```
% Written by Irina Yareshko and Luca Breggion, Skoltech 2022

close all
clear
clc

set(0,'defaulttextInterpreter','latex');
set(groot,'defaultAxesTickLabelInterpreter','latex');
set(groot,'defaultLegendInterpreter','latex');

%%

%Initial data
T = 2; N = 26;
InititalState = [13500/sqrt(2); -50; 13500/sqrt(2); -45];
sigma_D = 20;      %variance of range noise of measurements
sigma_beta = 0.02; %variance of azimuth noise of measurements

%Creating of arrays for running Kalman filter M times
M = 500;                          %number of runs
Err_range_filtered  = zeros(M,N); %Filtraion error of range
Err_range_forecast  = zeros(M,N); %Predicted error of range
Err_azimut_filtered = zeros(M,N); %Filtraion error of azimut
Err_azimut_forecast = zeros(M,N); %Predicted error of azimut
Condition_nums      = zeros(M,N); %array of average condition number
K_matr_values       = zeros(M,N);

Mean_Z_x     = zeros(1,N);
Mean_azimuth = zeros(1,N);

Mean_range = zeros(1,N);
for i=1:M
    %Generation of deterministic trajectory and its measurements
    [TruePolar, TrueCart, Z_c, Z_p] = ...
        Generation_true_determ(N, T, sigma_D, sigma_beta, InititalState);
    %Applying or Kalman filter for measurements
    [Z_filtered, P, P_pred, ...
        range_fe, azimuth_fe, CondMatr, K_matr] = ...
        Kalman(Z_c, Z_p, T, sigma_D, sigma_beta);

    Err_range_filtered(i,:)  = (TruePolar(1,:) - range_fe(1,:)).^2;
    Err_range_forecast(i,:)  = (TruePolar(1,:) - range_fe(2,:)).^2;
    Err_azimut_filtered(i,:) = (TruePolar(2,:) - azimuth_fe(1,:)).^2;
    Err_azimut_forecast(i,:) = (TruePolar(2,:) - azimuth_fe(2,:)).^2;

    Condition_nums(i,:) = CondMatr;
    K_matr_values(i,:) = K_matr(1,1,:);

    Mean_Z_x = Mean_Z_x + Z_filtered(1,2:N+1);
    Mean_azimuth = Mean_azimuth + azimuth_fe(1,:);
end
Mean_Z_x = Mean_Z_x/M;
Mean_azimuth = Mean_azimuth/M;
```

```matlab

FinalErr_range_filtered  = sqrt(1/(M - 1)*sum(Err_range_filtered));
FinalErr_range_forecast  = sqrt(1/(M - 1)*sum(Err_range_forecast));
FinalErr_azimut_filtered = sqrt(1/(M - 1)*sum(Err_azimut_filtered));
FinalErr_azimut_forecast = sqrt(1/(M - 1)*sum(Err_azimut_forecast));
Final_CN = sum(Condition_nums)/M;
%Final_K_matr = sum(K_matr_values)/M;

%% Point 2 (Generated motion in polar coordinate system)
figure(1)
polarplot(TruePolar(2,:), TruePolar(1,:),'m', 'LineWidth', 1.5)
grid on; grid minor
legend('True motion', 'FontSize', 30);
%title('Object moves uniformly', 'FontSize', 20);

%% Point 10 (Errors of extrapolation and filtration estimates of range and azimuth)

figure(2)
plot(3:N,FinalErr_range_filtered(3:N),'m',...
     3:N,FinalErr_range_forecast(3:N),'c',...
     3:N,sigma_D*ones(1, N-2), 'black', 'LineWidth', 1.2);
grid on; grid minor
legend('True filtration error','True extrapolation error','$\sigma_D$', 'FontSize', 30, 'interpreter', 'latex')
%title('a) Errors of range', 'FontSize', 20);
xlabel('Step', 'FontSize', 30)
ylabel('Errors', 'FontSize', 30)

figure(3)
plot(3:N,FinalErr_azimut_filtered(3:N),'m', ...
