ASSIGNMENT 4

1. EXTENDED KALMAN FILTER

Program:

function Extended\_Kalman\_filter

dt = 0.1;

N = 60;

time = [0:dt:N];

iter = N/dt;

x\_actual = zeros(4,iter+1);

x\_actual(:,1) = [0;0;50;50];

x\_estimated = zeros(4,iter+1);

x\_estimated(:,1) = x\_actual(:,1);

y = zeros(2,iter+1);

error = zeros(4,iter+1);

error\_sqr = zeros(4,iter+1);

% assigning all constants

P = zeros(4,4);

P\_values= zeros(4,iter+1);

P\_values(:,1) = [0;0;0;0];

N1 = 20;

E1 = 0;

N2 = 0;

E2 = 20;

Q = [0 0 0 0; 0 0 0 0; 0 0 4 0; 0 0 0 4];

R = [1 0; 0 1];

n\_1 = randn(1,iter);

n\_2 = randn(1,iter);

for i=2:iter+1

noise = [0;0;2\*n\_1(1,i-1);2\*n\_2(1,i-1)];

F = [1 0 0.1 0; 0 1 0 0.1;0 0 1 0; 0 0 0 1];

% actual state estimation

x\_actual(:,i) = F\*x\_actual(:,i-1)+noise;

y1 = sqrt((x\_actual(1,i)-N1)^2 + (x\_actual(2,i)-E1)^2);

y2 = sqrt((x\_actual(1,i)-N2)^2 + (x\_actual(2,i)-E2)^2);

y(:,i) = [y1;y2]+[n\_1(1,i-1);n\_2(1,i-1)];

% extended kalman filter estimate

x\_estimated(:,i) = F\*x\_estimated(:,i-1);

P\_values(:,i) = [P(1,1);P(2,2);P(3,3);P(4,4)];

P = F\*P\*F'+ Q;

h1 = sqrt((x\_estimated(1,i)-N1)^2 + (x\_estimated(2,i)-E1)^2);

h2 = sqrt((x\_estimated(1,i)-N2)^2 + (x\_estimated(2,i)-E2)^2);

H = [(x\_estimated(1,i)-N1)/h1 (x\_estimated(2,i)-E1)/h1 0 0;

(x\_estimated(1,i)-N2)/h2 (x\_estimated(2,i)-E2)/h2 0 0 ];

K = (P\*H')\*inv(H\*P\*H'+R);

y\_hat = [h1;h2];

x\_estimated(:,i) = x\_estimated(:,i)+K\*(y(:,i)-y\_hat);

P = (eye(4)- K\*H)\*P;

error(:,i) = abs(x\_actual(:,i)-x\_estimated(:,i));

error\_sqr(:,i) = error(:,i).^2;

end

theor\_1 = mean(sqrt(P\_values(1,:)))

pract\_1 = std(error(1,:))

theor\_2 = mean(sqrt(P\_values(2,:)))

pract\_2 = std(error(2,:))

theor\_3 = mean(sqrt(P\_values(3,:)))

pract\_3 = std(error(3,:))

theor\_4 = mean(sqrt(P\_values(4,:)))

pract\_4 = std(error(4,:))

figure;

subplot(2,2,1)

plot(time,error(1,:));

subpot(2,2,2)

plot(time,error(2,:));

subplot(2,2,3)

plot(time,error(3,:));

subpot(2,2,4)

figure;

hold on

plot(time,x\_estimated(1,:));

plot(time,x\_actual(1,:));

hold off

figure;

hold on

plot(time,x\_estimated(2,:));

plot(time,x\_actual(2,:));

hold off

figure;

hold on

plot(time,x\_estimated(3,:));

plot(time,x\_actual(3,:));

hold off

figure;

