```m
% 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;
```

```
 54
 55 FinalErr_range_filtered  = sqrt(1/(M - 1)*sum(Err_range_filtered));
 56 FinalErr_range_forecast  = sqrt(1/(M - 1)*sum(Err_range_forecast));
 57 FinalErr_azimut_filtered = sqrt(1/(M - 1)*sum(Err_azimut_filtered));
 58 FinalErr_azimut_forecast = sqrt(1/(M - 1)*sum(Err_azimut_forecast));
 59 Final_CN = sum(Condition_nums)/M;
 60 %Final_K_matr = sum(K_matr_values)/M;
 61
 62 %% Point 2 (Generated motion in polar coordinate system)
 63 figure(1)
 64 polarplot(TruePolar(2,:), TruePolar(1,:),'m', 'LineWidth', 1.5)
 65 grid on; grid minor
 66 legend('True motion', 'FontSize', 30);
 67 %title('Object moves uniformly', 'FontSize', 20);
 68
 69 %% Point 10 (Errors of extrapolation and filtration estimates of range and azimuth)
 70
 71 figure(2)
 72 plot(3:N,FinalErr_range_filtered(3:N),'m',...
 73      3:N,FinalErr_range_forecast(3:N),'c',...
 74      3:N,sigma_D*ones(1, N-2), 'black', 'LineWidth', 1.2);
 75 grid on; grid minor
 76 legend('True filtration error','True extrapolation error','$\sigma_D$', 'FontSize', 30, 'interpreter', 'latex')
 77 %title('a) Errors of range', 'FontSize', 20);
 78 xlabel('Step', 'FontSize', 30)
 79 ylabel('Errors', 'FontSize', 30)
 80
 81 figure(3)
 82 plot(3:N,FinalErr_azimut_filtered(3:N),'m', ...
 83      3:N,FinalErr_azimut_forecast(3:N),'c', ...
 84      3:N,sigma_beta*ones(1, N-2), 'black', 'LineWidth', 1.2);
 85 grid on; grid minor
 86 legend('True filtration error','True extrapolation error','$\sigma_\beta$', 'FontSize', 30, 'interpreter', 'latex');
 87 %title('b) Errors of azimuth', 'FontSize', 20);
 88 xlabel('Step', 'FontSize', 30)
 89 ylabel('Errors', 'FontSize', 30)
 90
 91 %% Point 12 (Building of plot of dependence of coordinate x on azimuth b)
 92
 93 figure(4)
 94 plot(Mean_azimuth(1,:), Mean_Z_x(1,:),'c',TruePolar(2,:), TrueCart(1,:), 'k--', 'LineWidth', 1.2);
 95 grid on; grid minor
 96 legend('x = f($\beta$) filtered values', 'x = g($\beta$) true data', 'FontSize', 30, 'interpreter', 'latex');
 97 %title('Dependance of coordinate x on azimuth $\beta$', 'FontSize', 30, 'interpreter', 'latex');
 98 xlabel('Azimuth', 'FontSize', 30)
 99 ylabel('Coordinate x', 'FontSize', 30)
100
```

```
101 %% Point 12 (Building of plot of dinamics of condotion number)
102
103 figure(5)
104 plot(1:N, Final_CN,'r', 'LineWidth', 1.2);
105 grid on; grid minor
106 legend('Condition number', 'FontSize', 30);
107 %title('Dinamics of condition number', 'FontSize', 20);
108 xlabel('Step', 'FontSize', 30)
109 ylabel('Condition number', 'FontSize', 30)
110
111 %% Point 13 (Building of plot of dinamics of filter gain)
112 figure()
113 plot(1:N, K_matr_values(66,:),'g', 'LineWidth', 1.2);
114 grid on; grid minor
115 legend('Filter gain','FontSize', 20);
116 %title('Dinamics of filter gain', 'FontSize', 20);
117 xlabel('Step','FontSize', 30)
118 ylabel('K','FontSize', 30)
119
120 %% Point 14 (Quite close distance from an observer)
121 InititalState = [3500/sqrt(2); -50; 3500/sqrt(2); -45];
122 for i=1:M
123     %Generation of deterministic trajectory and its measurements
124     [TruePolar, TrueCart, Z_c, Z_p] = ...
125         Generation_true_determ(N, T, sigma_D, sigma_beta, InititalState);
126     %Applying or Kalman filter for measurements
127     [Z_filtered, P, P_pred, ...
128         range_fe, azimuth_fe, CondMatr, K_matr] = ...
129         Kalman(Z_c, Z_p, T, sigma_D, sigma_beta);
130
131     Err_range_filtered(i,:)  = (TruePolar(1,:) - range_fe(1,:)).^2;
132     Err_range_forecast(i,:)  = (TruePolar(1,:) - range_fe(2,:)).^2;
133     Err_azimut_filtered(i,:) = (TruePolar(2,:) - azimuth_fe(1,:)).^2;
134     Err_azimut_forecast(i,:) = (TruePolar(2,:) - azimuth_fe(2,:)).^2;
135
136     Condition_nums(i,:) = CondMatr;
137     K_matr_values(i,:) = K_matr(1,1,:);
138
139     Mean_Z_x = Mean_Z_x + Z_filtered(1,2:N+1);
140     Mean_azimuth = Mean_azimuth + azimuth_fe(1,:);
141     Mean_range = Mean_range + range_fe(1,:);
142 end
143 Mean_Z_x = Mean_Z_x/M;
144 Mean_azimuth = Mean_azimuth/M;
145 Mean_range = Mean_range/M;
146
147 FinalErr_range_filtered  = sqrt(1/(M-1)*sum(Err_range_filtered));
148 FinalErr_range_forecast  = sqrt(1/(M-1)*sum(Err_range_forecast));
149 FinalErr_azimut_filtered = sqrt(1/(M-1)*sum(Err_azimut_filtered));
150 FinalErr_azimut_forecast = sqrt(1/(M-1)*sum(Err_azimut_forecast));
151 Final_CN = sum(Condition_nums)/M;
152 Final_K_matr = sum(K_matr_values)/M;
153
```

```
154 % Generated motion in polar coordinate system (Quite close)
155 figure(7)
156 polarplot(TruePolar(2,:), TruePolar(1,:),'k', 'LineWidth', 1.2)
157 grid on; grid minor
158 legend('True motion', 'FontSize', 20);
159 %title('Object moves uniformly (Quite close)');
160
161 %% Point 15 (Errors of extrapolation and filtration estimates of (Quite close))
162 figure(8)
163 % subplot(1,2,1);
164 plot(3:N, FinalErr_range_filtered(3:N),'m',...
165      3:N, FinalErr_range_forecast(3:N),'c',...
166      3:N, sigma_D*ones(1, N-2), 'black', 'LineWidth', 1.2);
167 grid on; grid minor
168 legend('True filtration error','True extrapolation error','$\sigma_D$',↙
'FontSize', 30, 'interpreter', 'latex');
169 %title('a) Errors of range (Quite close)');
170 xlabel('Step')
171 ylabel('Errors')
172
173 figure(9)
174 plot(3:N, FinalErr_azimut_filtered(3:N),'m',...
175      3:N, FinalErr_azimut_forecast(3:N),'c',...
