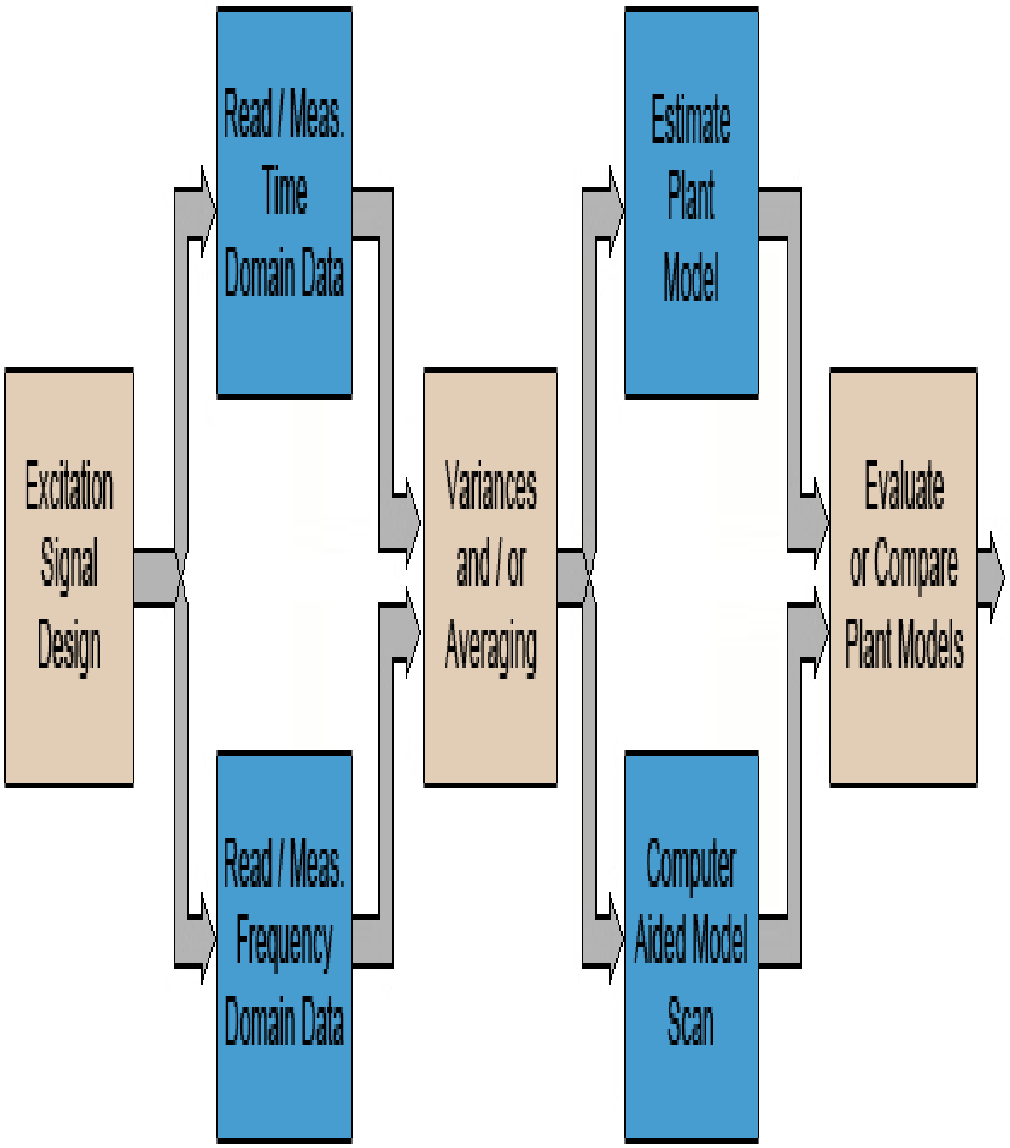
TP D'IDENTIFICATION DE SYSTEMES LINÉAIRES



Professeur: **Jamal DAAFOUZ**
Étudiant: **Arouna GANOU**

Les codes matlab sont en annexes.
Ce document une synthèse des deux séances de travaux pratiques d'**Identification de systèmes linéaires.**

On a une série de mesures. Il faut d'abord diviser cette série en deux données: L'une pour l'identification et l'autre pour la validation du modèle.

