Table of Contents

ENSC180-Assignment2

```
% Student Name 1: Nicholas Chu

% Student 1 #: 301440034

% Student 1 userid (email): nmc10@sfu.ca

% Student Name 2:

% Student 2 #:

% Student 2 userid (email):

% Below, edit to list any people who helped you with the assignment,
%      or put 'none' if nobody helped (the two of) you.

% Helpers: none
```

Instructions:

- Put your name(s), student number(s), userid(s) in the above section.
- Edit the "Helpers" line.
- Your group name should be "A2_<userid1>_<userid2>" (eg. A2_stu1_stu2)
- Form a group as described at: https://courses.cs.sfu.ca/docs/students
- Replace "% your work here" below, or similar, with your own answers and work.
- Nagvigate to the "PUBLISH" tab (located on top of the editor) * Click on the "Publish" dropdown and choose pdf as "Output file format" under "Edit Publishing Options..." * Click "Publish" button. Ensure a report is automatically generated
- You will submit THIS file (assignment2.m), and the PDF report (assignment2.pdf). Craig Scratchley, Spring 2021

main

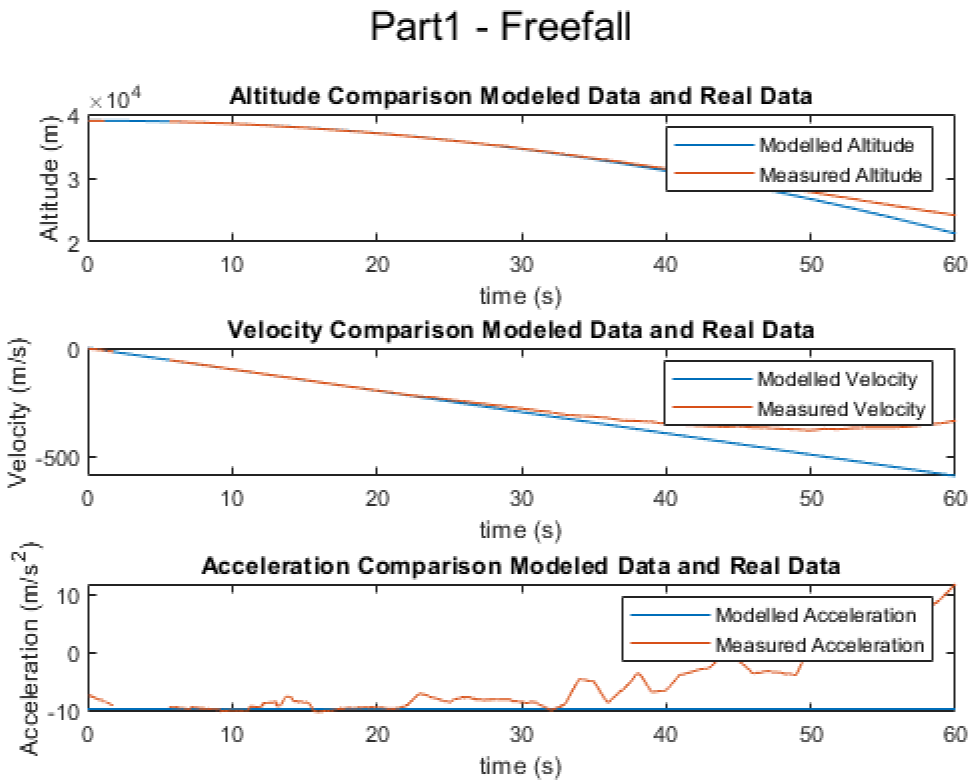
```
function main

% clf

% constants -- you can put constants for the program here
%MY_CONST = 123;

GRAV_CONSTANT = 6.67430 * 10^-11;
EARTH_MASS = 5.972 * 10^24;
EARTH_RADIUS = 6371 * 10^3;

% variables -- you can put variables for the program here
%myVar = 456;

% prepare the data
% <place your work here>

filename = 'data_clean_more_fixed.xlsx';
Measured_data = xlsread(filename);

temp = Measured_data(:,3);      % v
temp = temp./3.6;
Measured_data(:,3) = -temp;

% ... = xlsread()/csvread()/readtable()
% ...
% myVector(isnan(myVector))=[];

% <put here any conversions that are necessary>
```

