Προσομοίωση και Μοντελοποίηση Δυναμικών Συστημάτων

Δεύτερη εργασία

Θεόδωρος Κατζάλης
ΑΕΜ: 9282
katzalis@auth.gr

24/05/2021

Περιεχόμενα

On line εκτίμηση άγνωστων παραμέτρων

Σκοπός της εργαστηριακής άσκησης είναι η on line εκτίμηση άγνωστων παραμέτρων με τη μέθοδο της κλίσης και τη μέθοδο Lyapunov. Θεωρήστε το σύστημα:

$$\dot{x} = -\alpha x + bu, \quad x(0) = 0 \tag{1}$$

'Οπου x είναι η κατάσταση του συστήματος, $u = 5sin(3t)$ είναι η είσοδος και $a = 2, b = 1$ σταθερές παράμετροι τις οποίες θέλουμε να εκτιμήσουμε on line.

Θέμα 1

Σχεδιάστε έναν εκτιμητή πραγματικού χρόνου των άγνωστων παραμέτρων βασισμένο στη μέθοδο κλίσης και προσομοιώστε τη λειτουργία του. Δημιουργείστε τις γραφικές παραστάσεις των x, $\hat{x}$, καθώς και τις εκτιμήσεις των $\hat{\alpha}$, $\hat{b}$ των α, b.

Μαθηματική ανάλυση

Η μέθοδος κλίσης εφαρμόζεται σε γραμμικά παραμετροποιημένο σύστημα, οπότε από την (1) έχουμε:

$$\begin{aligned}
\dot{x} &= -\alpha x + bu && (2)\\
\dot{x} &= \pm\alpha_m - \alpha x + bu && (3)\\
\dot{x} + \alpha_m &= (\alpha_m - \alpha)x + bu && (4)\\
sx + \alpha_m x &= (\alpha_m - a)x + bu && (5)\\
x(s + \alpha_m) &= (\alpha_m - a)x + bu && (6)\\
x &= \frac{1}{s + \alpha_m}[(\alpha_m - \alpha)x + bu] && (7)\\
& && (8)
\end{aligned}$$

Στην ουσία για να γραμμικοποιήσουμε το σύστημα, χρησιμοποιήσαμε ένα ευσταθές φίλτρο $\Lambda(s) = s + \alpha_m$, όπου $\alpha_m > 0$. Οπότε έχουμε:

$$\begin{aligned}
\theta^* &= \begin{bmatrix}\alpha_m - \alpha & b\end{bmatrix}^T && (9)\\
\phi &= \begin{bmatrix}(\frac{1}{s+\alpha_m})x & (\frac{1}{s+\alpha_m})u\end{bmatrix}^T && (10)\\
x &= \theta^{*T}\phi && (11)
\end{aligned}$$

Ορίζουμε σύστημα αναγνώρισης:

$$\hat{x} = \hat{\theta}^T\phi \tag{12}$$

Για να βρούμε την βέλτιστη επιλογή παραμέτρων ορίζουμε ένα μέτρο αξιολόγησης (σφάλμα) και μια συνάρτηση κόστους που θέλουμε να ελαχιστοποιήσουμε. Πιο συγκεκριμένα έχουμε:

$$e = x - \hat{x} = x - \theta^T \phi \tag{13}$$

$$K(\hat{\theta}) = \frac{e^2}{2} = \frac{(x - \hat{\theta}^T \phi)^2}{2} \tag{14}$$

Σύμφωνα με τον ορισμό της μεθόδου της κλίσης έχουμε:

$$\dot{\hat{\theta}} = -\gamma \nabla K \tag{15}$$

$$\nabla K = -(x - \hat{\theta}^T \phi)\phi = -e\phi \tag{16}$$

όπου γ το βήμα του αλγορίθμου $\gamma > 0$. Απο (15) και (16):

$$\dot{\hat{\theta}} = \gamma e \phi, \quad \gamma > 0 \tag{17}$$

Απο (10) και (17) τελικά έχουμε το εξής σύστημα διαφορικών εξισώσεων το οποίο καλούμαστε να λύσουμε για να εκτιμήσουμε τις παραμέτρους:

$$\dot{\phi}_1 = -\alpha_m \phi_1 + x, \quad \phi_1(0) = 0 \tag{18}$$

$$\dot{\phi}_2 = -\alpha_m \phi_2 + u, \quad \phi_2(0) = 0 \tag{19}$$

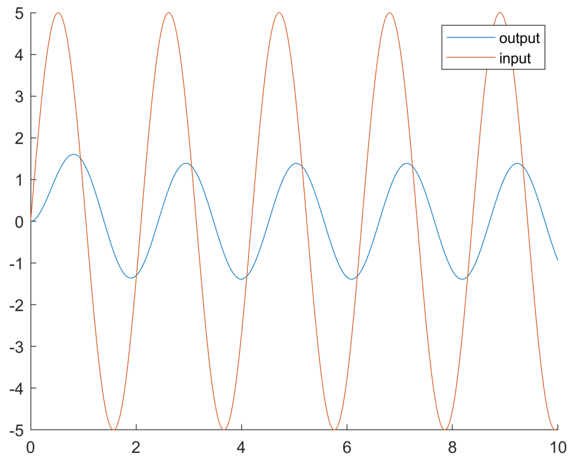
$$\dot{\hat{\theta}}_1 = \gamma e \phi_1, \quad \hat{\theta}_1(0) = 0 \tag{20}$$

$$\dot{\hat{\theta}}_2 = \gamma e \phi_2, \quad \hat{\theta}_2(0) = 0 \tag{21}$$

$$e = x - \hat{x} \tag{22}$$

όπου u και x μετρήσιμα μεγέθη. Η έξοδος του συστήματος είναι γνωστή λύνοντας την απλή διαφορική εξίσωση της σχέσης (1).

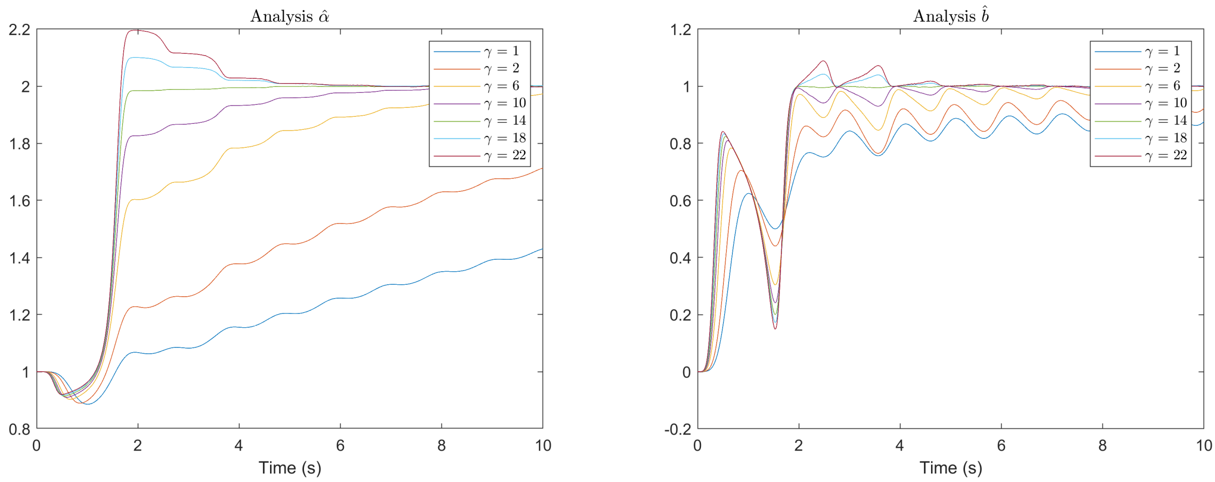
Διαγράμματα



Εικόνα 1: Συσχέτιση εισόδου και εξόδου ($t = 0 : 0.001 : 10$)

Όπως φαίνεται και απο τις εξισώσεις (18) - (22), τα μεγέθη που μπορούν να επηρεάσουν την επίλυση του προβλήματος και μπορούμε να τα ρυθμίσουμε (**tuning**), είναι το γ και το α_m, δηλαδή το βήμα και ο πόλος του φίλτρου. Βέβαια και οι αρχικές συνθήκες του $\hat{\theta}$ και του ϕ επηρεάζουν την λύση του προβλήματος. Για χάριν ευκολίας, θα προτιμήσουμε την τυπική επιλογή των μηδενικών αρχικών συνθηκών.

