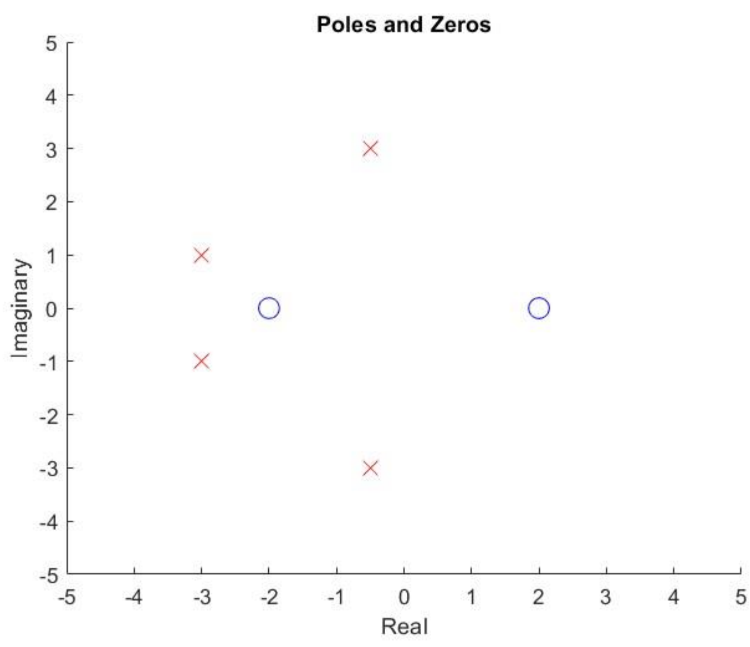
Θεωρία Εκτίμησης και Στοχαστικός Έλεγχος

ΚΑΤΗΓΟΡΙΑ 2

Όνομα	Τόσκα Αλέξανδρος
ΑΜ	1066625
Έτος	5^o
Τομέας	Τεχνολογία της Πληροφορίας

Μέρος Α)

Επιλέγουμε 4 πόλους και 2 μηδενικά όπως φαίνονται παρακάτω:



Οι δύο κυρίαρχοι, πιο αργοί και πιο απομακρυσμένοι είναι οι:

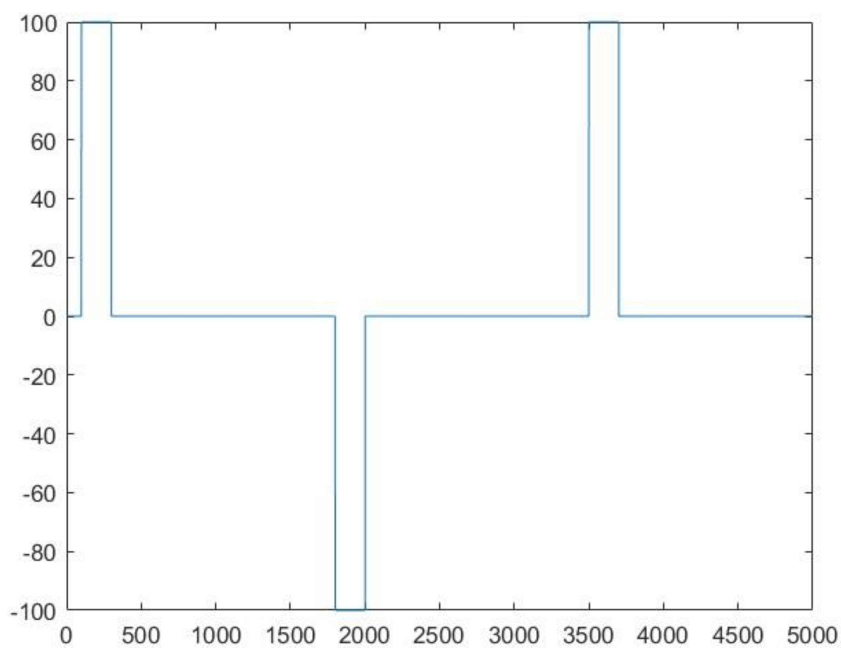
-0.5-3i, **-0.5+3i**

Το σύστημα αυτό έχει 4 διαστάσεις (x = [x1 x2 x3 x4]) και είναι της μορφής:

$x' = Ax + Bu$

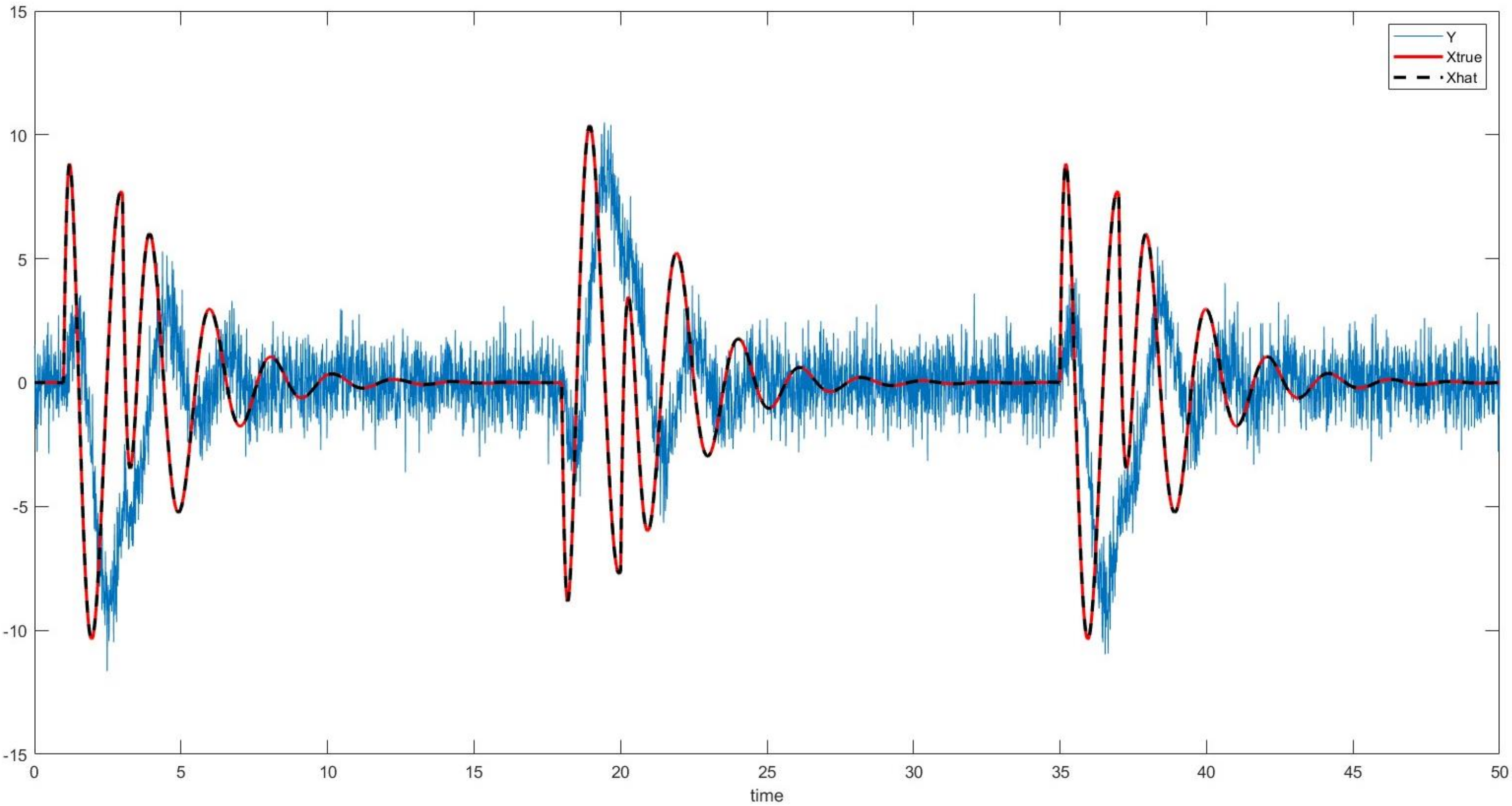
$y = Cx + Du$

Η είσοδος που βάζουμε στο σύστημα είναι ο παρακάτω παλμός:



Η υλοποίηση του φίλτρου Kalman στη MATLAB θα γίνει με την εντολή lqe() και τους αντίστοιχους πίνακες ως εισόδους.

