```
IC := r(0)=L0, r'(0)=0
```

$$r(0) = L0,\ r'(0) = 0$$

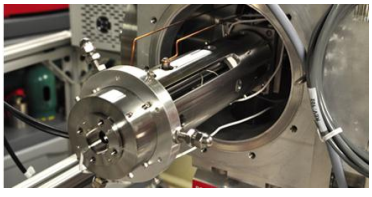
```
solve(ode({diffeq,IC}, r(t)),IgnoreSpecialCases)
```

$$\left\{\frac{e^{-\Omega t}\,(L0\,m\,\Omega^2 + L0\,k)}{2\,\Omega^2\,m} - \frac{L0\,k}{\Omega^2\,m} + \frac{L0\,e^{\Omega t}\,(m\,\Omega^2 + k)}{2\,\Omega^2\,m}\right\}$$

Now we get an exponential solution – instead of vibrating, the mass just flies off to infinity….

[3 POINTS]

2. The picture shows a 'quadrupole mass filter' (from the Rowland Institute). It consists of four parallel rods, which are charged to induce a special time-dependent electric field in the space between them. As this

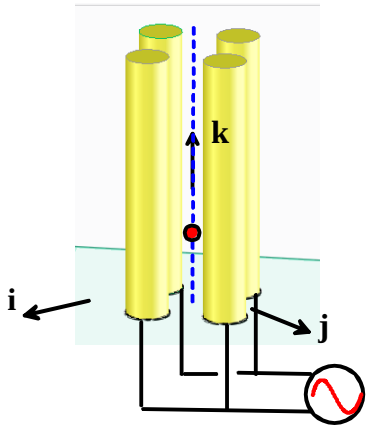


problem will show, only ions with a charge to mass ratio within a range are able to pass from one end of the filter to the other without hitting one of the rods.

2.1 The motion of the ion will be described by the components of its position vector (*x,y,z).* Write down the velocity and acceleration of the ion in terms of these variables.

$$\mathbf{v}=\frac{dx}{dt}\mathbf{i}+\frac{dy}{dt}\mathbf{j}+\frac{dz}{dt}\mathbf{k} \qquad \mathbf{a}=\frac{d^2x}{dt^2}\mathbf{i}+\frac{d^2y}{dt^2}\mathbf{j}+\frac{d^2z}{dt^2}\mathbf{k}$$

[1 POINT]



2.2 The ion has mass *m*, and is subjected to a force

$$\mathbf{F}=Q\mathbf{E}$$

where *Q* is its charge, and

$$\mathbf{E}=E_0\frac{1+\beta\cos(\omega t)}{d}\left(x\mathbf{i}+y\mathbf{j}\right)$$

is a time dependent electric field vector. Here, E_0,β,d are constants that describe the magnitude and geometry of the electric fields. Use Newton's law to show that the components of the position vector of the ion satisfies the following equations of motion

$$\frac{d^2x}{dt^2}=\Omega^2(1+\beta\cos(\omega t))x \qquad \frac{d^2y}{dt^2}=\Omega^2(1+\beta\cos(\omega t))y \qquad \frac{d^2z}{dt^2}=0$$

where

$$\Omega=\sqrt{\frac{QE_0}{md}}$$

is a parameter (which has units of frequency).

Newton's law **F**=m**a** gives

$$\frac{QE_0(1+\beta\cos\omega t)}{d}\left(x\mathbf{i}+y\mathbf{j}\right)=m\left(\frac{d^2x}{dt^2}\mathbf{i}+\frac{d^2y}{dt^2}\mathbf{j}+\frac{d^2z}{dt^2}\mathbf{k}\right)$$

The **i,j,k** components of this expression reduce to the expression given.