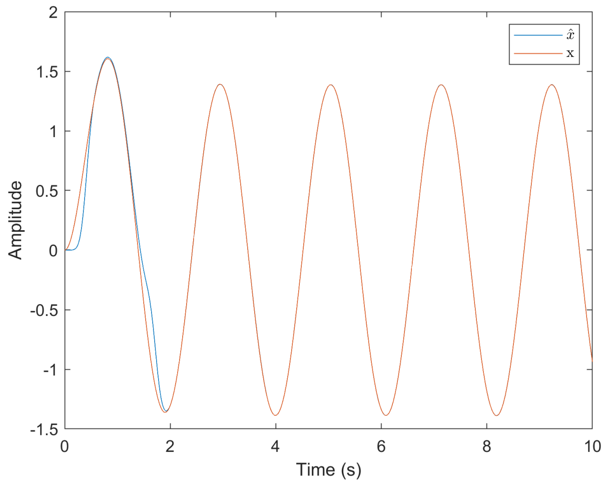
Σχετικά με το α_m, αξίζει να σημειωθεί ότι λόγω της σχέσης $\theta^* = \begin{bmatrix}\alpha_m - a & b\end{bmatrix}^T$ και εξαιτίας των μηδενικών αρχικών συνθηκών του $\hat{\theta}$, η εκτίμηση του α θα ξεκινάει πάντα απο την τιμή α_m. Με άλλα λόγια, το $\hat{\theta}$ θα προσπαθεί να πλησιάζει το θ^* και όσο πιο μεγάλη είναι η τιμή του α_m, τόσο πιο μεγάλο θα είναι το χρονικό διάστημα της προσέγγισης της πραγματικής τιμής, συμπεριλαμβάνοντας ότι το $\hat{\theta}$ έχει μηδενικές αρχικές συνθήκες. Οπότε γνωρίζοντας ότι $\alpha = 2$, για $\alpha_m = 2$ θα έχουμε $\theta_1^* = 0$ και επειδή όπως είπαμε $\hat{\theta}_1 = 0$, συμπεραίνουμε ότι $\alpha_m = 2$ είναι η καλύτερη επιλογή. Φυσικά, σε πραγματικό πρόβλημα δεν γνωρίζουμε τις παραμέτρους, οπότε δεν έχει νόημα η προηγούμενη ανάλυση. Θα έπρεπε με trial and error, να βρίσκαμε για ποια τιμή του πόλου συγκλίνει πιο γρήγορα η εκτίμηση. Στην δικιά μας ανάλυση, χρησιμοποιήσαμε $\boldsymbol{\alpha_m = 1}$. Βέβαια αξίζει να σημειωθεί ότι για λόγους ευστάθειας θα πρέπει $\alpha_m > 0$. Συνεπώς, θα δώσουμε βαρύτητα κυρίως στην επιλογή του γ και έχουμε την ακόλουθη παραμετρική ανάλυση:



(a) $\alpha = 2$ (b) b = 1

Εικόνα 2: Παραμετρική ανάλυση των εκτιμήσεων

Μπορούμε να παρατηρήσουμε ότι η καλύτερη τιμή, σχετικά με το πόσο γρήγορα σταθεροποιείται η εκτίμηση, είναι $\gamma = 14$. Για αυτήν την τιμή έχουμε επίσης το ακόλουθο διάγραμμα:



Εικόνα 3: Σύγκριση εκτίμησης και πραγματικής εξόδου ($t = 0 : 0.001 : 10$)

Από το παραπάνω διάγραμμα παρατηρούμε ότι στην αρχή έχουμε μια μικρή απόκλιση μέχρι να σταθεροποιηθούν οι εκτιμήσεις την χρονική στιγμή $t = 2$.

Στη συνέχεια παραθέτουμε τον κώδικα `Matlab` για όλα τα παραπάνω.

Matlab

simulation.m

```
%% SISO

clc;
clear;
close all;

% the parameters I want to estimate
a = 2; b = 1;

% input
u = @(t) 5*sin(3*t);

% output
start = 0; step = 0.001; finish = 10;
t = start:step:finish;
dx = @(t,x) -a*x + b*u(t);
[t, x] = ode45(dx,t,0);

% convert vector to function of time
x = @(t) x(round((t - start)/step + 1));

% PLOT
figure
hold on
tPlot = 0:0.01:min(10,finish);
plot(tPlot,x(tPlot))
plot(tPlot,u(tPlot))
xlabel('Time (s)'),ylabel('Amplitude'),legend('output','input')
```

```
print('in_out','-dpng','-r300')

% GRADIENT DESCENT

% the root of the stable filter
eig = -1;

gammaRange = [1 2:4:24];
xHat = zeros(numel(t),numel(gammaRange));
aHat = zeros(numel(t),numel(gammaRange));
bHat = zeros(numel(t),numel(gammaRange));
counter = 1;
for i=gammaRange
    % learning rate
    gamma = i;

    [phi,aHatPerGamma,bHatPerGamma] = gradientDescent(x,u,t,eig,gamma);
    phi1 = phi(:,1); phi2 = phi(:,2);

    % convert vectors to function of time
    theta1 = @(t) -eig -aHatPerGamma(round((t - start)/step + 1));
    theta2 = @(t) bHatPerGamma(round((t - start)/step + 1));
    phi1 = @(t) phi1(round((t - start)/step + 1));
    phi2 = @(t) phi2(round((t - start)/step + 1));

    % estimate output
    xHatPerGamma = @(t) theta1(t).*phi1(t) + theta2(t).*phi2(t);

    xHat(:,counter) = xHatPerGamma(t);
    aHat(:,counter) = aHatPerGamma;
    bHat(:,counter) = bHatPerGamma;

    legendInfo{counter} = ['gamma = ' num2str(i)];
    counter = counter + 1;
end

% gamma = 14, index = 5, best choice
figure
plot(t,xHat(:,5),t,x(t)),legend('estimation','real')
xlabel('Time (s)'),ylabel('Amplitude')
print('out_est','-dpng','-r300')

% plot
figure,plot(t,aHat),legend(legendInfo),xlabel('Time (s)')
title('Analysis estimation a')
print('a_param','-dpng','-r300')
figure,plot(t,bHat),legend(legendInfo),xlabel('Time (s)')
title('Analysis estimation b')
print('b_param','-dpng','-r300')
```

gradientDescent.m

```
function [phi,aHat,bHat] = gradientDescent(x,u,t,eig,gamma)
[t, y] = ode45(@(t,y)diffSystem(t,y,u,x,gamma,eig),t,[0 0 0 0]);
phi = [y(:,1) y(:,2)];
aHat = -eig -y(:,3);
bHat = y(:,4);
end

function dy = diffSystem(t,y,u,x,gamma,eig)
phi1 = y(1); phi2 = y(2); theta(1) = y(3); theta(2) = y(4);
```

```
% thetaDot = gamma*error*phi
dPhi1 = eig*phi1 + x(t);
dPhi2 = eig*phi2 + u(t);
error = x(t) - theta(1)*phi1 - theta(2)*phi2;
dtheta(1) = gamma*error*phi1;
dtheta(2) = gamma*error*phi2;

dy = [dPhi1; dPhi2; dtheta(1); dtheta(2)];
end
```

Θέμα 2

Για το (1) να σχεδιαστεί εκτιμητής πραγματικού χρόνου των άγνωστων παραμέτρων βασισμένο στη μέθοδο κλίσης και προσομοιώστε τη λειτουργία του. Δημιουργείστε τις γραφικές παραστάσεις των x,$\hat{x}$, καθώς και τις εκτιμήσεις $\hat{a}$,$\hat{b}$ των a,b.

Μαθηματική ανάλυση

Η μέθοδος **Lyapunov** εφαρμόζεται σε συστήματα τα οποία περιγράφονται απο τις εξισώσεις κατάστασεις του συστήματος, χωρίς να υπάρχει ανάγκη γραμμικοποίησης όπως συμβαίνει στην μέθοδο κλίσης που είδαμε προηγουμένως. Ακόμη ανάλογα το σύστημα αναγνώρισης που χρησιμοποιούμε για να εκτιμήσουμε τις παραμέτρους, υπάρχουν δύο κατηγορίες: η **παράλληλη** (Π) και η **μικτή** (Μ).

