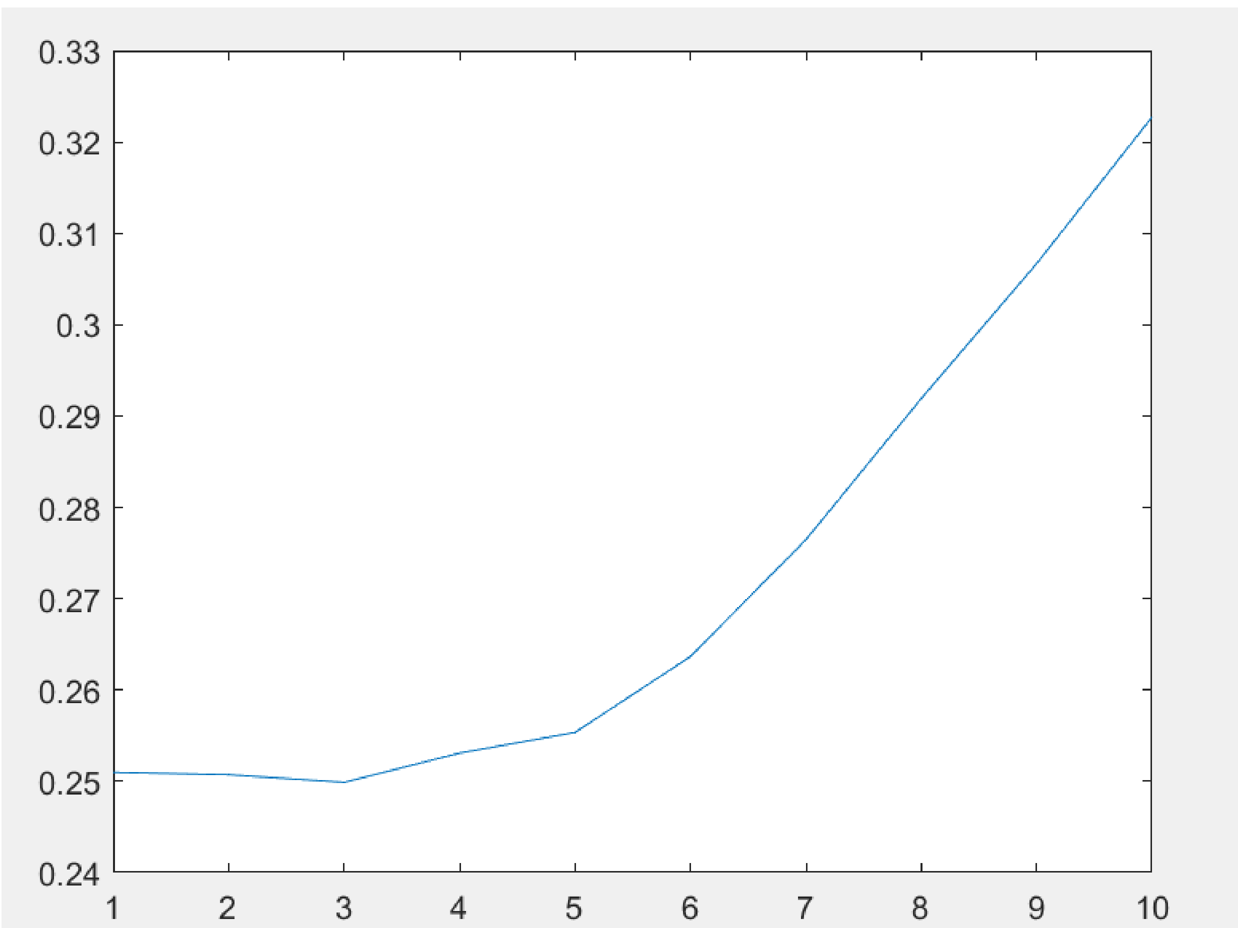
Mid Semester Exam

Results:

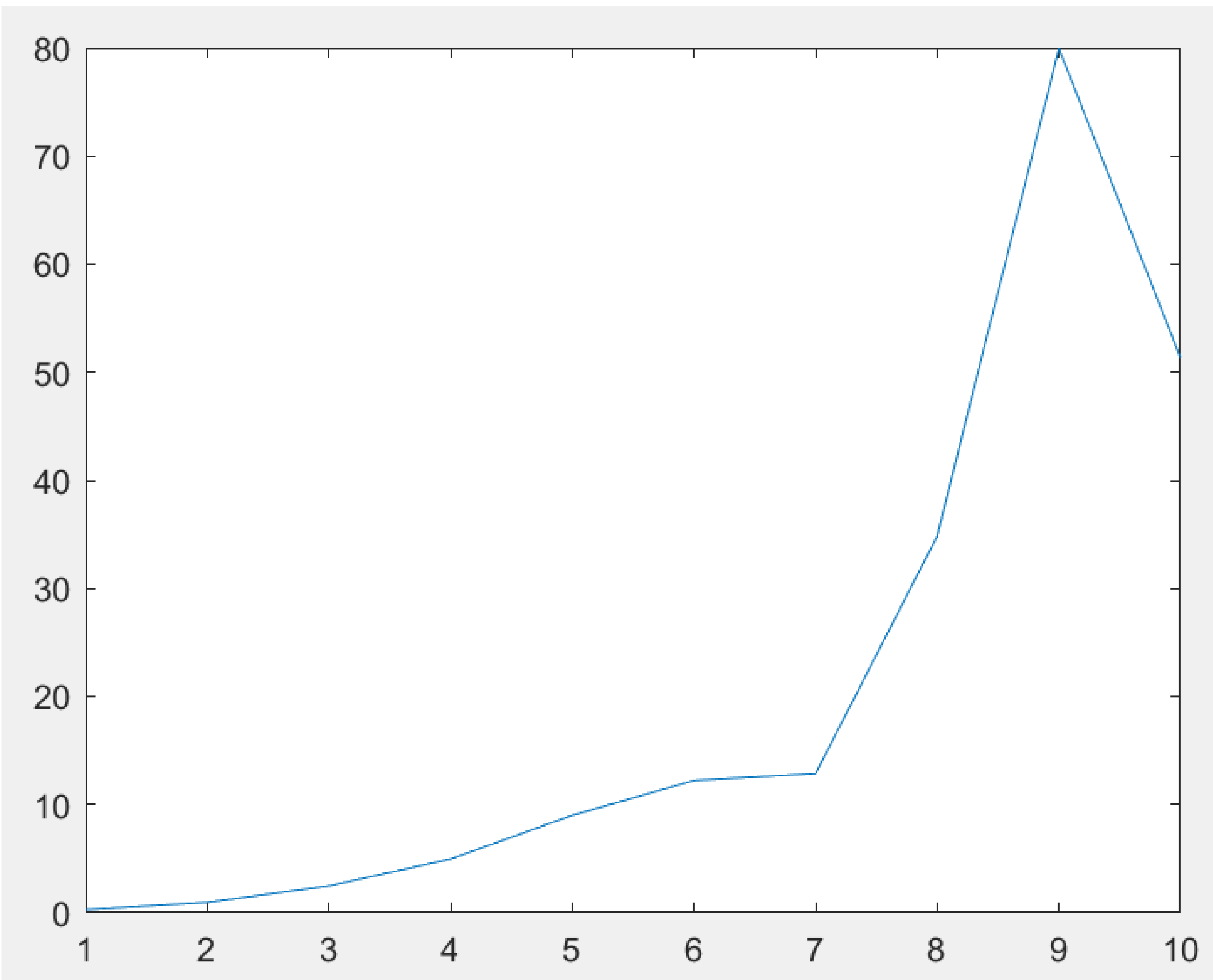
Question 1:

Both the plots have similar increases in trend. The increase in the percentage of lying inside triangles is because as the error increases the size of 3 corners of triangles increases. That is the reason why the percentage is increasing with time but the derived point is still far from the actual point as standard deviation increases.

Percentage vs standard deviation

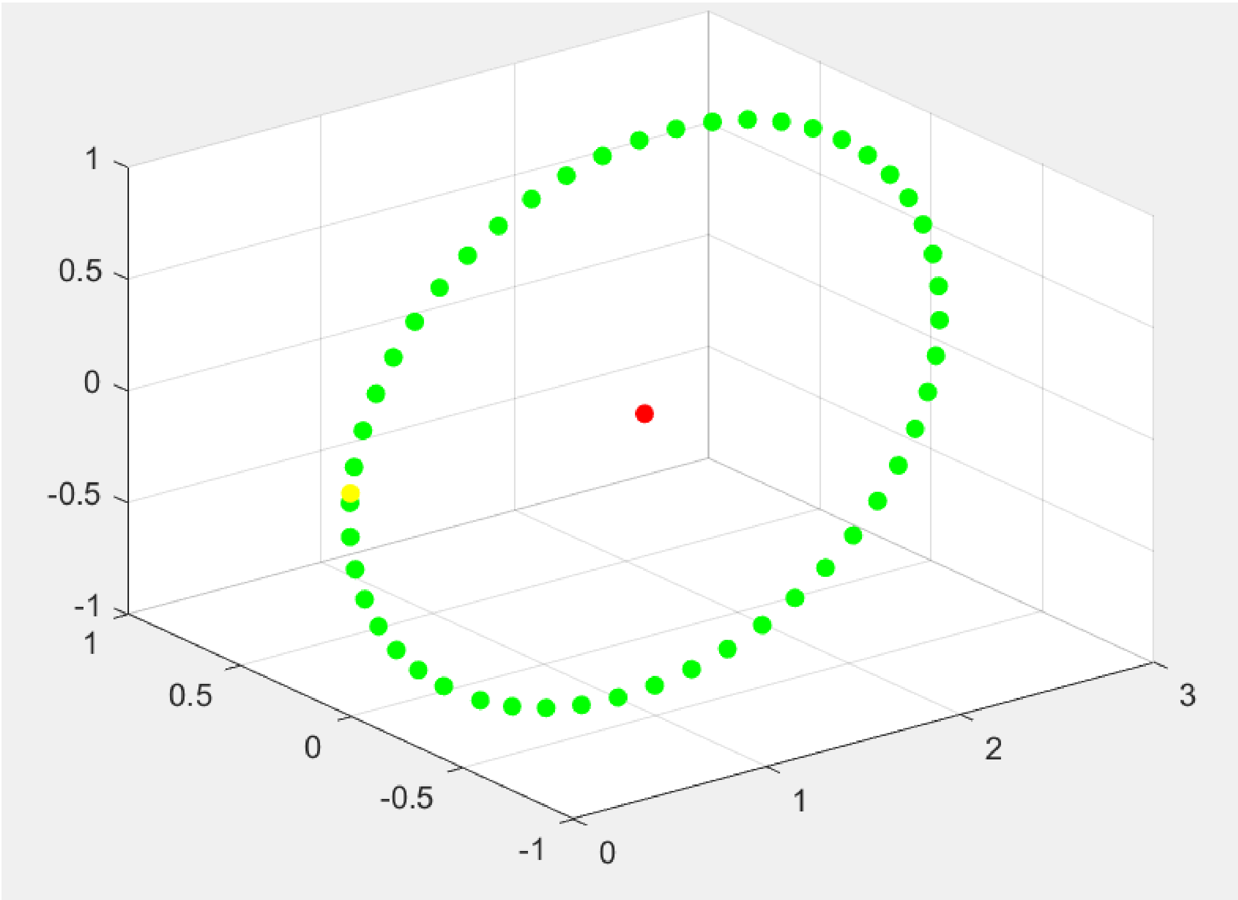


average distance of P to $\hat{c}$ versus σ.