176      3:N, sigma_beta*ones(1, N-2), 'black');
177 grid on; grid minor
178 legend('True filtration error','True extrapolation error','$\sigma_\beta$',↙
'FontSize', 30, 'interpreter', 'latex');
179 %title('b) Errors of azimuth (Quite close)');
180 xlabel('Step')
181 ylabel('Errors')
182
183 %% Point 16 (Dependence of coordinate x on azimuth b (Quite close))
184
185 figure(10)
186 plot(Mean_azimuth(1,:), Mean_Z_x(1,:),'m',TruePolar(2,:), TrueCart(1,:), 'c--',↙
'LineWidth', 1.2);
187 grid on; grid minor
188 legend('x = f($\beta$) filtered values', 'x = g($\beta$) true data','FontSize',↙
30, 'interpreter', 'latex');
189 %title('Dependence of coordinate x on azimuth $\beta$ (Quite↙
close)','FontSize', 30, 'interpreter', 'latex');
190 xlabel('Azimuth','FontSize', 30)
191 ylabel('Coordinate x','FontSize', 30)
192
193 %% Point 17 (Building of plot of dinamics of condotion number (Quite close))
194
195 figure(11)
196 plot(1:N,Final_CN,'r', 'LineWidth', 1.2);
197 grid on; grid minor
198 legend('Condition number','FontSize', 30);
199 %title('Dinamics of condition number (Quite close)','FontSize', 30);
200 xlabel('Step','FontSize', 30)
201 ylabel('Condition number','FontSize', 30)
```

```
202
203 %% Point 19 quite close distance from an observer and other values of variances
204 sigma_D=50; %variance of range noise of measurements
205 sigma_beta=0.0015; %variance of azimuth noise of measurements
206 for i=1:M
207     %Generation of deterministic trajectory and its measurements
208     [TruePolar,TrueCart,Z_c,Z_p] = ...
209         Generation_true_determ(N,T,sigma_D,sigma_beta,InititalState);
210     %Applying or Kalman filter for measurements
211     [Z_filtered, P, P_pred, ...
212         range_fe, azimuth_fe, CondMatr, K_matr] = ...
213         Kalman(Z_c,Z_p,T,sigma_D,sigma_beta);
214
215     Err_range_filtered(i,:) = (TruePolar(1,:) - range_fe(1,:)).^2;
216     Err_range_forecast(i,:) = (TruePolar(1,:) - range_fe(2,:)).^2;
217     Err_azimut_filtered(i,:)= (TruePolar(2,:) - azimuth_fe(1,:)).^2;
218     Err_azimut_forecast(i,:)= (TruePolar(2,:) - azimuth_fe(2,:)).^2;
219
220     Condition_nums(i,:) = CondMatr;
221     K_matr_values(i,:) = K_matr(1,1,:);
222
223     Mean_Z_x = Mean_Z_x + Z_filtered(1,2:N+1);
224     Mean_azimuth = Mean_azimuth + azimuth_fe(1,:);
225 end
226 Mean_Z_x = Mean_Z_x/M;
227 Mean_azimuth = Mean_azimuth/M;
228
229 FinalErr_range_filtered =sqrt(1/(M-1)*sum(Err_range_filtered));
230 FinalErr_range_forecast =sqrt(1/(M-1)*sum(Err_range_forecast));
231 FinalErr_azimut_filtered = sqrt(1/(M-1)*sum(Err_azimut_filtered));
232 FinalErr_azimut_forecast = sqrt(1/(M-1)*sum(Err_azimut_forecast));
233 Final_CN=sum(Condition_nums)/M;
234 Final_K_matr=sum(K_matr_values)/M;
235
236 % Generated motion in polar coordinate system (Quite close, other variances)
237 figure(12)
238 polarplot(TruePolar(2,:),TruePolar(1,:), 'LineWidth', 1.5)
239 grid on; grid minor
240 legend('True motion','FontSize',30);
241 %title('Object moves uniformly (Quite close, other variances)','FontSize', 30);
242
243 %Errors of extrapolation and filtration estimates of (Quite close, other↙
variances)
244 figure(13)
245 plot(3:N,FinalErr_range_filtered(3:N),'m',...
246      3:N,FinalErr_range_forecast(3:N),'c',...