Part 1

Answer some questions here in these comments... How accurate is the model for the first portion of the minute?

```
% The model is sufficiently accurate for the first portion of the
 minute,
% the lines of the measured and modelled, are nearly identical

% How accurate is the model for the last portion of that first minute?

% As it gets closer the to the end of the first minute the model
 becomes
% less accurate. We can see the modelled and measured lines diverging.

% Comment on the acceleration calculated from the measured data.

% Is there any way to smooth the acceleration calculated from the
 data?
% We can use the smoothdata function to smooth out the acceleration
 from
```

```
% the data. We could also decrease the increment in time to get a more
% continuous source of data

part = 1;
% model Felix Baumgartner’s altitude, velocity, and acceleration for
 the
% first minute after he jumped from  meters above sea level

 figure(1)
 [T, M] = ode45(@fall, [0, 60], [38969.4, 0]);
 plotComparisons(60, "Part1 - Freefall", T, M, Measured_data, 1);
```

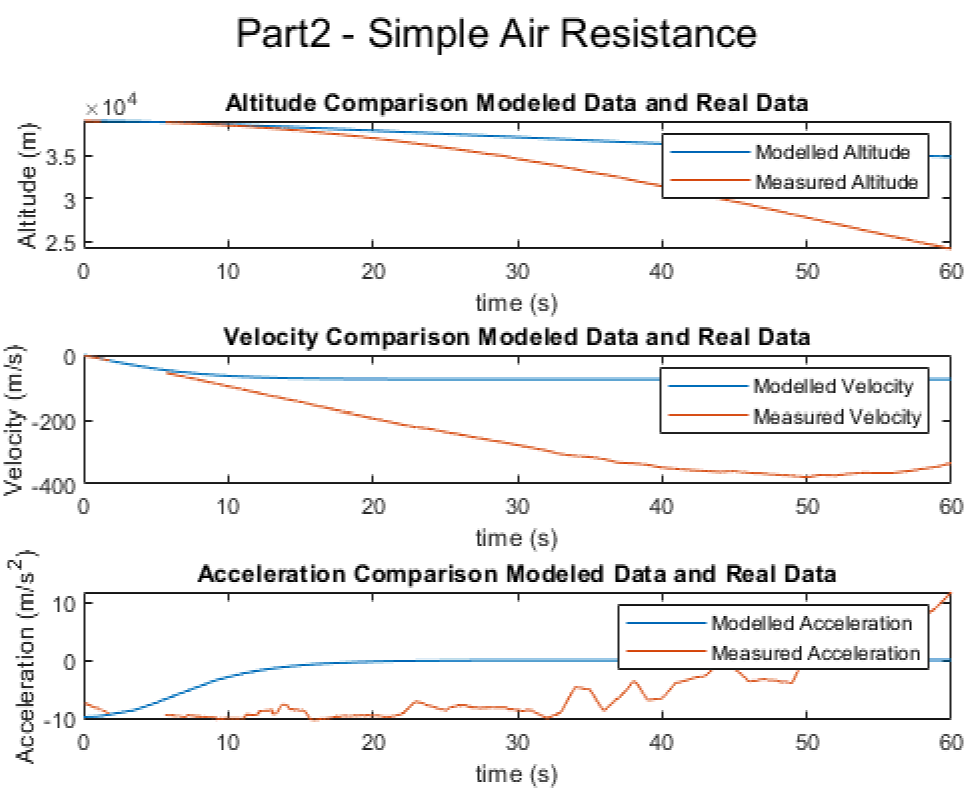


Part 2

Answer some questions here in these comments... Estimate your uncertainty in the mass that you have chosen (at the beginning of the jump).

```
% Felix's Mass was determined to be 118kg Based on research about
% skydiving/freefall gear. There is a +-10% uncertainty in his mass,
 which
% includes his suit, O2 tank and parachute

% How sensitive is the velocity and altitude reached after 60 seconds
 to
%    changes in the chosen mass?
```

```
% The sensitivity of the velocity and altitude due to changes in mass
 are
% fairly insignificant. The difference between the upper and lower
 bounds
% of the mass estimate is less than 1%.

part = 2;

figure(2)
[T, M] = ode45(@fall, [0, 60], [38969.4, 0]);
plotComparisons(60, "Part2 - Simple Air Resistance", T, M,
 Measured_data, 1);


% <call here your function to create your plots>
%plotComparisons(<...>, 'Part2 - Simple Air Resistance', T, M <, ...>)
```