[2 POINTS]

2.3 Re-write the equations of motion as 6 first-order differential equations that can be integrated using MATLAB.

$$\frac{d}{dt}\begin{bmatrix}x\\y\\z\\v_x\\v_y\\v_z\end{bmatrix}=\begin{bmatrix}v_x\\v_y\\v_z\\\Omega^2(1+\beta\cos(\omega t))x\\\Omega^2(1+\beta\cos(\omega t))y\\0\end{bmatrix}$$

Introducing v_x,v_y,v_z as new unknowns and substituting into **F**=m**a** in the usual way gives this result

[1 POINT]

2.4 Write a MATLAB script that will solve the equations of motion to determine x,y,z,v_x,v_y,v_z as a function of time. You need not submit a solution to this problem. **(There is no need to submit a solution to this problem)**

```
function quadrupole_filter
  close all
  omega = 10;
  beta = 20;
  Omega = 1.1;
  w0 = [0.005,0.005,0,0,0,0.01];

  options = odeset('Event',@event);
 [times,sols] = ode45(@eom,[0:1:100],w0,options);
  animate_quadrupole(times,sols)

  for i=1:100
      Omega = 0.6 + 1.15*(i-1)/99;
      [times,sols] = ode45(@eom,[0:1:100],w0,options);

     rval(i) = sqrt(sols(end,1)^2 + sols(end,2)^2);
     Omegaval(i) = Omega;
  end
  figure1 = figure
  axes1 = axes('Parent',figure1,'FontSize',14);
  ylim(axes1,[0 1]);
  box(axes1,'on');
  hold(axes1,'all');
  plot(Omegaval,rval,'LineWidth',2,'Color',[1 0 0]);
  xlabel({'\Omega'},'FontSize',16);
  ylabel({'r = (x^2 +y^2 )^(^1^/^2^) at end of simulation'},'FontSize',14);

    function dwdt = eom(t,w)
        x= w(1); y=w(2); z=w(3);
        vx = w(4); vy=w(5); vz=w(6);

        dxdt = vx;
        dydt = vy;
        dzdt = vz;
        dvxdt = Omega^2*(1+beta*cos(omega*t))*x;
        dvydt = Omega^2*(1+beta*cos(omega*t))*y;
        dvzdt = 0;

        dwdt = [dxdt;dydt;dzdt;dvxdt;dvydt;dvzdt];

    end
    function [eventval,stopcalc,dir] = event(t,w)
       z = w(3);
       eventval = z-1;
       stopcalc = 1;
       dir = -1;
    end
end
function animate_quadrupole(times,sols)
```

```
    [XX,YY,ZZ] = cylinder(0.1);
    XX1 = XX - 0.15;
    YY1 = YY - 0.15;
    XX2 = XX + 0.15;
    YY2 = YY + 0.15;
    figure1 = figure;
    for i=1:length(times)
        clf
     axes1 = axes('Parent',figure1,'DataAspectRatio',[1 1 1]);

     view(axes1,[-79.5 22]);
     grid(axes1,'on');
     xlim([-0.6 0.6]);
     ylim([-0.6 0.6]);
     zlim([0 1.2]);
     hold on
     surf(XX1,YY1,ZZ)
     surf(XX2,YY2,ZZ)
     surf(XX1,YY2,ZZ)
     surf(XX2,YY1,ZZ)
     plot3(sols(1:i,1),sols(1:i,2),sols(1:i,3),...
         'LineWidth',2,'Color',[0 1 0]);
     plot3(sols(i,1),sols(i,2),sols(i,3),...
         'ro','MarkerSize',11,'MarkerFaceColor','r');
     pause(0.01)
    end
end
```

[0 POINTS]

2.5 Analyze the motion of a charged particle that enters the rods at *z=0* with initial position very slightly off the axis (take the initial position and velocity to be $(0.005, 0.005, 0)$ and $(0, 0, 0.01)$, respectively) with the following parameters: $\omega = 10, \beta = 20$. Run the computation for 100 time units. Try a solution with $\Omega = 1.1$, and plot the trajectory of the ion. If you like, you can download a script that will animate the motion of the ion as it moves through the filter. To see the animation use the following syntax

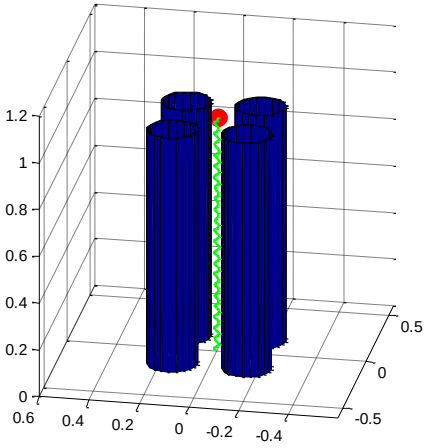
```
[times,sols] = ode45(@eom,[0:1:100],w0,options);
animate_quadrupole(times,sols)
```