     3:N,FinalErr_azimut_forecast(3:N),'c', ...
     3:N,sigma_beta*ones(1, N-2), 'black', 'LineWidth', 1.2);
grid on; grid minor
legend('True filtration error','True extrapolation error','$\sigma_\beta$', 'FontSize', 30, 'interpreter', 'latex');
%title('b) Errors of azimuth', 'FontSize', 20);
xlabel('Step', 'FontSize', 30)
ylabel('Errors', 'FontSize', 30)

%% Point 12 (Building of plot of dependence of coordinate x on azimuth b)

figure(4)
plot(Mean_azimuth(1,:), Mean_Z_x(1,:),'c',TruePolar(2,:), TrueCart(1,:), 'k--', 'LineWidth', 1.2);
grid on; grid minor
legend('x = f($\beta$) filtered values', 'x = g($\beta$) true data', 'FontSize', 30, 'interpreter', 'latex');
%title('Dependance of coordinate x on azimuth $\beta$', 'FontSize', 30, 'interpreter', 'latex');
xlabel('Azimuth', 'FontSize', 30)
ylabel('Coordinate x', 'FontSize', 30)

```

```
%% Point 12 (Building of plot of dinamics of condotion number)

figure(5)
plot(1:N, Final_CN,'r', 'LineWidth', 1.2);
grid on; grid minor
legend('Condition number', 'FontSize', 30);
%title('Dinamics of condition number', 'FontSize', 20);
xlabel('Step', 'FontSize', 30)
ylabel('Condition number', 'FontSize', 30)

%% Point 13 (Building of plot of dinamics of filter gain)
figure()
plot(1:N, K_matr_values(66,:),'g', 'LineWidth', 1.2);
grid on; grid minor
legend('Filter gain','FontSize', 20);
%title('Dinamics of filter gain', 'FontSize', 20);
xlabel('Step','FontSize', 30)
ylabel('K','FontSize', 30)

%% Point 14 (Quite close distance from an observer)
InititalState = [3500/sqrt(2); -50; 3500/sqrt(2); -45];
for i=1:M
    %Generation of deterministic trajectory and its measurements
    [TruePolar, TrueCart, Z_c, Z_p] = ...
        Generation_true_determ(N, T, sigma_D, sigma_beta, InititalState);
    %Applying or Kalman filter for measurements
    [Z_filtered, P, P_pred, ...
        range_fe, azimuth_fe, CondMatr, K_matr] = ...
        Kalman(Z_c, Z_p, T, sigma_D, sigma_beta);

    Err_range_filtered(i,:)  = (TruePolar(1,:) - range_fe(1,:)).^2;
    Err_range_forecast(i,:)  = (TruePolar(1,:) - range_fe(2,:)).^2;
    Err_azimut_filtered(i,:) = (TruePolar(2,:) - azimuth_fe(1,:)).^2;
    Err_azimut_forecast(i,:) = (TruePolar(2,:) - azimuth_fe(2,:)).^2;

    Condition_nums(i,:) = CondMatr;
    K_matr_values(i,:) = K_matr(1,1,:);

    Mean_Z_x = Mean_Z_x + Z_filtered(1,2:N+1);
    Mean_azimuth = Mean_azimuth + azimuth_fe(1,:);
    Mean_range = Mean_range + range_fe(1,:);
end
Mean_Z_x = Mean_Z_x/M;
Mean_azimuth = Mean_azimuth/M;
Mean_range = Mean_range/M;

FinalErr_range_filtered  = sqrt(1/(M-1)*sum(Err_range_filtered));
FinalErr_range_forecast  = sqrt(1/(M-1)*sum(Err_range_forecast));
FinalErr_azimut_filtered = sqrt(1/(M-1)*sum(Err_azimut_filtered));
FinalErr_azimut_forecast = sqrt(1/(M-1)*sum(Err_azimut_forecast));
Final_CN = sum(Condition_nums)/M;
Final_K_matr = sum(K_matr_values)/M;

```

```matlab
% Generated motion in polar coordinate system (Quite close)
figure(7)
polarplot(TruePolar(2,:), TruePolar(1,:),'k', 'LineWidth', 1.2)
grid on; grid minor
legend('True motion', 'FontSize', 20);
%title('Object moves uniformly (Quite close)');

%% Point 15 (Errors of extrapolation and filtration estimates of (Quite close))
figure(8)
% subplot(1,2,1);
plot(3:N, FinalErr_range_filtered(3:N),'m',...