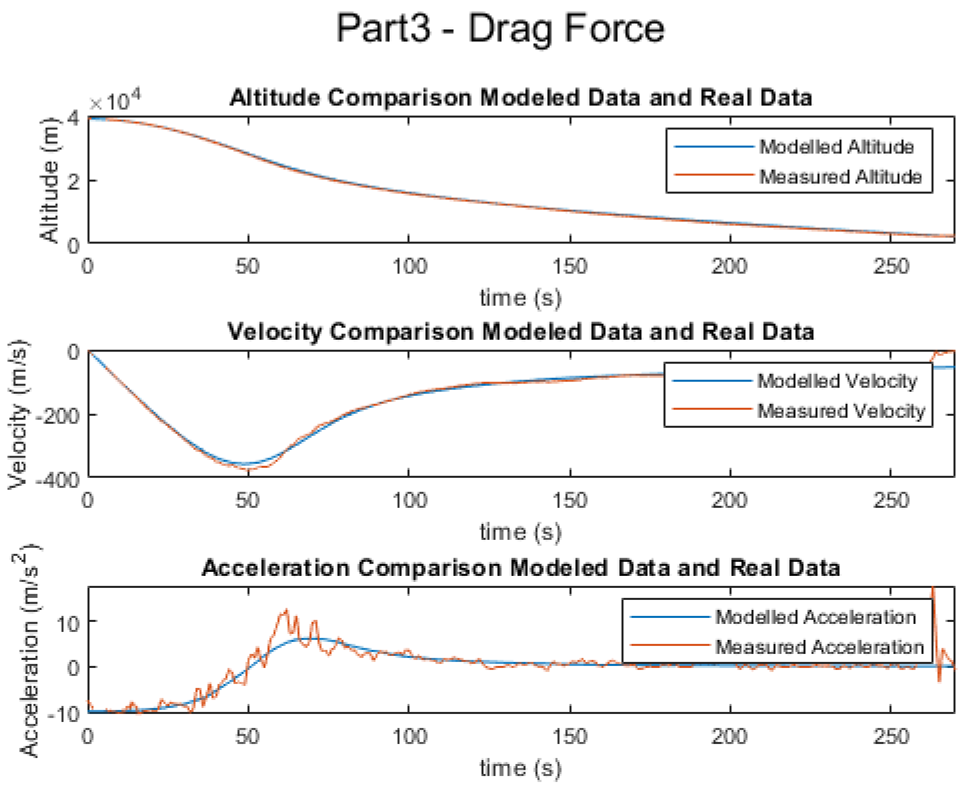
Part 3

Answer some questions here in these comments... Felix was wearing a pressure suit and carrying oxygen. Why?

```
% He needed his suit in order to protect himself from solar radiations
% (similarly to astronauts) and for protection from the cold. He was
 carrying
% oxygen due to decreased air density in the stratosphere. A lower air
```

```
% density means lower oxygen concentration, so the oxygen tank was needed
% for him to breathe.

%     What can we say about the density of air in the stratosphere?

% the density of air in the stratosphere is significantly less than the
% density of air at sea level. Air density generally decreases with altitude

%     How is the density of air different at around 39,000 meters than it
%     is on the ground?

% The density of air at around 39,000 meters is 0.003996 kg/m^3.
% The density of air at sea level is around 1.225 kg/m^3
% <put your answer here in these comments>

% What are the factors involved in calculating the density of air?
% How do those factors change when we end up at the ground but start
% at the stratosphere?
% Please explain how calculating air density up
% to the stratosphere is more complicated than say just in the troposphere.
% <put your answer here in these comments>

% The factors involved with calculating the density of air are: Air
% composition, but primarily humidity, Temperature, and Pressure

%As we descend, the air pressure increases exponentially
%Density at drop height : 0.003996 kg/m^3
%Density at surface Surface: 1.225 kg/m^3

%As we descend the temperature decreases. The temperature is warmer on the
%surface, but precise data depends upon geographical factors.
%At drop height: max cold temp ~ -57 degrees celsius

%As we descend in altitude, the humidity generally becomes greater as 99%
%of water vapour exists in the troposphere.
%Also, humidity can increase while passing through clouds when descending

% Lower in the troposphere, the factors mentioned above remain relatively
% more constant with predictable behaviour. However in the troposphere the
% factors change less predictably. As you descend from 40,000 meters,
% temperatures decrease until about 20,000 meters where it stagnates until
% we reach the troposphere. In the troposphere, the temperature generally
```

```
% increase steadily with decreasing altitude. Furthermore, UV-B and
 UV-C
% light interacts with O3 (ozone) and O2 in the stratosphere, creating
 an
% ever changing air composition. In the troposphere, the air
 composition
% remains more constant.

% What method(s) can we employ to estimate [the ACd] product?
% We can look at the approximate cross sectional area of Felix by
 looking
% at his height, and multiplying it by the approximate drag
 coefficient for
% a human freediver

% What is your estimated [ACd] product?
% Estimated ACd = 0.8
%
% [Given what we are told in the textbook about the simple drag
 constant, b,]
%   does the estimate for ACd seem reasonable?
%
% Yes, it seems reasonable because the ACd product should be around 1.

part = 3;

 figure(3)
 [T, M] = ode45(@fall, [0, 270], [38969.4, 0]);
 plotComparisons(270, "Part3 - Drag Force", T, M, Measured_data, 1);

% <place your work here>
```