Μετρήσεις - Αποτελέσματα:



Εδώ βλέπουμε 3 γραφικές:

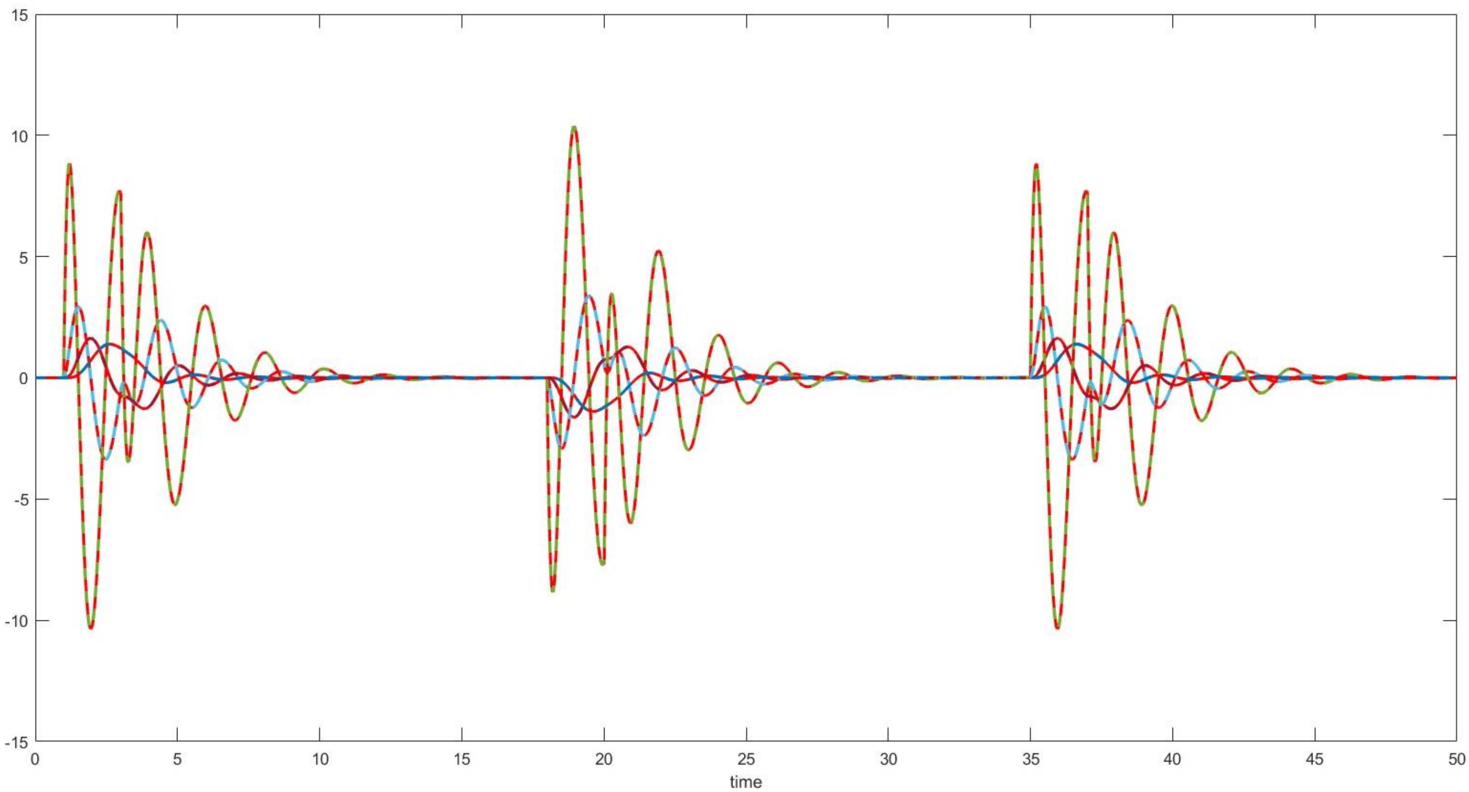
Η μπλε -> οι μετρήσεις Υ που κάνουμε.

Η κόκκινη -> οι πραγματικές τιμές Xtrue (1 διάστασης) χωρίς θόρυβο από το Full State System

Η διακεκομένη μαύρη -> Είναι η εκτίμηση Xhat (1 διάστασης) από το φίλτρο Kalman.

Βλέπουμε ότι παρόλο που οι μετρήσεις είναι αρκετά θορυβώδεις το φίλτρο εκτιμά με μεγάλη ακρίβεια τις καταστάσεις του συστήματος.

