Problem 1

```
%script_simaircraft05.m
%
%  Copyright (c) 2019 Mark L. Psiaki.  All rights reserved.
%
%  This Matlab script simulates the motion of an
%  aircraft by using the design and control input
%  data in maneuver02_data.mat
%  and the point-mass translational aircraft
%  dynamics model in ffunctaircraft03.m.
%
%  This script uses N-point Euler numerical
%  integration in order to do the numerical
%  integration.
%
%  This script also makes plots of the flight time history.
%
%  Clear the Matlab workspace.
%
   clear
%
%  Load the aircraft parameters, the thrust, angle-of-attack,
%  and roll/bank-angle input time histories, and the initial
%  state vector.
%
   load maneuver02_data
%
%  Define the aircraft dynamics function handle
%  in a form that is suitable for input to ode45.m
%  or to an Euler numerical integration.
%
   ffunctode45_03 = @(tdum,xdum) ...
             ffunctaircraft03(tdum,xdum,m,S,CLalpha,CD0,oneoverpiARe,...
                              tinhist,Thist,alphahist,phihist);
%
%  Define the time span of the simulation, computing outputs
%  every half second.
%
   t0 = tinhist(1,1);
   tf = tinhist(end,1);
%
%  Compute the ode45.m results using a very
%  precise relative tolerance.
%
   tspan = (t0:0.5:tf)';
   optionsode45 = odeset('RelTol',1.e-12);
   tic
   [thist03,xhist03] = ode45(ffunctode45_03,tspan,x0,optionsode45);
   timetoode45 = toc
%
%  Set up 3 different N values and prepare to store 3 different
%  time histories generated by Euler integration.
%
   Nvec = [1000;10000;100000];
   thist03_euler_cell = cell(3,1);
   xhist03_euler_cell = cell(3,1);
```

```
   timetoeuler_vec = zeros(3,1);
%
%  Select N and perform N steps of Euler numerical integration
%  to go from time tmin to time tmax.
%
   n = size(x0,1);
   for jj = 1:3
      N = Nvec(jj,1)
      deltat = (tf-t0)/N;
      Np1 = N + 1;
      thist03_euler = zeros(Np1,1);
      xhist03_euler = zeros(Np1,n);
      thist03_euler(1,1) = t0;
      xhist03_euler(1,:) = x0';
      clear Np1
      tic
      tkp1 = t0;
      xkp1 = x0;
      for k = 0:(N-1);
         tk = tkp1;
         xk = xkp1;
         fk = ffunctode45_03(tk,xk);
         tkp1 = tk+deltat;
         xkp1 = xk+deltat*fk;
         kp2 = k + 2;
         thist03_euler(kp2,1) = tkp1;
         xhist03_euler(kp2,:) = xkp1';
      end
      clear k tk xk fk tkp1 xkp1 kp2
      timetoeuler = toc
      timetoeuler_vec(jj,1) = timetoeuler;
      thist03_euler_cell{jj,1} = thist03_euler;
      xhist03_euler_cell{jj,1} = xhist03_euler;
   end
   clear jj N deltat thist03_euler xhist03_euler timetoeuler
%
%  Plot the ground track.
%
   figure(1)
   hold off
   plot(xhist03(:,2)*0.001,xhist03(:,1)*0.001,'b-','LineWidth',3)
   hold on
   plot(xhist03_euler_cell{1,1}(:,2)*0.001,...
        xhist03_euler_cell{1,1}(:,1)*0.001,'k:','LineWidth',1.5)
   plot(xhist03_euler_cell{2,1}(:,2)*0.001,...
        xhist03_euler_cell{2,1}(:,1)*0.001,'g--','LineWidth',1.5)
   plot(xhist03_euler_cell{3,1}(:,2)*0.001,...
        xhist03_euler_cell{3,1}(:,1)*0.001,'r-.','LineWidth',1.5)
   hold off
   grid
   axis('equal')
   xlabel('Eastward Displacement (km)')
   ylabel('Northward Displacment (km)')
   title('Ground Tracks for simaircraft05.mat')
   legend('ode45.m',...
          ['Euler integration w/',int2str(Nvec(1,1)),' steps'],...
          ['Euler integration w/',int2str(Nvec(2,1)),' steps'],...
          ['Euler integration w/',int2str(Nvec(3,1)),' steps'])
%
```

```
%  Plot the altitude, airspeed, flight-path angle,
%  and heading angle time histories.
%
   figure(2)
   subplot(411)
   hold off
   plot(thist03,-xhist03(:,3),'b-','LineWidth',3)
   hold on
   plot(thist03_euler_cell{1,1},-xhist03_euler_cell{1,1}(:,3),...
        'k:','LineWidth',1.5)
   plot(thist03_euler_cell{2,1},-xhist03_euler_cell{2,1}(:,3),...
        'g--','LineWidth',1.5)
   plot(thist03_euler_cell{3,1},-xhist03_euler_cell{3,1}(:,3),...
        'r-.','LineWidth',1.5)
   hold off
   grid
   ylabel('Altitude above Airport (m)')
   title('State time histories for simaircraft05.mat')
   legend('ode45.m',...
          ['Euler integration w/',int2str(Nvec(1,1)),' steps'],...
          ['Euler integration w/',int2str(Nvec(2,1)),' steps'],...
          ['Euler integration w/',int2str(Nvec(3,1)),' steps'])
   subplot(412)
   hold off
   plot(thist03,xhist03(:,4),'b-','LineWidth',3)
   hold on
   plot(thist03_euler_cell{1,1},xhist03_euler_cell{1,1}(:,4),...
        'k:','LineWidth',1.5)
   plot(thist03_euler_cell{2,1},xhist03_euler_cell{2,1}(:,4),...
        'g--','LineWidth',1.5)
   plot(thist03_euler_cell{3,1},xhist03_euler_cell{3,1}(:,4),...
        'r-.','LineWidth',1.5)
   hold off
   grid
   ylabel('Airspeed (m/sec)')
   subplot(413)
   hold off
   plot(thist03,xhist03(:,5)*(180/pi),'b-','LineWidth',3)
   hold on
   plot(thist03_euler_cell{1,1},...
        xhist03_euler_cell{1,1}(:,5)*(180/pi),'k:','LineWidth',1.5)
   plot(thist03_euler_cell{2,1},...
        xhist03_euler_cell{2,1}(:,5)*(180/pi),'g--','LineWidth',1.5)
   plot(thist03_euler_cell{3,1},...
        xhist03_euler_cell{3,1}(:,5)*(180/pi),'r-.','LineWidth',1.5)
   hold off
   grid
   ylabel('Flight Path Angle (deg)')
   subplot(414)
   hold off
   plot(thist03,xhist03(:,6)*(180/pi),'b-','LineWidth',3)
   hold on
   plot(thist03_euler_cell{1,1},...
        xhist03_euler_cell{1,1}(:,6)*(180/pi),'k:','LineWidth',1.5)
   plot(thist03_euler_cell{2,1},...
        xhist03_euler_cell{2,1}(:,6)*(180/pi),'g--','LineWidth',1.5)
   plot(thist03_euler_cell{3,1},...
        xhist03_euler_cell{3,1}(:,6)*(180/pi),'r-.','LineWidth',1.5)
   hold off
```

```
   grid
   ylabel('Heading Angle (deg)')
   xlabel('Time (seconds)')
%
%  Plot the thrust, angle-of-attack, and roll/bank-angle
%  time histories.
%
   figure(3)
   subplot(311)
   hold off
   plot(tinhist,Thist,'LineWidth',1.5)
   grid
   ylabel('Thrust (N)')
   title('Control input time histories for simaircraft05.mat')
   subplot(312)
   hold off
   plot(tinhist,alphahist*(180/pi),'LineWidth',1.5)
   grid
   ylabel('Angle-of-Attack (deg)')
   subplot(313)
   hold off
   plot(tinhist,phihist*(180/pi),'LineWidth',1.5)
   grid
   ylabel('Roll/Bank-Angle (deg)')
   xlabel('Time (seconds)')
%
%  Display final state error.
%
   format long
   errorxfinal_1000 = xhist03_euler_cell{1,1}(end,:)' - xhist03(end,:)'
   errorxfinal_10000 = xhist03_euler_cell{2,1}(end,:)' - xhist03(end,:)'
   errorxfinal_100000 = xhist03_euler_cell{3,1}(end,:)' - xhist03(end,:)'
%
%  Save the results.
%
   textcommands = ['These data have been generated by the',...
                   ' commands in script_simaircraft05.m'];
   save simaircraft05
   disp('errorxfinal_10000./errorxfinal_100000')
   disp(errorxfinal_10000./errorxfinal_100000)
```