Part 4

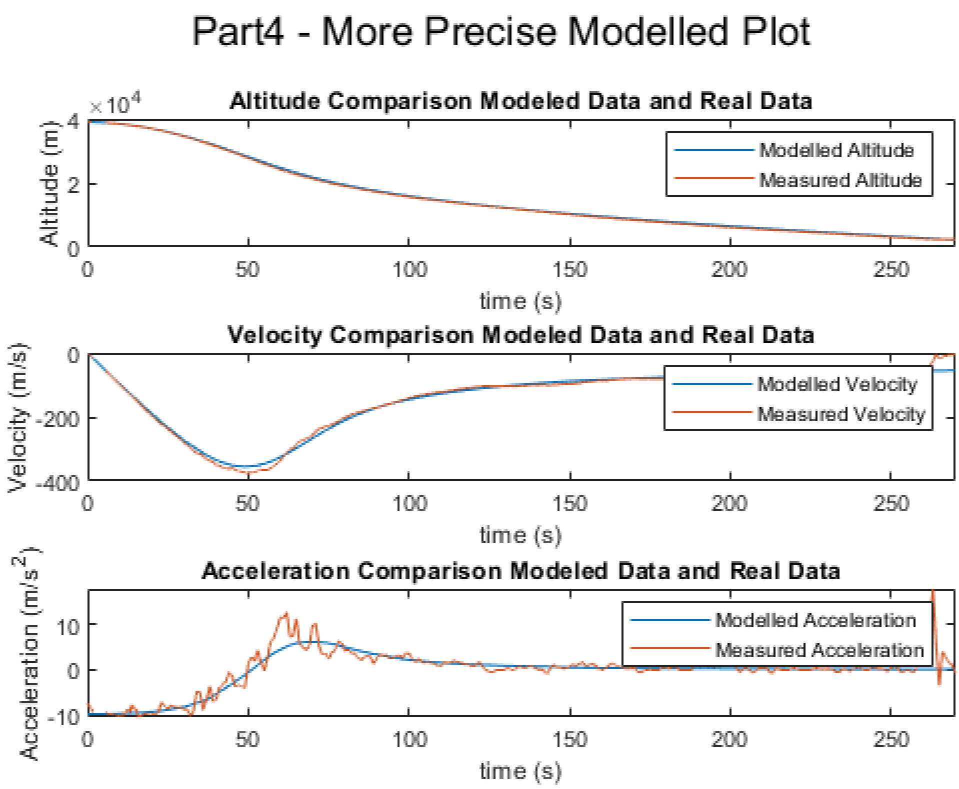
Answer some questions here in these comments... What is the actual gravitational field strength around 39,000 meters?