Παρακάτω φαίνονται και οι 4 καταστάσεις του συστήματος



Και στις 4 γράμμες οι εκτιμήσεις του φίλτρου (διακεκομένες γραμμές) σχεδόν ταυτίζονται με τις πραγματικές τιμές.

Την απόδοση του αλγόριθμου θα την υπολογίσουμε με βάση το μέσο τετραγωνικό σφάλμα (MSE) των X_true και X_hat.

MSE = 0.0011

Τη συμμεταβλητότητα σφάλματος θα την υπολογίσουμε ως το άθροισμα των διαγώνιων στοιχείων του πίνακα P (που είναι το μέτρο της συμμεταβλητότητας του σφάλματος εκτίμησης)

Error_Covariance = 0.1887

Το κέρδος του αλγόριθμου Kalman είναι ένας πίνακας ο οποίος αποφασίζει πόσο βάρος να δώσει σε κάθε μέτρηση για την εκτίμηση της κατάστασης του. Συγκεκριμένα είναι το βέλτιστο κέρδος όπου ελαχιστοποιεί το (MSE) μεταξυ X_true και X_hat.

Kf = [-0.0013

0.0296

0.0014

-0.0064]

Μέρος Β)

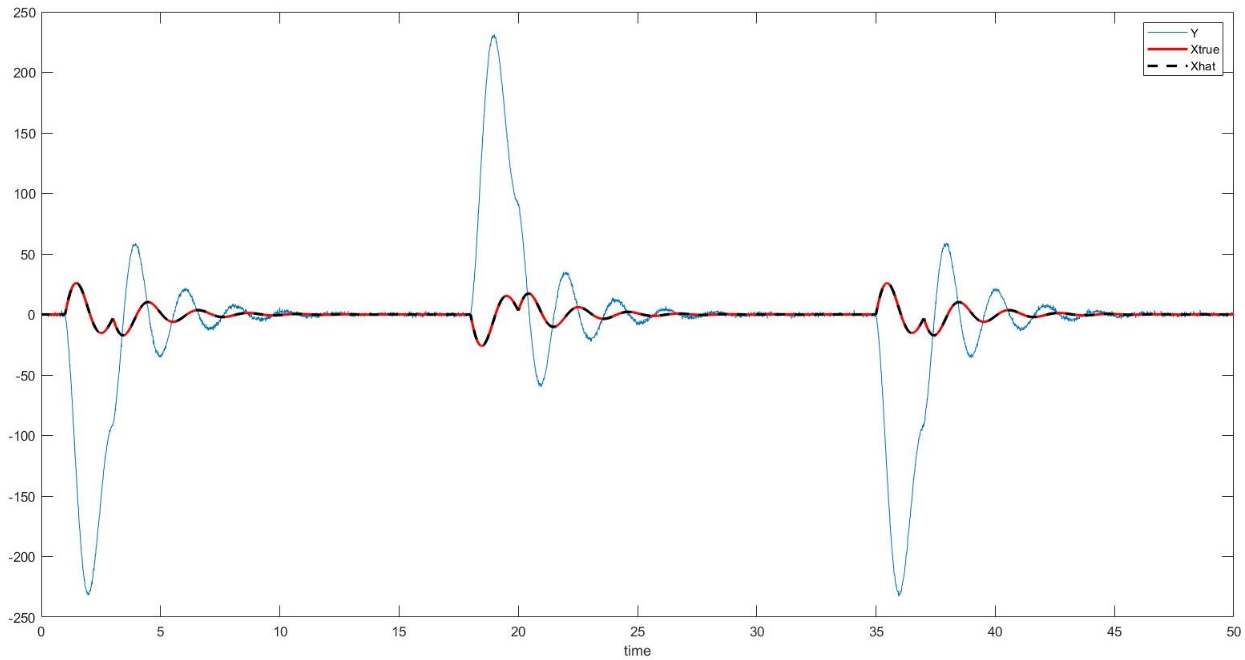
Εδώ θα κρατήσουμε τους πόλους p1 = -0.5 -3i,p2 = -0.5 +3i.