hold on

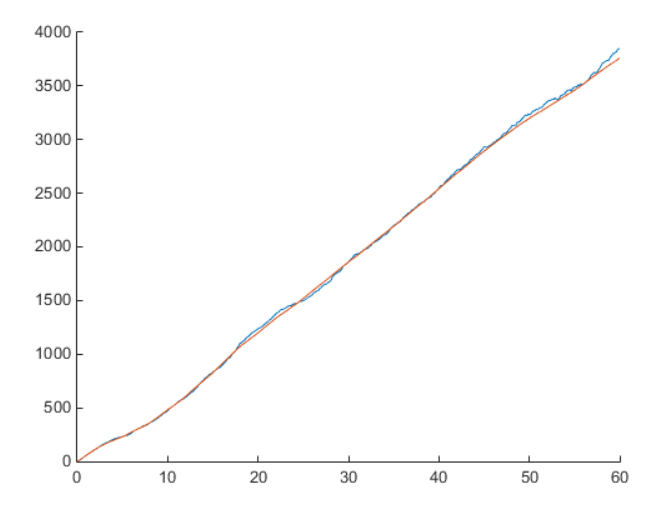
plot(time,x\_estimated(4,:));

plot(time,x\_actual(4,:));

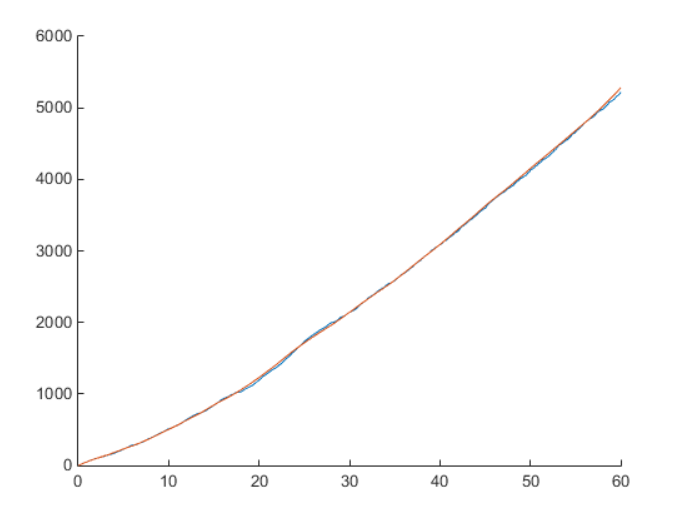
hold off

GRAPH :

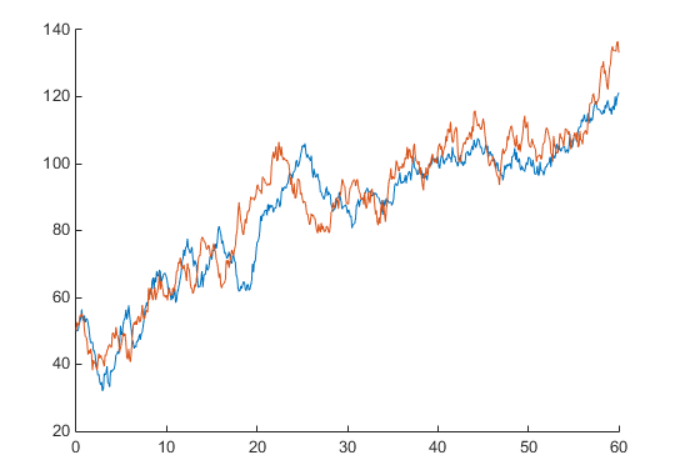
X\_1:



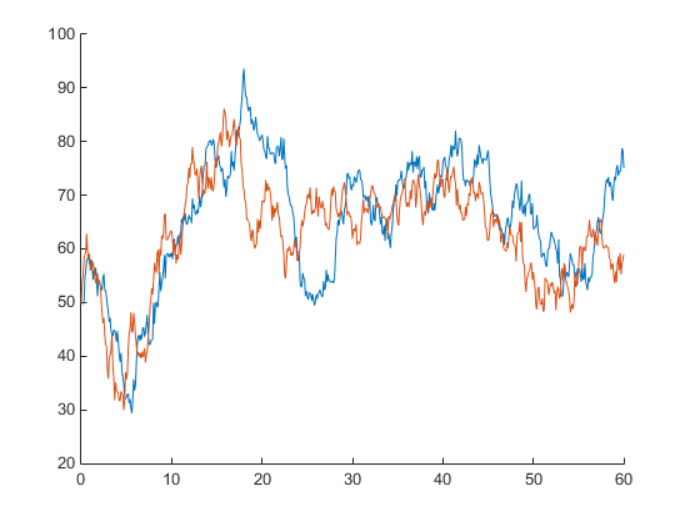
X\_2



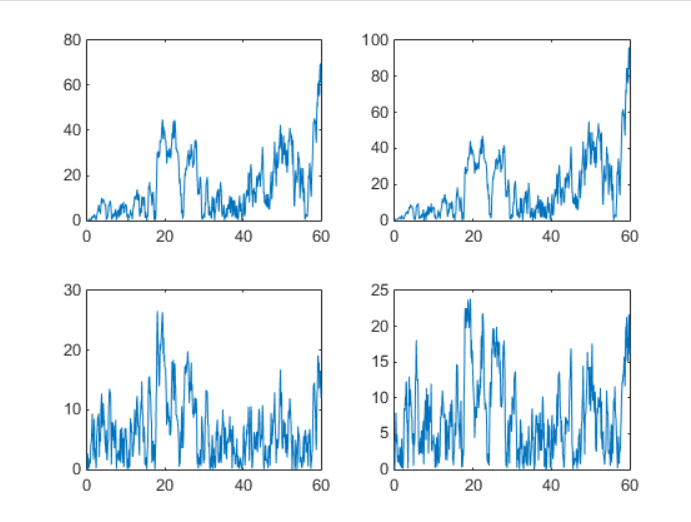
X\_3:



X4:



Error plot:



Standard deviation:

>> Extended\_Kalman\_filter

theor\_1 =

4.6389

pract\_1 =

4.6126

theor\_2 =

20.2034

pract\_2 =

17.4914

theor\_3 =

4.2043

pract\_3 =

2.7991

theor\_4 =

9.7098

pract\_4 =

7.7784

Unscented Kalman Filter:

Program:

function unscented\_kalman\_filter

dt = 0.1;

N = 60;

time = [0:dt:N];

iter = N/dt;

x\_actual = zeros(4,iter+1);

x\_actual(:,1) = [0;0;50;50];

x\_ukf = zeros(4,iter+1);

x\_ukf(:,1) = x\_actual(:,1);

y = zeros(2,iter+1);

error = zeros(4,iter+1);

error\_sqr = zeros(4,iter+1);

% assigning all constants

p\_ukf = zeros(4,4);

p\_ukf(1,1) = 0.00001;

p\_ukf(2,2) = 0.00001;

p\_ukf(3,3) = 0.00001;

p\_ukf(4,4) = 0.00001;

P\_values= zeros(4,iter+1);

P\_values(:,1) = [0;0;0;0];

N1 = 20;

E1 = 0;

N2 = 0;

E2 = 20;

Q = [0 0 0 0; 0 0 0 0; 0 0 4 0; 0 0 0 4];

R = [1 0; 0 1];

n\_1 = randn(1,iter);

n\_2 = randn(1,iter);

for k=2:iter+1

k

noise = [0;0;2\*n\_1(1,k-1);2\*n\_2(1,k-1)];

F = [1 0 0.1 0; 0 1 0 0.1;0 0 1 0; 0 0 0 1];

% actual state estimation

x\_actual(:,k) = F\*x\_actual(:,k-1)+noise;

y1 = sqrt((x\_actual(1,k)-N1)^2 + (x\_actual(2,k)-E1)^2);

y2 = sqrt((x\_actual(1,k)-N2)^2 + (x\_actual(2,k)-E2)^2);

y(:,k) = [y1;y2]+[1\*n\_1(1,k-1);1\*n\_2(1,k-1)];

% Unscented Kalman Filter

% time update equation

P\_values(:,k) = [p\_ukf(1,1);p\_ukf(2,2);p\_ukf(3,3);p\_ukf(4,4)];

mat = chol(p\_ukf);

for j = 1:8

if(j<=4)

x\_i(:,j) = x\_ukf(:,k-1)+sqrt(4).\*mat(:,j);

else

x\_i(:,j) = x\_ukf(:,k-1)-sqrt(4).\*mat(:,j-4);

end

x\_i(:,j) = F\*x\_i(:,j);

end

x\_ukf(:,k) = mean(x\_i,2);