PREMIERE PARTIE

1. Méthode graphique

IL suffit de regarder les allures des données et identifier les paramètres d'une fonction de transfert dont on a déjà fait les hypothèses sur l'ordre et les pôles.

Hypothèse: Ordre 1: $H(p)=\frac{K}{1+\tau p}$.

Il s'agit donc de regarder sur les courbes pour voir les valeurs de K et tau. On peut les identifier sur la courbe car $K=\frac{y\infty}{u\infty}$. On sait qu'à $t=\tau$, la réponse vaut $0{,}63*y\infty$. Donc sans bruit dans les données, il est simple à identifier un modèle.

Hypothèse: Ordre 2: $H(p)=\frac{K}{(1+T1\,p)*(1+T2\,p)}$.

Il s'agit donc de regarder sur les courbes pour voir les valeurs de K et tau. On peut les identifier sur la courbe car $K=\frac{y\infty}{u\infty}$. On détermine T1 et T2 avec la méthode du point d'inflexion.

On utilise ce algorithme:
On utilise la réponse à un échelon unité(quitte à normaliser).

1. On détermine t_i l'instant auquel il y a un point d'inflexion dans la réponse.
2. On détermine ensuite s_i la valeur de la réponse à cet instant t_i (comme on normalisé $0\leqslant s_i\leqslant 1$.)
3. On utilise les abaques pour déterminer $x=\frac{T1}{T2}$ (ou résoudre l'équation $s_i=1-(1+x)x^{\frac{x}{1-x}}$ qui est clairement assez difficile à résoudre)
4. Sachant que $t_i=\frac{T2}{x-1}\ln(x)$, donc on déduit la valeur de T_2 , ensuite cette de T_1 avec la relation $x=\frac{T1}{T2}$.

A la fin de l'algorithme, on a notre fonction de transfert.