Μετά από κάποιες δοκιμές καταλήξαμε στα:

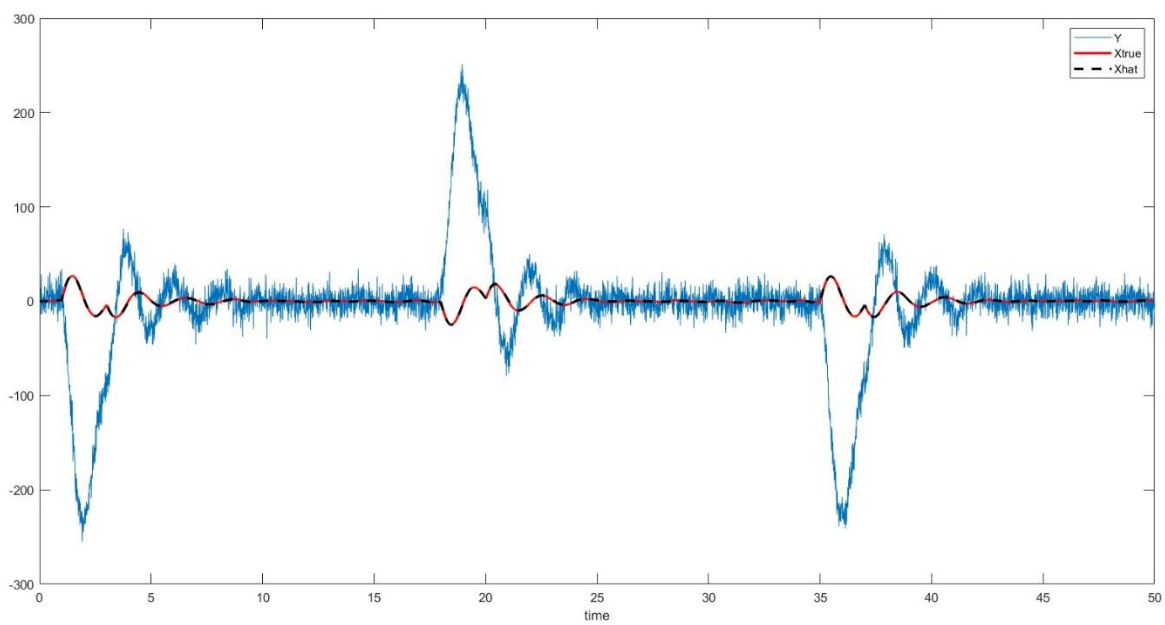
Vd = [1.5 0; 0 1.5], Vn = 10

Στο Α ερώτημα είχαμε (Vd=[0.1 0 0 0; 0 0.1 0 0; 0 0 0.1 0; 0 0 0 0.1], Vn=1)

Το δύο δαστάσεων σύστημα στο Β με τον θόρυβο και πίνακα συμμεταβλητότητας του ερωτήματος Α έχει ως αποτέλεσμα:

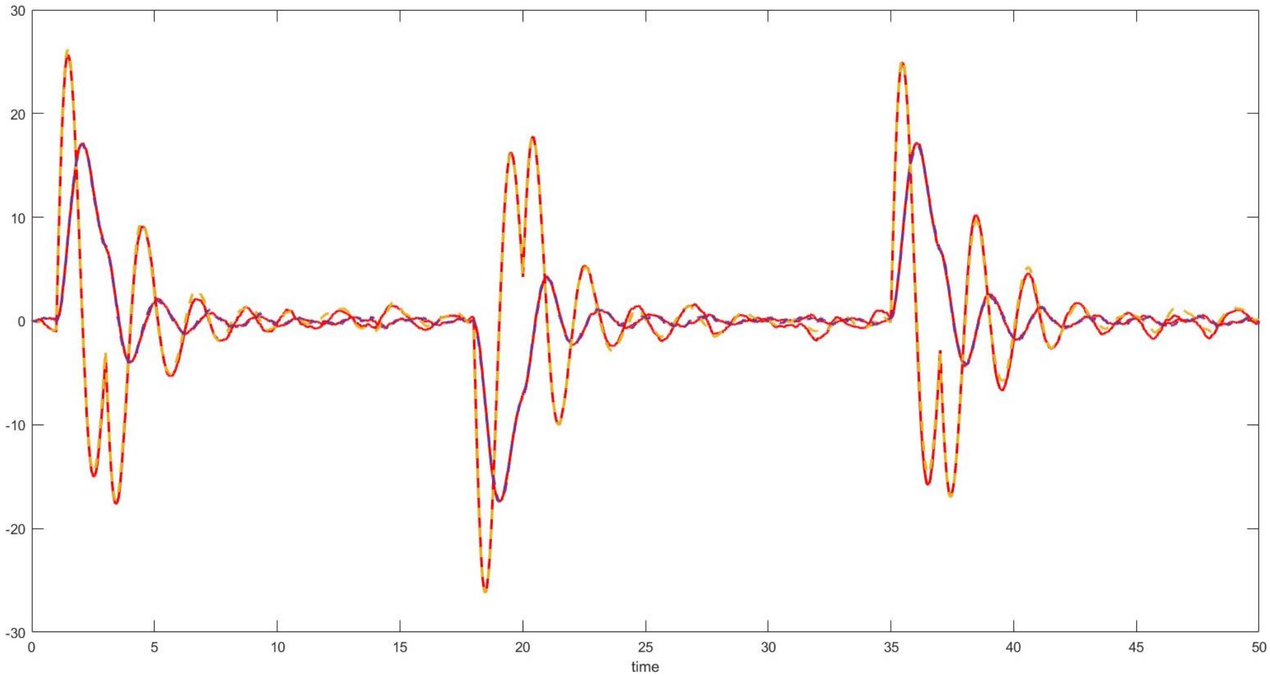


Το ίδιο σύστημα με τις νέες τιμές Vd, Vn έχει ως αποτέλεσμα:

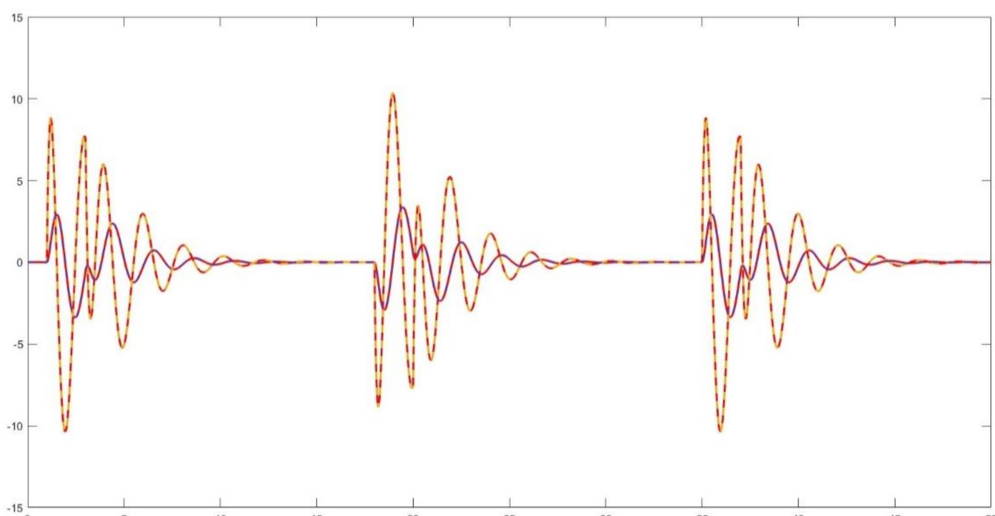


Φαίνεται πως το σύστημα είναι πιο κοντά σε αυτό του Α ερωτήματος.

Οι δύο καταστάσεις μαζί με τις εκτιμήσεις του φίλτρου:



Και παρακάτω είναι μονο οι 2 πρώτες καταστάσεις του συστήματος του ερωτήματος Α:



Βλέπουμε πως το σύστημα στο ερώτημα Β πλησιάζει το Α αλλά κάνει μεγαλύτερα λάθη στις εκτιμήσεις του.

MSE = 0.1003

Error_Covariance = 4.4346

Kf = [-0.0652; -0.5355]

Το Α ερώτημα μόνο για τις 2 πρώτες καταστάσεις είχε:

MSE = 0.0022, Error_Cov = 0.1850 και Kf=[-0.0013; 0.0296]

Βλέπουμε ότι και το MSE και το Error Covariance είναι μεγαλύτερα από αυτά του Α ερωτήματος δείχνοντας ότι το φίλτρο έχει χειρότερη απόδοση.

Μέρος Γ)

Το σήμα καινοτομίας για το φίλτρο ορίζεται ως

innovation_signal = y(measured) – y_hat

οπου y_hat = C * X_hat.

Για να ελέγξουμε αν το σήμα καινοτομίας είναι κοντά σε λευκό θόρυβο θα χρησιμοποιήσουμε το Ljung-Box test, το οποίο είναι ένα στατιστικό τεστ όπου ελέγχει ένα σύνολο αυτοσυσχετίσεων του σήματος αν είναι διάφορο του μηδενός. Στο matlab η συνάρτηση lbqtest() επιστρέφει μία τιμή h η οποία όταν είναι 0 τότε το σήμα είναι λευκός θόρυβος και όταν είναι 1 δεν είναι.