The actual gravitational field strength is around 9.70085 N/kg

```
% How sensitive is the altitude reached after 4.5 minutes to simpler
 and
% more complicated ways of modelling the gravitational field strength?
% the change in gravitational force from 2399 meters (altitude after
 4.5
% minutes) to 0 meters only differs by 0.075% when using more
 complicated
% ways of modelling gravity. Therefore, the altitude is not very
 sensitive
% to changes

% What other changes could we make to our model? Refer to, or at least
% attempt to explain, the physics behind any changes that you propose.
% We could factor in the changes in the temperature in the atmosphere
 which
% affects air density (and subsequently drag force) by this formula:
%
% Fd = 1/2 * rho * v^2 * Cd * A
%
```

```
% where rho is the density of the fluid, and is proportional to
 temperature
% change
%
% We could also take into account cross winds and updrafts experienced
% during the descent. An updraft would cause the acceleration in the
% negative y direction to decrease, due to the forces from the updraft
% opposing the direction of gravity.


% What is a change that we could make to our model that would result
 in
%   insignificant changes to the altitude reached after 4.5 minutes?

%   After 4.5 minutes, Felix has pulled his chute, so the way he
 positions
%   his body would have an insignifcant effect on drag when compared
 to the
%   parachute. Futhermore, the change in gravitational force from 2399
 meters
%   (altitude after 4.5 minutes) to 0 meters only differs by 0.075%,
 which
%   is also slightly insignificant for graphing purposes. The small
 change
%   in mass from when he pulled his parachute would also have an
%   insignifcant change on the acceleration, and consequently the
 altitude
%   after 4.5 minutes.

% How can we decide what change is significant and what change is
%   insignificant?
% We can perform calulations or determine by inspection which changes
 will
% have a noticeable effect on acceleration.

% [What changes did you try out to improve the model?  (Show us your
 changes
%   even if they didn't make the improvement you hoped for.)]
% I tried finding information about the changes in Felix's mass when
 he
% deployed his parachute, but determined that the changes were
% insignificant. I tried experimenting with different surface areas of
% felix depending on his bodily position (Arms to side vs arms out)
 but the
% changes were once again insignificant. I also tried to find
 information
% about any wind, but could not find any quantitive data on this. I
 believe
% that this would have a meaningful impact on the jump, if we can
 acquire
% this data. I just used the recalculated gravity and the recalculated
 drag
% for part 4 as shown below.
```

```
part = 4;

figure(4)
[T, M] = ode45(@fall, [0, 270], [38969.4, 0]);
plotComparisons(270, "Part4 - More Precise Modelled Plot", T, M,
 Measured_data, 1);
```



