
Problem 1

```
function f = ffunctaircraft03(t,x,m,S,CLalpha,CD0,oneoverpiARe,...
                              tinhist,Thist,alphahist,phihist)

%
% Copyright (c) 2019 Mark L. Psiaki. All rights reserved.
%
% This function implements a nonlinear dynamic model
% of a point-mass airplane flying over a flat Earth
% in an atmosphere whose air density decays exponentially
% with altitude. This function models the effects of
% time-varying thrust, angle-of-attack, and roll/bank-angle
% inputs.
%
% The dynamic model takes the form:
%
%     xdot = f(t,x)
%
% where xdot is the time rate of change of the 6-by-1
% state vector x and where the 6-by-1 vector function
% f(t,x) is the output of this Matlab function.
%
% Note: The aerodynamic model does not include stall.
%
% This particular function models the coordinate frame as
% being centered at the center of the runway of the Blacksburg,
% VA airport, which has coordinates latitude = 37.2076389 deg,
% longitude = -80.4078333 deg, altitude = 649.7 m. While it
% uses a flat-Earth (i.e., constant) gravity field, its
% constant gravity takes into account the Earth's J2
% oblateness effect at the coordinate system center and
% it subtracts off the centrifugal acceleration of the
% coordinate system center as caused by the Earth's rotation.
% This model also accounts for the additional centrifugal
% effects of the Earth's rotation as caused by rotation of
% the reference frame and any position offset from the
% original of the reference frame. In addition, it accounts
% for the Coriolis acceleration that is caused by the Earth's
% rotation.
%
%
% Inputs:
%
%     t                The time, in seconds, at which f is
%                      to be computed.
%
%     x                = [X;Y;Z;V;gamma;psi], the 6-by-1 state
%                      vector of this system. The first three
%                      elements give the Cartesian position
%                      vector of the aircraft's center of
%                      mass in local coordinates, in meters
%                      units, with X being the northward
```

```

% displacement from a reference position,
% Y being the eastward displacement from
% a reference position, and -Z being the
% altitude above the Blacksburg, VA
% airport (so that a positive value
% of x(3,1) indicates
% flight below the altitude of the
% Blacksburg airport. The fourth element
% of x is airspeed (and the Earth-relative
% speed assuming no wind) in meters/second.
% The fifth element is the flight path
% angle in radians. The sixth element is
% the heading angle in radians (0 is due
% north, +pi/2 radians is due east).
%
% m The aircraft mass in kg.
%
% S The wing area, in meters^2, which is
% the aerodynamic model's reference area.
%
% CLalpha The lift curve slope, dCL/dalpha, which
% is non-dimensional.
%
% CD0 The drag at zero lift, which is non-
% dimensional.
%
% oneoverpiARe = 1/(pi*AR*e), where AR is the non-
% dimensional aspect ratio of the wing
% and e is the Oswald efficiency factor.
% This composite input quantity is non-
% dimensional. It is the coefficient
% of CL^2 in the drag coefficient model.
%
% tinhist = [tin0;tin1;tin2;...;tinM], the
% (M+1)-by-1 vector of times, in seconds,
% at which the airplane control inputs in
% Thist, alphahist, and phihist are
% defined. This must be a monotonically
% increasing vector. Also, it is required
% that tinhist(1,1) = tin0 <= t <= tinM = ...
% tinhist(M+1,1). Otherwise, an error
% condition will occur.
%
% Thist = [T0;T1;T2;...;TM], the (M+1)-by-1 vector
% of thrust inputs that apply at the times
% in tinhist, in Newtons.
%
% alphahist = [alpha0;alpha1;alpha2;...;alphaM], the
% (M+1)-by-1 vector of angle-of-attack
% inputs that apply at the times in tinhist,
% in radians.
%
% phihist = [phi0;phi1;phi2;...;phiM], the (M+1)-by-1
% vector of roll/bank-angle inputs that apply