- Μετά από αρκετές δοκιμές το σήμα το ερωτήματος Α βγαίνει h=0 κάθε φορά άρα το σήμα καινοτομίας είναι λευκός θόρυβος.
- Ενώ για το ερώτημα Β μετά από 10 δοκιμές, 8/10 -> h=0 και 2/10 -> h=1, άρα συμπεραίνουμε ότι το σήμα καινοτομίας δεν είναι καθαρά λευκός θόρυβος αλλά έχει μία μικρή αυτοσυσχέτιση.

Έτσι φαίνεται ότι το σήμα του ερωτήματος Α είναι πιο κοντά από το σήμα του Β στον ιδανικό λευκό θόρυβο.

Για να ελέξουμε αν το σήμα καινοτομίας είναι κοντά στην κανονική κατανομή θα χρησιμοποιήσουμε το Jarque-Bera test. Το τεστ αυτό βασίζεται στο skewness και το kurtosis. Στο matlab η συνάρτηση jbtest() που πραγματοποιεί τον έλεγχο επιστρέφει μία h τιμή η οποία όταν είναι 0 το σήμα ανήκει στην κανονική κατανομή και όταν είναι 1 δεν ανήκει.

- Μετά από αρκετές δοκιμές και τα δύο σήματα καινοτομίας έχουν ως αποτέλεσμα h=0 άρα και τα δύο σήματα ανήκουν στη κανονική κατανομή.

Matlab – Code

First file: Περιέχει το Μέρος Α και στο τέλος έχει κάποια δεδόμενα που χρησιμοποιούνται στο Μέρος Β,Γ και υπάρχουν σχόλια: *%% B, %% C*

```
clc;
clear all;
close all;
%
p1 = -0.5 + 3i; % slower
p2 = -0.5 - 3i; % slower
p3 = -3 + 1i;
p4 = -3 - 1i;
z1 = -2 + 0i;
z2 = 2 + 0i;

poles = [p1, p2, p3, p4];
zeros_ = [z1, z2];
m = 4; % number of poles

% Plot the poles and zeros
figure
hold on
plot(real(p1),imag(p1),'x','MarkerSize',10,'Color','red')
plot(real(p2),imag(p2),'x','MarkerSize',10,'Color','red')
plot(real(p3),imag(p3),'x','MarkerSize',10,'Color','red')
plot(real(p4),imag(p4),'x','MarkerSize',10,'Color','red')
plot(real(z1),imag(z1),'o','MarkerSize',10,'Color','blue')
plot(real(z2),imag(z2),'o','MarkerSize',10,'Color','blue')
axis([-5 5 -5 5])
xlabel('Real')
ylabel('Imaginary')
title('Poles and Zeros')
%

% Create the transfer function object from the poles and zeros
gain=1;
sys = zpk(zeros_, poles, gain);
% get num and den from sys
[num, den] = tfdata(sys);
num = cell2mat(num);
den = cell2mat(den);

% convert to state space model
[A, B, C, D] = tf2ss(num, den);

% Augment the system with disturbances and noise
Vd = 0.1* eye(m); % disturbance covariance
Vn = 1; % noise covariance

BF = [B Vd 0*B]; % augment inputs with disturbances and noise

sysC = ss(A, BF, C, [0 0 0 0 0 Vn]); % build state space system
```

```
% system with full state output, disturbances, no noise
sysFullOutput = ss(A, BF, eye(m), zeros(m,size(BF,2)));

% build kalman filter
[Kf, P, E] = lqe(A,Vd,C,Vd,Vn); % design Kalman filter

sysKf = ss(A-Kf*C, [B Kf], eye(m), 0*[B Kf]); % kalman filter

% Estimate linearized system
T = 0.01;
t = 0:T:50;

uDIST = randn(m, size(t,2));
uNOISE = randn(size(t));

% impulse
u = 0*t;
u(100:300) = 100;
u(1800:2000) = -100;
u(3500:3700) = 100;
uAUG = [u; Vd*Vd*uDIST; uNOISE];

[y,t] = lsim(sysC, uAUG, t);
figure
plot(t,y);
%
[xtrue,t] = lsim(sysFullOutput, uAUG,t);
hold on
plot(t,xtrue(:,1),'r','LineWidth',2.0)

% Kalman filter estimate
[x_hat,t] = lsim(sysKf, [u; y'],t);
plot(t,x_hat(:,1),'k--', 'LineWidth', 2.0)
xlabel('time')
legend('Y', 'Xtrue', 'Xhat')

%
figure
plot(t,xtrue,'r',t, x_hat,'--','LineWidth',2)
xlabel('time')

% calculate MSE of X_hat
mse = immse(xtrue, x_hat)

% calculate Error Covariance
err_cov = trace(P)

% Kalman Gain
Kf

%% For B question
figure
plot(t,xtrue(:,1:2),'r',t, x_hat(:,1:2),'--','LineWidth',2)
title('First 2 states used for B question')
% Calculate MSE -> performance
```

```
mse_B = immse(xtrue(:,1:2),x_hat(:,1:2))
err_cov = trace(P(1:2,1:2))

%% For C question

% innovation signal
y_hat = (C*x_hat')';
innov_signal = y - y_hat;

% max lag -> 1/4-1/3 of length t=5001
max_lag = 1250; % ~ 50001/4;

% white noise
[h1, p1, q1, stats1] = lbqtest(innov_signal, 'Lags', max_lag)


% normal distribution
[h2, p2, jbstat2, critval2] = jbtest(innov_signal)
```