Output

```
timetoode45 =    1.5511


N =        1000
timetoeuler =    0.2197


N =       10000
timetoeuler =    2.1907


N =      100000
timetoeuler =   21.5372
```

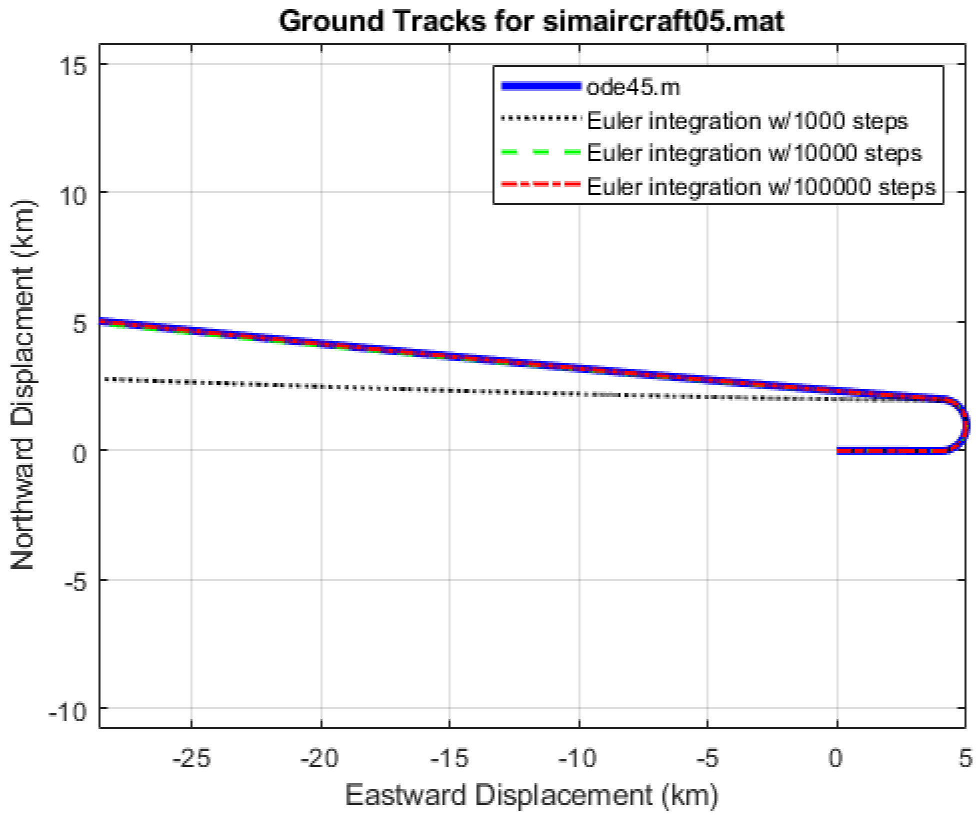
```
errorxfinal_1000 =

   1.0e+03 *

  -2.249895849628666
   0.251280539653530
   0.152176155278537
   0.003678238682550
   0.000056219697832
  -0.000066881570922


errorxfinal_10000 =

   1.0e+02 *

  -1.056444894361602
  -0.038021493036576
   0.020757723483021
   0.000410954780697
   0.000011519644391
  -0.000031502241030


errorxfinal_100000 =

  -9.952207111369717
  -0.365993885796343
   0.177905214820271
   0.003264838969415
   0.000100452336104
  -0.000296849328265

errorxfinal_10000./errorxfinal_100000
  10.615181964558259
  10.388559621384907
  11.667855551053442
  12.587290967402469
  11.467771520357749
  10.612198859979557
```

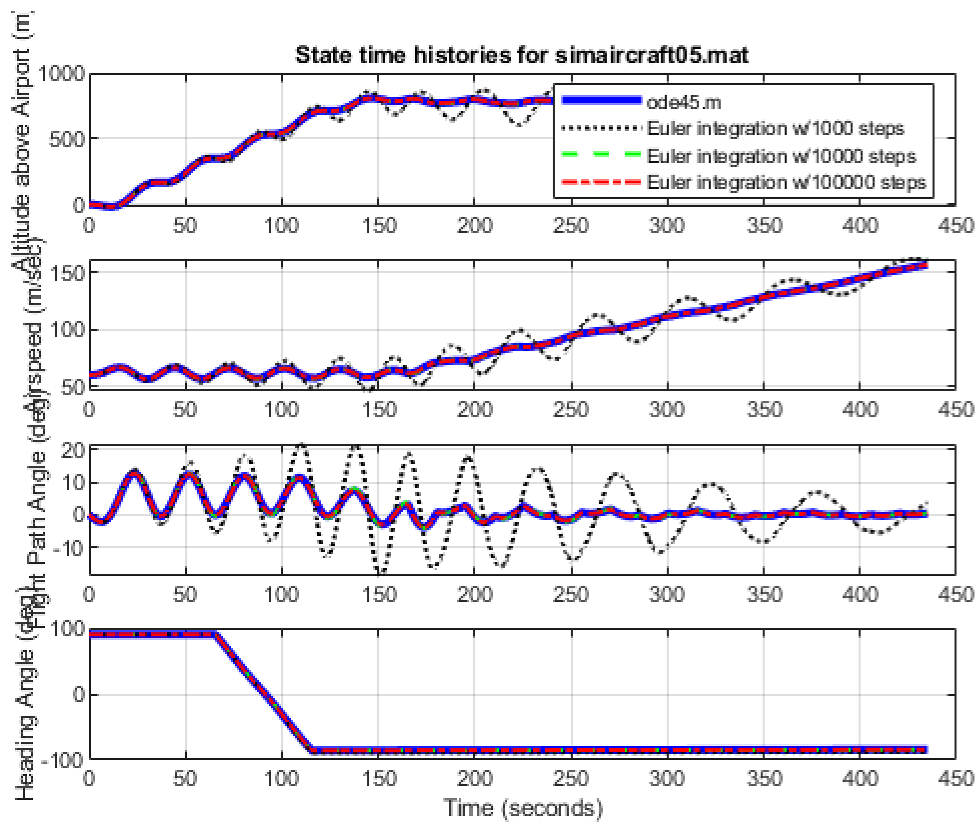
Q) How does Euler integration compare to `ode45.m` in terms of execution speed?