247      3:N,sigma_D*ones(1,N-2), 'black', 'LineWidth', 1.2);
248 grid on; grid minor
249 legend('True filtration error','True extrapolation↙
error','$\sigma_D$','FontSize', 30, 'interpreter', 'latex');
250 %title('a) Errors of range (Quite close, other variances)','FontSize', 20);
251 xlabel('Step','FontSize', 20)
252 ylabel('Errors','FontSize', 20)
```

```

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;
```

```
299         V_x(i) = V_x(i - 1);
300         Y(i) = Y(i - 1) + V_y(i - 1)*T;
301         V_y(i) = V_y(i - 1);
302     end
303     True_Cartesian = [X; V_x; Y; V_y];
304     %Generation of true values of range D and aimuth b
305     D = sqrt(X.^2 + Y.^2);
306     b = atan(X./Y);
307     True_polar = [D; b];
308     %Generation of measurements
309     D_m = zeros(1, N); %array of measurements of range
310     b_m = zeros(1, N); %array of measurements of azimuth
311     x_m = zeros(1, N); %array of pseudo-measurements of x
312     y_m = zeros(1, N); %array of pseudo-measurements of y
313     Z_c = zeros(2, N); %array of pseudo-measurements in Cartesian coordinates
314     Z_p = zeros(2, N); %array of measurements in polar coordinates
315     for i = 1:N
316         D_m(i) = D(i) + randn*sigma_D;
317         b_m(i) = b(i) + randn*sigma_b;
318
319         x_m(i) = D_m(i)*sin(b_m(i));
320         y_m(i) = D_m(i)*cos(b_m(i));
321
322         Z_p(:,i) = [D_m(i); b_m(i)];
323         Z_c(:,i) = [x_m(i); y_m(i)];
324     end
325 end
326
327 function [Z_filtered, P, P_pred, range_fe, azimuth_fe, CondMatr, K] = Kalman↙
(Z_cart,...
328     Z_polar, T, sigma_D, sigma_beta)
329
330     N = length(Z_cart);
331     Z_filtered = zeros(4, N + 1); %Filtered data
332     Z_forecast = zeros(4, N);     %Forecast data
333
334     P = zeros(4, 4, N + 1);       %Filtration error covariance matrix
335     P_pred = zeros(4, 4, N + 1);  %Prediction error covariance matrix
336     Fi = [1 T 0 0; 0 1 0 0; 0 0 1 T; 0 0 0 1];
337     H = [1 0 0 0; 0 0 1 0];
338
339     Z_filtered(:, 1) = [40000; -20; 40000; -20]; %Initian state vector
340     P(:, :, 1) = [10^10 0 0 0; 0 10^10 0 0; 0 0 10^10 0; 0 0 0 10^10];
341     R = zeros(2,2);
342     K = zeros(4, 2, N);
343     range_fe = zeros(2,N);   %array of filtered and extrapolated range
344     azimuth_fe = zeros(2,N); %array of filtered and extrapolated azimuth
345     CondMatr = zeros(1,N);
346
347     for i = 2:N + 1
348         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;
349         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;
350         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);
351         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);
352         %Prediction part
353         Z_forecast(:,i - 1) =  Fi * Z_filtered(:,i - 1);
354         P_pred(:,:,i-1) = Fi * P(:,:,i - 1) * Fi';
355         %Filtrarion part
356         K(:,:,i - 1) = P_pred(:,:,i - 1)*(H') * (H*P_pred(:,:,i - 1)*H' + R)^↙
(-1);
357         P(:, :, i) = (eye(4) - K(:,:,i - 1)*H)*P_pred(:,:,i - 1);
358         Z_filtered(:,i) = Z_forecast(:,i - 1) + K(:,:,i - 1)*(Z_cart(:,i - 1) -↙
H*Z_forecast(:,i - 1));
359
360         range_fe(1,i - 1) = sqrt( Z_filtered(1,i)^2 + Z_filtered(3,i)^2 );
361         range_fe(2,i - 1) = sqrt( Z_forecast(1,i - 1)^2 + Z_forecast(3,i - 1)↙
^2);
362
363         azimuth_fe(1,i - 1) = atan(Z_filtered(1,i)/Z_filtered(3,i));
364         azimuth_fe(2,i - 1) = atan(Z_forecast(1,i - 1)/Z_forecast(3,i - 1));
365
366         if (sigma_D^2) > (Z_polar(1,i - 1)^2*sigma_beta^2)
367             CondMatr(i - 1) = (sigma_D^2)/(Z_polar(1,i - 1)^2*sigma_beta^2);
368         else
369             CondMatr(i - 1) = (Z_polar(1,i - 1)^2*sigma_beta^2)/(sigma_D^2);
370         end
371     end
372 end
373
374 function [X, P, D_K, b_K, condition_num, K] = Kalman_filter_determ(Z_c, Z_p,...