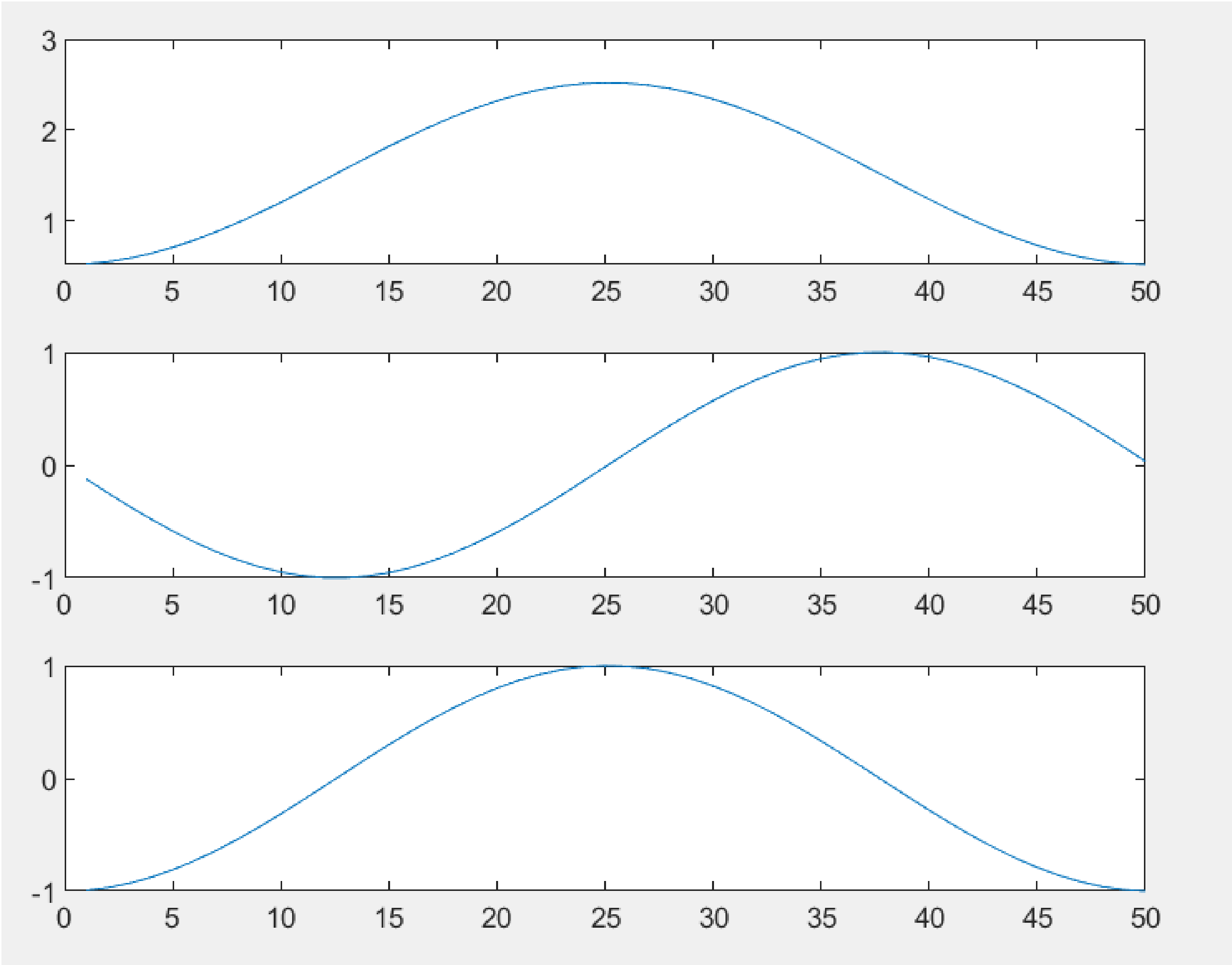


Question 2:

Obtained solutions are shown below in green

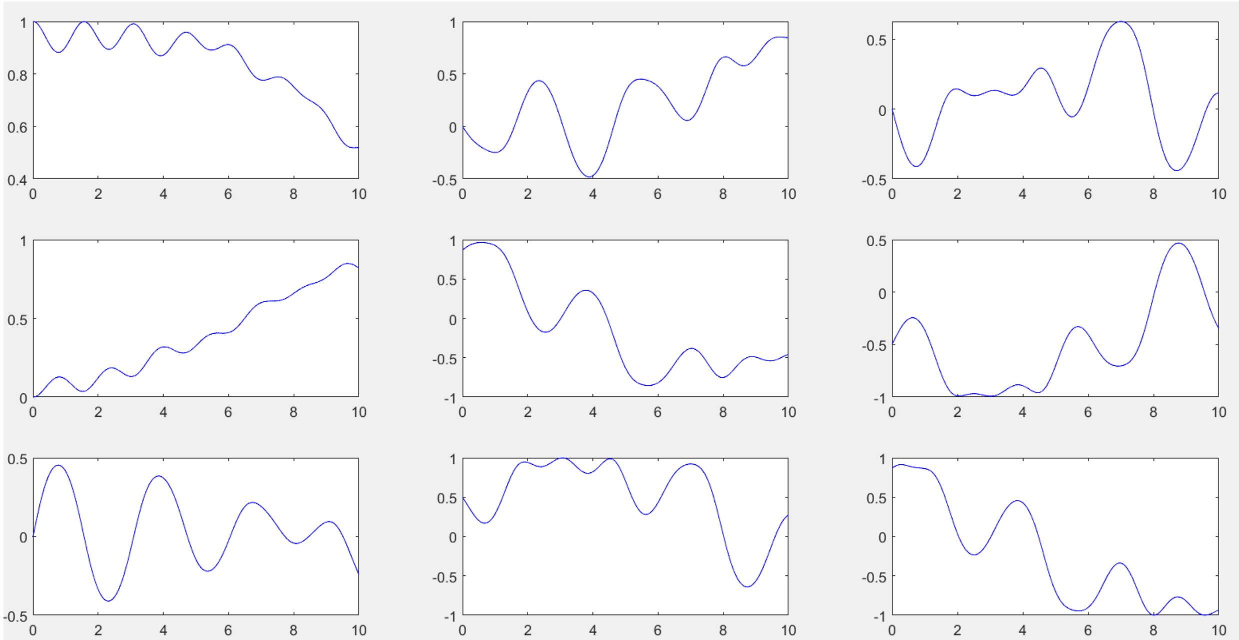


x,y,z plots shown below:

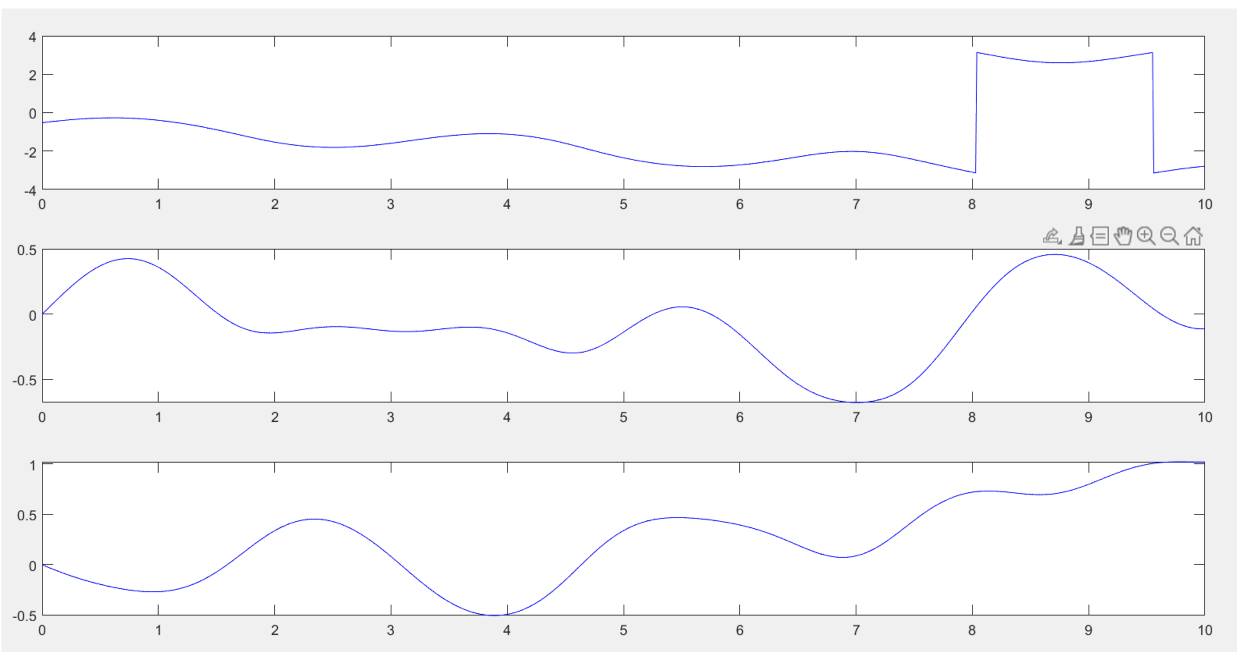


Question 3:

Coefficients of matrix vs time

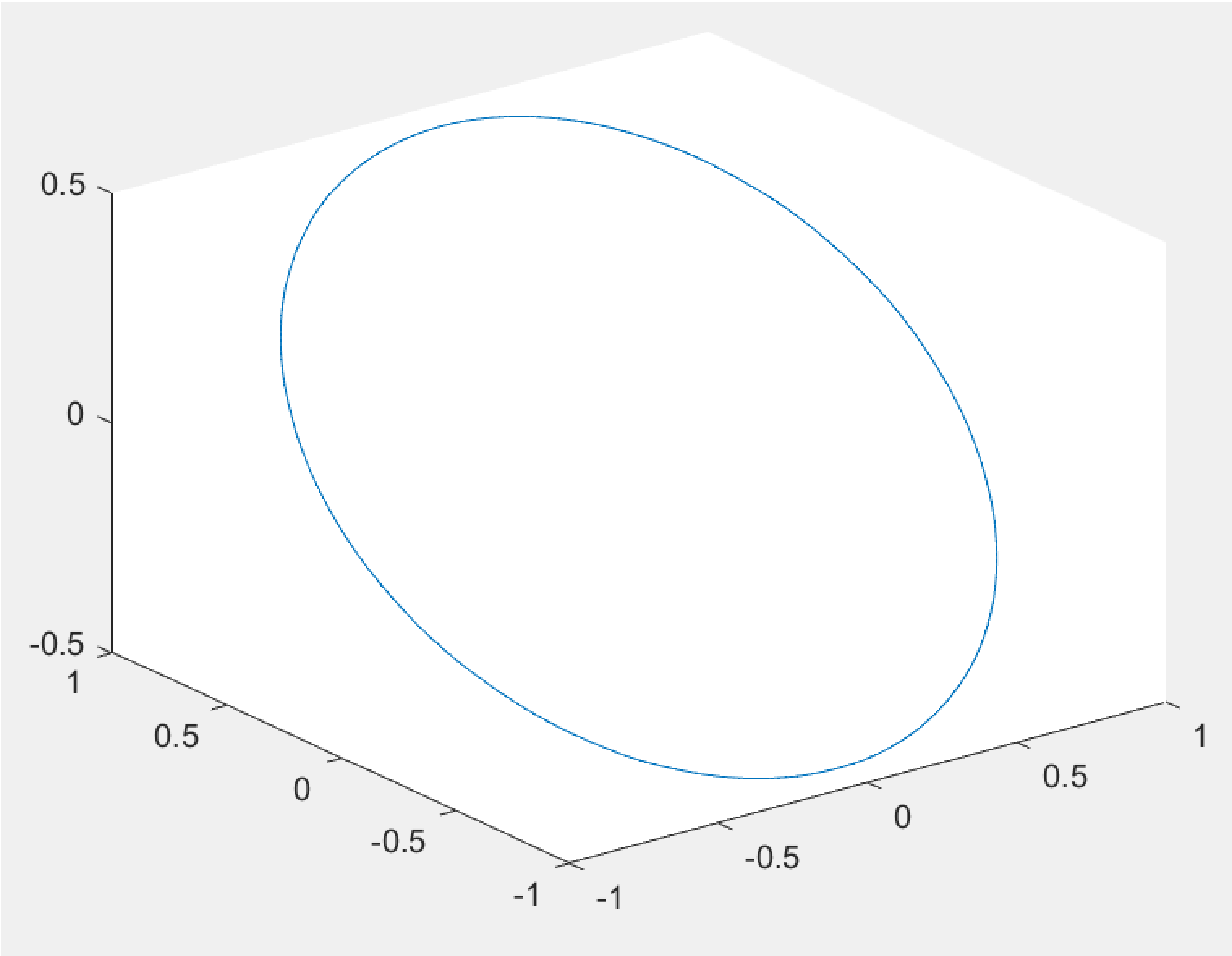


Psi theta phi shown below:

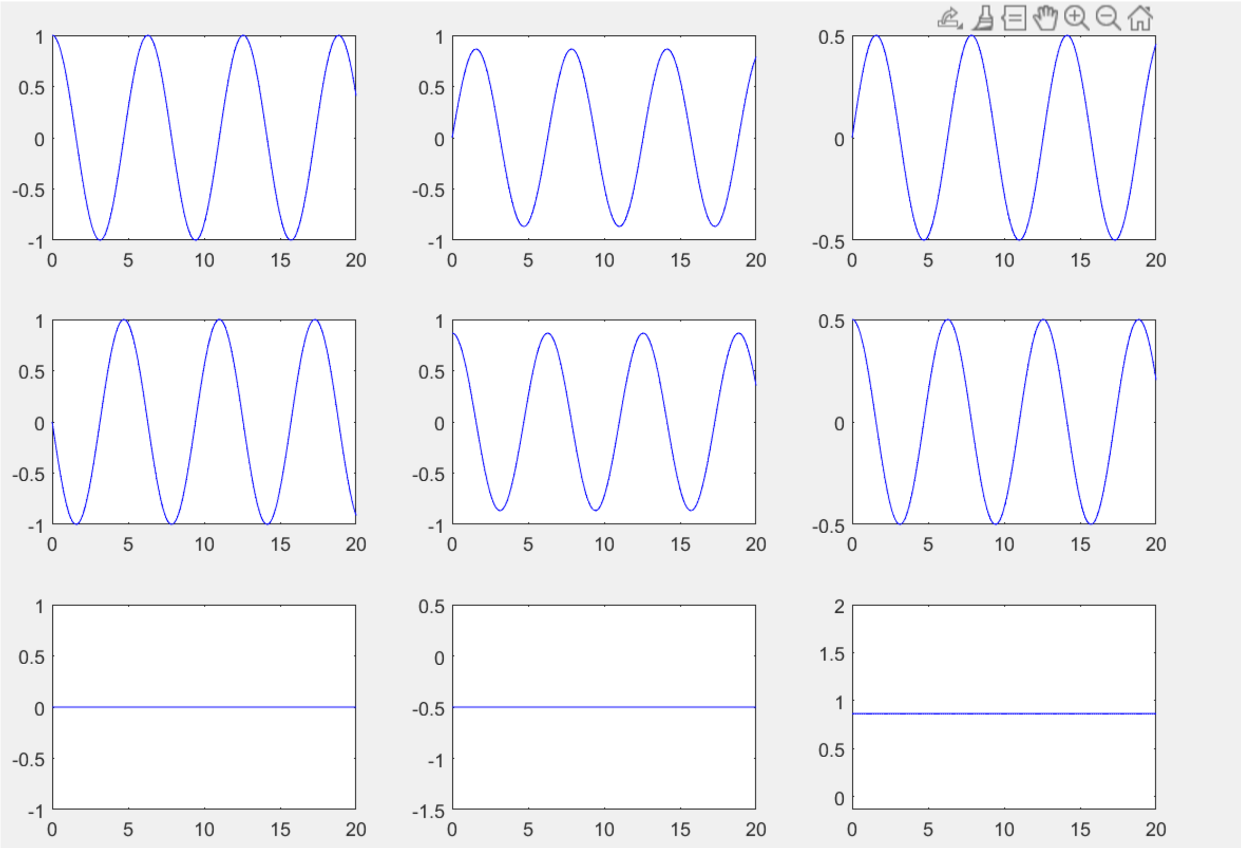


Question 4:

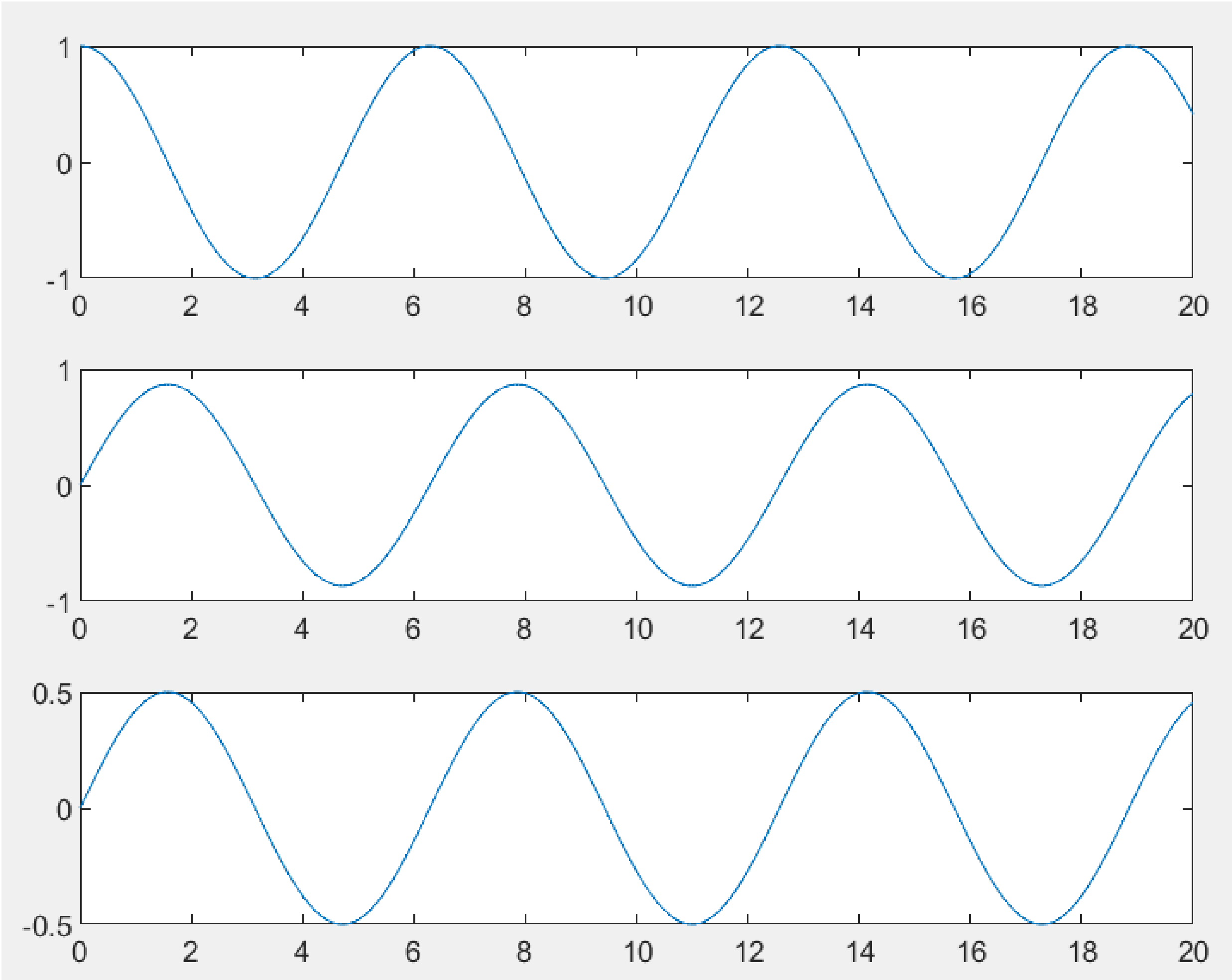
Trajectory obtained:



O_B/E vs time:



x,y,z wrt time:



Codes:

Question 1:

```
%% Defining parameters
clear all;
clc ;
p1 = [0,0,0];
p2 = [4,2,0];
p3 = [1,4,0];
p_ori = [2.5,2,0];
%%
```

```
theta1 = get_thetas(p_ori,p1);
theta2 = get_thetas(p_ori,p2);
theta3 = get_thetas(p_ori,p3);
%%
runs = 100000;
avgs = [];
percents = [];
for jj =1:10
    sigma = jj*0.1;
    count = 0;
    solns = [];
    for i = 1:runs
        noise1 = sigma*randn();
        noise2 = sigma*randn();
        noise3 = sigma*randn();
        [p_sol, p_sol1, p_sol2, p_sol3] = get_pos(p1,theta1+noise1, p2, theta2+noise2, p3,
theta3+noise3);
        flag = isInside(p_sol1, p_sol2, p_sol3,p_ori);
        err = norm(p_sol-p_ori(1:2)');
        solns = [solns,err];
        count = count + flag;
    end
    percents = [percents,count/runs];
    avgs = [avgs,mean(solns)];
end
%%
plot(percents);
fig = figure();
plot(avgs);


function p = pos_fix(p1,p2, T1, T2)

A = [T1, -1; T2, -1];

B = [(T1*p1(1))-p1(2);(T2*p2(1))-p2(2)];

p = inv(A)*B;

end

function plot_sol(p1,p2,p3,p_sol)
fig = figure();
scatter(p1(1),p1(2),'g','filled', 'o');
```

```
hold on;
scatter(p2(1),p2(2),'g','filled', 'o');
scatter(p3(1),p3(2),'g','filled', 'o');
scatter(p_sol(1),p_sol(2),'r','filled', 'o');
axis equal
hold off

end

function flag = isInside(P1,P2,P3,P)
A = area_cal (P1,P2,P3);
A1 = area_cal (P,P2,P3);
A2 = area_cal (P1,P,P3);
A3 = area_cal (P1,P2,P);
flag = 0;
if(abs(A - (A1+A2+A3))<1e-6)
   flag = 1;
end

function theta = get_thetas(P,P1)

T = (P(2)-P1(2))/(P(1)-P1(1));
theta = (pi/2) - atan(T);
end

function [p_mean, p_sol1, p_sol2, p_sol3] = get_pos(p1,theta1, p2, theta2, p3, theta3)

T1 = tan((pi/2)-theta1);
T2 = tan((pi/2)-theta2);
T3 = tan((pi/2)-theta3);

p_sol1 = pos_fix(p1,p2,T1,T2);
p_sol2 = pos_fix(p2,p3,T2,T3);
p_sol3 = pos_fix(p3,p1,T3,T1);

p_mean = (p_sol3 + p_sol2 + p_sol1)/3;

end

function Area = area_cal(P1, P2, P3)
Area = abs(((P1(1) * (P2(2) - P3(2))) + (P2(1) * (P3(2) - P1(2)))+ (P3(1) * (P1(2) - P2(2))))/ 2.0);
End
```

Question 2:

```
load 'AE584_Midterm_P2.mat'
p1 = [1.52;0;0];
p2 = [0;0;0];
p0 = [0.52;0;-1];
soln = [];
p = p0;
for i=1:50
    theta = subAngL1L2(i);
    psi1= bearingL2St1(i);
    psi2 = bearingL2St2(i);

    cost = @(p) the_cost(p, p1, p2, theta, psi1, psi2);
    options = optimoptions(@fminunc,'OptimalityTolerance',1e-2);
    [sol1,~] = fminunc(cost,p);
    p = sol1;

    soln = [soln;mat2cell(sol1,3,1)];
end

px = [];
py = [];
pz = [];
fig = figure();
scatter3(p1(1), p1(2), p1(3),'r','filled', 'o');
hold on;
scatter3(p2(1), p2(2), p2(3),'yellow','filled', 'o');
for i=1:length(soln)
    pp = cell2mat(soln(i));
    scatter3(pp(1), pp(2), pp(3),'g','filled', 'o');
    px = [px;pp(1)];
    py = [py;pp(2)];
    pz = [pz;pp(3)];
end
hold off;


fig = figure();
subplot(3,1,1);
plot(px);
subplot(3,1,2);
plot(py);
subplot(3,1,3);
plot(pz);
```

```
function cost = the_cost(p, p1, p2, theta, psi1, psi2)

r_hat1 = [0;1;0];
r_hat2 = [0;0;1];

l1 = sqrt((p-p1)'*(p-p1));
l2 = sqrt((p-p2)'*(p-p2));

l = sqrt((p1-p2)'*(p1-p2));

part1 = (p-p1)'*(p2-p1);
part2 = l1*l2*cos((theta));
part3 = l1*l1;

E2 = part3 - part2 - part1;

part4 = (p2-p)'*r_hat1;
part5 = l2*cos((psi1));

E3 = part5 - part4;

part6 = (p2-p)'*r_hat2;
part7 = l2*cos((psi2));

E1 = part6 - part7;


flag =0;
if(l1<=0.01 || l2<=0.01)
    flag = 1;
end
cost = 100*flag+ E2^2 + E3^2 + E1^2 ;

end
```