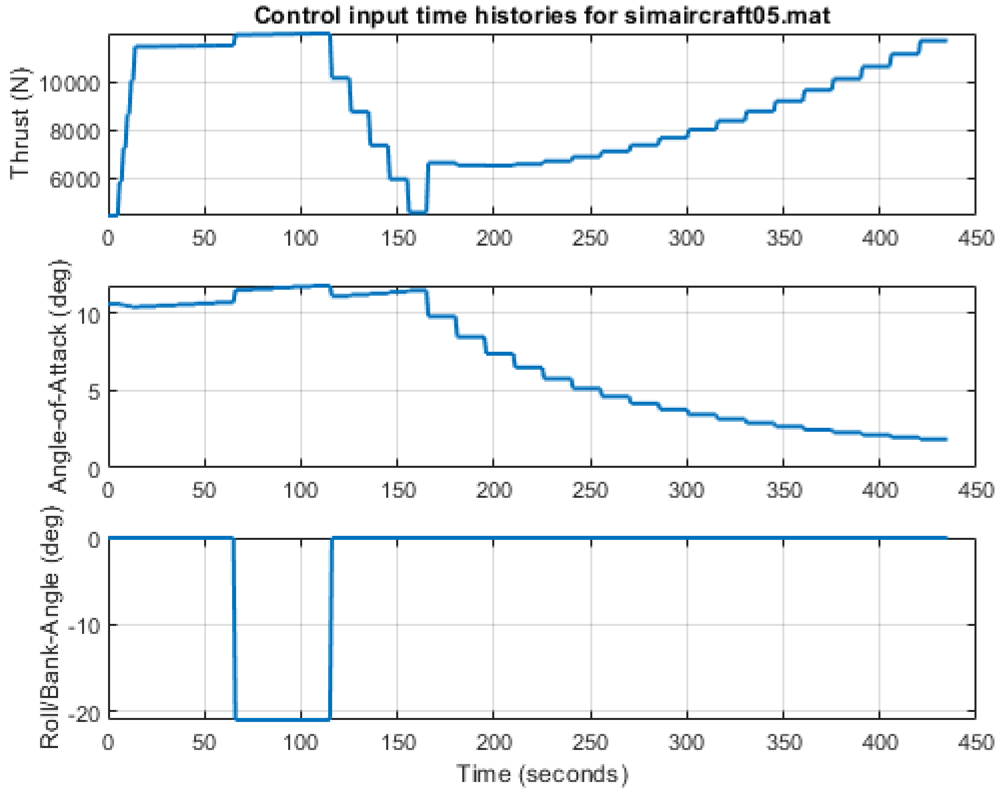
Ans) Given the accuracy considered in the ode45.m versus the accuracy achieved by Euler integration, the ode45 is very fast. The 1000 step Euler is fast but it is inaccurate.

Q) The theory of Euler's method predicts that these ratios should be about 10. Is that true?

Ans) Yes. It is close to 10, or at least 10.


Ground Tracks for simaircraft05.mat
ode45.m
Euler integration w/1000 steps
Euler integration w/10000 steps
Euler integration w/100000 steps
Northward Displacment (km)
Eastward Displacement (km)


State time histories for simaircraft05.mat
ode45.m
Euler integration w/1000 steps
Euler integration w/10000 steps
Euler integration w/100000 steps
Altitude above Airport (m)
Airspeed (m/sec)
Flight Path Angle (deg)
Heading Angle (deg)
Time (seconds)


Control input time histories for simaircraft05.mat
Thrust (N)
10000
8000
6000
Angle-of-Attack (deg)
10
5
0
Roll/Bank-Angle (deg)
0
-10
-20
0
50
100
150
200
250
300
350
400
450
Time (seconds)