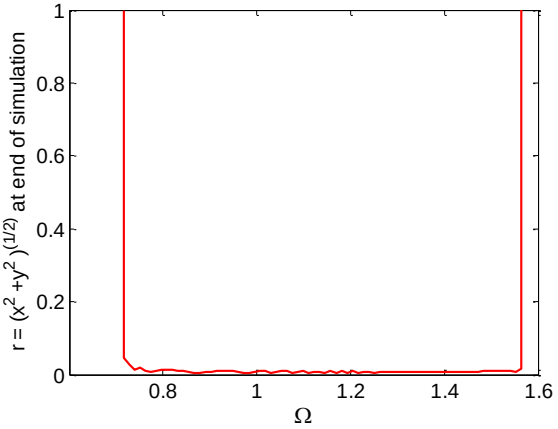
(this assumes that the function you are using to compute the time derivatives in the differential equation is called 'eom' and that you have set the 'options' variable to use an event function).

Show that with these settings, the ion will arrive at *z=1* without a large radial deflection if $0.73 < \Omega < 1.55$ (you can do this by trial and error, or a better way would be to plot the value of

$r = \sqrt{x^2 + y^2}$ at the end of the simulation as a function of Ω. Plotting this graph is quite challenging, however, so just trying a few values by hand and reporting the behavior you observe in the trajectories is fine.).

The solution for $\Omega = 1.1$ is shown in the figure (from the animation). The second figure shows the plot of *r* as a function of Ω

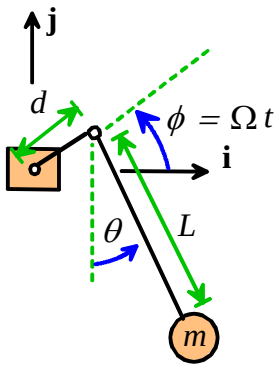




The plot could be refined around the critical Ω values to get the values a bit more accurately, but with the accuracy in the figure we find that *r* is small at the end of the simulation as long as $0.73 < \Omega < 1.55$.

[6 POINTS]

3. The figure shows a pendulum mounted on a crank that rotates at constant angular speed Ω. The pendulum shaft AB pivots freely at A, so the angle θ varies with time as the pendulum swings. The goal of this problem is to derive the differential equation governing θ and solve it with Matlab.



3.1 Write down the position vector of the mass *m* in terms of $d, L, \phi = \Omega t, \theta$. Hence, find an expression for its acceleration, in terms of d, L, Ω and θ and its time derivatives.

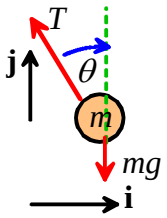
$$\mathbf{r} = (d\cos\phi + L\sin\theta)\mathbf{i} + (d\sin\phi - L\cos\theta)\mathbf{j}$$

$$\mathbf{v} = \left(-d\Omega\sin\phi + L\frac{d\theta}{dt}\cos\theta\right)\mathbf{i} + \left(d\Omega\cos\phi + L\frac{d\theta}{dt}\sin\theta\right)\mathbf{j}$$

$$\mathbf{a} = \left(-d\Omega^2\cos\phi + L\frac{d^2\theta}{dt^2}\cos\theta - L\left[\frac{d\theta}{dt}\right]^2\sin\theta\right)\mathbf{i} + \left(-d\Omega^2\sin\phi + L\frac{d^2\theta}{dt^2}\sin\theta + L\left[\frac{d\theta}{dt}\right]^2\cos\theta\right)\mathbf{j}$$

[3 POINTS]

3.2 Draw a free body diagram showing the forces acting on the mass *m*, and hence write down Newton's law of motion in terms of d, L, Ω and θ and its time derivatives, as well as the unknown reaction force in the pendulum shaft.