Συστήματα αναγνώρισης:

$$(\Pi): \quad \dot{\hat{x}} = -\hat{\theta}_1\hat{x} + \hat{\theta}_2, \quad \hat{x}(0) = \hat{x}_o \tag{1}$$

$$(M): \quad \dot{\hat{x}} = -\hat{\theta}_1 x + \hat{\theta}_2 u + \theta_m(x - \hat{x}) \quad \hat{x}(0) = \hat{x}_o \tag{2}$$

Στη συνέχεια θα χρησιμοποιήσουμε για την μαθηματική μας ανάλυση το σύστημα αναγνώρισης της μικτής κατηγορίας. Αρχικά επιλέγουμε υποψήφια συνάρτηση **Lyapunov**, μέσω της οποίας ο στόχος μας είναι να βρούμε τις διαφορικές εξισώσεις για την σχεδίαση μας και έχουμε:

$$\dot{V} = e\dot{e} + \frac{1}{\gamma_1}\tilde{\theta}_1\dot{\hat{\theta}}_1 + \frac{1}{\gamma_2}\tilde{\theta}_2\dot{\hat{\theta}}_2 \tag{3}$$

$$= \underbrace{-\theta_m e^2}_{1} + \underbrace{\tilde{\theta}_1 ex}_{2} - \underbrace{\tilde{\theta}_2 eu}_{3} + \underbrace{\frac{1}{\gamma_1}\tilde{\theta}_1\dot{\hat{\theta}}_1}_{4} + \underbrace{\frac{1}{\gamma_2}\tilde{\theta}_2\dot{\hat{\theta}}_2}_{5} \tag{4}$$

όπου $\theta_m > 0$ (δικιά μας επιλογή) και $e = x - \hat{x}$. Παρατηρούμε ότι μόνο ο πρώτος όρος της σχέσης (4) έχει γνωστό πρόσημο. Έτσι σχεδιάζω το σύστημα μου εξισώνοντας τον δεύτερο και τον τρίτο όρο, με τον τέταρτο και τον πέμπτο αντίστοιχα για να διώξω τους όρους με αμφίβολο πρόσημο. Έτσι προκύπτουν οι ακόλουθες διαφορικές εξισώσεις για την σχεδίαση μας:

$$\dot{\hat{\theta}}_1 = -\gamma_1 ex, \quad \gamma_1 > 0 \tag{5}$$

$$\dot{\hat{\theta}}_2 = +\gamma_2 eu, \quad \gamma_2 > 0 \tag{6}$$

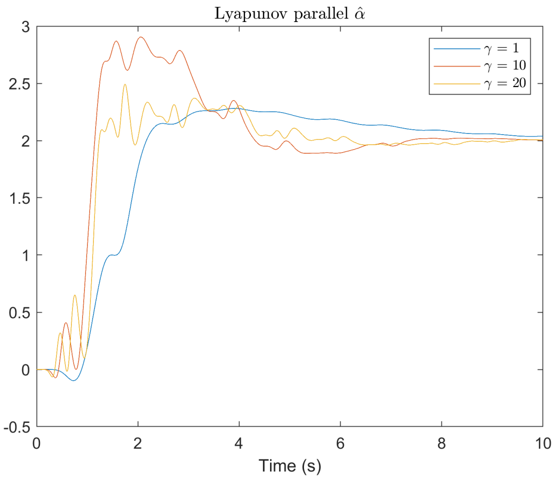
Με παρόμοιο τρόπο προκύπτει ότι για το σύστημα αναγνώρισης (Π):

$$\dot{\hat{\theta}}_1 = -\gamma_1 e \hat{x} \tag{7}$$

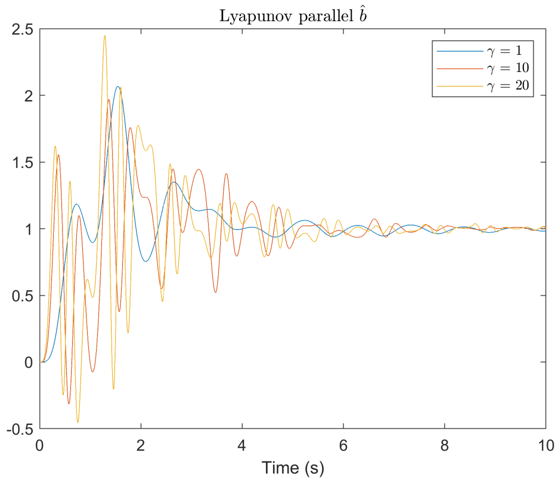
$$\dot{\hat{\theta}}_2 = +\gamma_2 e u \tag{8}$$

Διαγράμματα

Απο την προηγούμενη μαθηματική ανάλυση καταλαβαίνουμε ότι οι παράμετροι με τους οποίους μπορούμε να πειραματιστούμε για να έχουμε καλύτερα αποτελέσματα είναι οι θ_m, γ_1, γ_2. Έτσι λοιπόν με trial and error, δημιουργήσαμε κάποιες παραμετρικές αναλύσεις. Για λόγους ευκολίας θεωρήσαμε $\gamma_1 = \gamma_2$ και $\theta_m = 1$.

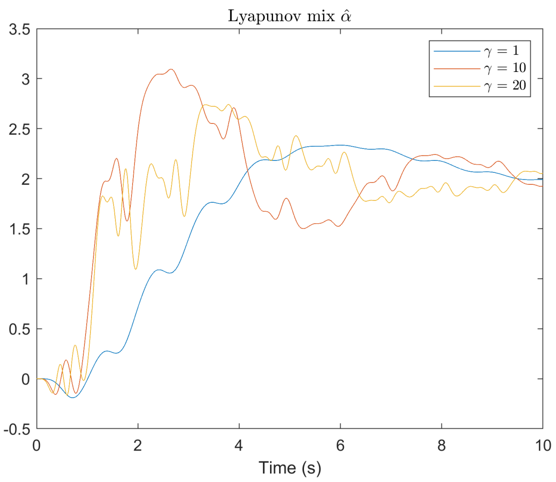


(a) $\alpha = 2$

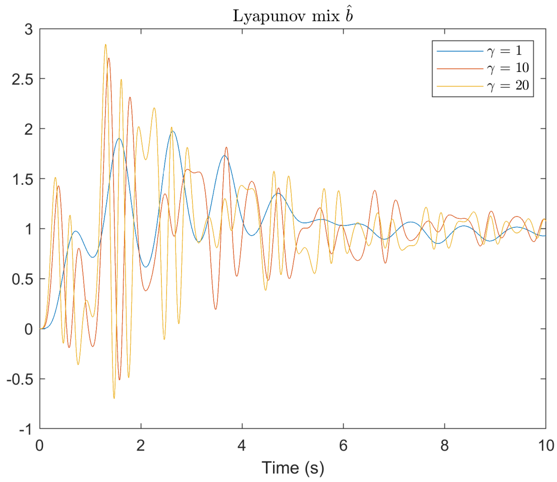


(b) b = 1

Εικόνα 4: Παραμετρική ανάλυση των εκτιμήσεων ως προς γ, Lyapunov παράλληλη μέθοδος



(a) $\alpha = 2$



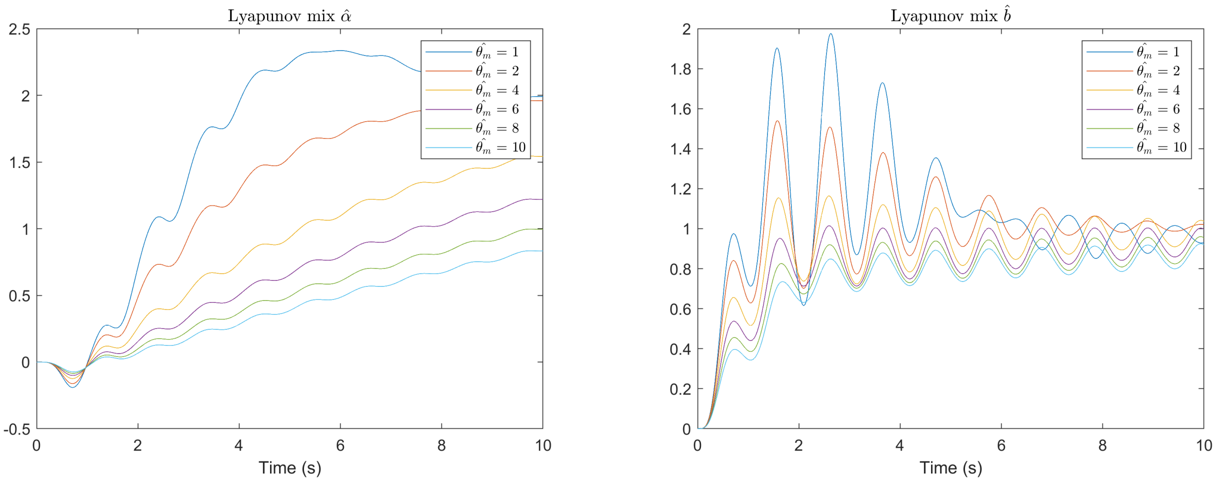
(b) b = 1

Εικόνα 5: Παραμετρική ανάλυση των εκτιμήσεων ως προς γ, Lyapunov μικτή μέθοδος

Παρατηρώντας τα διαγράμματα, σε αντίθεση με το Θέμα 1, δεν μπορούμε να διακρίνουμε μια ξεκάθαρη επίδραση των συντελεστών γ όσον αφορά την ταχύτητα με την οποία προσεγγίζουν την πραγματική τιμή. Ωστόσο μπορούμε να εντοπίσουμε έντονες ταλαντώσεις καθώς αυξάνεται το γ και στις εκτιμήσεις του α αλλά και του b, το οποίο μπορούμε να το δικαιολογήσουμε εξαιτίας της αύξησης του "άλματος". Τελικά, προτιμήσαμε να κρατήσουμε $\gamma_1 = 1$ και $\gamma_2 = 1$.

Σχετικά με την σύγκριση των δύο μεθόδων, παράλληλης και μικτής, μπορούμε να ισχυριστούμε ότι υπάρχει παρόμοια συμπεριφορά. Ιδιαίτερο ενδιαφέρον, όπως θα δούμε και στη συνέχεια, έχει η επίδραση του θορύβου σε αυτές τις δύο μεθόδους.

Για να μελετήσουμε την επίδραση του θ_m, θα κρατήσουμε σταθερούς τους γ συντελεστές ($\gamma = 1$).

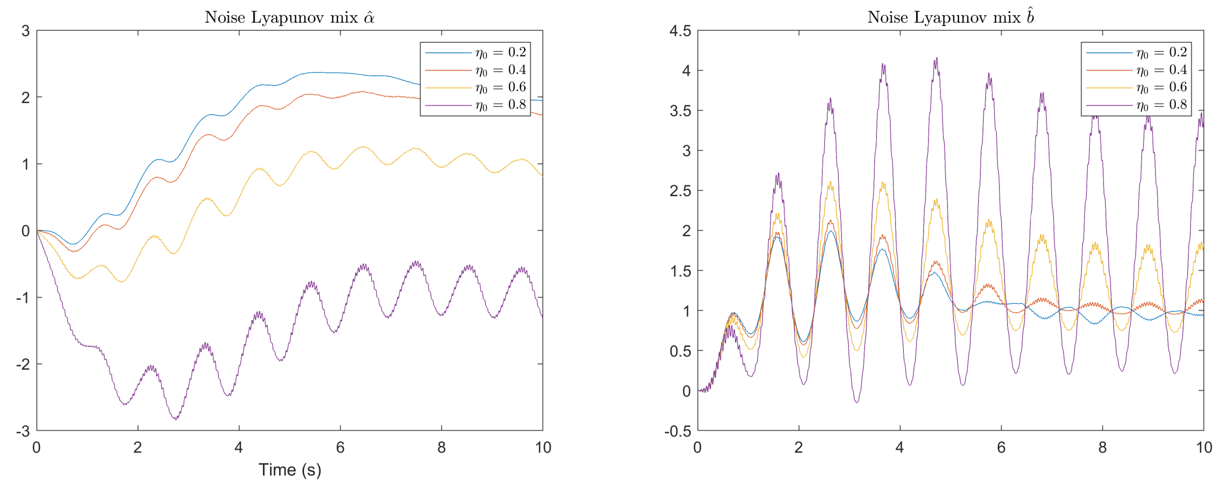


(a) $\alpha = 2$ (b) b = 1

Εικόνα 6: Παραμετρική ανάλυση των εκτιμήσεων ως προς θ_m, Lyapunov μικτή μέθοδος

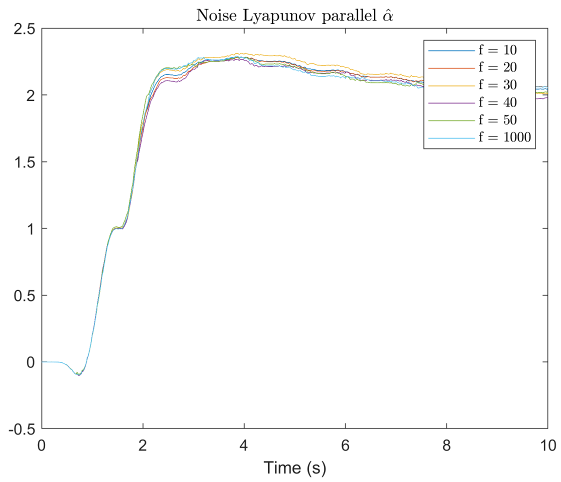
Παρατηρώντας τα παραπάνω μπορούμε να διακρίνουμε ότι η αύξηση του θ_m επιταχύνει την σταθεροποίηση της εκτίμησης. Θα συνεχίσουμε την ανάλυση μας για $\theta_m = 1$.

Σχετικά με την προσθήκη του θορύβου, $u = \eta_0 sin(2\pi f t)$, στην έξοδο των μετρήσεων x, με $\gamma_1 = \gamma_2 = 1$ και για την μικτή μέθοδο επιπλέον με $\theta_m = 1$, έχουμε τα ακόλουθα διαγράμματα:

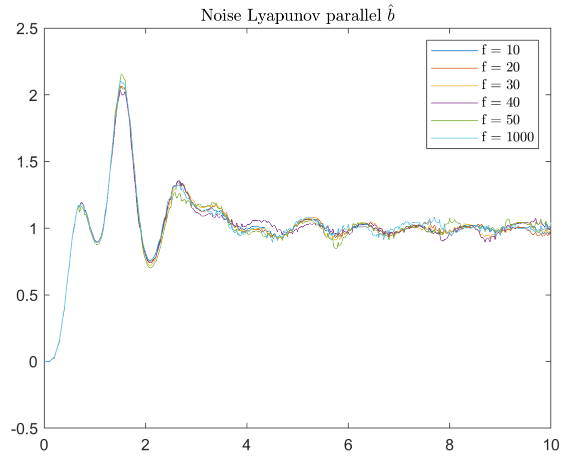


(a) $\alpha = 2$ (b) b = 1

Εικόνα 7: Επίδραση του θορύβου ως προς το πλάτος, Lyapunov παράλληλη μέθοδος

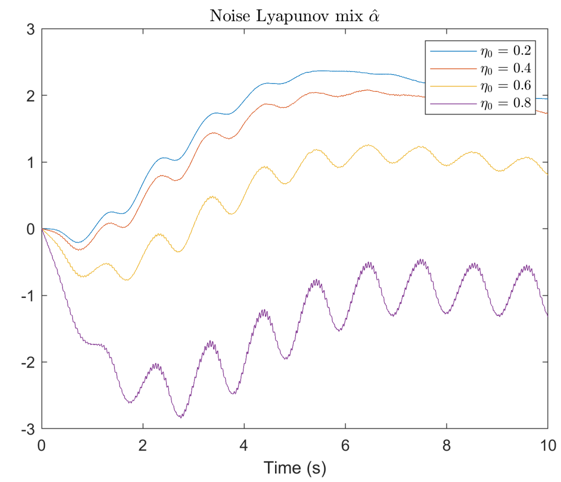


(a) $\alpha = 2$

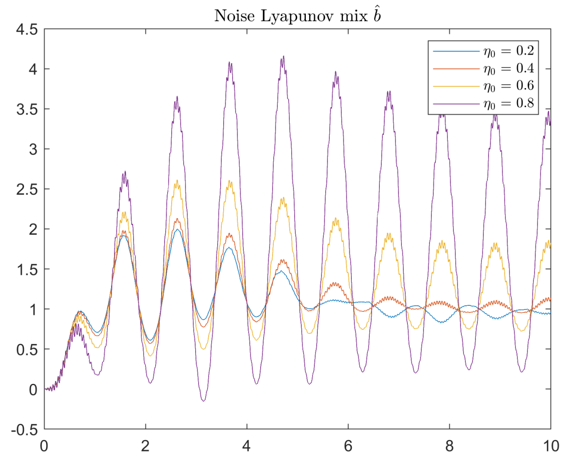


(b) b = 1

Εικόνα 8: Επίδραση του θορύβου ως προς την συχνότητα, Lyapunov παράλληλη μέθοδος

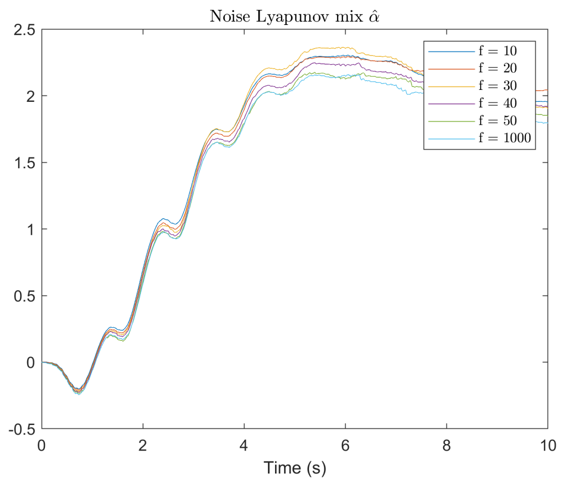


(a) $\alpha = 2$

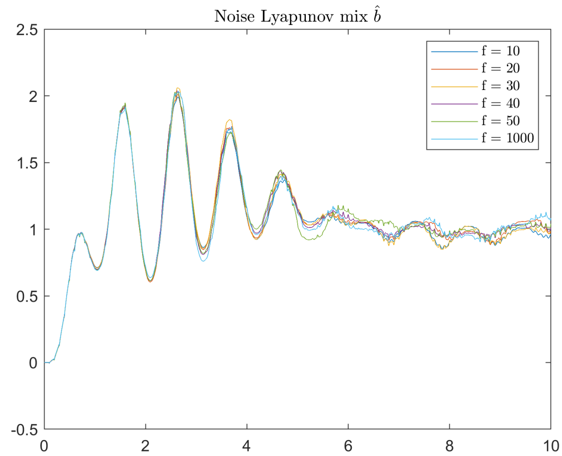


(b) b = 1

Εικόνα 9: Επίδραση του θορύβου ως προς το πλάτος, Lyapunov μικτή μέθοδος



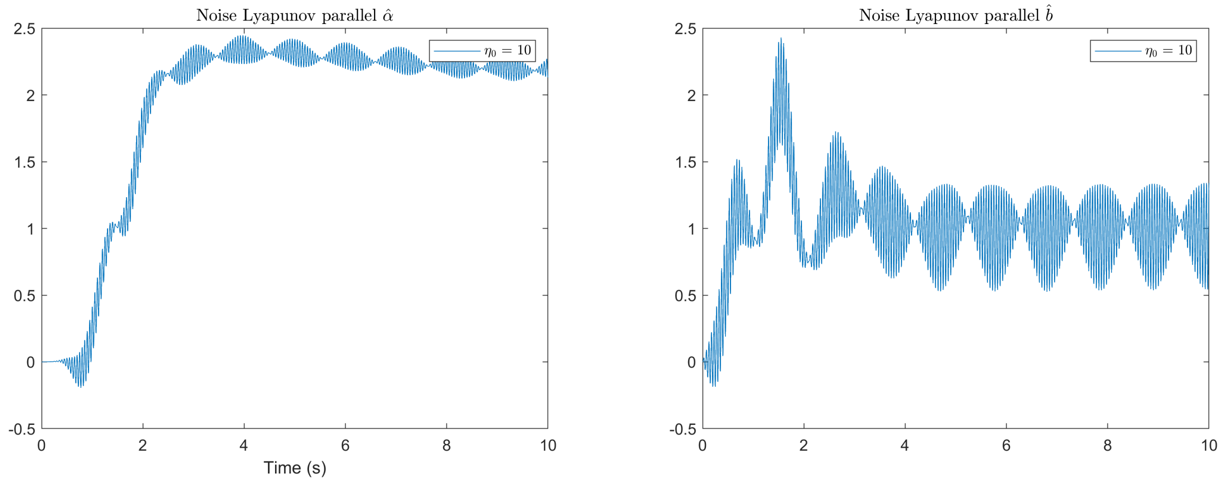
(a) $\alpha = 2$



(b) b = 1

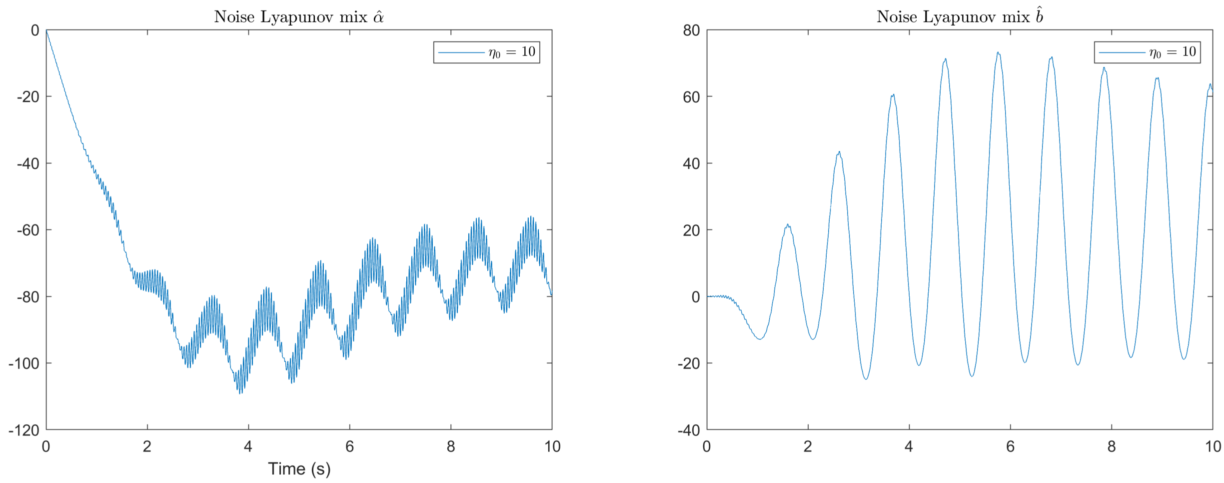
Εικόνα 10: Επίδραση του θορύβου ως προς την συχνότητα, Lyapunov μικτή μέθοδος

Είναι εμφανές ότι η επίδραση του θορύβου συναρτήσει του πλάτους, επηρεάζει σε σημαντικό βαθμό την μικτή μέθοδο. Για να το κάνουμε ακόμα πιο εμφανές, ας δούμε το αποτέλεσμα του θορύβου για μια αρκετά μεγάλη τιμή πλάτους $\eta_0 = 10$:



(a) $\alpha = 2$ (b) b = 1

Εικόνα 11: Επίδραση του θορύβου ως προς το πλάτος, Lyapunov παράλληλη μέθοδος



(a) $\alpha = 2$ (b) b = 1

Εικόνα 12: Επίδραση του θορύβου ως προς το πλάτος, Lyapunov μικτή μέθοδος

Βλέπουμε ότι πλέον η μικτή μέθοδος έχει πολύ μεγάλη απόκλιση από τον στόχο της εκτίμησης της πραγματικής τιμής. Αυτή λοιπόν η απόκλιση οφείλεται στην επίδραση που έχει η έξοδος στο σύστημα αναγνώρισης της μικτής μεθόδου. Πιο συγκεκριμένα:

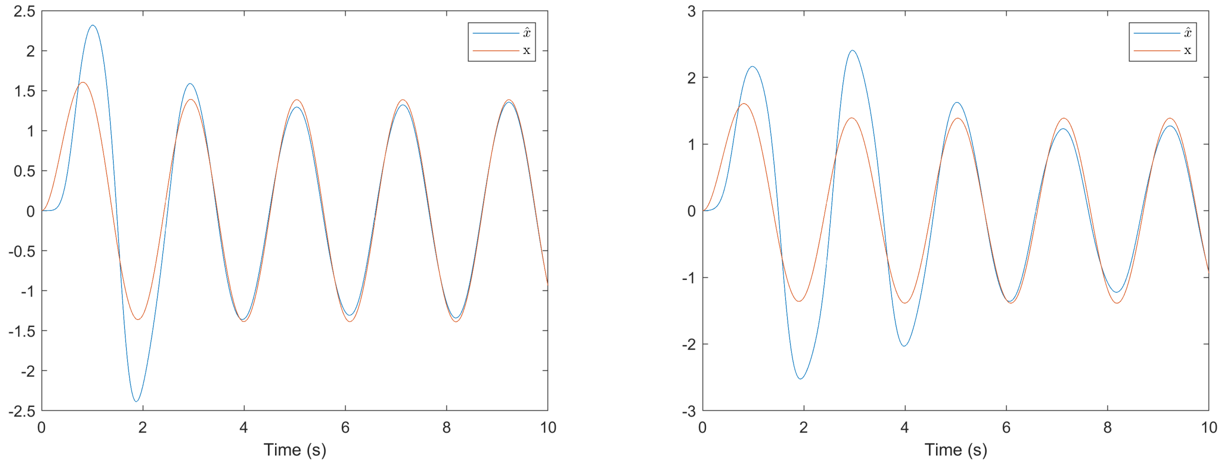
$$\dot{\hat{\theta}}_1 = -\gamma_1 e\hat{x} \xrightarrow{e=x-\hat{x}} \dot{\hat{\theta}}_1 = -\gamma_1 x\hat{x} + \gamma_1 \hat{x}^2 \quad (\Pi) \tag{9}$$

$$\dot{\hat{\theta}}_1 = -\gamma_1 ex \xrightarrow{e=x-\hat{x}} \dot{\hat{\theta}}_1 = \underbrace{-\gamma_1 x^2} + \gamma_1 x\hat{x} \quad (M) \tag{10}$$

Βλέποντας τις σχέσεις (9) και (10), παρατηρούμε ότι η βασική διαφορά μεταξύ των δύο μεθόδων είναι σε ποιό μέγεθος εμφανίζεται ο τετραγωνικός όρος. Στην (Π) μέθοδο, έχουμε τετράγωνο στην εκτίμηση, ενώ στην (Μ) έχουμε τετράγωνο στην έξοδο. Οπότε η προσθήκη θορύβου στην έξοδο, αναμένουμε να επηρεάσει σε πολύ μεγαλύτερο βαθμό την (Μ) μέθοδο σε σχέση με την (Π).