Problem 2

```
%script_simaircraft06.m
%
%  Copyright (c) 2019 Mark L. Psiaki.  All rights reserved.
%
%  This Matlab script simulates the motion of an
%  aircraft by using the design and control input
%  data in maneuver02_data.mat
%  and the point-mass translational aircraft
%  dynamics model in ffunctaircraft03.m.
%
%  This script uses N-point trapezoidal numerical
%  integration in order to do the numerical
%  integration.
%
%  This script also makes plots of the flight time history.
%
%  Clear the Matlab workspace.
%
   clear
%
%  Load the aircraft parameters, the thrust, angle-of-attack,
%  and roll/bank-angle input time histories, and the initial
%  state vector.
%
   load maneuver02_data
%
%  Define the aircraft dynamics function handle
%  in a form that is suitable for input to ode45.m
%  or to a trapezoidal numerical integration.
%
   ffunctode45_03 = @(tdum,xdum) ...
            ffunctaircraft03(tdum,xdum,m,S,CLalpha,CD0,oneoverpiARe,...
                             tinhist,Thist,alphahist,phihist);
%
%  Define the time span of the simulation, computing outputs
%  every half second.
%
   t0 = tinhist(1,1);
   tf = tinhist(end,1);
%
%  Compute the ode45.m results using a very
%  precise relative tolerance.
%
   tspan = (t0:0.5:tf)';
   optionsode45 = odeset('RelTol',1.e-12);
   tic
   [thist03,xhist03] = ode45(ffunctode45_03,tspan,x0,optionsode45);
   timetoode45 = toc
%
%  Set up 3 different N values and prepare to store 3 different
%  time histories generated by trapezoidal integration.
%
   Nvec = [500;2000;8000];
   thist03_trapez_cell = cell(3,1);
   xhist03_trapez_cell = cell(3,1);
```

```
   timetotrapez_vec = zeros(3,1);
%
%  Select N and perform N steps of trapezoidal numerical integration
%  to go from time tmin to time tmax.
%
   n = size(x0,1);
   for jj = 1:3
      N = Nvec(jj,1)
      deltat = (tf-t0)/N;
      Np1 = N + 1;
      thist03_trapez = zeros(Np1,1);
      xhist03_trapez = zeros(Np1,n);
      thist03_trapez(1,1) = t0;
      xhist03_trapez(1,:) = x0';
      clear Np1
      tic
      tkp1 = t0;
      xkp1 = x0;
      for k = 0:(N-1);
         tk = tkp1;
         xk = xkp1;
         tak = tk;
         xak = xk;
         fak = ffunctode45_03(tak,xak);
         tbk = tk+deltat;
         xbk = xk+deltat*fak;
         fbk = ffunctode45_03(tbk,xbk);
         tkp1 = tk+deltat;
         xkp1 = xk+(deltat/2)*(fak+fbk);
         kp2 = k + 2;
         thist03_trapez(kp2,1) = tkp1;
         xhist03_trapez(kp2,:) = xkp1';
      end
      clear k tk xk tak xak fak tbk xbk fbk tkp1 xkp1 kp2
      timetotrapez = toc
      timetotrapez_vec(jj,1) = timetotrapez;
      thist03_trapez_cell{jj,1} = thist03_trapez;
      xhist03_trapez_cell{jj,1} = xhist03_trapez;
   end
   clear jj N deltat thist03_trapez xhist03_trapez timetotrapez
%
%  Plot the ground track.
%
   figure(1)
   hold off
   plot(xhist03(:,2)*0.001,xhist03(:,1)*0.001,'b-','LineWidth',3)
   hold on
   plot(xhist03_trapez_cell{1,1}(:,2)*0.001,...
        xhist03_trapez_cell{1,1}(:,1)*0.001,'k:','LineWidth',1.5)
   plot(xhist03_trapez_cell{2,1}(:,2)*0.001,...
        xhist03_trapez_cell{2,1}(:,1)*0.001,'g--','LineWidth',1.5)
   plot(xhist03_trapez_cell{3,1}(:,2)*0.001,...
        xhist03_trapez_cell{3,1}(:,1)*0.001,'r-.','LineWidth',1.5)
   hold off
   grid
   axis('equal')
   xlabel('Eastward Displacement (km)')
   ylabel('Northward Displacment (km)')
   title('Ground Tracks for simaircraft06.mat')
```

```
    legend('ode45.m',...
           ['Trapezoidal integration w/',int2str(Nvec(1,1)),' steps'],...
           ['Trapezoidal integration w/',int2str(Nvec(2,1)),' steps'],...
           ['Trapezoidal integration w/',int2str(Nvec(3,1)),' steps'])
%
%  Plot the altitude, airspeed, flight-path angle,
%  and heading angle time histories.
%
    figure(2)
    subplot(411)
    hold off
    plot(thist03,-xhist03(:,3),'b-','LineWidth',3)
    hold on
    plot(thist03_trapez_cell{1,1},-xhist03_trapez_cell{1,1}(:,3),...
         'k:','LineWidth',1.5)
    plot(thist03_trapez_cell{2,1},-xhist03_trapez_cell{2,1}(:,3),...
         'g--','LineWidth',1.5)
    plot(thist03_trapez_cell{3,1},-xhist03_trapez_cell{3,1}(:,3),...
         'r-.','LineWidth',1.5)
    hold off
    grid
    ylabel('Altitude above Airport (m)')
    title('State time histories for simaircraft06.mat')
    legend('ode45.m',...
           ['Trapezoidal integration w/',int2str(Nvec(1,1)),' steps'],...
           ['Trapezoidal integration w/',int2str(Nvec(2,1)),' steps'],...
           ['Trapezoidal integration w/',int2str(Nvec(3,1)),' steps'])
    subplot(412)
    hold off
    plot(thist03,xhist03(:,4),'b-','LineWidth',3)
    hold on
    plot(thist03_trapez_cell{1,1},xhist03_trapez_cell{1,1}(:,4),...
         'k:','LineWidth',1.5)
    plot(thist03_trapez_cell{2,1},xhist03_trapez_cell{2,1}(:,4),...
         'g--','LineWidth',1.5)
    plot(thist03_trapez_cell{3,1},xhist03_trapez_cell{3,1}(:,4),...
         'r-.','LineWidth',1.5)
    hold off
    grid
    ylabel('Airspeed (m/sec)')
    subplot(413)
    hold off
    plot(thist03,xhist03(:,5)*(180/pi),'b-','LineWidth',3)
    hold on
    plot(thist03_trapez_cell{1,1},...
         xhist03_trapez_cell{1,1}(:,5)*(180/pi),'k:','LineWidth',1.5)
    plot(thist03_trapez_cell{2,1},...
         xhist03_trapez_cell{2,1}(:,5)*(180/pi),'g--','LineWidth',1.5)
    plot(thist03_trapez_cell{3,1},...
         xhist03_trapez_cell{3,1}(:,5)*(180/pi),'r-.','LineWidth',1.5)
    hold off
    grid
    ylabel('Flight Path Angle (deg)')
    subplot(414)
    hold off
    plot(thist03,xhist03(:,6)*(180/pi),'b-','LineWidth',3)
    hold on
    plot(thist03_trapez_cell{1,1},...
         xhist03_trapez_cell{1,1}(:,6)*(180/pi),'k:','LineWidth',1.5)
```

```
   plot(thist03_trapez_cell{2,1},...
        xhist03_trapez_cell{2,1}(:,6)*(180/pi),'g--','LineWidth',1.5)
   plot(thist03_trapez_cell{3,1},...
        xhist03_trapez_cell{3,1}(:,6)*(180/pi),'r-.','LineWidth',1.5)
   hold off
   grid
   ylabel('Heading Angle (deg)')
   xlabel('Time (seconds)')
%
%  Plot the thrust, angle-of-attack, and roll/bank-angle
%  time histories.
%
   figure(3)
   subplot(311)
   hold off
   plot(tinhist,Thist,'LineWidth',1.5)
   grid
   ylabel('Thrust (N)')
   title('Control input time histories for simaircraft06.mat')
   subplot(312)
   hold off
   plot(tinhist,alphahist*(180/pi),'LineWidth',1.5)
   grid
   ylabel('Angle-of-Attack (deg)')
   subplot(313)
   hold off
   plot(tinhist,phihist*(180/pi),'LineWidth',1.5)
   grid
   ylabel('Roll/Bank-Angle (deg)')
   xlabel('Time (seconds)')
%
%  Display final state error.
%
   format long
   errorxfinal_500 = xhist03_trapez_cell{1,1}(end,:)' - xhist03(end,:)'
   errorxfinal_2000 = xhist03_trapez_cell{2,1}(end,:)' - xhist03(end,:)'
   errorxfinal_8000 = xhist03_trapez_cell{3,1}(end,:)' - xhist03(end,:)'
%
%  Save the results.
%
   textcommands = ['These data have been generated by the',...
                   ' commands in script_simaircraft06.m'];
   save simaircraft06
   disp('errorxfinal_2000./errorxfinal_8000')
   disp(errorxfinal_2000./errorxfinal_8000)
```

Output

```
timetoode45 =   1.379359100000000


N =   500
timetotrapez =   0.237054400000000


N =        2000
timetotrapez =   0.917606000000000
```

```
N =        8000
timetotrapez =   3.608178800000000


errorxfinal_500 =

   1.0e+02 *

   2.470874862165565
   0.312780779269015
  -0.020084760958699
  -0.001686353740498
   0.000002202438762
   0.000076213603099


errorxfinal_2000 =

   4.784498364122555
   0.478795131981315
  -0.085716895745350
  -0.007187758640072
   0.000037796464585
   0.000149039574289


errorxfinal_8000 =

   0.302894297890816
   0.029262601630762
  -0.004806028911162
  -0.000405483572820
   0.000002360428262
   0.000009412494139

errorxfinal_2000./errorxfinal_8000
  15.795934084725539
  16.362015176326043
  17.835285082508793
  17.726386768487977
  16.012545345590301
  15.834227580369248
```

Q) How do `errorxfinal_2000` and `errorxfinal_8000` for this run compare `errorxfinal_10000` and `errorxfinal_100000` for the Euler integration run?