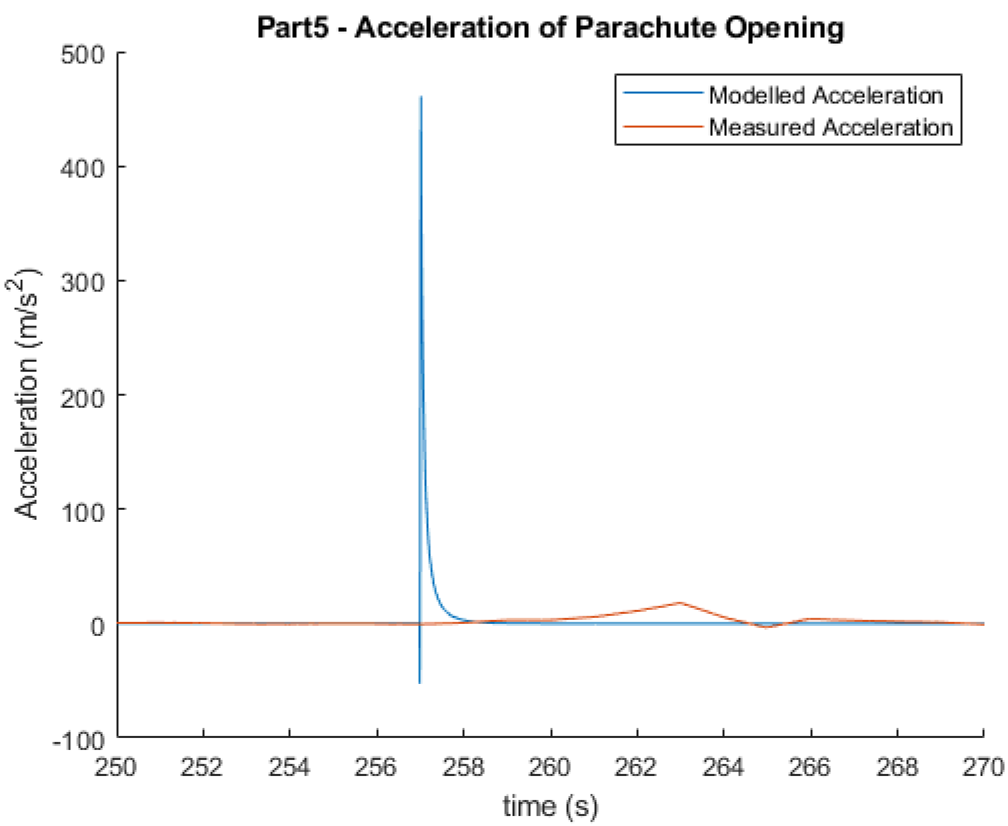
Part 5

Answer some questions here in these comments... At what altitude does Felix pull the ripcord to deploy his parachute? At an altitude of about 2680m

```
% Recalculate the CdA product with the parachute open, and modify your
%   code so that you use one CdA product before and one after this
 altitude.
%   According to this version of the model, what is the maximum
 magnitude
%   of acceleration that Felix experiences?
% The model where Felix's parachute opens instantly, has Felix
% experiencing an acceleration of over 460 m/s^2

%   How safe or unsafe would such an acceleration be for Felix?
% The highest acceleration a human can withstand is 9g's (88.2
% m/s^2). With the parachute deploying instantly, Felix would
 experience
```

```
% over 47g's, which would be incredibly unsafe, and would result in
 his
% death.

part = 5;


%Make a single acceleration-plot figure that includes, for each of the
%model and the acceleration calculated from measurements, the moment
 when
%the parachute opens and the following 10 or so seconds. If you have
%trouble solving this version of the model, just plot the acceleration
%calculated from measurements.


figure(5)
[T, M] = ode45(@fall, [0, 270], [38969.4, 0]);
plotComparisons(270, "Part5 - Acceleration of Parachute Opening", T,
 M, Measured_data, 0);
```

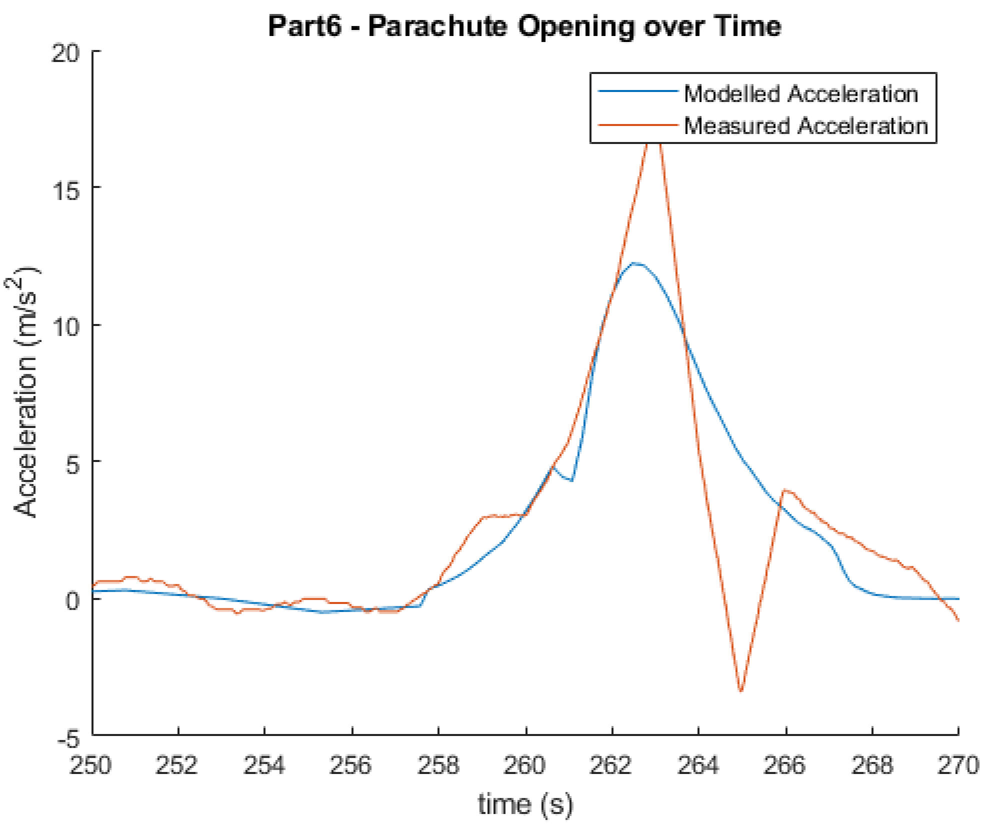
Warning: One or more altitudes above upper limit.