     3:N, FinalErr_range_forecast(3:N),'c',...
     3:N, sigma_D*ones(1, N-2), 'black', 'LineWidth', 1.2);
grid on; grid minor
legend('True filtration error','True extrapolation error','$\sigma_D$', 'FontSize', 30, 'interpreter', 'latex');
%title('a) Errors of range (Quite close)');
xlabel('Step')
ylabel('Errors')

figure(9)
plot(3:N, FinalErr_azimut_filtered(3:N),'m',...
     3:N, FinalErr_azimut_forecast(3:N),'c',...
     3:N, sigma_beta*ones(1, N-2), 'black');
grid on; grid minor
legend('True filtration error','True extrapolation error','$\sigma_\beta$', 'FontSize', 30, 'interpreter', 'latex');
%title('b) Errors of azimuth (Quite close)');
xlabel('Step')
ylabel('Errors')

%% Point 16 (Dependence of coordinate x on azimuth b (Quite close))

figure(10)
plot(Mean_azimuth(1,:), Mean_Z_x(1,:),'m',TruePolar(2,:), TrueCart(1,:), 'c--', 'LineWidth', 1.2);
grid on; grid minor
legend('x = f($\beta$) filtered values', 'x = g($\beta$) true data','FontSize', 30, 'interpreter', 'latex');
%title('Dependence of coordinate x on azimuth $\beta$ (Quite close)','FontSize', 30, 'interpreter', 'latex');
xlabel('Azimuth','FontSize', 30)
ylabel('Coordinate x','FontSize', 30)

%% Point 17 (Building of plot of dinamics of condotion number (Quite close))

figure(11)
plot(1:N,Final_CN,'r', 'LineWidth', 1.2);
grid on; grid minor
legend('Condition number','FontSize', 30);
%title('Dinamics of condition number (Quite close)','FontSize', 30);
xlabel('Step','FontSize', 30)
ylabel('Condition number','FontSize', 30)
```

```matlab

%% Point 19 quite close distance from an observer and other values of variances
sigma_D=50; %variance of range noise of measurements
sigma_beta=0.0015; %variance of azimuth noise of measurements
for i=1:M
    %Generation of deterministic trajectory and its measurements
    [TruePolar,TrueCart,Z_c,Z_p] = ...
        Generation_true_determ(N,T,sigma_D,sigma_beta,InititalState);
    %Applying or Kalman filter for measurements
    [Z_filtered, P, P_pred, ...
        range_fe, azimuth_fe, CondMatr, K_matr] = ...
        Kalman(Z_c,Z_p,T,sigma_D,sigma_beta);

    Err_range_filtered(i,:) = (TruePolar(1,:) - range_fe(1,:)).^2;
    Err_range_forecast(i,:) = (TruePolar(1,:) - range_fe(2,:)).^2;
    Err_azimut_filtered(i,:)= (TruePolar(2,:) - azimuth_fe(1,:)).^2;
    Err_azimut_forecast(i,:)= (TruePolar(2,:) - azimuth_fe(2,:)).^2;

    Condition_nums(i,:) = CondMatr;
    K_matr_values(i,:) = K_matr(1,1,:);

    Mean_Z_x = Mean_Z_x + Z_filtered(1,2:N+1);
    Mean_azimuth = Mean_azimuth + azimuth_fe(1,:);
end
Mean_Z_x = Mean_Z_x/M;
Mean_azimuth = Mean_azimuth/M;

FinalErr_range_filtered =sqrt(1/(M-1)*sum(Err_range_filtered));
FinalErr_range_forecast =sqrt(1/(M-1)*sum(Err_range_forecast));
FinalErr_azimut_filtered = sqrt(1/(M-1)*sum(Err_azimut_filtered));
FinalErr_azimut_forecast = sqrt(1/(M-1)*sum(Err_azimut_forecast));
Final_CN=sum(Condition_nums)/M;
Final_K_matr=sum(K_matr_values)/M;

% Generated motion in polar coordinate system (Quite close, other variances)
figure(12)
polarplot(TruePolar(2,:),TruePolar(1,:), 'LineWidth', 1.5)
grid on; grid minor
legend('True motion','FontSize',30);
%title('Object moves uniformly (Quite close, other variances)','FontSize', 30);

%Errors of extrapolation and filtration estimates of (Quite close, other variances)
figure(13)
plot(3:N,FinalErr_range_filtered(3:N),'m',...