Σχετικά με την μεταβολή της **συχνότητας του θορύβου**, παρατηρούμε ότι δεν επηρεάζεται σε σημαντικό βαθμό η πορεία της εκτίμησης.

Τελικά η εκτίμηση της εξόδου χωρίς θόρυβο με τιμές $\gamma_1 = \gamma_2 = 1$ (Π) και επιπλέον $\theta_m = 1$ για την (Μ) είναι:



(a) Εκτίμηση εξόδου, Lyapunov μέθοδος παράλληλη (b) Εκτίμηση εξόδου, Lyapunov μέθοδος μικτή

Εικόνα 13

Matlab

simulation.m

```
%% LYAPUNOV PARALLEL

gammaRange = [1 10:10:20];

[xHat,aHat,bHat,legendInfo] = ...
    gammaAnalysis(@(x,u,t,gamma,thetaM)lyapunovParallel(x,u,t,gamma,thetaM),...
     gammaRange,x,u,t);

% plot
figure,plot(t,aHat),legend(legendInfo,'interpreter','latex')
title('Lyapunov parallel $\hat{\alpha}$','interpreter','latex')
xlabel('Time (s)')
print('lyap_paral_a','-dpng','-r300')
figure,plot(t,bHat),legend(legendInfo,'interpreter','latex')
title('Lyapunov parallel $\hat{b}$','interpreter','latex')
xlabel('Time (s)')
print('lyap_paral_b','-dpng','-r300')

% estimate output
figure,plot(t,xHat(:,1),t,x(t))
legend('$\hat{x}$', 'x', 'interpreter', 'latex')
xlabel('Time (s)')
print('lyap_parallel_x','-dpng','-r300')

%% LYAPUNOV MIX

[xHat,aHat,bHat,legendInfo] = ...
    gammaAnalysis(@(x,u,t,gamma,thetaM)lyapunovMix(x,u,t,gamma,thetaM),...
     gammaRange,x,u,t);

% plot
figure,plot(t,aHat),legend(legendInfo,'interpreter','latex')
```

```matlab
title('Lyapunov mix $\hat{\alpha}$','interpreter','latex')
xlabel('Time (s)')
print('lyap_mix_a','-dpng','-r300')
figure,plot(t,bHat),legend(legendInfo,'interpreter','latex')
title('Lyapunov mix $\hat{b}$','interpreter','latex')
xlabel('Time (s)')
print('lyap_mix_b','-dpng','-r300')

% thetaM analysis
thetamRange = [1 2:2:10];
[xHat,aHat,bHat,legendInfo] = thetamAnalysis(thetamRange,x,u,t);

% plot
figure,plot(t,aHat),legend(legendInfo,'interpreter','latex')
title('Lyapunov mix $\hat{\alpha}$','interpreter','latex')
xlabel('Time (s)')
print('lyap_mix_a_thetam','-dpng','-r300')
figure,plot(t,bHat),legend(legendInfo,'interpreter','latex')
title('Lyapunov mix $\hat{b}$','interpreter','latex')
xlabel('Time (s)')
print('lyap_mix_b_thetam','-dpng','-r300')

% estimate output
figure,plot(t,xHat(:,1),t,x(t))
legend('$\hat{x}$', 'x', 'interpreter', 'latex')
xlabel('Time (s)')
print('lyap_mix_x','-dpng','-r300')

%% AMPLITUDE NOISE

amplitudeRange = 0.2:0.2:0.8;
% LYAPUNOV PARALLEL NOISE
[xHat,aHat,bHat,legendInfo] = ...
    amplitudeNoiseAnalysis(@(x,u,t,gamma,thetaM)lyapunovParallel(x,u,t,gamma,thetaM)...
     ,amplitudeRange,x,u,t);

% plot
figure,plot(t,aHat),legend(legendInfo,'interpreter','latex')
title('Noise Lyapunov parallel $\hat{\alpha}$','interpreter','latex')
xlabel('Time (s)')
print('lyap_parallel_a_noise_ampl','-dpng','-r300')
figure,plot(t,bHat),legend(legendInfo,'interpreter','latex')
title('Noise Lyapunov parallel $\hat{b}$','interpreter','latex')
print('lyap_parallel_b_noise_ampl','-dpng','-r300')

% LYAPUNOV MIX NOISE
[xHat,aHat,bHat,legendInfo] = ...
    amplitudeNoiseAnalysis(@(x,u,t,gamma,thetaM)lyapunovMix(x,u,t,gamma,thetaM)...
     ,amplitudeRange,x,u,t);

% plot
figure,plot(t,aHat),legend(legendInfo,'interpreter','latex')
title('Noise Lyapunov mix $\hat{\alpha}$','interpreter','latex')
xlabel('Time (s)')
print('lyap_mix_a_noise_ampl','-dpng','-r300')
figure,plot(t,bHat),legend(legendInfo,'interpreter','latex')
title('Noise Lyapunov mix $\hat{b}$','interpreter','latex')
print('lyap_mix_b_noise_ampl','-dpng','-r300')
xlabel('Time (s)')

%% FREQUENCY NOISE

freqRange = [10:10:50 1000];
```

```
% LYAPUNOV PARALLEL NOISE
[xHat,aHat,bHat,legendInfo] = ...
    freqNoiseAnalysis(@(x,u,t,gamma,thetaM)lyapunovParallel(x,u,t,gamma,thetaM)...
     ,freqRange,x,u,t);

% plot
figure,plot(t,aHat),legend(legendInfo,'interpreter','latex')
title('Noise Lyapunov parallel $\hat{\alpha}$','interpreter','latex')
xlabel('Time (s)')
print('lyap_parallel_a_noise_freq','-dpng','-r300')
figure,plot(t,bHat),legend(legendInfo,'interpreter','latex')
title('Noise Lyapunov parallel $\hat{b}$','interpreter','latex')
print('lyap_parallel_b_noise_freq','-dpng','-r300')

% LYAPUNOV MIX NOISE
[xHat,aHat,bHat,legendInfo] = ...
    freqNoiseAnalysis(@(x,u,t,gamma,thetaM)lyapunovMix(x,u,t,gamma,thetaM)...
     ,freqRange,x,u,t);

% plot
figure,plot(t,aHat),legend(legendInfo,'interpreter','latex')
title('Noise Lyapunov mix $\hat{\alpha}$','interpreter','latex')
xlabel('Time (s)')
print('lyap_mix_a_noise_freq','-dpng','-r300')
figure,plot(t,bHat),legend(legendInfo,'interpreter','latex')
title('Noise Lyapunov mix $\hat{b}$','interpreter','latex')
print('lyap_mix_b_noise_freq','-dpng','-r300')
xlabel('Time (s)')
```

lyapunovParallel.m

```
function [xHat,aHat,bHat] = lyapunovParallel(x,u,t,gamma,thetaM)
[t,y] = ode45(@(t,y)diffSystem(x,u,t,y,gamma), t, [0 0 0]);
xHat = y(:,1);
aHat = y(:,2);
bHat = y(:,3);
end

function dy = diffSystem(x,u,t,y,gamma)
xHat=y(1); theta(1) = y(2); theta(2) = y(3);

dxHat = -theta(1)*xHat + theta(2)*u(t);
error = x(t) - xHat;
dtheta(1) = -gamma(1)*error*xHat;
dtheta(2) = gamma(2)*error*u(t);

dy(1) = dxHat; dy(2) = dtheta(1); dy(3) = dtheta(2);
dy = dy';
end
```

lyapunovMix.m

```
function [xHat,aHat,bHat] = lyapunovMix(x,u,t,gamma,thetaM)
[t,y] = ode45(@(t,y)diffSystem(x,u,t,y,gamma,thetaM), t, [0 0 0]);
xHat = y(:,1);
aHat = y(:,2);
bHat = y(:,3);
end

function dy = diffSystem(x,u,t,y,gamma,thetaM)
```

```
xHat = y(1); theta(1) = y(2); theta(2) = y(3);

error = x(t) - xHat;
dxHat = -theta(1)*x(t) + theta(2)*u(t) + thetaM*error;
dtheta(1) = -gamma(1)*error*x(t);
dtheta(2) = gamma(2)*error*u(t);

dy(1) = dxHat; dy(2) = dtheta(1); dy(3) = dtheta(2);
dy = dy';
end
```

Θέμα 3

Θεωρείστε το σύστημα δεύτερης τάξης

$$\dot{x} = \begin{bmatrix} \alpha_{11} & \alpha_{12} \\ \alpha_{21} & \alpha_{22} \end{bmatrix} x + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} u, \quad x(0) = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{1}$$

Όπου x είναι οι καταστάσεις, $u = 10sin(2t) + 5sin(7.5t)$ είναι η είσοδος και $\alpha_{11} = -0.25, \alpha_{12} = 3, \alpha_{21} = -5, \alpha_{22} = -1, b_1 = 1, b_2 = 2.2$ σταθερές άγνωστες παράμετροι.