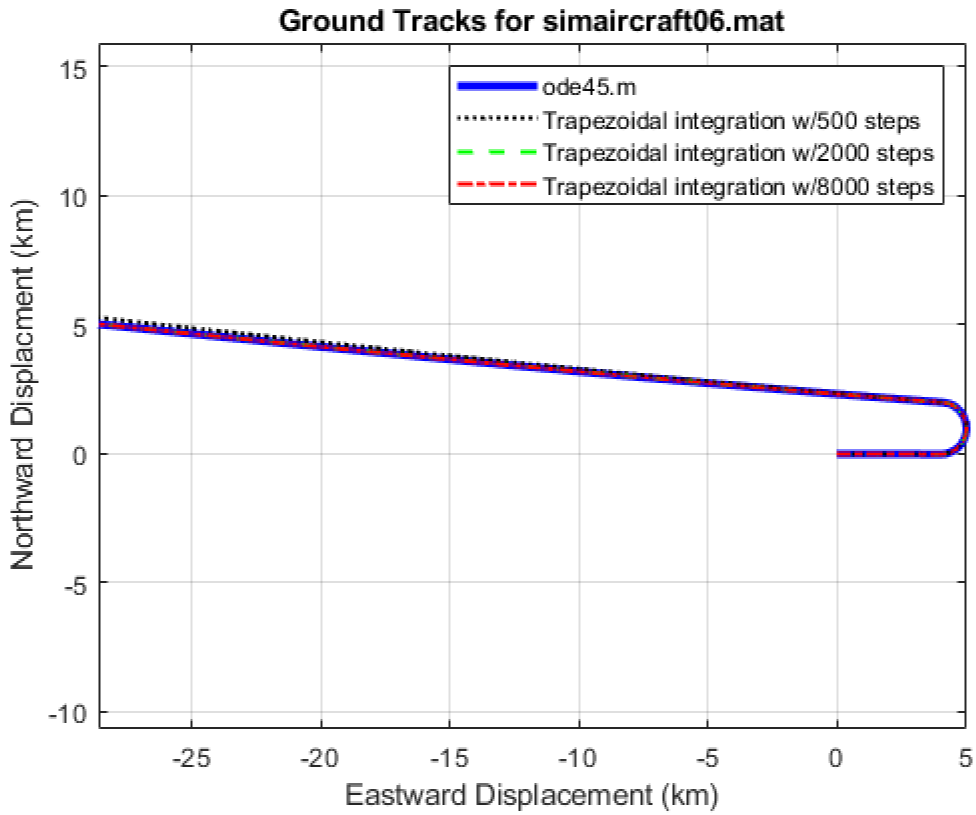
Ans) The are off by 2 orders of magnitude. About 100 to 40 times lower in Trapezoidal integration.

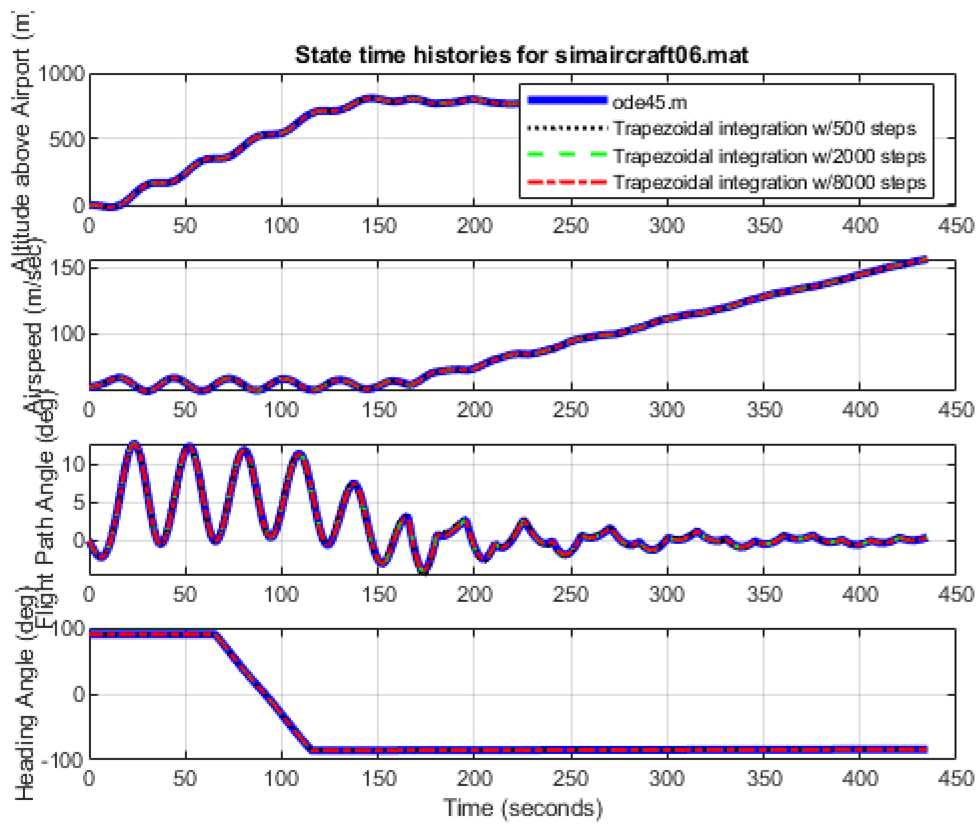
Q) How does trapezoidal integration compare to `ode45.m` in terms of execution speed?

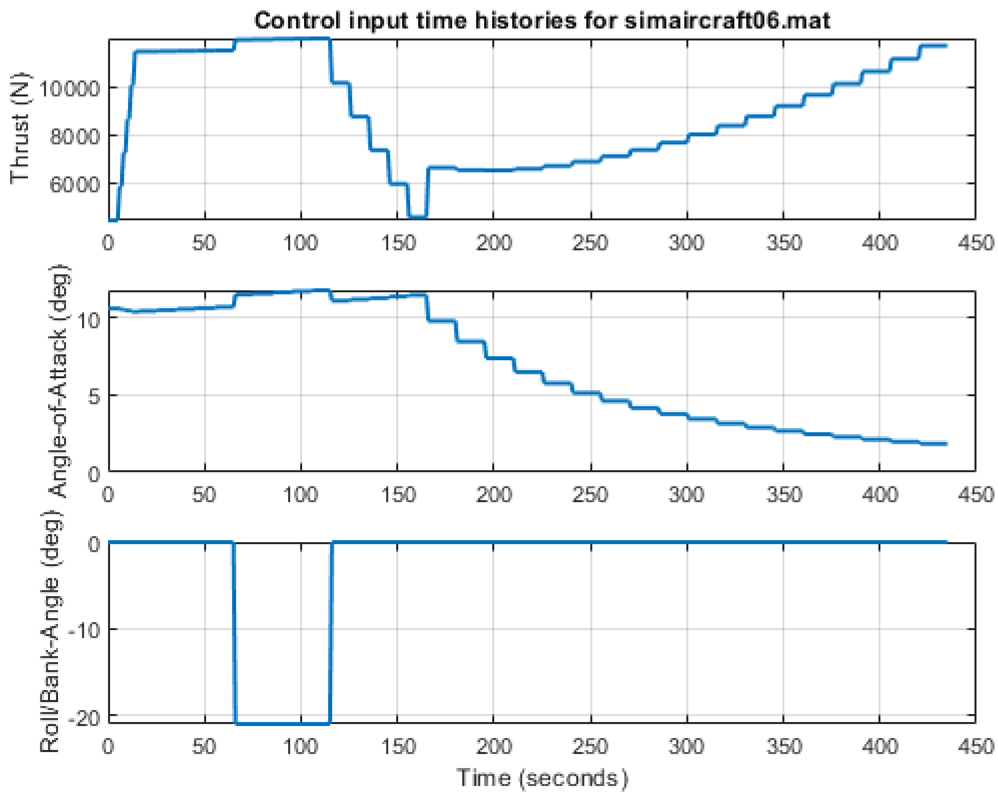
Ans) The speeds are comparable.

Q) The theory of Euler's method predicts that these ratios should be about 10. Is that true?

Ans) Yes. It is close to 16.