     3:N,FinalErr_range_forecast(3:N),'c',...
     3:N,sigma_D*ones(1,N-2), 'black', 'LineWidth', 1.2);
grid on; grid minor
legend('True filtration error','True extrapolation error','$\sigma_D$','FontSize', 30, 'interpreter', 'latex');
%title('a) Errors of range (Quite close, other variances)','FontSize', 20);
xlabel('Step','FontSize', 20)
ylabel('Errors','FontSize', 20)
```

```matlab

figure(14)
plot(3:N,FinalErr_azimut_filtered(3:N),'m',...
     3:N,FinalErr_azimut_forecast(3:N),'c',...
     3:N,sigma_beta*ones(1,N-2), 'black', 'LineWidth', 1.2);
grid on; grid minor
legend('True filtration error','True extrapolation error','$\sigma_\beta$','FontSize', 30,'interpreter', 'latex')
%title('b) Errors of azimuth (Quite close, other variances)','FontSize', 20);
xlabel('Step','FontSize', 20)
ylabel('Errors','FontSize', 20)

%Dependence of coordinate x on azimuth b (Quite close, other variances)
figure(15)
plot(Mean_azimuth(1,:), Mean_Z_x(1,:),'r',TruePolar(2,:), TrueCart(1,:), 'b--', 'LineWidth', 1.2);
grid on; grid minor
legend('x = f($\beta$) filtered values', 'x = g($\beta$) true data','FontSize', 20,'interpreter', 'latex');
%title('Dependence of coordinate x on azimuth $\beta$ (Quite close, other variances)','FontSize', 20, 'interpreter', 'latex');
xlabel('Azimuth','FontSize', 20)
ylabel('Coordinate x','FontSize', 20)

% Building of plot of dinamics of condotion number (Quite close, other variances)
figure(16)
plot(1:N, Final_CN,'b', 'LineWidth', 1.2);
grid on; grid minor
legend('Condition number','FontSize', 20);
%title('Dinamics of condition number (Quite close, other variances)','FontSize', 20);
xlabel('Step','FontSize', 20)
ylabel('Condition number','FontSize', 20)
%xlim([1 26])


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                           %
%                                FUNCTION                                   %
%                                                                           %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [True_polar, True_Cartesian, Z_c, Z_p] = Generation_true_determ(N, T, ...
    sigma_D, sigma_b, InititalState)
% Generation of true trajectory
    X   = zeros(1, N); X(1)   = InititalState(1);
    Y   = zeros(1, N); Y(1)   = InititalState(3);
    V_x = zeros(1, N); V_x(1) = InititalState(2);
    V_y = zeros(1, N); V_y(1) = InititalState(4);
    for i = 2:N
        X(i) = X(i - 1) + V_x(i - 1)*T;
```

```
        V_x(i) = V_x(i - 1);
        Y(i) = Y(i - 1) + V_y(i - 1)*T;
        V_y(i) = V_y(i - 1);
    end
    True_Cartesian = [X; V_x; Y; V_y];
    %Generation of true values of range D and aimuth b
    D = sqrt(X.^2 + Y.^2);
    b = atan(X./Y);
    True_polar = [D; b];
    %Generation of measurements
    D_m = zeros(1, N); %array of measurements of range
    b_m = zeros(1, N); %array of measurements of azimuth
    x_m = zeros(1, N); %array of pseudo-measurements of x
    y_m = zeros(1, N); %array of pseudo-measurements of y
    Z_c = zeros(2, N); %array of pseudo-measurements in Cartesian coordinates
    Z_p = zeros(2, N); %array of measurements in polar coordinates
    for i = 1:N
        D_m(i) = D(i) + randn*sigma_D;
        b_m(i) = b(i) + randn*sigma_b;

        x_m(i) = D_m(i)*sin(b_m(i));
        y_m(i) = D_m(i)*cos(b_m(i));

        Z_p(:,i) = [D_m(i); b_m(i)];
        Z_c(:,i) = [x_m(i); y_m(i)];
    end
end

function [Z_filtered, P, P_pred, range_fe, azimuth_fe, CondMatr, K] = Kalman(Z_cart,...