Σχεδιάστε έναν εκτιμητή πραγματικού χρόνου παράλληλης δομής, των άγνωστων παραμέτρων στη μέθοδο σχεδίασης Lyapunov και προσομοιώστε τη λειτουργία του. Δημιουργείστε τις γραφικές παραστάσεις των x, $\hat{x}$ καθώς και τις εκτιμήσεις για τις άγνωστες παραμέτρους.

Μαθηματική ανάλυση

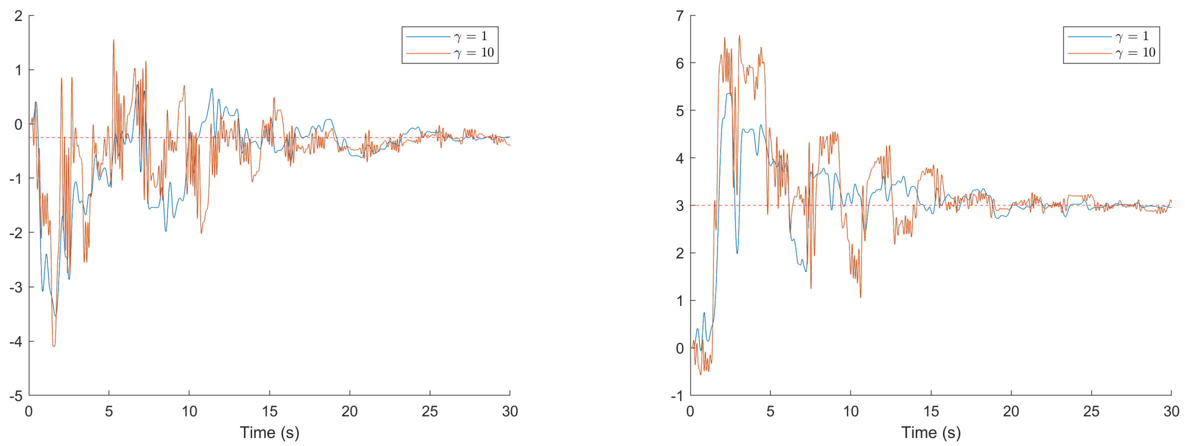
Μέχρι στιγμής μελετούσαμε συστήματα μονής εισόδου. Γενικεύοντας την μαθηματική ανάλυση που κάναμε στο Θέμα 2, για την παράλληλη δομή σε σύστημα πολλών εισόδων έχουμε:

$$\dot{\hat{A}} = \gamma_1 \hat{x} e^T \tag{2}$$

$$\dot{\hat{B}} = \gamma_2 \hat{u} e^T \tag{3}$$

όπου $e = x - \hat{x}$.

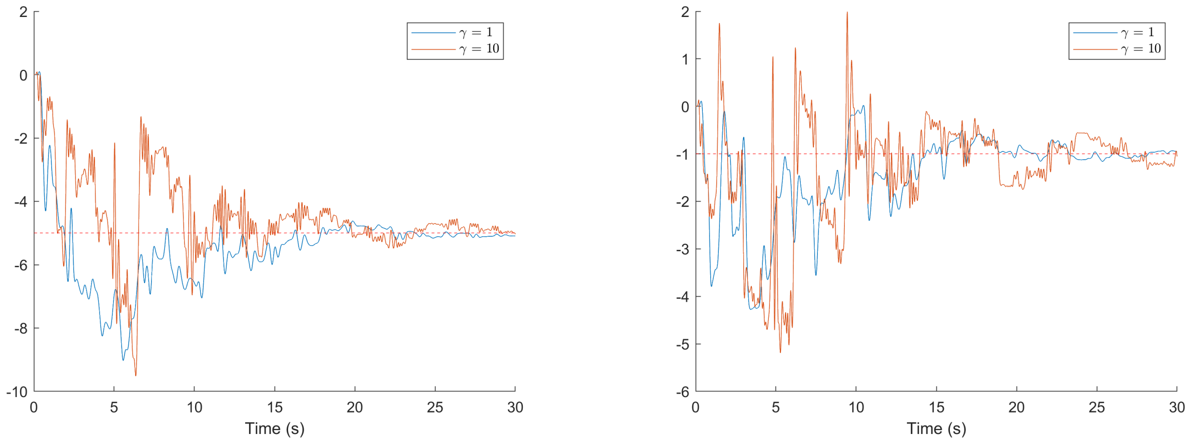
Διαγράμματα



(a) $\alpha_{11} = -0.25$

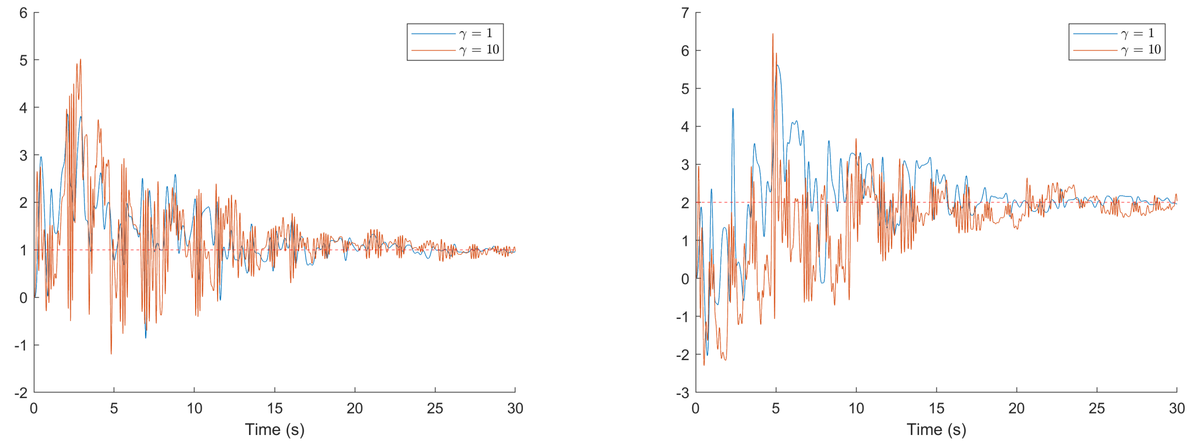
(b) α_{12} = 3

Εικόνα 14



(a) $\alpha_{21} = -5$ (b) $\alpha_{22} = -1$

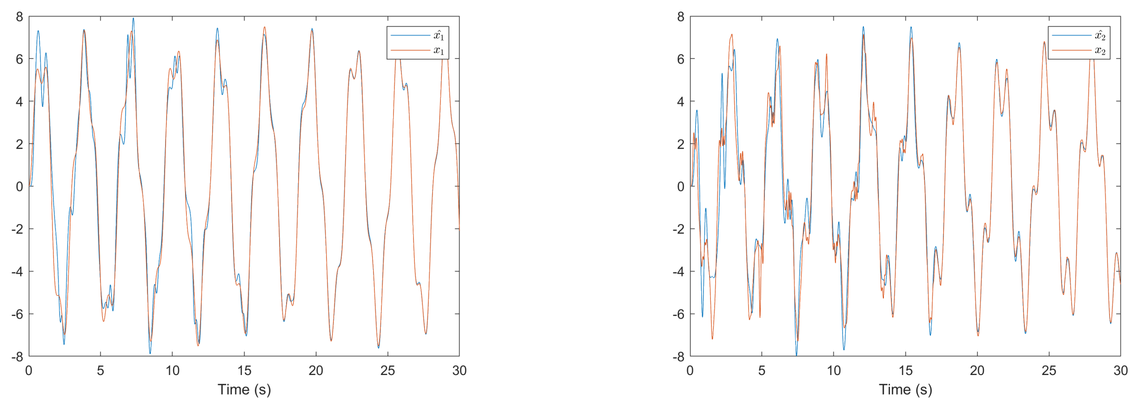
Εικόνα 15



(a) $b_1 = 1$ (b) $b_2 = 2.2$

Εικόνα 16

Με παρόμοιο τρόπο με τα προηγούμενα θέματα, δοκιμάσαμε διαφορετικές τιμές του γ. Μπορούμε να διαπιστώσουμε, ότι η αύξηση του γ στην συγκεκριμένη περίπτωση δεν βελτίωσε την ταχύτητα σύγκλισης προς την πραγματική τιμή αλλά δημιούργησε έντονες ταλαντώσεις. Αυτό συμβαίνει επειδή το "άλμα" είναι αρκετά μεγάλο με αποτέλεσμα να ξεφεύγουμε απο την επιθυμητή τιμή της εύρεσης του ελαχίστου σφάλματος.