Newton's law gives

$$-T\sin\theta\mathbf{i} + (T\cos\theta - mg)\mathbf{j}$$

$$= m\left(-d\Omega^2\cos\phi + L\frac{d^2\theta}{dt^2}\cos\theta - L\left[\frac{d\theta}{dt}\right]^2\sin\theta\right)\mathbf{i} + m\left(-d\Omega^2\sin\phi + L\frac{d^2\theta}{dt^2}\sin\theta + L\left[\frac{d\theta}{dt}\right]^2\cos\theta\right)\mathbf{j}$$

[2 POINTS]

3.3 Eliminate the unknown reaction force in the crank from the equation of motion to show that θ is governed by the following differential equation

$$\frac{d^2\theta}{dt^2} - \frac{d}{L}\Omega^2(\cos\phi\cos\theta + \sin\phi\sin\theta) + \frac{g}{L}\sin\theta = 0$$

The quickest way to eliminate the tension force is to take the dot product of both sides of this equation with $\mathbf{e} = \cos\theta\mathbf{i} + \sin\theta\mathbf{j}$. This gives

$$-mg\sin\theta$$

$$= m\left(-d\Omega^2\cos\phi + L\frac{d^2\theta}{dt^2}\cos\theta - L\left[\frac{d\theta}{dt}\right]^2\sin\theta\right)\cos\theta + m\left(-d\Omega^2\sin\phi + L\frac{d^2\theta}{dt^2}\sin\theta + L\left[\frac{d\theta}{dt}\right]^2\cos\theta\right)\sin\theta$$

$$= mL\frac{d^2\theta}{dt^2} - md\Omega^2(\cos\phi\cos\theta + \sin\phi\sin\theta)$$

$$\Rightarrow \frac{d^2\theta}{dt^2} - \frac{d}{L}\Omega^2(\cos\phi\cos\theta + \sin\phi\sin\theta) + \frac{g}{L}\sin\theta = 0$$

[3 POINTS]

3.4 Show that the equation of motion can be re-arranged into the following form

$$\frac{d}{dt}\begin{bmatrix}\theta \\ \omega\end{bmatrix} = \begin{bmatrix}\omega \\ \frac{d}{L}\Omega^2(\cos\phi\cos\theta + \sin\phi\sin\theta) - \frac{g}{L}\sin\theta\end{bmatrix}$$

3.5 Write a MATLAB script that will solve the equations of motion, and plot a graph of θ as a function of time. You can download a script that will animate the motion of the forced pendulum to help visualize its behavior. To call the function, use the syntax

```
animate_pendulum(times,sols,L,d,Omega)
```

where times,sols are the solutions returned by the ODE solver, and `L,d,Omega` are the values of L, d, Ω. **(There is no need to submit a solution to this problem)**

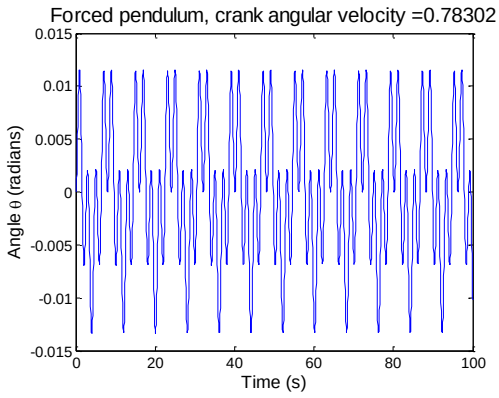
```
function crankpendulum

   close all

   d = 0.1;
   L = 1;
   g = 9.81;
   Omega = 1.85*sqrt(g/L);
   theta0 = 0;
   omega0 = 0;
   options = odeset('RelTol',0.00001);
   [times,sols] = ode45(@eom,[0,100],[theta0,omega0],options);

   figure1 = figure;
   axes1 = axes('Parent',figure1,'FontSize',12);
   box(axes1,'on');
   hold(axes1,'all');
   plot(times,sols(:,1));
   xlabel({'Time (s)'},'FontSize',14);
   ylabel({'Angle \theta (radians)'},'FontSize',14);
   titval = strcat('Forced pendulum, crank angular velocity
=',num2str(Omega));
   title(titval,'FontSize',16);

%   animate_pendulum(times,sols,L,d,Omega)

    function dwdt = eom(t,w)
        theta = w(1); omega = w(2);
        phi = Omega*t;
        dthetadt = omega;
        domegadt = d*Omega^2/L*(cos(phi)*cos(theta)+sin(phi)*sin(theta))-
g*sin(theta)/L;
        dwdt = [dthetadt;domegadt];
    end

end
function animate_pendulum(times,sols,L,d,Omega)
xa = d*cos(Omega*times);
ya = d*sin(Omega*times);
xp = xa + L*sin(sols(:,1));
yp = ya - L*cos(sols(:,1));
maxv = max(max(max(xp),max(xa)),max(max(ya),max(yp)));
minv = min(min(min(xa),min(xp)),min(min(ya),min(yp)));
%if (minv>0) minv = -0.1; end
%if (maxv<0) maxv = 0.1; end
dx = 0.505*(maxv-minv);
```