    Z_polar, T, sigma_D, sigma_beta)

    N = length(Z_cart);
    Z_filtered = zeros(4, N + 1); %Filtered data
    Z_forecast = zeros(4, N);     %Forecast data

    P = zeros(4, 4, N + 1);       %Filtration error covariance matrix
    P_pred = zeros(4, 4, N + 1);  %Prediction error covariance matrix
    Fi = [1 T 0 0; 0 1 0 0; 0 0 1 T; 0 0 0 1];
    H = [1 0 0 0; 0 0 1 0];

    Z_filtered(:, 1) = [40000; -20; 40000; -20]; %Initian state vector
    P(:, :, 1) = [10^10 0 0 0; 0 10^10 0 0; 0 0 10^10 0; 0 0 0 10^10];
    R = zeros(2,2);
    K = zeros(4, 2, N);
    range_fe = zeros(2,N);    %array of filtered and extrapolated range
    azimuth_fe = zeros(2,N); %array of filtered and extrapolated azimuth
    CondMatr = zeros(1,N);

    for i = 2:N + 1
        R(1,1) = sin(Z_polar(2,i - 1))^2*sigma_D^2 + (Z_polar(1,i - 1))^2*cos(Z_polar(2,i - 1))^2*sigma_beta^2;
        R(2,2) = cos(Z_polar(2,i - 1))^2*sigma_D^2 + (Z_polar(1,i - 1))^2*sin
```

```
(Z_polar(2,i - 1))^2*sigma_beta^2;
        R(1,2) = sin(Z_polar(2,i - 1))*cos(Z_polar(2,i - 1))*(sigma_D^2 - (Z_polar(1,i - 1))^2*sigma_beta^2);
        R(2,1) = sin(Z_polar(2,i - 1))*cos(Z_polar(2,i - 1))*(sigma_D^2 - (Z_polar(1,i - 1))^2*sigma_beta^2);
        %Prediction part
        Z_forecast(:,i - 1) =  Fi * Z_filtered(:,i - 1);
        P_pred(:,:,i-1) = Fi * P(:,:,i - 1) * Fi';
        %Filtrarion part
        K(:,:,i - 1) = P_pred(:,:,i - 1)*(H') * (H*P_pred(:,:,i - 1)*H' + R)^(-1);
        P(:, :, i) = (eye(4) - K(:,:,i - 1)*H)*P_pred(:,:,i - 1);
        Z_filtered(:,i) = Z_forecast(:,i - 1) + K(:,:,i - 1)*(Z_cart(:,i - 1) - H*Z_forecast(:,i - 1));

        range_fe(1,i - 1) = sqrt( Z_filtered(1,i)^2 + Z_filtered(3,i)^2 );
        range_fe(2,i - 1) = sqrt( Z_forecast(1,i - 1)^2 + Z_forecast(3,i - 1)^2);

        azimuth_fe(1,i - 1) = atan(Z_filtered(1,i)/Z_filtered(3,i));
        azimuth_fe(2,i - 1) = atan(Z_forecast(1,i - 1)/Z_forecast(3,i - 1));

        if (sigma_D^2) > (Z_polar(1,i - 1)^2*sigma_beta^2)
            CondMatr(i - 1) = (sigma_D^2)/(Z_polar(1,i - 1)^2*sigma_beta^2);
        else
            CondMatr(i - 1) = (Z_polar(1,i - 1)^2*sigma_beta^2)/(sigma_D^2);
        end
    end
end

function [X, P, D_K, b_K, condition_num, K] = Kalman_filter_determ(Z_c, Z_p, ...