```

```

%                               at the times in tinhist, in radians.
%
%                               Note: a piecewise cubic hermite
%                               interpolating polynomial is used
%                               to interpolate between times in tinhist
%                               in order to compute the thrust, angle-of-
%                               attack, and roll/bank angle that apply at
%                               time t. These interpolations are computed
%                               using interp1.m.
%
% Outputs:
%
%   f                               = [Xdot;Ydot;Zdot;Vdot;gammadot;psidot],
%                               the 6-by-1 vector that contains the
%                               computed time derivatives of the state
%                               from the kinematics and dynamics models
%                               of the aircraft. f(1:3,1) is given
%                               in meters/second. f(4,1) is given in
%                               meters/second^2, and f(5:6,1) is given
%                               in radians/second.
%
%
% Compute the thrust, angle-of-attack, and roll/bank-angle
% inputs that apply at time t. It is more
% efficient to do all three piecewise cubic hermite
% interpolations simultaneously, as is done here.
%
alphaTphi = interp1(tinhist,[alphahist,Thist,phihist],t,'pchip');
alpha = alphaTphi(1,1);
T = alphaTphi(1,2);
phi = alphaTphi(1,3);
%
% Compute the air density using a decaying exponential
% model. This model is good to about 1500 m altitude
% (about 5000 ft). This model recognizes that - x(3,1) + 649.7
% is the aircraft altitude above sea level in meters.
%
rho_sealevel = 1.225; % kg/m^3
hscale = 10230.; % meters
haltitude = - x(3,1) + 649.7;
rho = rho_sealevel*exp(-haltitude/hscale); % kg/m^3
%
% Determine the airspeed.
%
V = x(4,1);
%
% Determine the dynamic pressure.
%
qbar = 0.5*rho*(V^2);
%
% Compute the lift and drag coefficients.
%
CL = CLalpha*alpha;

```

```

    CD = CD0 + (CL^2)*oneoverpiARE;
%
% Determine the lift and drag forces.
%
qbar_S = qbar*S;
L = qbar_S*CL;
D = qbar_S*CD;
%
% Set the flat-Earth gravitational acceleration at the
% Blacksburg airport minus the effects of centrifugal
% acceleration at the Blacksburg airport due to the
% Earth's rotation vector, i.e., at the coordinate frame
% origin.
%
g = 9.79721; % meters/second^2
%
% Compute the kinematics part of the model.
%
cospsi = cos(x(6,1));
sinpsi = sin(x(6,1));
cosgamma = cos(x(5,1));
singamma = sin(x(5,1));
V_cosgamma = V*cosgamma;
Xdot = V_cosgamma*cospsi;
Ydot = V_cosgamma*sinpsi;
Zdot = - V*singamma;
%
% Compute the additional centrifugal acceleration term due
% to the Earth's rotation rate vector and any offset of the
% aircraft position from the coordinate frame center.
% Also determine the Coriolis acceleration term due
% to the Earth's rotation rate vector and any velocity
% of the aircraft. Compute these two accelerations in
% North/East/Down coordinates.
%
omegaEarth_ned = [5.80780002889625e-05;0;-4.40958072123465e-05];
r_ned = x(1:3);
v_ned = [Xdot;Ydot;Zdot];
a_deltacentrifugal_ned =
cross(omegaEarth_ned,cross(omegaEarth_ned,r_ned));
a_coriolis_ned = 2.*cross(omegaEarth_ned,v_ned);
%
% Transform the net acceleration of the delta centrifugal and
% Coriolis terms from North/East/Down coordinates into
% navigation axes.
%
% Compute the gamma rotation matrix about the jhat axis.
%
R2atgamma = [cosgamma 0 -singamma;0 1 0;singamma 0 cosgamma];
%
% Compute the yaw rotation matrix about the khat axis.
%
R3atpsi = [cospsi sinpsi 0;-sinpsi cospsi 0;0 0 1];

```

```

%
% Complete the transformation from ned to nav axes.
%
    a_centrifugalscoriolis_ned = a_deltacentrifugal_ned +
    a_coriolis_ned;
    a_centrifugalscoriolis_nav =
    R2atgamma*R3atpsi*a_centrifugalscoriolis_ned;
%
% Compute the dynamics part of the model.
%
    cosphi = cos(phi);
    sinphi = sin(phi);
    cosalpha = cos(alpha);
    sinalpha = sin(alpha);
    oneoverm = 1/m;
    Vdot = oneoverm*(T*cosalpha - D) - g*singamma - ...
        a_centrifugalscoriolis_nav(1);
    T_sinalpha_plus_L = T*sinalpha + L;
    gammadot = (1/V)*((cosphi*oneoverm)*T_sinalpha_plus_L - g*cosgamma
+ ...
        a_centrifugalscoriolis_nav(3));
    psidot = (1/V*cosgamma)*((sinphi*oneoverm)*T_sinalpha_plus_L - ...
        a_centrifugalscoriolis_nav(2));
%
% Assemble the computed state time derivative elements
% into the output vector.
%
    f = [Xdot;Ydot;Zdot;Vdot;gammadot;psidot];

```

Not enough input arguments.

Error in ffunctaircraft03 (line 134)

```
alphaTphi = interp1(tinhist,[alphahist,Thist,phihist],t,'pchip');
```

Published with MATLAB® R2017a