Question 3:

```
%% Problem 3
%%
%% Initialise Parameters
t0=0; tf=10;
t_2 = linspace(0,10,1000);
```

```
phi_e = 0;
theta_e = pi/6;
psi_e = 0;
O = zeros(3,3);
O_b_i = [];
O_e_i = [];
O_b_e = [];
euler_angles = [];
dt = 0.01;

phi_dot =  sin(0.05*t_2);
theta_dot =  0.3*cos(0.01*t_2);
psi_dot =  0.5*sin(0.01*t_2);

orientation_matrix_initial = [1, 0, 0; 0, 1, 0; 0, 0, 1];



%% Part from I to B - Orientation Matrix using Poission Integral

options = odeset('RelTol',1e-12,'AbsTol',1e-12);
sol = ode45(@(t,O_linear) poisson_integral(t,O_linear),[t0, tf],orientation_matrix_initial,
options); %% Poission Integral using ode45
O_2_linear = deval(sol,t_2); % Obtaining Function for all reqired time of evaluation
for i=1:length(t_2)
   O(1,1) = O_2_linear(1,i);
   O(1,2) = O_2_linear(2,i);
   O(1,3) = O_2_linear(3,i);

   O(2,1) = O_2_linear(4,i);
   O(2,2) = O_2_linear(5,i);
   O(2,3) = O_2_linear(6,i);

   O(3,1) = O_2_linear(7,i);
   O(3,2) = O_2_linear(8,i);
   O(3,3) = O_2_linear(9,i);
   O_b_i = [O_b_i;(O)];
end
```

```
%% Part I to E
rot_an = [];
for i=1:length(t_2)
    matrix = orientation_from_angles([phi_e,theta_e, psi_e]);
    O_e_i = [O_e_i;matrix];
%     phi_e = phi_e+(phi_dot(i)*dt);
%     theta_e = theta_e+(theta_dot(i)*dt);
%     psi_e = psi_e+(psi_dot(i)*dt);
%
    phi_e = 20*(1-cos(0.05*t_2(i)));
    theta_e = (pi/6)+30*(sin(0.01*t_2(i)));
    psi_e = 50*(1-cos(0.01*t_2(i)));
    rot_an = [rot_an;[psi_e, theta_e, psi_e]];

end

%% Part B to E

for i=1:length(t_2)
    o_e_i = O_e_i(((i-1)*3)+1:i*3,1:3);
    o_b_i = O_b_i(((i-1)*3)+1:i*3,1:3);
    R = o_b_i*(inv(o_e_i));
    O_b_e = [O_b_e; R];
    eulers = euler_from_rotation(R);
    euler_angles = [euler_angles; eulers];
end
%% Plotting

fig4 = figure();
fig4.Position = [10 10 900 600];
ax4 = axes(fig4);
for i = 1:3
    for j = 1:3
        subplot(3,3,(i-1)*3+j);
        O_b = [];
        for ijk = 1:length(t_2)
            x = O_b_e(3*(ijk-1)+i,j);
            O_b = [O_b;x];
        end
        plot(t_2, O_b, 'b');
```

```
        ax = gca;
        ax.TitleFontSizeMultiplier = 0.5;
    end
end

fig5 = figure();
fig5.Position = [10 10 900 600];
ax5 = axes(fig5);
for i = 1:3
    subplot(3,1,i);
    O_b = [];
    for ijk = 1:length(t_2)
        x = euler_angles(ijk,i);
        O_b = [O_b;x];
    end
    plot(t_2, O_b, 'b');
    ax = gca;
    ax.TitleFontSizeMultiplier = 0.5;
end


%% Poisson's Integral Implementation
function O_dot_linear = poisson_integral(t,O_linear)
O = [O_linear(1), O_linear(2), O_linear(3);
     O_linear(4), O_linear(5), O_linear(6);
     O_linear(7), O_linear(8), O_linear(9);];
O_dot = zeros(3,3);
O_dot_linear = zeros(9,1);
omega = [cos(2*t),cos(2*t),0.025*t];
omega_cross = [0, -omega(3), omega(2);
               omega(3), 0, -omega(1);
               -omega(2), omega(1), 0];
omega_cross = omega_cross*-1;
%% O_dot elements

O_dot(1,1) =  omega_cross(1,1)*O(1,1) + omega_cross(1,2)*O(2,1) +
omega_cross(1,3)*O(3,1);
O_dot(1,2) =  omega_cross(1,1)*O(1,2) + omega_cross(1,2)*O(2,2) +
omega_cross(1,3)*O(3,2);
```

```
O_dot(1,3) =  omega_cross(1,1)*O(1,3) + omega_cross(1,2)*O(2,3) +
omega_cross(1,3)*O(3,3);

O_dot(2,1) =  omega_cross(2,1)*O(1,1) + omega_cross(2,2)*O(2,1) +
omega_cross(2,3)*O(3,1);
O_dot(2,2) =  omega_cross(2,1)*O(1,2) + omega_cross(2,2)*O(2,2) +
omega_cross(2,3)*O(3,2);
O_dot(2,3) =  omega_cross(2,1)*O(1,3) + omega_cross(2,2)*O(2,3) +
omega_cross(2,3)*O(3,3);

O_dot(3,1) =  omega_cross(3,1)*O(1,1) + omega_cross(3,2)*O(2,1) +
omega_cross(3,3)*O(3,1);
O_dot(3,2) =  omega_cross(3,1)*O(1,2) + omega_cross(3,2)*O(2,2) +
omega_cross(3,3)*O(3,2);
O_dot(3,3) =  omega_cross(3,1)*O(1,3) + omega_cross(3,2)*O(2,3) +
omega_cross(3,3)*O(3,3);
%%
O_dot_linear(1) = O_dot(1,1);
O_dot_linear(2) = O_dot(1,2);
O_dot_linear(3) = O_dot(1,3);

O_dot_linear(4) = O_dot(2,1);
O_dot_linear(5) = O_dot(2,2);
O_dot_linear(6) = O_dot(2,3);

O_dot_linear(7) = O_dot(3,1);
O_dot_linear(8) = O_dot(3,2);
O_dot_linear(9) = O_dot(3,3);

end

function matrix = orientation_from_angles(euler_angles)
matrix = zeros(3,3);
O_3 = zeros(3,3);
O_2 = zeros(3,3);
O_1 = zeros(3,3);
phi = euler_angles(1);
theta = euler_angles(2);
psi = euler_angles(3);
%% Orientation matrix elements
```

```
O_3 = [ cos(phi),  sin(phi),    0;
       -sin(phi), cos(phi),    0;
              0,         0,   1;
       ];


O_2 = [ 1,    0,    0;
              0,    cos(theta),    sin(theta);
       0,     -sin(theta),   cos(theta);
       ];

O_1 = [ cos(psi),  sin(psi),    0;
       -sin(psi), cos(psi),    0;
              0,         0,   1;
       ];
matrix = O_1*(O_2*O_3);

%%
end
function euler_angles = euler_from_rotation(R)
euler_angles = zeros(3,1);
if abs(R(3,1)) ~= 1

   %% theta vals
   theta1 = -asin(R(1,3));
   theta2 = pi - theta1;

   %% psi vals
   psi1 = atan2((R(2,3)/cos(theta1)) , (R(3,3)/cos(theta1)));
   psi2 = atan2((R(2,3)/cos(theta2)) , (R(3,3)/cos(theta2)));

   %% phi vals
   phi1 = atan2((R(1,2)/cos(theta1)) , (R(1,1)/cos(theta1)));
   phi2 = atan2((R(1,2)/cos(theta2)) , (R(1,1)/cos(theta2)));

   euler_angles = [psi1, theta1, phi1];
else
   phi = 0;
   if R(1,3)==-1
```

```
        theta = pi/2;
        psi = phi + atan2(R(2,1), R(3,1));
    else
        theta = -pi/2;
        psi = -phi + atan2(-R(2,1), -R(3,1));

    end
    euler_angles = [ psi,theta, phi];
end

end
```