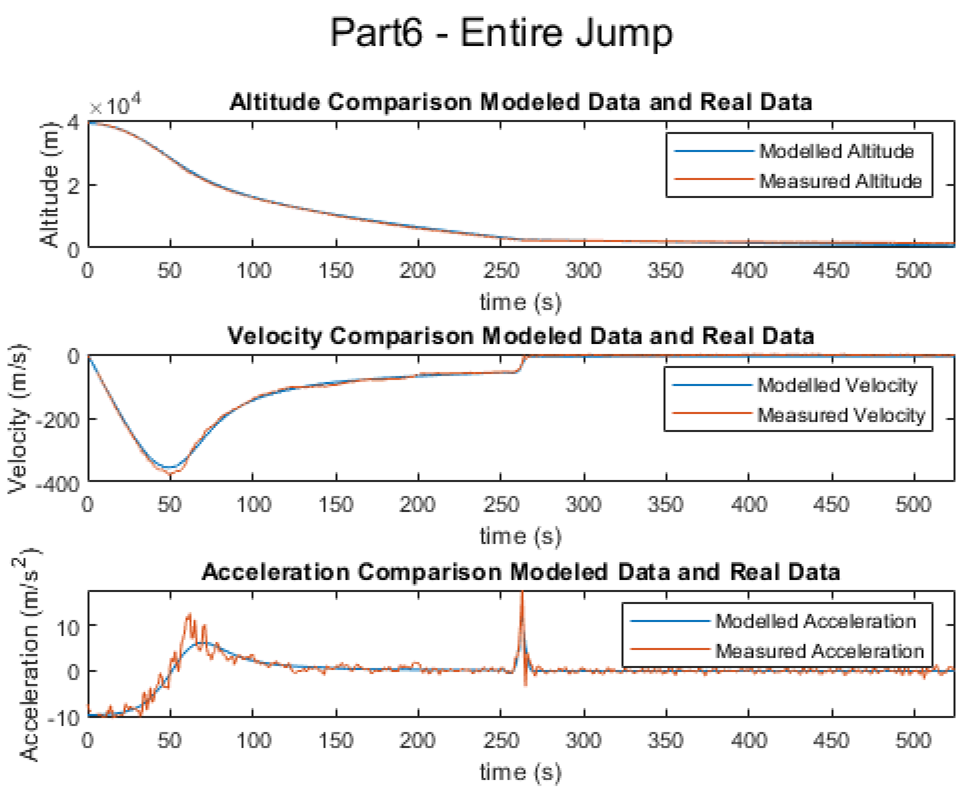


Part 6

Answer some questions here in these comments... How long does it take for Felix's parachute to open?

```
% about 13 seconds
% With this model, the maximum acceleration Felix experiences is about
% 12.2m/s^2, which is well within the safety limits for a human. Felix
% would be fine.
part = 6;

%Redraw the acceleration figure from the previous Part but using the
 new
%   model. Also, using your plotting function from Part 1, plot the
%   measured/calculated data and the model for the entire jump from
%   stratosphere to ground.

figure(6)
[T, M] = ode45(@fall, [0, 525], [38969.4, 0]);
plotComparisons(270, "Part6 - Parachute Opening over Time", T, M,
 Measured_data, 0);

figure(7)
plotComparisons(525, "Part6 - Entire Jump", T, M, Measured_data, 1);
```





nested functions

nested functions below are required for the assignment. see Appendix B of Physical Modeling in MATLAB for discussion of nested functions

```
function res = fall(t, X)
    %FALL <Summary of this function goes here>
    %   <Detailed explanation goes here>

    % do not modify this function unless required by you for some
 reason!

    y = X(1); % the first element is position
    v = X(2); % the second element is velocity

    dydt = v; % velocity: the derivative of position w.r.t. time
    dvdt = acceleration(t, y, v); % acceleration: the derivative of
 velocity w.r.t. time

    res = [dydt; dvdt]; % pack the results in a column vector
end

function res = acceleration(t, y, v)
    % <insert description of function here>
    % input...
    %   t: time
```

```
    %   y: altitude
    %   v: velocity
    % output...
    %   res: acceleration

    % do not modify this function unless required by you for some
 reason!


    grav = gravityEst(y);

    if part == 1
        res = -grav;
    else
        m = mass(t, v);
        temp_grav = -grav;                 % acceleration due to
 gravity in m/s^2
        drag_force = drag(t, y, v, m);      % call to drag function
        temp_drag = drag_force/m;           % calculate drag
 acceleration
        res = temp_grav + temp_drag;        % total acceleration

    end
end

function grav = gravityEst(y)
    % estimate the acceleration due to gravity as a function of
 altitude, y
    g_SEA = 9.807;  % gravity at sea level in m/s^2

    if part <= 3
        grav = g_SEA;
    else
        % Universal Gravity equation: F = (G * M1 * M2)/(r^2)
        grav = (GRAV_CONSTANT * EARTH_MASS)/((y +
 EARTH_RADIUS)^2); %Felix's mass cancels: a=F/m
    end
end

function res = mass(~, ~)
    %Give mass of Felix
    res = 118;    %mass in kg of Felix and all his equipment

end

function res = drag(t, y, v, m)
%calculate drag force, depending on part number

    if part <= 4

        if part == 2                    % b = 0.2 given
            b = 0.2;
            res = -b * v^2 * sign(v);
```

```
       else
           res = before(t, y);
       end
   end


   if part == 5

       if(t < 257)                  %Time before pulled rip cord
           res = before(t, y);

       else                         %Time after pulled rip cord
           res = after(t,y);

       end
   end


   if part == 6

      if(t <= 254)                    % Before Chute Opens
           res = before(t, y);

      elseif(t > 254 && t <= 267)   % During Chute Opening
          Rho = stdatmo(y);
          completion = ((t - 267) + 13)/13;    % Relative completion
of opening 0 to 100%

          if 25*(completion)^5.5 < 0.7          % If chute area is
less than felix area, we add felix area
              area = 25*(completion)^5.5 + 0.7;   % a goes from 0 to
~1.4
              Cd = 1 + completion/10 ;          % Cd = 1 with a little
extra from chute


          else
              area = 25*(completion)^5.5;
              Cd = 1.7;                % Cd of chute is ~1.75, 1.7
plots nicer
          end

           res = 0.5 * Rho * v^2 * Cd * area;

      else                             % After Chute Opening
           res = after(t,y);
      end
   end

   function res = before(t, y)
           Rho = stdatmo(y);
           res = 0.5 * Rho * v^2 * 1 * 0.8; %Fd = 1/2 * p * v^2 * Cd
* A
   end
```

```
    function res = after(t, y)
            Rho = stdatmo(y);
            res = 0.5 * Rho * v^2 * 1.7 * 25; %Fd = 1/2 * p * v^2 * Cd
 * A
    end
end
```