    T, sigma_D, sigma_beta)

    N = length(Z_c);
    Fi = [1 T 0 0; 0 1 0 0; 0 0 1 T; 0 0 0 1]; %transition matrix
    H = [1 0 0 0; 0 0 1 0]; %observation matrix

    %Kalman filter development
    %Creating of arrays
    X = zeros(4,2,N+1);    %filtered and smoothed data
    P = zeros(4,8,N+1);    %filtration and prediction error covariance matrix
    K = zeros(4,2,N);      %filter gain
    D_K = zeros(2,N);      %array of filtered and extrapolated range
    b_K = zeros(2,N);      %array of filtered and extrapolated azimuth
    lambda1 = zeros(1,N); %array of first eigenvalues
    lambda2 = zeros(1,N); %array of second eigenvalues
    condition_num = zeros(1,N); %array of condition numbers

    %Initial conditions
    X(:,:,1)=[40000,0; -20,0; 40000,0; -20,0];
    P(:,1:4,1)=[10^10 0 0 0; 0 10^10 0 0; 0 0 10^10 0; 0 0 0 10^10];
    R = zeros(2, 2);
    for i=2:N+1 %0 1 2 3 .... N N+1
```

```
        R(1, 1) = sin(Z_p(2,i - 1))^2*sigma_D^2 + (Z_p(1,i - 1))^2*cos(Z_p(2,i - 1))^2*sigma_beta^2;
        R(2, 2) = cos(Z_p(2,i - 1))^2*sigma_D^2 + (Z_p(1,i - 1))^2*sin(Z_p(2,i - 1))^2*sigma_beta^2;
        R(1, 2) = sin(Z_p(2,i - 1))*cos(Z_p(2,i - 1))*(sigma_D^2 - (Z_p(1,i - 1))^2*sigma_beta^2);
        R(2, 1) = sin(Z_p(2,i - 1))*cos(Z_p(2,i - 1))*(sigma_D^2 - (Z_p(1,i - 1))^2*sigma_beta^2);
        %Prediction of state vector at time i using i-1 measurements
        X(:,2,i - 1) = Fi*X(:,1,i-1);
        %Prediction error covariance matrix
        P(:,5:8,i - 1) = Fi*P(:,1:4,i-1)*Fi.';
        %Filter gain, weight of residual
        K(:,:,i-1) = P(:,5:8,i-1)*H.'*(H*P(:,5:8,i-1)*H.'+R)^(-1);
        %Improved estimate by incorporating a new measurement
        X(:,1,i) = X(:,2,i-1) + K(:,:,i-1)*(Z_c(i-1) - H*X(:,2,i-1));
        %Filtration error covariance matrix
        P(:,1:4,i) = (eye(4) - K(:,:,i-1)*H)*P(:,5:8,i-1);
        %Evaluation of predicted and extrapolated range and azimuth
        D_K(1,i-1) = sqrt(X(1,1,i)^2 + X(3,1,i)^2);
        D_K(2,i-1) = sqrt(X(1,2,i-1)^2 + X(3,2,i-1)^2);
        b_K(1,i-1) = atan(X(1,1,i)/X(3,1,i));
        b_K(2,i-1) = atan(X(1,2,i-1)/X(3,2,i-1));
        lambda1(i-1) = sigma_D^2; %const 400
        lambda2(i-1) = Z_p(1,i-1)^2*sigma_beta^2; %descrise 10^4
        if lambda1(i-1)>lambda2(i-1)
            condition_num(i-1) = lambda1(i-1)/lambda2(i-1);
        else
            condition_num(i-1) = lambda2(i-1)/lambda1(i-1);
        end
    end
end

```