Question 4:

```
%% Initialising variables
phi = pi/6;
r_a = [1;0;0];
r_a_dot = [0,cos(phi),sin(phi)];
O_b_a_ini = [1,          0,    0;
            0,    cos(phi), sin(phi);
            0,   -sin(phi), cos(phi);];
O_a_b_ini = (O_b_a_ini);

x0 = [r_a(1);r_a(2);r_a(3);
     r_a_dot(1);r_a_dot(2);r_a_dot(3);
     O_a_b_ini(1,1);O_a_b_ini(1,2);O_a_b_ini(1,3);
     O_a_b_ini(2,1);O_a_b_ini(2,2);O_a_b_ini(2,3);
     O_a_b_ini(3,1);O_a_b_ini(3,2);O_a_b_ini(3,3);];

tspan = [0 20];
options = odeset('RelTol',1e-12,'AbsTol',1e-12);
sol = ode45(@(t,x) drone_dyn(t,x),tspan,x0,options); % Integrating 4.10.10 using ode45
t_1 = linspace(0,20,1000); % Time frame
x_total = deval(sol,t_1); % Obtaining Function for all reqired time of evaluation


plot3(x_total(1,:),x_total(2,:),x_total(3,:));
% fig = figure()
% subplot(


function euler_angles = euler_from_rotation(R)
euler_angles = zeros(3,1);
```

```
if abs(R(3,1)) ~= 1

    %% theta vals
    theta1 = -asin(R(1,3));
    theta2 = pi - theta1;

    %% psi vals
    psi1 = atan2((R(2,3)/cos(theta1)) , (R(3,3)/cos(theta1)));
    psi2 = atan2((R(2,3)/cos(theta2)) , (R(3,3)/cos(theta2)));

    %% phi vals
    phi1 = atan2((R(1,2)/cos(theta1)) , (R(1,1)/cos(theta1)));
    phi2 = atan2((R(1,2)/cos(theta2)) , (R(1,1)/cos(theta2)));

    euler_angles = [psi1, theta1, phi1];
else
    phi = 0;
    if R(1,3)==-1
        theta = pi/2;
        psi = phi + atan2(R(2,1), R(3,1));
    else
        theta = -pi/2;
        psi = -phi + atan2(-R(2,1), -R(3,1));

    end
    euler_angles = [ psi,theta, phi];
end

end
function x_dot = drone_dyn(t,x)
%% intialising
x_dot = zeros(15,1);
phi = pi/6;
g = 9.80665;
O = [ x(7),  x(8),  x(9);
      x(10), x(11), x(12);
      x(13), x(14), x(15);];
% an = euler_from_rotation(O);
% phi = an(3);
omega = [0;0;1];
a_m = [(-1-(g*sin(phi)*sin(t)));
        -(g*sin(phi)*cos(t));
        -g*cos(phi)];
G = [0;0;-g];
```

```
%% Velocity
x_dot(1) = x(4);
x_dot(2) = x(5);
x_dot(3) = x(6);
%% Acceleration
a_b = ((a_m - O*G));
a = inv(O)*a_b;
x_dot(4) = a(1);
x_dot(5) = a(2);
x_dot(6) = a(3);

%% Oriention
omega_cross = [0, -omega(3), omega(2);
               omega(3), 0, -omega(1);
               -omega(2), omega(1), 0];
omega_cross = omega_cross*-1;

O_dot = omega_cross*O;

for i =1:3
    for j=1:3
        x_dot(6+((i-1)*3)+j) = O_dot(i,j);
    end
end

end
```