Ground Tracks for simaircraft06.mat
ode45.m
Trapezoidal integration w/500 steps
Trapezoidal integration w/2000 steps
Trapezoidal integration w/8000 steps
Northward Displacment (km)
Eastward Displacement (km)
15
10
5
0
-5
-10
-25
-20
-15
-10
-5
0
5


State time histories for simaircraft06.mat
ode45.m
Trapezoidal integration w/500 steps
Trapezoidal integration w/2000 steps
Trapezoidal integration w/8000 steps
Altitude above Airport (m)
Airspeed (m/sec)
Flight Path Angle (deg)
Heading Angle (deg)
Time (seconds)
1000
500
0
150
100
10
5
0
100
0
-100
0
50
100
150
200
250
300
350
400
450



Published with MATLAB® R2019a

Problem 3

```
%script_simaircraft07.m
%
%  Copyright (c) 2019 Mark L. Psiaki.  All rights reserved.
%
%  This Matlab script simulates the motion of an
%  aircraft by using the design and control input
%  data in maneuver02_data.mat
%  and the point-mass translational aircraft
%  dynamics model in ffunctaircraft03.m.
%
%  This script uses N-point 4th-order Runge-Kutta numerical
%  integration in order to do the numerical integration.
%
%  This script also makes plots of the flight time history.
%
%  Clear the Matlab workspace.
%
   clear
%
%  Load the aircraft parameters, the thrust, angle-of-attack,
%  and roll/bank-angle input time histories, and the initial
%  state vector.
%
   load maneuver02_data
%
%  Define the aircraft dynamics function handle
%  in a form that is suitable for input to ode45.m
%  or to a 4th order Runge-Kutta numerical integration.
%
   ffunctode45_03 = @(tdum,xdum) ...
             ffunctaircraft03(tdum,xdum,m,S,CLalpha,CD0,oneoverpiARe,...
                                  tinhist,Thist,alphahist,phihist);
%
%  Define the time span of the simulation, computing outputs
%  every half second.
%
   t0 = tinhist(1,1);
   tf = tinhist(end,1);
%
%  Compute the ode45.m results using a very
%  precise relative tolerance.
%
   tspan = (t0:0.5:tf)';
   optionsode45 = odeset('RelTol',1.e-12);
   tic
   [thist03,xhist03] = ode45(ffunctode45_03,tspan,x0,optionsode45);
   timetoode45 = toc
%
%  Set up 3 different N values and prepare to store 3 different
%  time histories generated by 4th order Runge-Kutta integration.
%
   Nvec = [100;400;1600];
   thist03_4thOrdRK_cell = cell(3,1);
   xhist03_4thOrdRK_cell = cell(3,1);
   timeto4thOrdRK_vec = zeros(3,1);
```

```
%
%  Select N and perform N steps of 4th order Runge-Kutta
%  numerical integration to go from time tmin to time tmax.
%
   n = size(x0,1);
   for jj = 1:3
      N = Nvec(jj,1)
      deltat = (tf-t0)/N;
      Np1 = N + 1;
      thist03_4thOrdRK = zeros(Np1,1);
      xhist03_4thOrdRK = zeros(Np1,n);
      thist03_4thOrdRK(1,1) = t0;
      xhist03_4thOrdRK(1,:) = x0';
      clear Np1
      tic
      tkp1 = t0;
      xkp1 = x0;
      for k = 0:(N-1);
         tk = tkp1;
         xk = xkp1;
         tak = tk;
         xak = xk;
         fak = ffunctode45_03(tak,xak);
         tbk = tk + deltat/2;
         xbk = xk + fak*deltat/2;
         fbk = ffunctode45_03(tbk,xbk);
         tck = tk + deltat/2;
         xck = xk + fbk*deltat/2;
         fck = ffunctode45_03(tck,xck);
         tdk = tk + deltat;
         xdk = xk + fck*deltat;
         fdk = ffunctode45_03(tdk,xdk);
         tkp1 = tk + deltat;
         xkp1 = xk + (deltat/6)*(fak+2*fbk+2*fck+fdk);
         kp2 = k + 2;
         thist03_4thOrdRK(kp2,1) = tkp1;
         xhist03_4thOrdRK(kp2,:) = xkp1';
      end
      clear k tk xk tak xak fak tbk xbk fbk tck xck fck ...
            tdk xdk fdk tkp1 xkp1 kp2
      timeto4thOrdRK = toc
      timeto4thOrdRK_vec(jj,1) = timeto4thOrdRK;
      thist03_4thOrdRK_cell{jj,1} = thist03_4thOrdRK;
      xhist03_4thOrdRK_cell{jj,1} = xhist03_4thOrdRK;
   end
   clear jj N deltat thist03_4thOrdRK xhist03_4thOrdRK timeto4thOrdRK
%
%  Plot the ground track.
%
   figure(1)
   hold off
   plot(xhist03(:,2)*0.001,xhist03(:,1)*0.001,'b-','LineWidth',3)
   hold on
   plot(xhist03_4thOrdRK_cell{1,1}(:,2)*0.001,...
        xhist03_4thOrdRK_cell{1,1}(:,1)*0.001,'k:','LineWidth',1.5)
   plot(xhist03_4thOrdRK_cell{2,1}(:,2)*0.001,...
        xhist03_4thOrdRK_cell{2,1}(:,1)*0.001,'g--','LineWidth',1.5)
   plot(xhist03_4thOrdRK_cell{3,1}(:,2)*0.001,...
        xhist03_4thOrdRK_cell{3,1}(:,1)*0.001,'r-.','LineWidth',1.5)
```

```
    hold off
    grid
    axis('equal')
    xlabel('Eastward Displacement (km)')
    ylabel('Northward Displacment (km)')
    title('Ground Tracks for simaircraft07.mat')
    legend('ode45.m',...
           ['4th Order RK integration w/',int2str(Nvec(1,1)),' steps'],...
           ['4th Order RK integration w/',int2str(Nvec(2,1)),' steps'],...
           ['4th Order RK integration w/',int2str(Nvec(3,1)),' steps'])
%
%  Plot the altitude, airspeed, flight-path angle,
%  and heading angle time histories.
%
    figure(2)
    subplot(411)
    hold off
    plot(thist03,-xhist03(:,3),'b-','LineWidth',3)
    hold on
    plot(thist03_4thOrdRK_cell{1,1},-xhist03_4thOrdRK_cell{1,1}(:,3),...
         'k:','LineWidth',1.5)
    plot(thist03_4thOrdRK_cell{2,1},-xhist03_4thOrdRK_cell{2,1}(:,3),...
         'g--','LineWidth',1.5)
    plot(thist03_4thOrdRK_cell{3,1},-xhist03_4thOrdRK_cell{3,1}(:,3),...
         'r-.','LineWidth',1.5)
    hold off
    grid
    ylabel('Altitude above Airport (m)')
    title('State time histories for simaircraft07.mat')
    legend('ode45.m',...
           ['4th Order RK integration w/',int2str(Nvec(1,1)),' steps'],...
           ['4th Order RK integration w/',int2str(Nvec(2,1)),' steps'],...
           ['4th Order RK integration w/',int2str(Nvec(3,1)),' steps'])
    subplot(412)
    hold off
    plot(thist03,xhist03(:,4),'b-','LineWidth',3)
    hold on
    plot(thist03_4thOrdRK_cell{1,1},xhist03_4thOrdRK_cell{1,1}(:,4),...
         'k:','LineWidth',1.5)
    plot(thist03_4thOrdRK_cell{2,1},xhist03_4thOrdRK_cell{2,1}(:,4),...
         'g--','LineWidth',1.5)
    plot(thist03_4thOrdRK_cell{3,1},xhist03_4thOrdRK_cell{3,1}(:,4),...
         'r-.','LineWidth',1.5)
    hold off
    grid
    ylabel('Airspeed (m/sec)')
    subplot(413)
    hold off
    plot(thist03,xhist03(:,5)*(180/pi),'b-','LineWidth',3)
    hold on
    plot(thist03_4thOrdRK_cell{1,1},...
         xhist03_4thOrdRK_cell{1,1}(:,5)*(180/pi),'k:','LineWidth',1.5)
    plot(thist03_4thOrdRK_cell{2,1},...
         xhist03_4thOrdRK_cell{2,1}(:,5)*(180/pi),'g--','LineWidth',1.5)
    plot(thist03_4thOrdRK_cell{3,1},...
         xhist03_4thOrdRK_cell{3,1}(:,5)*(180/pi),'r-.','LineWidth',1.5)
    hold off
    grid
    ylabel('Flight Path Angle (deg)')
```

```
    subplot(414)
    hold off
    plot(thist03,xhist03(:,6)*(180/pi),'b-','LineWidth',3)
    hold on
    plot(thist03_4thOrdRK_cell{1,1},...
         xhist03_4thOrdRK_cell{1,1}(:,6)*(180/pi),'k:','LineWidth',1.5)
    plot(thist03_4thOrdRK_cell{2,1},...
         xhist03_4thOrdRK_cell{2,1}(:,6)*(180/pi),'g--','LineWidth',1.5)
    plot(thist03_4thOrdRK_cell{3,1},...
         xhist03_4thOrdRK_cell{3,1}(:,6)*(180/pi),'r-.','LineWidth',1.5)
    hold off
    grid
    ylabel('Heading Angle (deg)')
    xlabel('Time (seconds)')
%
%   Plot the thrust, angle-of-attack, and roll/bank-angle
%   time histories.
%
    figure(3)
    subplot(311)
    hold off
    plot(tinhist,Thist,'LineWidth',1.5)
    grid
    ylabel('Thrust (N)')
    title('Control input time histories for simaircraft07.mat')
    subplot(312)
    hold off
    plot(tinhist,alphahist*(180/pi),'LineWidth',1.5)
    grid
    ylabel('Angle-of-Attack (deg)')
    subplot(313)
    hold off
    plot(tinhist,phihist*(180/pi),'LineWidth',1.5)
    grid
    ylabel('Roll/Bank-Angle (deg)')
    xlabel('Time (seconds)')
%
%   Display final state error.
%
    format long
    errorxfinal_100 = xhist03_4thOrdRK_cell{1,1}(end,:)' - xhist03(end,:)'
    errorxfinal_400 = xhist03_4thOrdRK_cell{2,1}(end,:)' - xhist03(end,:)'
    errorxfinal_1600 = xhist03_4thOrdRK_cell{3,1}(end,:)' - xhist03(end,:)'
%
%   Save the results.
%
    textcommands = ['These data have been generated by the',...
                    ' commands in script_simaircraft07.m'];
    save simaircraft07
    disp('errorxfinal_400./errorxfinal_1600')
    disp(errorxfinal_400./errorxfinal_1600)
```