Additional nested functions

Nest any other functions below.

```
%Do not put functions in other files when you submit, except you can
 use
%    the stdatmo function in file stdatmo.m which has been provided to
 you.

function plotComparisons(elapsed, Title, T, M, Measured_data,
 Triple_Plot)

%measured Data
measured_Time = Measured_data(:,1);             % Time in first column
measured_Altitude = Measured_data(:,2);         % Altitude in second
 column
measured_Velocity = Measured_data(:,3);         % Velocity in third
 column
measTime_delta = diff(measured_Time);           % For acceleration
measVelocity_delta = diff(measured_Velocity);   % For accelertation
measured_Acceleration = measVelocity_delta./measTime_delta; % a = v/t

measured_Acceleration = smoothdata(measured_Acceleration);  % smoothen
measured_Acceleration = smoothdata(measured_Acceleration);  % smoothen

accel_measured_Time = measured_Time(2:size(measured_Time,1)); % For x
 axis


%modeled Data
modelled_Altitude = M(:,1);                  % Altitude in first
 column
modelled_Velocity = M(:,2);                  % Velocity in the second
 column
time_delta = diff(T);                        % For acceleration
velocity_delta = diff(modelled_Velocity);    % For acceleration
modelled_Acceleration = velocity_delta ./ time_delta;      % a = v/t

sizeofTime = size(T, 1);                    % For x axis
accel_T = T(2:sizeofTime);
size(accel_T);
size(modelled_Acceleration);


clear title;
```

```
hold on;

if Triple_Plot == true

%Altitude Plot
subplot(3,1,1);
plot(T, modelled_Altitude, measured_Time, measured_Altitude);
title('Altitude Comparison Modeled Data and Real Data');
legend('Modelled Altitude', 'Measured Altitude')
xlabel('time (s)')
ylabel('Altitude (m)')
xlim([0, elapsed]);

%Velocity Plot
subplot(3,1,2)
plot(T, modelled_Velocity, measured_Time, measured_Velocity);
title('Velocity Comparison Modeled Data and Real Data')
legend('Modelled Velocity', 'Measured Velocity')
xlabel('time (s)')
ylabel('Velocity (m/s)')
xlim([0, elapsed]);

%Acceleration Plot
subplot(3,1,3)
plot(accel_T, modelled_Acceleration, accel_measured_Time,
 measured_Acceleration);
title('Acceleration Comparison Modeled Data and Real Data');
legend('Modelled Acceleration', 'Measured Acceleration')
xlabel('time (s)')
ylabel('Acceleration (m/s^2)')
xlim([0, elapsed]);

sgt = sgtitle(Title);
sgt.FontSize = 15;

%Single Acceleration plot
else
    plot(accel_T, modelled_Acceleration, accel_measured_Time,
 measured_Acceleration);
    title(Title);
    legend('Modelled Acceleration', 'Measured Acceleration')
    xlabel('time (s)')
    ylabel('Acceleration (m/s^2)')
    xlim([250, 270]);
end
end

% end of nested functions

end % closes function main.
```

Published with MATLAB® R2020b