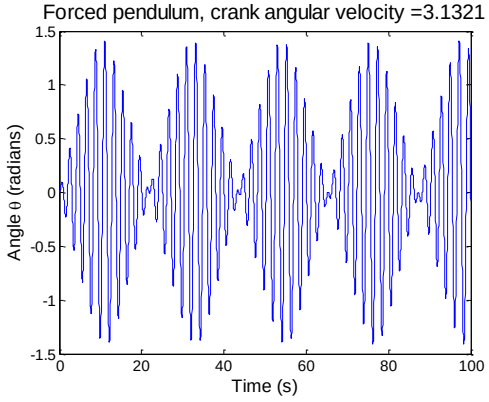
```
xcen = 0.5*(max(max(xa),max(xp))+min(min(xa),min(xp)));
ycen = 0.5*(max(max(ya),max(yp))+min(min(ya),min(yp)));
figure;
for i = 1:length(times)
    clf
    ylim([ycen-dx ycen+dx]);
    xlim([xcen-dx xcen+dx]);
    axis square;
    hold on;
    plot(xp(i),yp(i),'ro','MarkerSize',11,'MarkerFaceColor',[1,0,0]);
    plot([xa(i),xp(i)],[ya(i),yp(i)],'LineWidth',2)
    plot([0,xa(i)],[0,ya(i)],'LineWidth',2)
    pause(0.05)
end

end
```

3.6 Hence, plot graphs showing the variation of θ with time for *L=1m*, *d=0.1m*, and the f values of Ω listed below. Plot each solution for 0<t<100, and use $\theta = d\theta/dt = 0$ at time *t=0*.

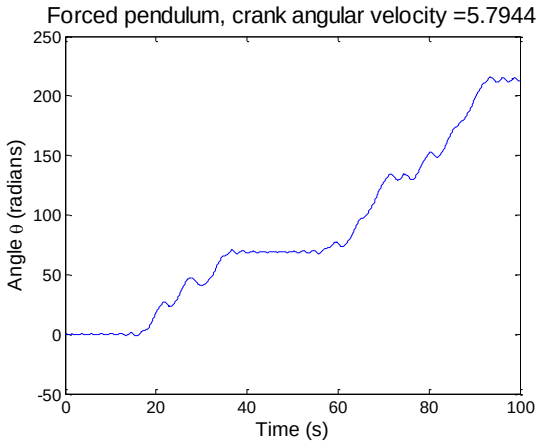
1. $\Omega = 0.25\sqrt{g/L}$
2. $\Omega = \sqrt{g/L}$
3. $\Omega = 1.85\sqrt{g/L}$. Try this solution with the default value of the tolerance for the ODE solver, then try RelTol = 0.0001 and RelTol = 0.00001. Notice that MATLAB gives an entirely different solution as the tolerance is made smaller – in fact it won't converge to a unique solution. This is because the behavior with this value of Ω is *chaotic* – very small changes in parameter values lead to completely different solutions if you let the simulation run for a while.

The solutions for 1 and 2 are shown below. ($\sqrt{g/L}$ is the natural frequency of vibration of the pendulum)

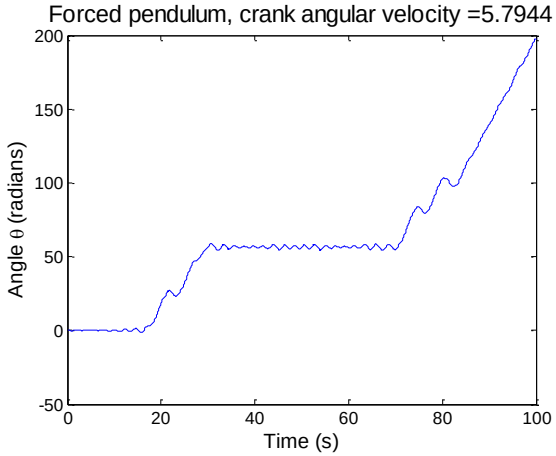




The solutions for 3 are below. Results may differ in detail because of the chaotic behavior.



(a) RelTol = 0.0001



(b) RelTol = 0.00001

[6 POINTS]

4. In this problem we will set up a more accurate calculation of Felix Baumgartner's space jump. We made two crude approximations in our earlier calculation: (i) the first phase of the jump was approximated using a rough guess for the variation of acceleration with time; and (ii) the variation of air density with altitude was approximated. With MATLAB, we can correct both these approximations. We will make the following assumptions:

- Experiments show that the air temperature varies with altitude y (approximately) according to the equation

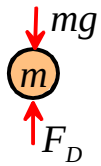
$$\theta(y) = \begin{cases} 298 - 120\,y/14000 & y < 14000 \\ 178 & 14000 < y < 20000 \\ 178 + 140(y - 20000)/30000 & y > 20000 \end{cases}$$

  with y in meters, and θ in Kelvin.
- If air is modeled as an ideal gas, the variation of air density with altitude y satisfies the equation

$$\frac{d\rho}{dy} = -\frac{\rho g}{R\theta(y)}$$

  Where g is the gravitational acceleration, R is the gas constant for air, and θ is the air temperature (in Kelvin) (The TAs will be happy to explain where this equation comes from!)
- The air drag force is given by $F_D = \frac{1}{2}\rho C_D A v^2$

4.1 Draw a free body diagram for the jumper during free-fall, and hence use Newton's law to find an equation for his acceleration in terms of $m, g, \rho, C_D, \theta, A, v$



Newton's law gives $\frac{d^2y}{dt^2} = F_D - mg$ (note that y is positive upwards)

This means that

$$\frac{d^2y}{dt^2} = \frac{1}{2}\rho C_D A v^2 - mg$$

[2 POINTS]

4.2 Show that the equations governing the jumper's altitude y, vertical speed v and the air density at the jumper's altitude can be expressed in the following form

$$\frac{d}{dt}\begin{bmatrix} y \\ v \\ \rho \end{bmatrix} = \begin{bmatrix} v \\ -g + \dfrac{1}{2m}\rho C_D A v^2 \\ -\dfrac{\rho g v}{R\theta(y)} \end{bmatrix}$$

Following the standard procedure, we introduce the velocity as an additional variable, and re-write the acceleration equation in terms of the velocity

$$\frac{dy}{dt} = v$$

$$\frac{dv}{dt} = \frac{1}{2}\rho C_D A v^2 - mg$$

We also need to re-write the equation for the density evolution in terms of time

$$\frac{d\rho}{dy} = \frac{d\rho}{dt}\frac{dt}{dy} = \frac{d\rho}{dt}\frac{1}{v} = -\frac{\rho g}{R\theta(y)}$$

$$\Rightarrow \frac{d\rho}{dt} = -\frac{\rho g v}{R\theta(y)}$$

Collecting these equations into a vector form gives the answer stated

[3 POINTS]

4.3 Hence, set up a MATLAB script that will calculate y, v, ρ as functions of time. Use the following parameter values g=9.81ms^{-2}, R=286.9 J/kgK, A=1.1m^{-2}, $C_D = 0.8$, m=80kg, and take $y = 39000m, v = 0, \rho = 0.002$ kg m^{-3} at time t=0. Add an 'event' function to your code to stop the calculation when y=1524m (this is when the parachute was deployed). **There is no need to submit a solution to this problem)**

A MATLAB script is listed below

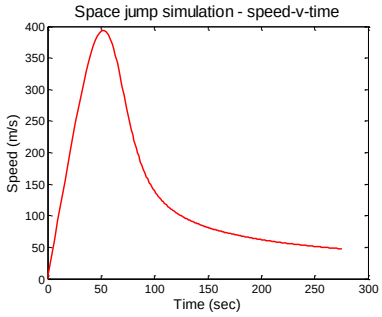
```
function spacejump

   close all

   g = 9.81; % Gravitational accel. m/s^2
   A = 1.1;  % Projected x-sect area, m
   Cd = 0.8; % Drag coefficient
   m = 80;   % Mass (kg)
   R = 286.9; % Gas constant, J/kg K

   options = odeset('Events',@event,'RelTol',0.00001);
   [times,sols] = ode45(@eom,[0,500],[39000,0,0.002],options);

   figure1 = figure;
   axes1 = axes('Parent',figure1,'FontSize',12);
```

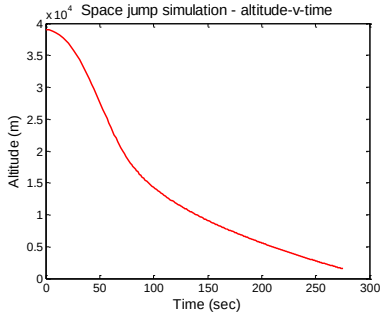
```
    box(axes1,'on');
    hold(axes1,'all');
    plot(times,-sols(:,2),'LineWidth',2,'Color',[1 0 0]);
    xlabel({'Time (sec)'},'FontSize',14);
    ylabel({'Speed (m/s)'},'FontSize',14);
    title('Space jump simulation - speed-v-time','FontSize',16);
    figure2 = figure;
    axes2 = axes('Parent',figure2,'FontSize',12);
    box(axes2,'on');
    hold(axes2,'all');
    plot(times,sols(:,1),'LineWidth',2,'Color',[1 0 0]);
    xlabel({'Time (sec)'},'FontSize',14);
    ylabel({'Altitude (m)'},'FontSize',14);
    title('Space jump simulation - altitude-v-time','FontSize',15);
    figure3 = figure;
    axes3 = axes('Parent',figure3,'FontSize',12);
    box(axes3,'on');
    hold(axes3,'all');
    plot(sols(:,1),sols(:,3),'LineWidth',2,'Color',[1 0 0]);
    xlabel({'Altitude (m)'},'FontSize',14);
    ylabel({'Density (kg/m^3)'},'FontSize',14);
    title('Space jump simulation - Density-v-altitude','FontSize',15);

    max_vel = max(-sols(:,2))
    jump_time = times(end)

     function dwdt = eom(t,w)
         y = w(1); v = w(2); rho= w(3);
         theta = temp(y);
         dydt = v;
         dvdt = -g + 0.5*rho*v^2*A*Cd/m;
         drhodt = -g*rho*v/(R*theta);
         dwdt = [dydt;dvdt;drhodt];
     end

     function [eventval,stopcalc,dir] = event(t,w)
        y = w(1);
        eventval = y-1524;
        stopcalc = 1;
        dir = -1;
     end

end
function theta = temp(y)
  if (y>20000)
      theta = 178 + 140*(y-20000)/30000;
  elseif (y>14000)
      theta = 178;
  else
      theta = 298 - 120*y/14000;
  end

end
```

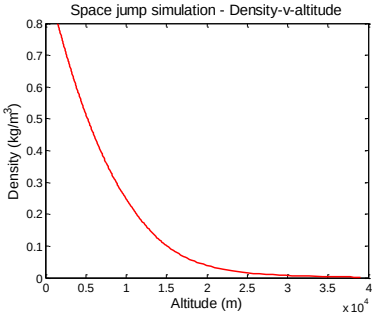
[0 POINTS]

4.4 Hence plot graphs showing (i) the variation of speed with time; (ii) the variation of altitude with time; and (iii) the variation of density with altitude.

The graphs are shown below







[6 POINTS]