Second File: Περιέχει το Μέρος Β και στο τέλος (σχόλια: **%% C**) περιέχει κάποια δεδομένα για το Μέρος Γ.

```
clc;
clear all;
close all;
%
p1 = -0.5 + 3i; % slower
p2 = -0.5 - 3i; % slower
%p3 = -3 + 1i;
%p4 = -3 - 1i;
z1 = -2 + 0i;
z2 = 2 + 0i;

poles = [p1, p2];
zeros_ = [z1, z2];
m = 2; % number of poles

% Create the transfer function object from the poles and zeros
gain=1;
sys = zpk(zeros_, poles, gain);
% get num and den from sys
[num, den] = tfdata(sys);
num = cell2mat(num);
den = cell2mat(den);

% convert to state space model
[A, B, C, D] = tf2ss(num, den);
```

```
% 1.5 - 10
% Augment the system with disturbances and noise
Vd = 1.5* eye(m); % disturbance covariance
Vn = 10; % noise covariance

BF = [B Vd 0*B]; % augment inputs with disturbances and noise

sysC = ss(A, BF, C, [0 0 0 Vn]); % build state space system

% system with full state output, disturbances, no noise
sysFullOutput = ss(A, BF, eye(m), zeros(m,size(BF,2)));

% build kalman filter
[Kf, P, E] = lqe(A,Vd,C,Vd,Vn); % design Kalman filter

sysKf = ss(A-Kf*C, [B Kf], eye(m), 0*[B Kf]); % kalman filter

% Estimate linearized system
T = 0.01;
t = 0:T:50;

uDIST = randn(m, size(t,2));
uNOISE = randn(size(t));

% impulse
u = 0*t;
u(100:300) = 100;
u(1800:2000) = -100;
u(3500:3700) = 100;
uAUG = [u; Vd*Vd*uDIST; uNOISE];

[y,t] = lsim(sysC, uAUG, t);
figure
plot(t,y);
%
[xtrue,t] = lsim(sysFullOutput, uAUG,t);
hold on
plot(t,xtrue(:,1),'r','LineWidth',2.0)

% Kalman filter estimate
[x_hat,t] = lsim(sysKf, [u; y'],t);
plot(t,x_hat(:,1),'k--', 'LineWidth', 2.0)
xlabel('time')
legend('Y', 'Xtrue', 'Xhat')

%
figure
plot(t,xtrue,'r',t, x_hat,'--','LineWidth',2)
xlabel('time')

% Calculate MSE -> performance
mse = immse(xtrue, x_hat)

% calculate Error Covariance
err_cov = trace(P)
```

```
% Kalman Gain
Kf

%% For C question

% innovatiion signal
y_hat = (C*x_hat')';
innov_signal = y - y_hat;

% max lag -> 1/4-1/3 of length t=5001
max_lag = 1250; % ~ 50001/4;

% white noise
[h1, p1, q1, stats1] = lbqtest(innov_signal, 'Lags', max_lag)

% normal distribution
[h2, p2, jbstat2, critval2] = jbtest(innov_signal)
```