% To caluculate the value of P -- deviation of all vaues from the mean

% value

for i= 1:8

if i == 1

P\_u = (x\_i(:,i)-x\_ukf(:,k))\*(x\_i(:,i)-x\_ukf(:,k))';

else

P\_u = P\_u+(x\_i(:,i)-x\_ukf(:,k))\*(x\_i(:,i)-x\_ukf(:,k))';

end

end

P\_u = (1/8).\*P\_u+Q;

% To generate value for y

% measurement update equation

% updating with best known result

mat = chol(P\_u);

for n = 1:8

if(n<=4)

x\_i(:,n) = x\_ukf(:,k)+sqrt(4).\*mat(:,n);

else

x\_i(:,n) = x\_ukf(:,k-1)-sqrt(4).\*mat(:,n-4);

end

y\_1(1,n) = sqrt((x\_i(1,n)-N1)^2+(x\_i(2,n)-E1)^2);

y\_2(1,n) = sqrt((x\_i(1,n)-N2)^2+(x\_i(2,n)-E2)^2);

y\_esti(:,n) = [y\_1(1,n);y\_2(1,n)];

end

y\_estimated(:,k) = mean(y\_esti,2);

for m= 1:8

if m == 1

P\_y = (y\_esti(:,m)-y\_estimated(:,k))\*(y\_esti(:,m)-y\_estimated(:,k))';

P\_xy = (x\_i(:,m)-x\_ukf(:,k))\*(y\_esti(:,m)-y\_estimated(:,k))';

else

P\_y = P\_y+(y\_esti(:,m)-y\_estimated(:,k))\*(y\_esti(:,m)-y\_estimated(:,k))';

P\_xy = P\_xy+(x\_i(:,m)-x\_ukf(:,k))\*(y\_esti(:,m)-y\_estimated(:,k))';

end

end

P\_y = (1/8).\*P\_y+R;

P\_xy =(1/8).\*P\_xy;

% To calculate gain

K\_k = P\_xy\*inv(P\_y);

x\_ukf(:,k)= x\_ukf(:,k)+K\_k\*(y(:,k)-y\_estimated(:,k));

p\_ukf = P\_u%-K\_k\*P\_y\*K\_k';

error(:,k) = abs(x\_actual(:,k)-x\_ukf(:,k));

error\_sqr(:,k) = error(:,k).^2;

end

theor\_1 = mean(sqrt(P\_values(1,:)))

pract\_1 = std(error(1,:))

theor\_2 = mean(sqrt(P\_values(2,:)))

pract\_2 = std(error(2,:))

theor\_3 = mean(sqrt(P\_values(3,:)))

pract\_3 = std(error(3,:))

theor\_4 = mean(sqrt(P\_values(4,:)))

pract\_4 = std(error(4,:))

plot(time,x\_ukf(1,:));

hold on

plot(time,x\_actual(1,:));

hold off

\*\*\* The program gives a error when the update is done using gain so it is neglected.

Standard deviation:

theor\_1 =

51.5122

pract\_1 =

17.6660

theor\_2 =

51.5122

pract\_2 =

18.3955

theor\_3 =

20.8542

pract\_3 =

17.7357

theor\_4 =

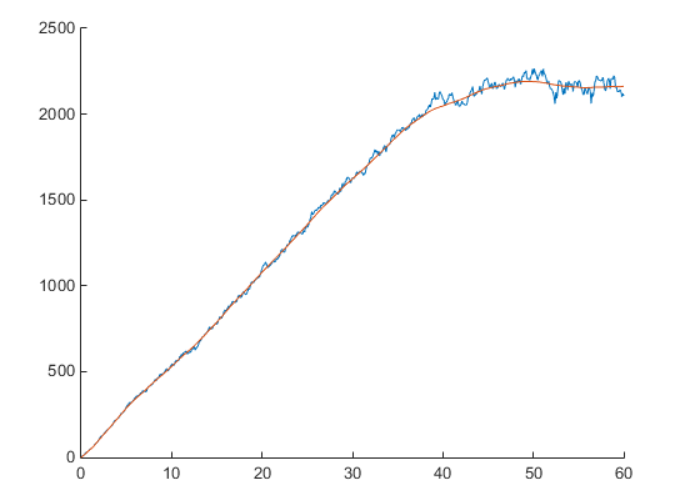
20.8542

pract\_4 =

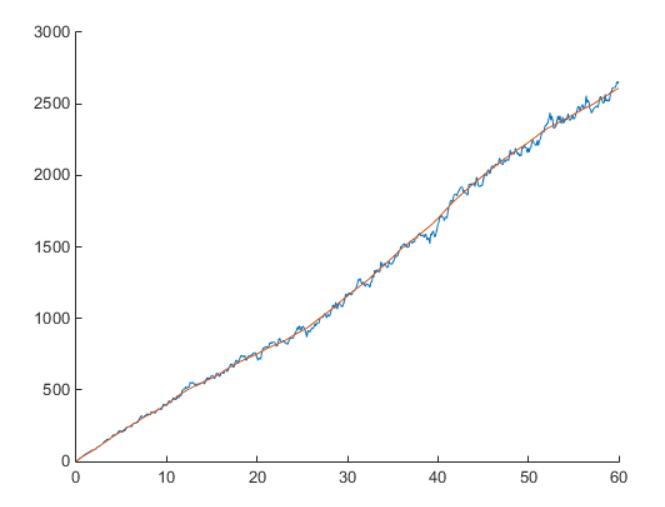
5.6008

GRAPH:

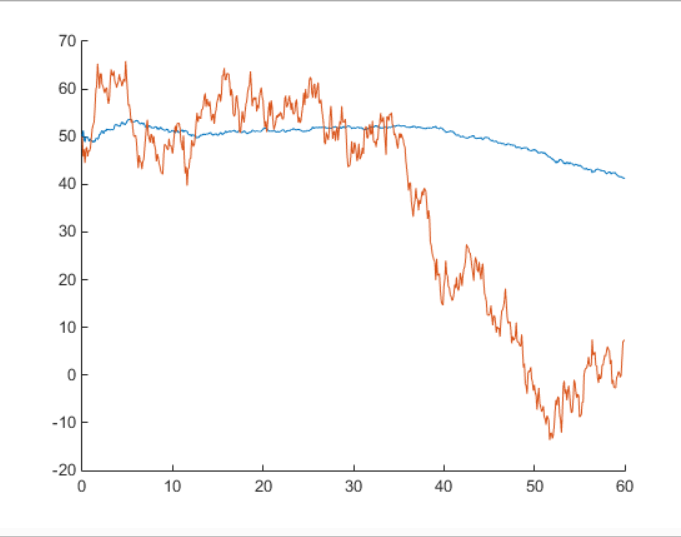
X1



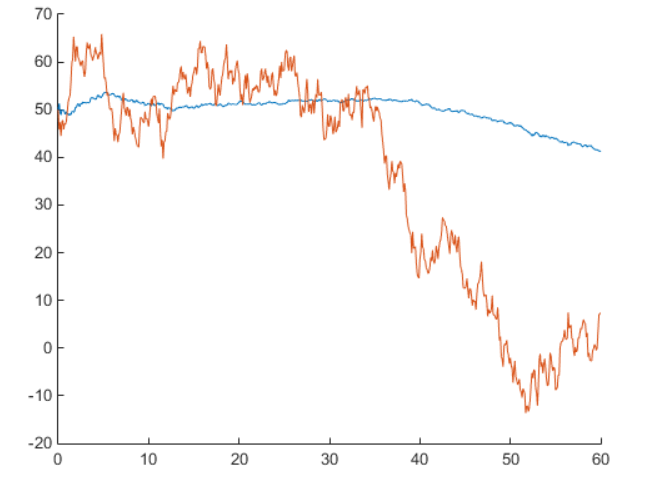
X2:



X3:



X4:



Error:

