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)
```

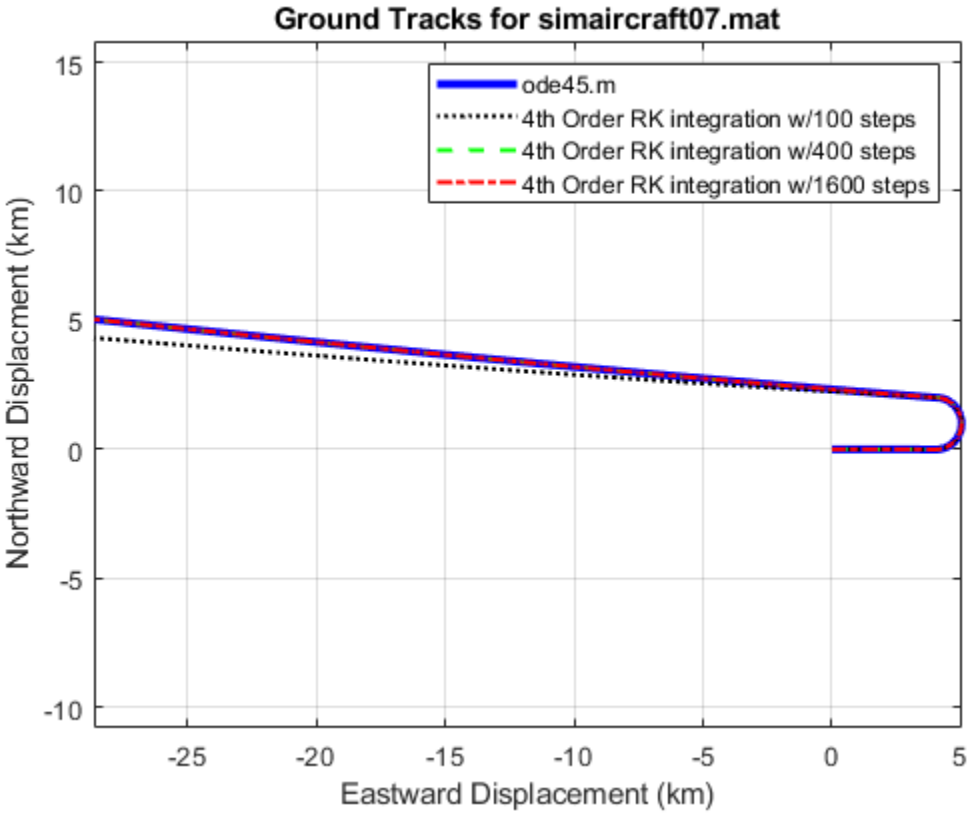
Output

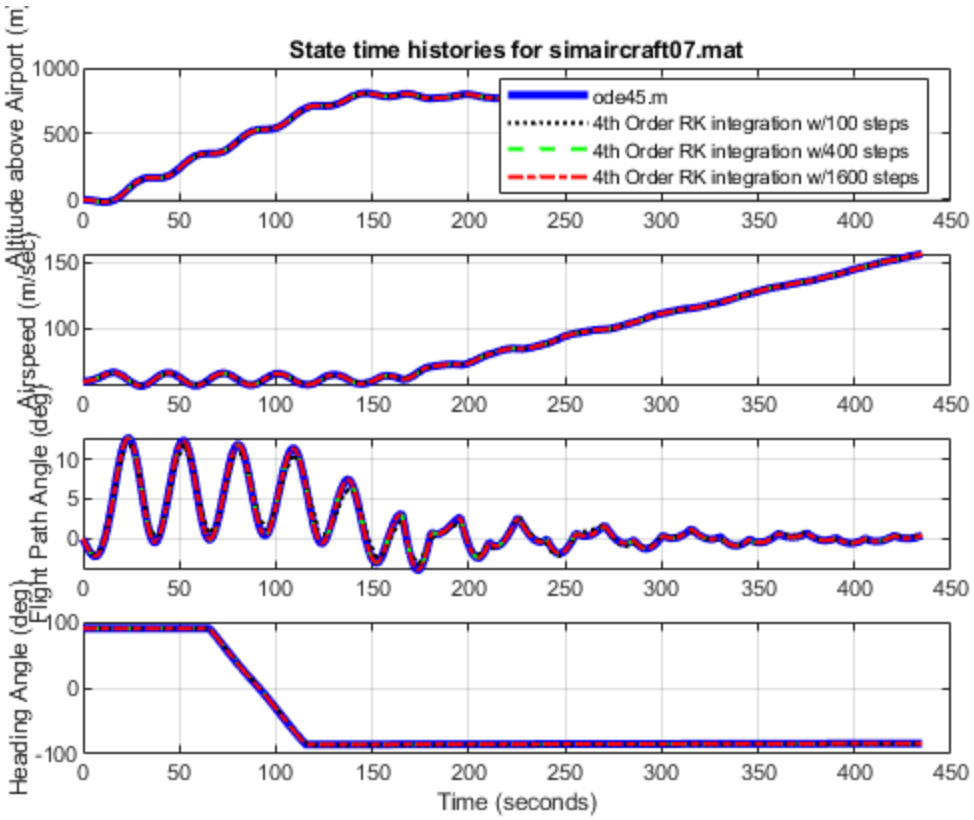
```
timetoode45 =   1.289840300000000


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
```

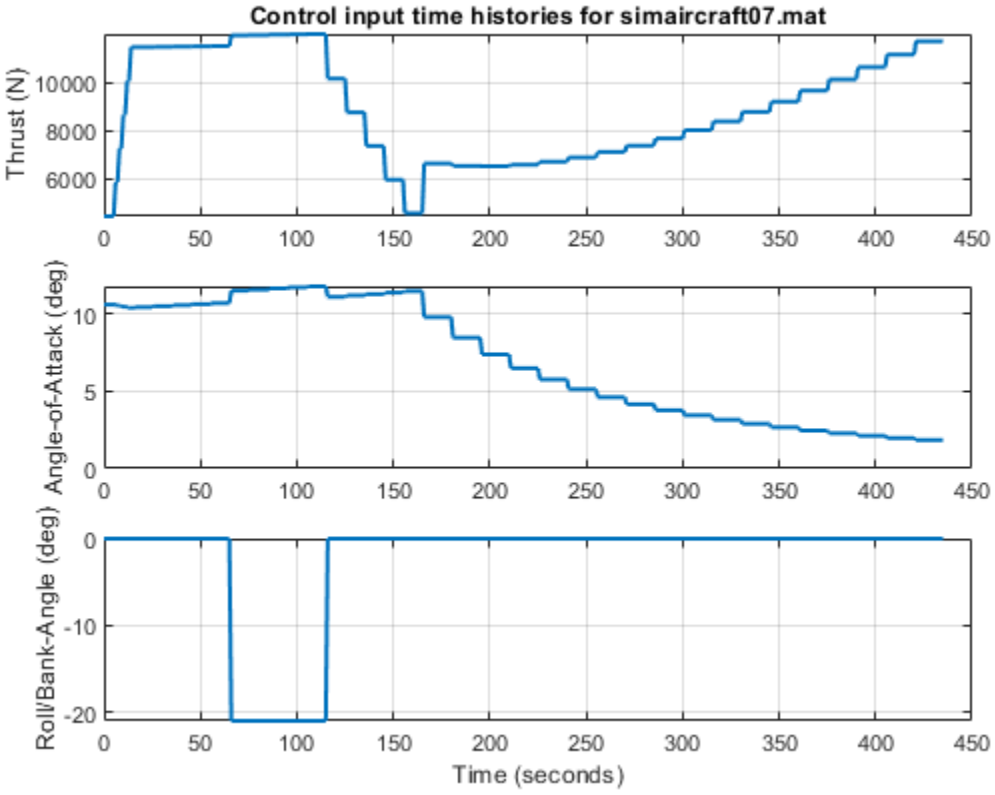
Q) How do errorxfinal_400 and errorxfinal_1600 for this run compare errorxfinal_2000 and errorxfinal_8000 for the trapezoidal integration run?
Ans) The errors are comparable. Same orders of magnitude.

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