375     T, sigma_D, sigma_beta)
376
377     N = length(Z_c);
378     Fi = [1 T 0 0; 0 1 0 0; 0 0 1 T; 0 0 0 1]; %transition matrix
379     H = [1 0 0 0; 0 0 1 0]; %observation matrix
380
381     %Kalman filter development
382     %Creating of arrays
383     X = zeros(4,2,N+1);    %filtered and smoothed data
384     P = zeros(4,8,N+1);    %filtration and prediction error covariance matrix
385     K = zeros(4,2,N);      %filter gain
386     D_K = zeros(2,N);      %array of filtered and extrapolated range
387     b_K = zeros(2,N);      %array of filtered and extrapolated azimuth
388     lambda1 = zeros(1,N); %array of first eigenvalues
389     lambda2 = zeros(1,N); %array of second eigenvalues
390     condition_num = zeros(1,N); %array of condition numbers
391
392     %Initial conditions
393     X(:,:,1)=[40000,0; -20,0; 40000,0; -20,0];
394     P(:,1:4,1)=[10^10 0 0 0; 0 10^10 0 0; 0 0 10^10 0; 0 0 0 10^10];
395     R = zeros(2, 2);
396     for i=2:N+1 %0 1 2 3 .... N N+1
```

```
397         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;
398         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;
399         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);
400         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);
401         %Prediction of state vector at time i using i-1 measurements
402         X(:,2,i - 1) = Fi*X(:,1,i-1);
403         %Prediction error covariance matrix
404         P(:,5:8,i - 1) = Fi*P(:,1:4,i-1)*Fi.';
405         %Filter gain, weight of residual
406         K(:,:,i-1) = P(:,5:8,i-1)*H.'*(H*P(:,5:8,i-1)*H.'+R)^(-1);
407         %Improved estimate by incorporating a new measurement
408         X(:,1,i) = X(:,2,i-1) + K(:,:,i-1)*(Z_c(i-1) - H*X(:,2,i-1));
409         %Filtration error covariance matrix
410         P(:,1:4,i) = (eye(4) - K(:,:,i-1)*H)*P(:,5:8,i-1);
411         %Evaluation of predicted and extrapolated range and azimuth
412         D_K(1,i-1) = sqrt(X(1,1,i)^2 + X(3,1,i)^2);
413         D_K(2,i-1) = sqrt(X(1,2,i-1)^2 + X(3,2,i-1)^2);
414         b_K(1,i-1) = atan(X(1,1,i)/X(3,1,i));
415         b_K(2,i-1) = atan(X(1,2,i-1)/X(3,2,i-1));
416         lambda1(i-1) = sigma_D^2; %const 400
417         lambda2(i-1) = Z_p(1,i-1)^2*sigma_beta^2; %descrise 10^4
418         if lambda1(i-1)>lambda2(i-1)
419             condition_num(i-1) = lambda1(i-1)/lambda2(i-1);
420         else
421             condition_num(i-1) = lambda2(i-1)/lambda1(i-1);
422         end
423     end
424 end
425
```