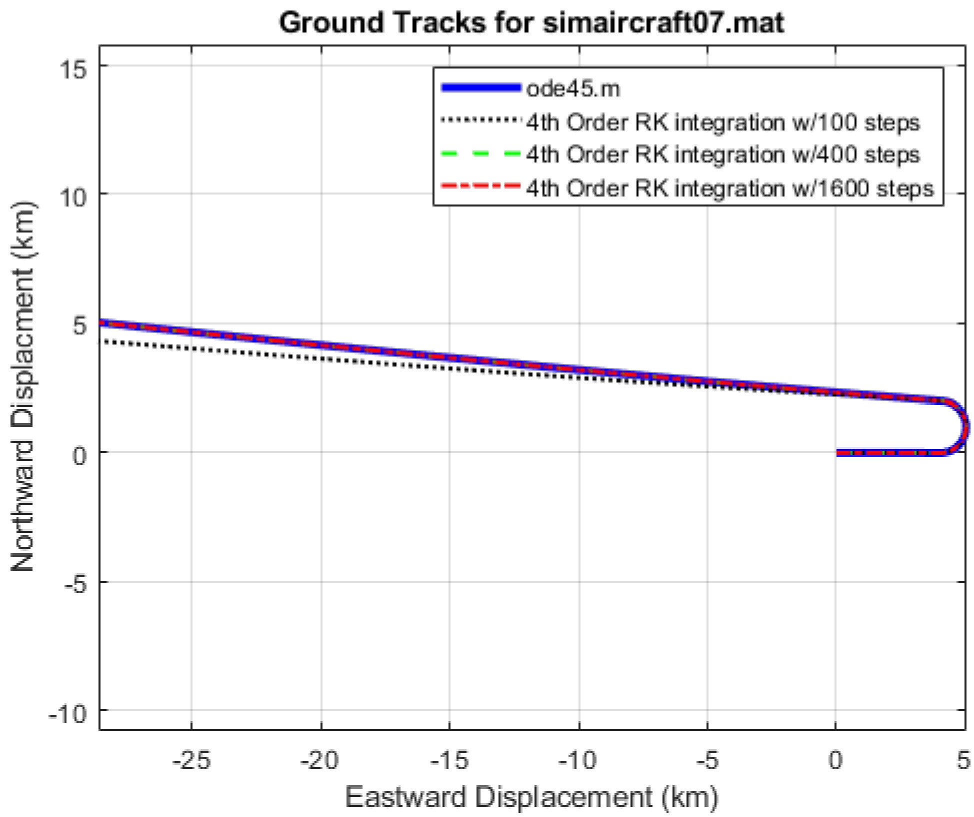
Output

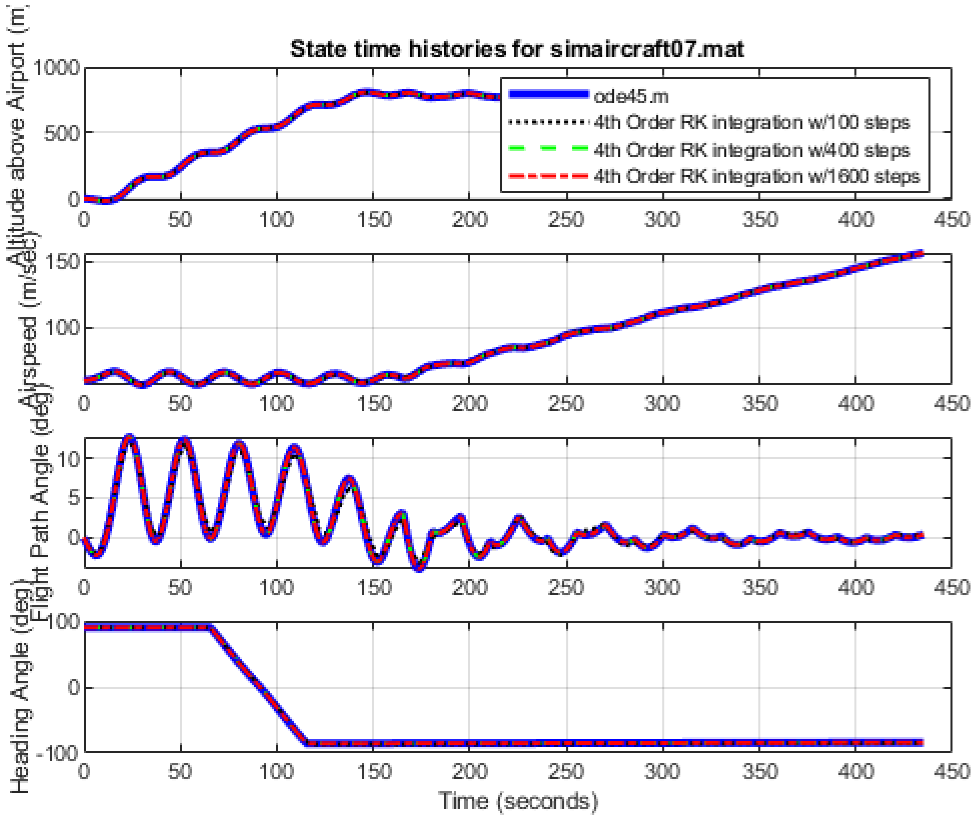
```
timetoode45 =   1.289840300000000


N =   100
timeto4thOrdRK =   0.113976200000000


N =   400
timeto4thOrdRK =   0.353902400000000


N =        1600
timeto4thOrdRK =   1.452845800000000


errorxfinal_100 =

   1.0e+02 *

  -7.127605809049656
  -0.264485502332536
  -0.035365079268283
  -0.002812286643644
   0.000014206432811
  -0.000218345311633


errorxfinal_400 =

  -4.535844163201546
  -5.700187308535533
  -0.484651160040698
  -0.020674546318020
  -0.000334735731401
  -0.000144124937576


errorxfinal_1600 =

   0.107024907014420
  -0.027637959086860
   0.005905547682346
   0.000350192809918
   0.000000553981200
   0.000003273830659

errorxfinal_400./errorxfinal_1600
   1.0e+02 *

  -0.423812016261823
   2.062448710710183
  -0.820670979407234
  -0.590376093753656
  -6.042366269435917
  -0.440233330851968
```