(a) $x(1)$ (b) $x(2)$

Εικόνα 17: Εκτίμηση των συνιστωσών της εξόδου

Matlab

simulation.m

```matlab
%% MIMO

% the parameters I want to estimate
a = [-0.25 3; -5 -1];
b = [1;2];

% input
u = @(t) 10*sin(2*t) + 5*sin(7.5*t);

% time
t = 0:0.001:30;

% learning rate
%gamma = [1 1];
gammaRange = [1 10];

% compare to the previous implementations, now we compute everything
% (output,estimations) within one ode function in lyapunovParallelMIMO()
[x,xHat,aHat,bHat,legendInfo] = gammaAnalysisMIMO(gammaRange,u,t,a,b);

% plot
a = a';
for j=1:4
    figure
    hold on
    yline(a(j),'r--','HandleVisibility','off');
    for i=1:numel(gammaRange)
        plot(t,aHat(:,j,i))
    end
    legend(legendInfo,'interpreter','latex')
    xlabel('Time (s)')
    print(['lyap_parallel_a_mimo_' num2str(j)],'-dpng','-r300')
end

for j=1:2
    figure
    hold on
    yline(b(j),'r--','HandleVisibility','off');
    for i=1:numel(gammaRange)
        plot(t,bHat(:,j,i))
    end
    legend(legendInfo,'interpreter','latex')
    xlabel('Time (s)')
    print(['lyap_parallel_b_mimo_' num2str(j)],'-dpng','-r300')
end

figure,plot(t,xHat(:,1,1),t,x(:,1))
legend('$\hat{x_1}$', '$x_1$', 'interpreter', 'latex')
xlabel('Time (s)')
print('mimo_x1','-dpng','-r300')

figure,plot(t,xHat(:,2,1),t,x(:,2))
legend('$\hat{x_2}$', '$x_2$', 'interpreter', 'latex')
xlabel('Time (s)')
print('mimo_x2','-dpng','-r300')

%% FUNCTIONS
```

```
function [xHat,aHat,bHat,legendInfo] = gammaAnalysis(func,gammaRange,x,u,t)
    xHat = zeros(numel(t),numel(gammaRange));
    aHat = zeros(numel(t),numel(gammaRange));
    bHat = zeros(numel(t),numel(gammaRange));
    counter = 1;
    for i=gammaRange
        % learning rate
        gamma = [i i];
        thetaM = 1;
        [xHatPerGamma,aHatPerGamma,bHatPerGamma] = func(x,u,t,gamma,thetaM);

        xHat(:,counter) = xHatPerGamma;
        aHat(:,counter) = aHatPerGamma;
        bHat(:,counter) = bHatPerGamma;

        legendInfo{counter} = ['$\gamma$ = ' num2str(i)];
        counter = counter + 1;
    end
end

function [xHat,aHat,bHat,legendInfo] = thetamAnalysis(thetamRange,x,u,t)
    xHat = zeros(numel(t),numel(thetamRange));
    aHat = zeros(numel(t),numel(thetamRange));
    bHat = zeros(numel(t),numel(thetamRange));
    counter = 1;
    for i=thetamRange
        % learning rate
        gamma = [1 1];
        [xHatPerGamma,aHatPerGamma,bHatPerGamma] = lyapunovMix(x,u,t,gamma,i);

        xHat(:,counter) = xHatPerGamma;
        aHat(:,counter) = aHatPerGamma;
        bHat(:,counter) = bHatPerGamma;

        legendInfo{counter} = ['$\hat{\theta_m}$ = ' num2str(i)];
        counter = counter + 1;
    end
end

function [xHat,aHat,bHat,legendInfo] = ...
    amplitudeNoiseAnalysis(func,amplitudeRange,x,u,t)
    xHat = zeros(numel(t),numel(amplitudeRange));
    aHat = zeros(numel(t),numel(amplitudeRange));
    bHat = zeros(numel(t),numel(amplitudeRange));
    counter = 1;
    for i=amplitudeRange
        amplitude = i;
        f = 20;
        noise = @(t) amplitude*sin(2*pi*f*t);
        x = @(t) x(t) + noise(t);

        gamma = [1 1];
        thetaM = 1;
        [xHatPerAmpl,aHatPerAmpl,bHatPerAmpl] = func(x,u,t,gamma,thetaM);

        xHat(:,counter) = xHatPerAmpl;
        aHat(:,counter) = aHatPerAmpl;
        bHat(:,counter) = bHatPerAmpl;

        legendInfo{counter} = ['$\eta_0$ = ' num2str(i)];
        counter = counter + 1;
    end
```

```
end

function [xHat,aHat,bHat,legendInfo] = ...
    freqNoiseAnalysis(func,amplitudeRange,x,u,t)
    xHat = zeros(numel(t),numel(amplitudeRange));
    aHat = zeros(numel(t),numel(amplitudeRange));
    bHat = zeros(numel(t),numel(amplitudeRange));
    counter = 1;
    for i=amplitudeRange
        amplitude = 0.15;
        f = i;
        noise = @(t) amplitude*sin(2*pi*f*t);
        x = @(t) x(t) + noise(t);

        gamma = [1 1];
        thetaM = 1;
        [xHatPerFreq,aHatPerFreq,bHatPerFreq] = func(x,u,t,gamma,thetaM);

        xHat(:,counter) = xHatPerFreq;
        aHat(:,counter) = aHatPerFreq;
        bHat(:,counter) = bHatPerFreq;

        legendInfo{counter} = ['f = ' num2str(i)];
        counter = counter + 1;
    end
end

function [x,xHat,aHat,bHat,legendInfo] = gammaAnalysisMIMO(gammaRange,u,t,a,b)
    xHat = zeros(numel(t),2,numel(gammaRange));
    aHat = zeros(numel(t),4,numel(gammaRange));
    bHat = zeros(numel(t),2,numel(gammaRange));
    counter = 1;
    for i=gammaRange
        % learning rate
        gamma = [i i];
        [x,xHatPerGamma,aHatPerGamma,bHatPerGamma] = ...
            lyapunovParallelMIMO(u,t,a,b,gamma);

        xHat(:,:,counter) = xHatPerGamma;
        aHat(:,:,counter) = aHatPerGamma;
        bHat(:,:,counter) = bHatPerGamma;

        legendInfo{counter} = ['$\gamma$ = ' num2str(i)];
        counter = counter + 1;
    end
end
```

lyapunovParallelMIMO.m

```
function [x,xHat,aHat,bHat] = lyapunovParallelMIMO(u,t,a,b,gamma)
[t,y] = ode45(@(t,y)diffSystem(t,y,u,a,b,gamma), t, zeros(10,1));
x    = [y(:,1) y(:,4)];
xHat = [y(:,3) y(:,4)];
aHat = [y(:,5) y(:,6) y(:,7) y(:,8)];
bHat = [y(:,9) y(:,10)];
end

function dy = diffSystem(t,y,u,a,b,gamma)
x    = [y(1);y(2)];
xHat = [y(3);y(4)];
aHat = [y(5) y(6);y(7) y(8)];
```

```
bHat = [y(9);y(10)];

dx = a*x + b*u(t);
e = [x(1) - xHat(1); x(2) - xHat(2)];
dxHat = aHat*xHat + bHat*u(t);
daHat = gamma(1) * (xHat*e');
dbHat = gamma(2)*u(t)*e';

dy = [dx(1); dx(2);dxHat(1); dxHat(2);...
     daHat(1);daHat(2);daHat(3);daHat(4);...
     dbHat(1);dbHat(2)];
end
```

Matlab

Όλα τα αρχεία Maltab που παραθέσαμε στην αναφορά μπορούν να βρεθούν συγκεντρωμένα εδώ.