6-DOF dynamic Simulation of Aircraft

Runge-Kutta 4 (RK4):

It is a numerical method for solving ordinary differential equations (ODEs) of the form:

dy/dt = f(t, y)   where, y is a function of t, and f is a known function.

RK4 is accurate up to terms of order h^5, where h is the step size.

The RK4 method works by using four estimates of the derivative of y at different points in the interval of integration to approximate the value of y at the end of the interval.

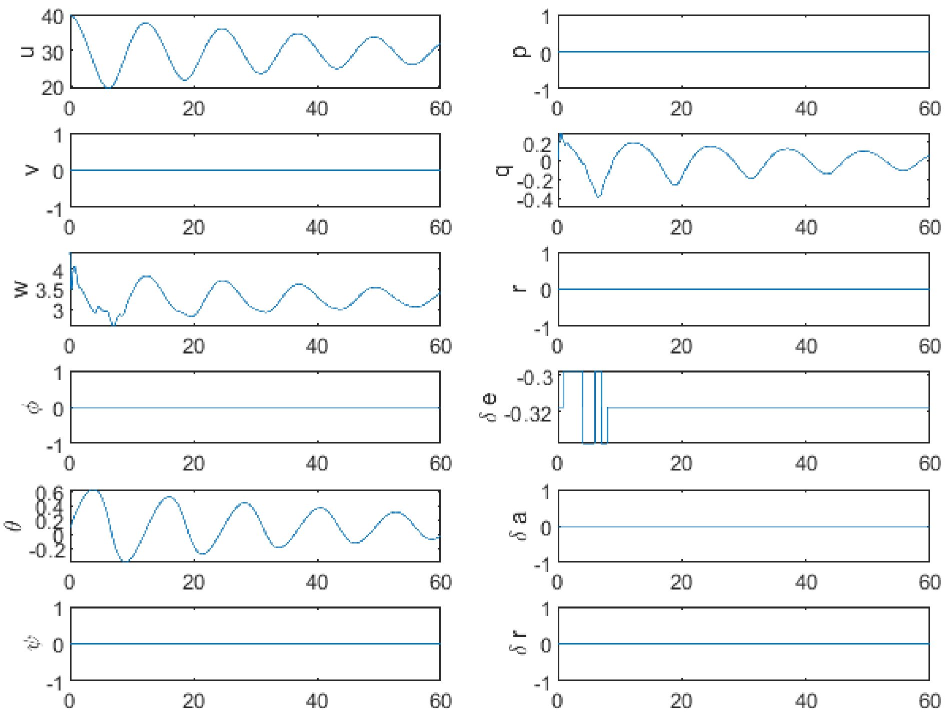
The algorithm can be summarized as follows:

✧ Given an initial value y0 at time t0, and a step size h, set y_n = y0 and tn = t0.
✧ Compute k1 = f(tn, y_n)
✧ Compute k2 = f(tn + h/2, y_n + h/2 * k1)
✧ Compute k3 = f(tn + h/2, y_n + h/2 * k2)
✧ Compute k4 = f(tn + h, y_n + h * k3)
✧ Compute y_n+1 = y_n + h/6 * (k1 + 2*k2 + 2*k3 + k4)
✧ Set tn+1 = tn + h.
✧ Repeat above until tn+1 = t_end, (where t_end is the final time.)

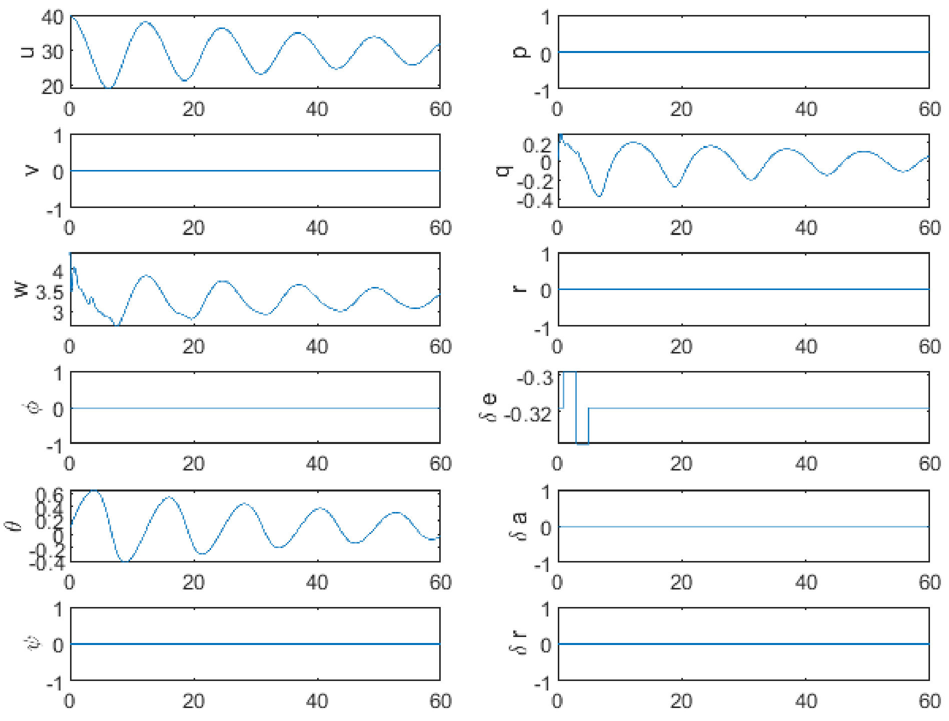
Q2.

(a) -

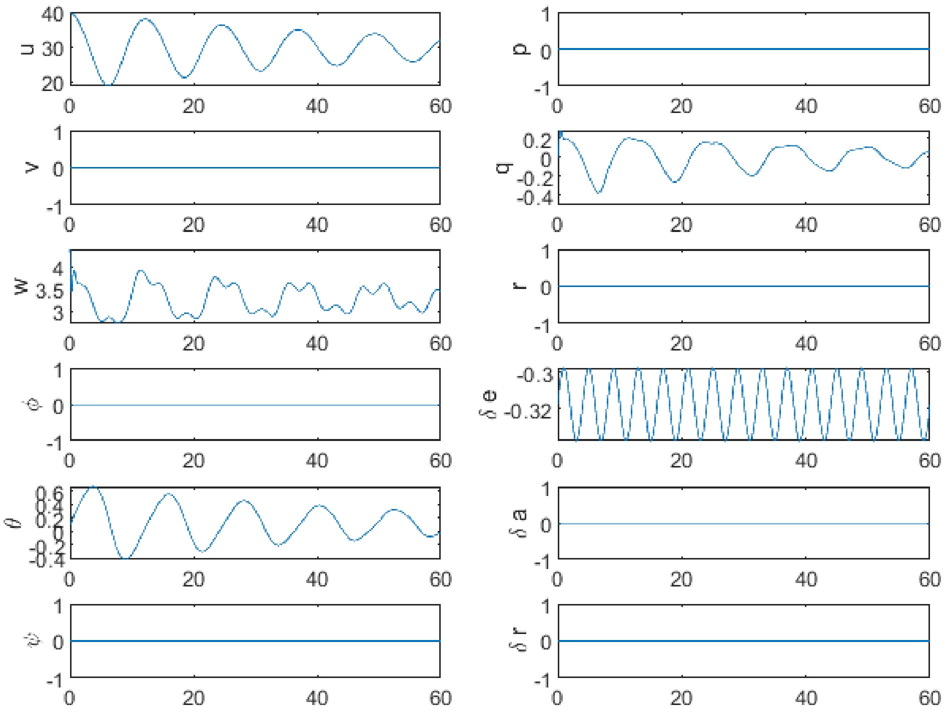
3-2-1-1 input to elevator:



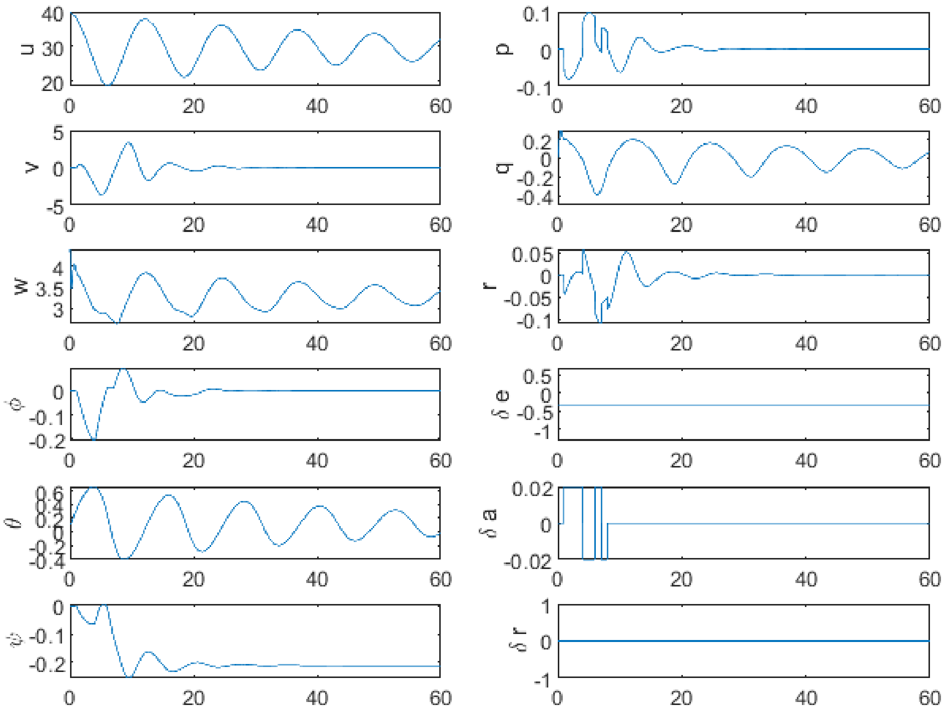
Doublet input to elevator:



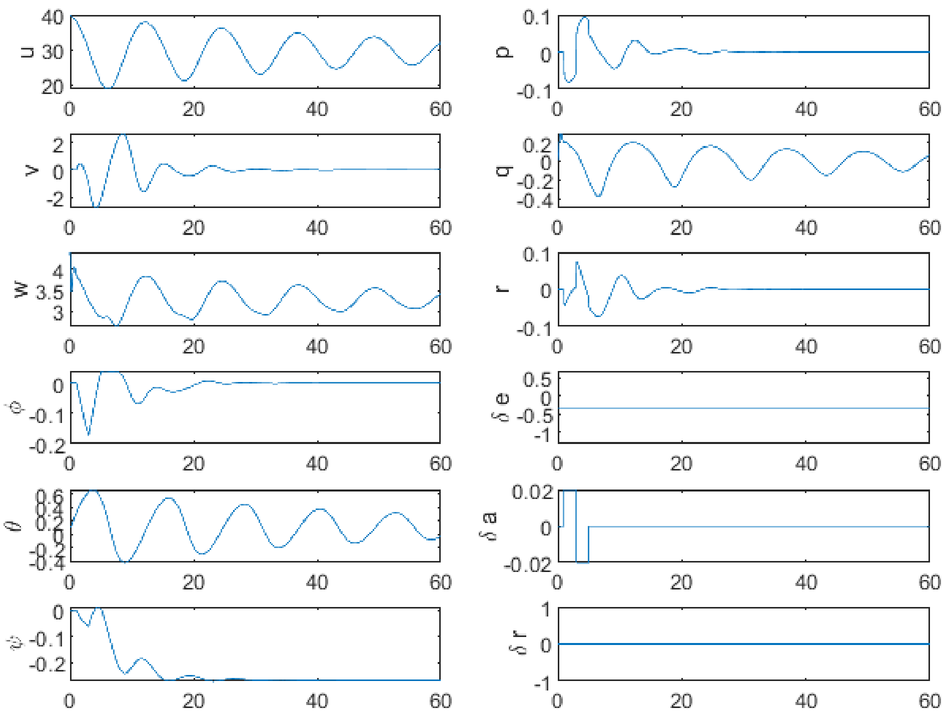
Sinusoidal input to elevator:



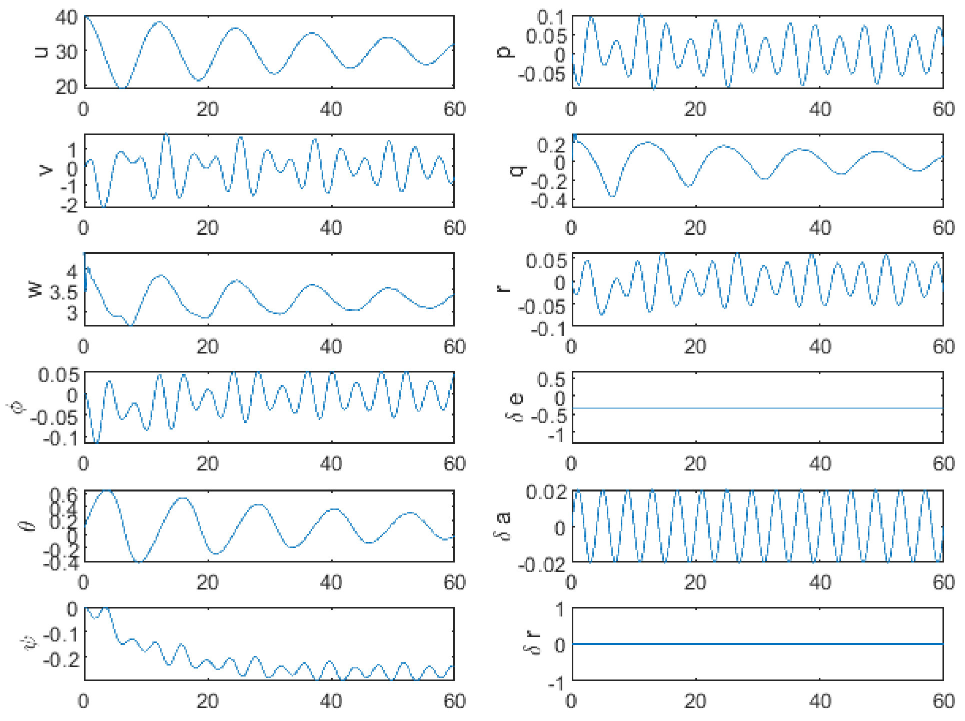
(b) -
3-2-1-1 input to aileron:



Doublet input to aileron:

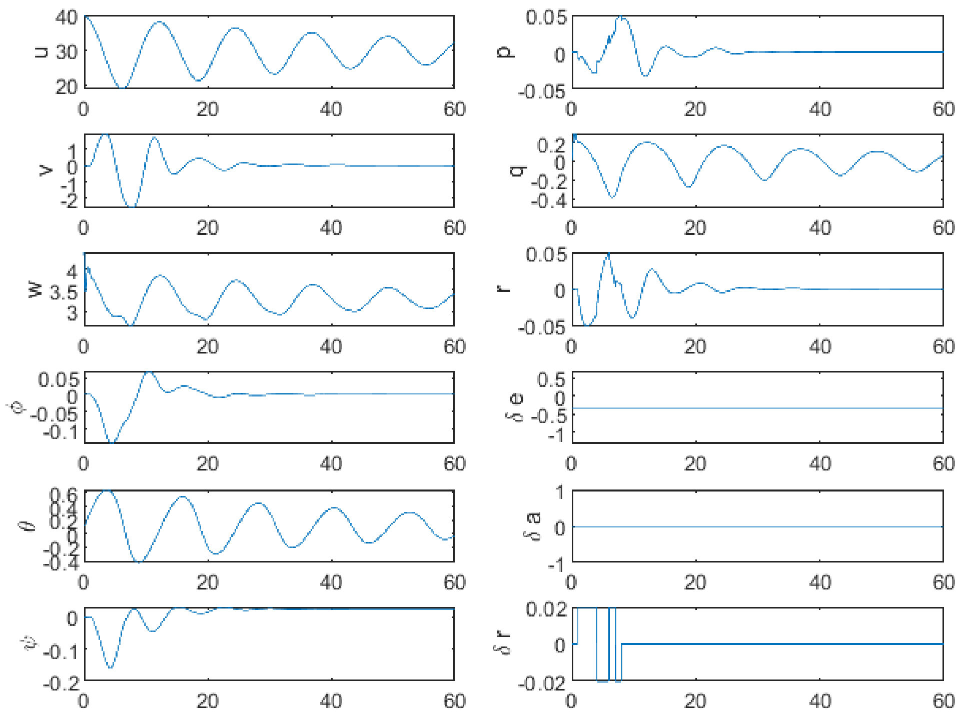


Sinusoidal input to aileron

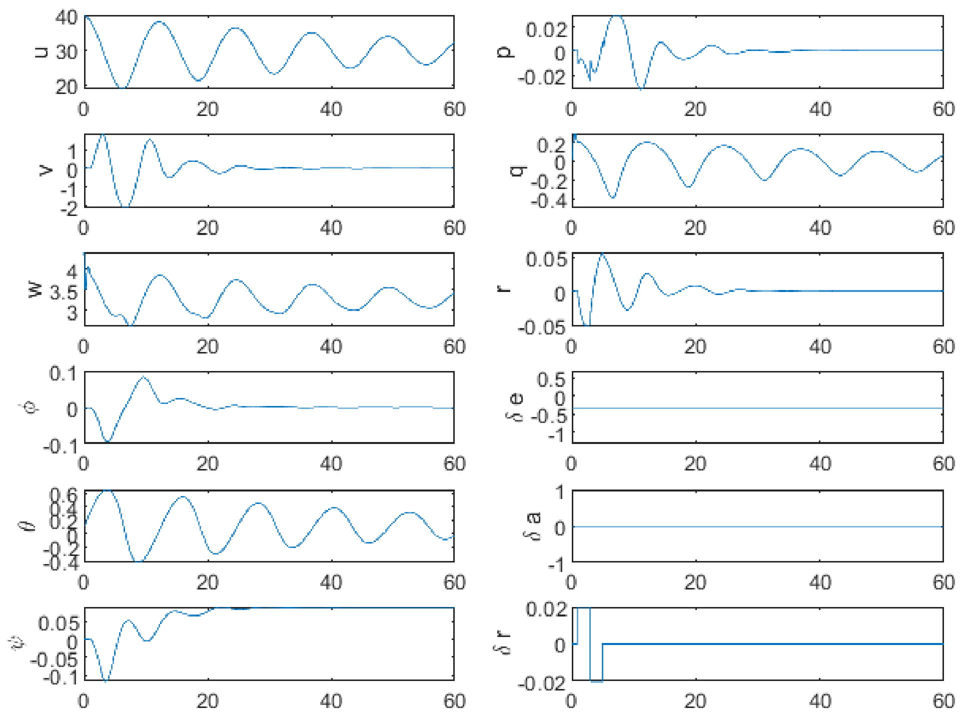


(c) -

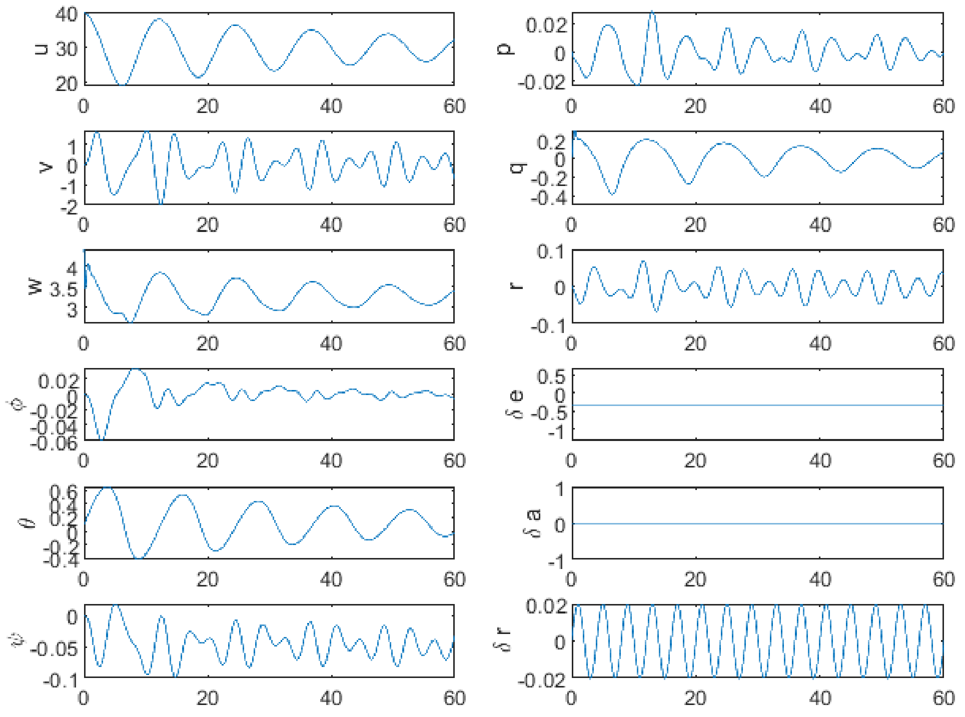
3-2-1-1 input to rudder



Doublet input to rudder



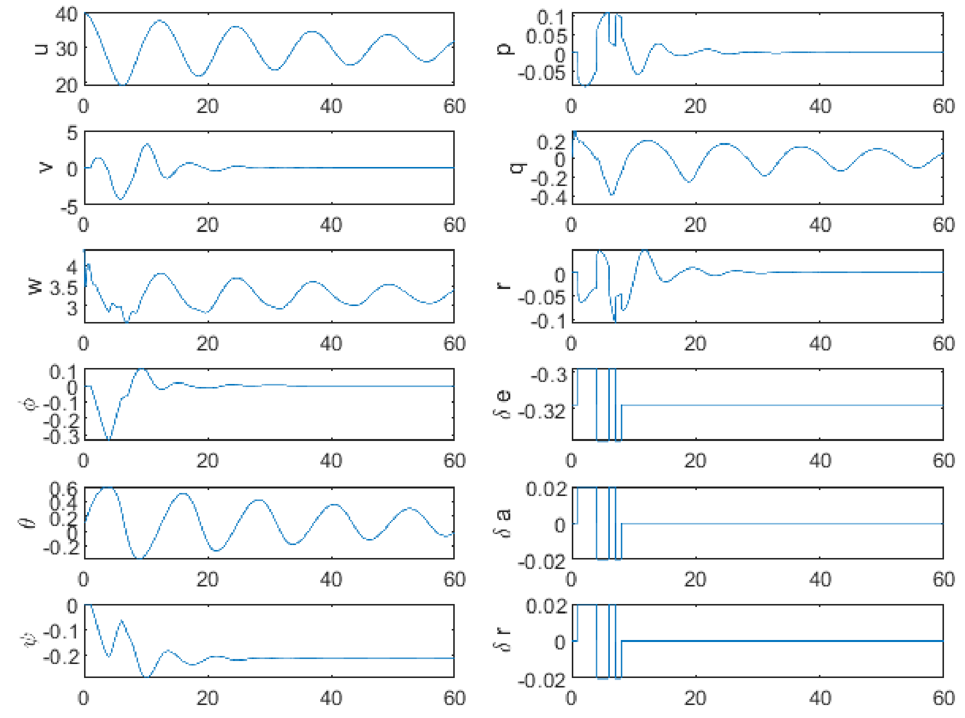
Sinusoidal input to rudder



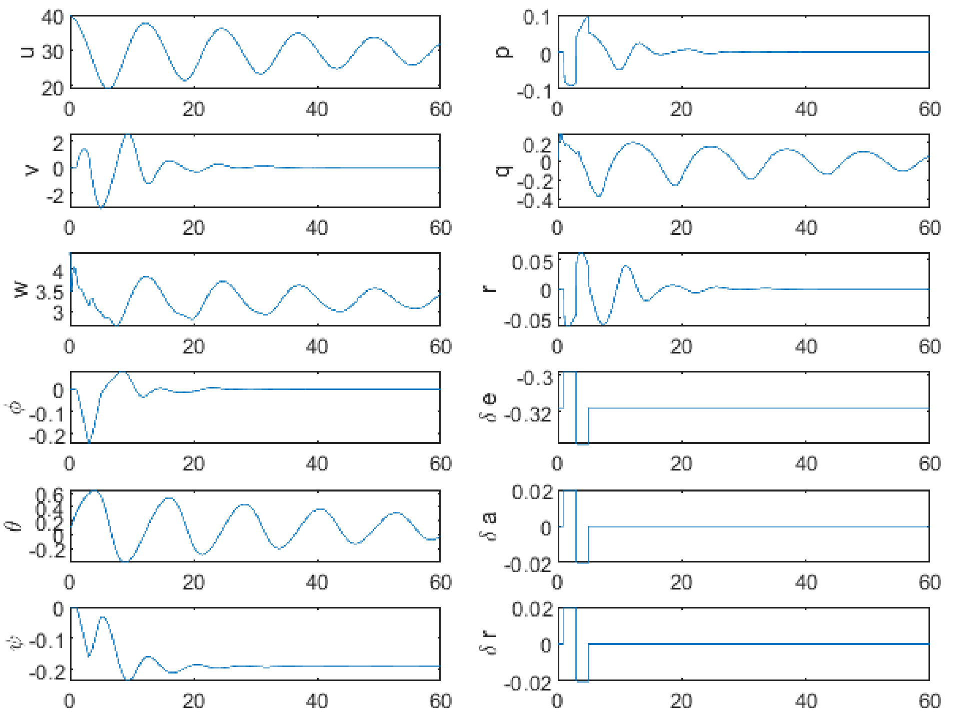
(d) - same as {a}

(e) -

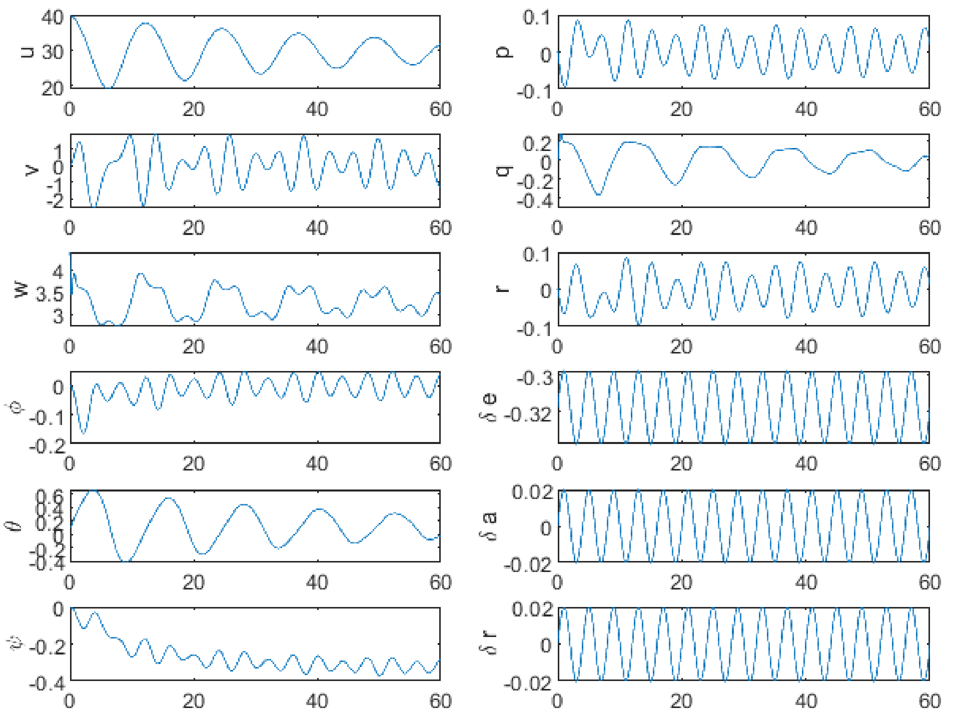
3-2-1-1 input to elevator, aileron, rudder simultaneously



(f) -
Doublet input to elevator, aileron, rudder simultaneously



(g) -
Sinusoidal input to elevator, aileron, rudder simultaneously



CODE of Q2:

%%% Main Code%%%

```
%% Initial Conditions
u0 = 39.7564; v0 = 0; w0 = 4.4076; % m/s
x0 = 0; y0 = 0; z0 = -3000; % m
p0 = 0; q0 = 0.001; r0 = 0; % rad/s
phi0 = 0; theta0 = 0.1104; psi0 = 0; % rad

states0 = [u0, v0, w0, x0, y0, z0, p0, q0, r0, phi0, theta0, psi0];

[t,y] = rk4(@six_dof_model, 60, 0.02, states0);

%% calculate control inputs
c = zeros(numel(t), 4);
for i=1:numel(t)
cc = cont(t(i), y(i));
c(i,:) = cc;
end

figure();
subplot(6,2,1);
plot(t,y(:,1));
ylabel('u');

subplot(6,2,3);
plot(t,y(:,2));
ylabel('v');

subplot(6,2,5);
plot(t,y(:,3));
ylabel('w');

subplot(6,2,2);
plot(t,y(:,7));
ylabel('p');

subplot(6,2,4);
plot(t,y(:,8));
ylabel('q');

subplot(6,2,6);
plot(t,y(:,9));
ylabel('r');

subplot(6,2,7);
plot(t,y(:,10));
ylabel('\phi');

subplot(6,2,9);
plot(t,y(:,11));
ylabel('\theta');

subplot(6,2,11);
plot(t,y(:,12));
ylabel('\psi');
```

```
subplot(6,2,8);
plot(t,c(:,1));
ylabel('\delta e');

subplot(6,2,10);
plot(t,c(:,2));
ylabel('\delta a');

subplot(6,2,12);
plot(t,c(:,3));
ylabel('\delta r');
```

%%% 6-DOF Model function %%%

```
function dstates = six_dof_model(t, states)

%% params
m = 750;g = 9.81; Ixx = 873; Iyy = 907; Izz = 1680; Ixz = 1144;S = 12.47; b
= 10.47; cb = 1.211;
c_d0 = 0.035; k = 0.045; cL_0 = 0.370; cL_a = 5.0; cL_q = 37.211; cL_de =
0.374;
cm_0 = 0.091; cm_a = -2.937; cm_q = -8.719; cm_de = -0.735;
cy_0 = 0; cy_b = -0.531; cy_p = -0.0571; cy_r = 0.4657; cy_dr = 0.1502;
cl_0 = 0; cl_b = -0.031; cl_p = -0.262; cl_r = -0.0541; cl_dr = 0.005;
cl_da = -0.153;
cn_0 = 0; cn_b = 0.01; cn_p = -0.007; cn_da = 0; cn_r = -0.067; cn_dr = -
0.047;
%%

statesCell = num2cell(states);
[u, v, w, x, y, z, p, q, r, phi, theta, psi] = statesCell{:};

%% Controlled Input function
ctrl = cont(t, states);

%% AeroF&M__Kinematics
dele = ctrl(1); dela = ctrl(2); delr = ctrl(3);

rho0 = 1.225; % kg/m^3
H0 = 10400; %m (scale height)

rho = rho0*exp(-z/H0);

V_inf = sqrt(u^2 + v^2 + w^2);
alpha = atan2(w,u);
beta = asin(v/V_inf);

ph = p*b/2/V_inf; qh = q*cb/2/V_inf; rh = r*b/2/V_inf;

cL = cL_0 + cL_a*alpha + cL_q*qh + cL_de*dele;
cD = c_d0 + k*cL^2;
cY = cy_0 + cy_b*beta + cy_dr*delr;
```

```
cM = cm_0 + cm_a*alpha + cm_q*qh + cm_de*dele;
cl = cl_0 + cl_b*beta + cl_p*ph + cl_r*rh + cl_da*dela + cl_dr*delr;
cN = cn_0 + cn_b*beta + cn_p*ph + cn_r*rh + cn_da*dela + cn_dr*delr;

cX = cL*sin(alpha) - cD*cos(alpha);
cZ = -(cL*cos(alpha) + cD*sin(alpha));

Q = 1/2*rho*V_inf^2;

Fx = Q*S*cX; Fy = Q*S*cY; Fz = Q*S*cZ;
l = Q*S*b*cl; M = Q*S*cb*cM; N = Q*S*b*cN;

T = ctrl(4);

du = Fx/m + T/m - q*w + r*v - g*sin(theta);
dv = Fy/m - r*u + p*w + g*cos(theta)*sin(phi);
dw = Fz/m - p*v + q*u + g*cos(theta)*cos(phi);

%% Rotation Matrix
Rpsi = [cos(psi) sin(psi) 0;
-sin(psi) cos(psi) 0;
0 0 1];

Rtheta = [cos(theta) 0 -sin(theta);
0 1 0;
sin(theta) 0 cos(theta)];

Rphi = [1 0 0;
0 cos(phi) sin(phi);
0 -sin(phi) cos(phi)];

b2i_mat = Rpsi'*Rtheta'*Rphi';

pqr2eul = [1 sin(phi)*tan(theta) cos(phi)*tan(theta);
0 cos(phi) -sin(phi);
0 sin(phi)/cos(theta) cos(phi)/cos(theta)];
%%
dxx = b2i_mat*[u; v; w];

dx = dxx(1); dy = dxx(2); dz = dxx(3);

dp = (l*Izz + N*Ixz - p*q*(Ixz*(Iyy-Ixx-Izz)) - q*r*(Izz*(Izz-
Iyy)+Ixz^2))/(Ixx*Izz - Ixz^2);
dq = (M + p*r*(Izz-Ixx) - Ixz*(p^2-r^2))/Iyy;
dr = (l*Ixz + N*Ixx + p*q*(Ixz^2+Ixx*(Ixx-Iyy)) + q*r*(Ixz*(Iyy-Ixx-
Izz)))/(Ixx*Izz - Ixz^2);

deul = pqr2eul*[p;q;r];
dphi = deul(1); dtheta = deul(2); dpsi = deul(3);

dstates = [du, dv, dw, dx, dy, dz, dp, dq, dr, dphi, dtheta, dpsi]';
end
```

%%% Control Input function %%%

```matlab
function ctrl = cont(t,states)

thrust = 586.5262; % N

%% trim condition
dele_trim = -0.3174; % rad
dela_trim = 0; % rad
delr_trim = 0; % rad

%% no control
% offset = 0; % rad

%% sinusoidal inputs
offset = 0.02*sin(pi/2*t); % rad

%% doublet inputs
% if t < 1
% offset = 0; % rad
% elseif t < 3
% offset = 0.02; % rad
% elseif t < 5
% offset = -0.02; % rad
% else
% offset = 0; % rad
% end

%% 3-2-1-1 inputs
% if t < 1
% offset = 0; % rad
% elseif t < 4
% offset = 0.02; % rad
% elseif t < 6
% offset = -0.02; % rad
% elseif t < 7
% offset = 0.02; % rad
% elseif t < 8
% offset = -0.02; % rad
% else
% offset = 0; % rad
% end


% dele = dele_trim;
dele = dele_trim + offset;
% dela = dela_trim;
dela = dela_trim + offset;
% delr = delr_trim;
delr = delr_trim + offset;

ctrl = [dele, dela, delr, thrust];
end
```

%%% RK4 function %%%

```
function [t,y] = rk4(dydt, tf, dt, y0)
% reshape y0 to column vector
y0 = reshape(y0, [], 1);

n = tf/dt;
d = size(y0,1);

t = zeros(n,1);
y = zeros(n,d);

tn = 0;
yn = y0;

t(1,:) = tn;
y(1,:) = yn';

for i=2:n+1
C1 = dydt(tn, yn);
C2 = dydt((tn + dt*0.5), (yn + dt*C1*0.5));
C3 = dydt((tn + dt*0.5), yn + (dt*C2*0.5));
C4 = dydt(tn + dt, yn + dt*C3);

yn = yn + (C1 + 2*C2 + 2*C3 + C4)*dt/6;
tn = tn + dt;

t(i) = tn;
y(i,:) = yn';
end
end
```