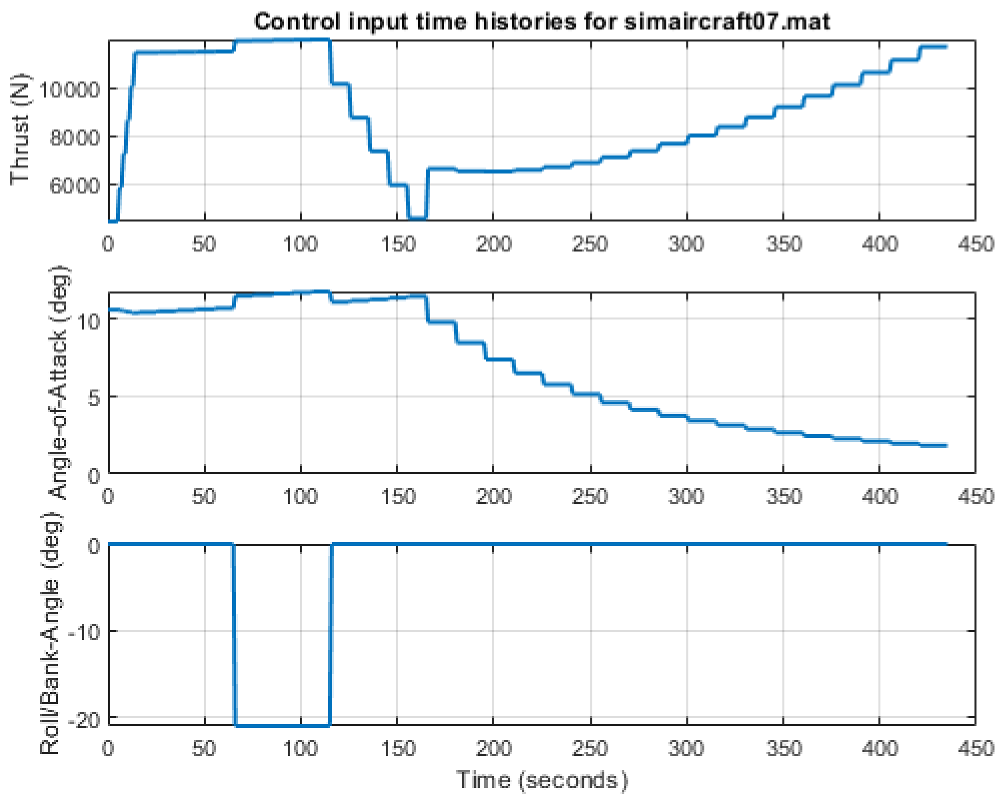
Q) How do errorxfinal_400 and errorxfinal_1600 for this run compare errorxfinal_2000 and errorxfinal_8000 for the trapezoidal integration run?
Ans) The errors are comparable. Same orders of magnitude.

Q) How does the 4th-order Runge-Kutta integration method compare to ode45.m in terms of execution speed?
Ans) Its comparable.

Q) The theory of Euler's method predicts that these ratios should be about 256. Is that true?
Ans) No. The error is not always 256. It varies between 600 and 40, close in order of magnitude.







Published with MATLAB® R2019a

Q4) $f_{bk} = f(t_{bk}, x_{bk}) = f\left(t_k + \frac{\Delta t}{2}, x_k + \frac{\Delta t}{2} f_{ak}\right)$

$f_{ak} = f(t_{ak}, x_{ak}) = f(t_k, x_k) \triangleq f_k$

So,

$$f_{bk} = f\left(t_k + \frac{\Delta t}{2}, x_k + \frac{\Delta t}{2} f_k\right)$$

The Taylor Series Expansion gives

$$f_{bk} = f_k + \left(\left.\frac{\partial f}{\partial t}\right|_{at\, x_k, t_k} \frac{\Delta t}{2}\right) + \left(\left.\frac{\partial f}{\partial \underline{x}}\right|_{at\, x_k, t_k} \frac{\Delta t}{2} f_k\right) + \mathcal{O}(\Delta t^2)$$

$$f_{bk} = f_k + \frac{\Delta t}{2} \underbrace{\left[\frac{\partial f}{\partial t} + \frac{\partial F}{\partial x} f_k\right]_{at\, t_k, x_k}}_{\ddot{x}(t_k)} + \mathcal{O}(\Delta t^2)$$

$$f_{bk} = \dot{x}(t_k) + \frac{\Delta t}{2} \ddot{x}(t_k) + \mathcal{O}(\Delta t^2)$$

From, the assignment Question

$$x_{k+1} = x_k + \Delta t(b_1 f_{ak} + b_2 f_{bk}) = x_k + \Delta t\, b_1 f_k + \Delta t\, b_2 f_{bk}$$

$$x_{k+1} = x_k + b_1 \Delta t f_k + b_2 \Delta t\left(\dot{x}(t_k) + \frac{\Delta t}{2} \ddot{x}(t_k) + \mathcal{O}(\Delta t^2)\right)$$

$$= x_k + \Delta t(b_1 + b_2)\dot{x}(t_k) + b_2 \frac{\Delta t^2}{2} \ddot{x}(t_k) + b_2 \mathcal{O}(\Delta t^3)$$

$\boxed{\text{If } b_1 = 0 \text{ \& } b_2 = 1}$ Ans

$$x_{k+1} = \underbrace{x_k + \Delta t\, \dot{x}(t_k) + \frac{\Delta t^2}{2} \ddot{x}(t_k)}_{\text{first 3 terms of Taylor Series}} + \mathcal{O}(\Delta t^3)$$

making it a 2nd order method
with the Error at $\mathcal{O}(\Delta t^3)$ → polynomial with at least power 3