Comparaison des résultats graphiques

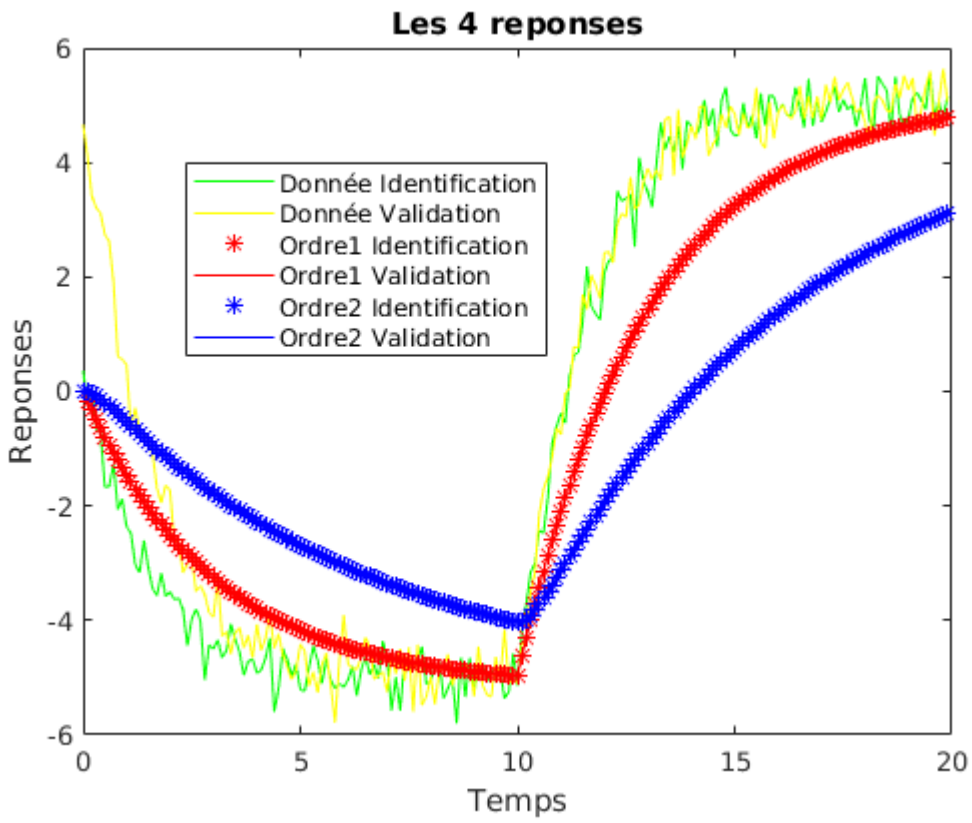


Figure 1: Allures des méthodes graphiques

On voit sur la courbe que le modèle d'ordre 1 s'approche mieux des données que le modèle d'ordre 2. Ceci ne veut pas dire que le modèle d'ordre 1 est plus adapté mais que le modèle d'ordre 2 est très mal identifié à travers les données. Nos données étant bruitées, la recherche du point d'inflexion n'est pas du tout rigoureuse.

Nous avons calculer le critère des moindres carrés par $$J=\frac{1}{N}\sum_{k=1}^{k=N}\left(y(k)-\hat{y}(k)\right)^2$$

Ordre du Modèle	Critère J Identification	Critère J Validation
Ordre 1	1.484436160664858	1.965030240863012
Ordre 2	7.91790952818800	7.706292458663141

Tableau1: Critères minimaux(au sens MCO)pour les modèles graphiques

Nous voyons que le critère J sur les données d'identification est plus petit que sur les données de validation. C'est normal car le modèle obtenu colle mieux avec les données ayant servi à l'identification qu'aux données de validation. C'est pourquoi on teste le modèle identifié par une

données qui n'a pas servi à l'identification.
Par contre le critère J pour l'ordre a un comportement qu'on n'a pas prévu. Le critère est plus faible sur la validation que sur l'identification. Ceci est dû au fait que l'identification d'un modèle d'ordre sur des données bruitées est quasi-impossible. En effet, on ne peut pas identifier un point d'inflexion sur des données bruitées.

Conclusion: Cette analyse nous montre que les méthodes graphiques que nous avons utilisées sont très peu robustes au bruit. Les deux méthodes utilisées sont donc biaisées.
On pouvait filtré le bruit avant de d'appliqué la méthode. On aurait peut-être de meilleurs résultats.

2. ARX1, ARX2, ARX3 et ARX4

Dans le modèle ARX(Autor-Regressive with eXternal inputs), on fait hypothèse que la fonction de transfert se met sous cette forme: $H(z)=\frac{\sum_{j=0}^{j=M} b_j z^{-j}}{1+\sum_{k=1}^{j=N} a_k z^{-k}}$ Avec N l'ordre du système et M le retard maximal.
En domaine temporel échantillonné, on a:

$$y_k=\sum_{j=1}^{j=M} b_j u_{(k-j)}-\sum_{i=1}^{i=N} a_i y_{(k-i)}$$

Nous avons supposé par la suite qu'il n'y a pas retard comme dit dans l'énoncé; et nous avons considéré successivement les ordres 1, 2, 3 et 4.
Donc M=0, et N=1;2;3 et 4.
Par estimateur des moindres carrées on a cette formule générique:

$$\Theta=\left[\Phi'\Phi\right]^{-1}\Phi'Y$$

Résultats:
On a utilisé un estimateur par moindre carré ordinaire donc le minimum de la fonction coût est exactement la somme des carrés des résidus:

$$J=\frac{1}{N}\sum_{k=1}^{k=N}\left(y(k)-\hat{y}(k)\right)^2$$

Modèle	Équation	Critère Identification	Critère Validation
ARX1	$y(t)=0{,}9259\,y(t-Ts)+0{,}1417\,u(t)$	0,2675	1,5250
ARX2	$y(t)=0{,}4542\,y(t-Ts)+0{,}4588\,y(t-2Ts)+0{,}1737\,u(t)$	0,1440	1,3039
ARX3	$y(t)=0{,}2824\,y(t-Ts)+0{,}2936\,y(t-2Ts)+0{,}3186\,y(t-3Ts)+0{,}2171\,u(t)$	0,1573	1,2944
ARX4	$y(t)=0{,}2550\,y(t-Ts)+0{,}1737\,y(t-2Ts)+0{,}2024\,y(t-3Ts)+0{,}2447\,y(t-4Ts)+0{,}2598u(t)$	0,1693	1,3407

Tableau2: Critères minimaux(au sens MCO) pour les modèles ARX

3. Simulation des ARX(ARX1, ARX2, ARX3 et ARX4)

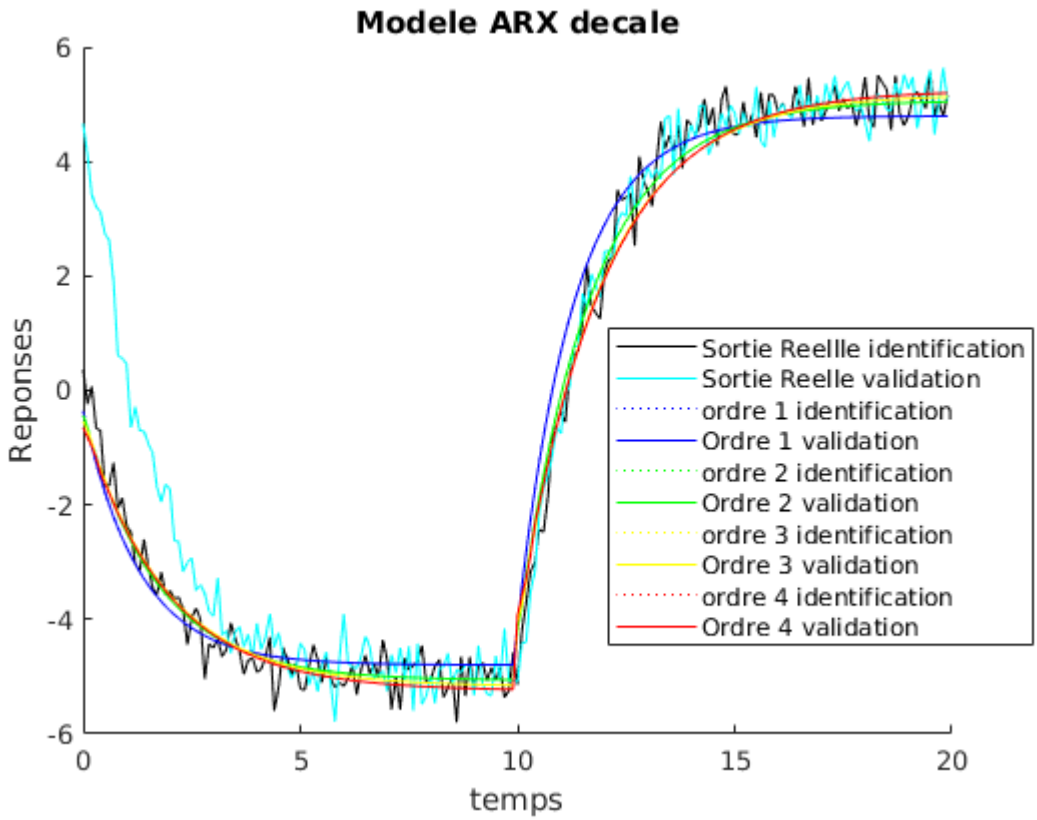


Figure 2: Allure des méthodes ARX

On a simulé les différentes fonctions de transfert obtenues avec les entrées de l'identification et de la validation. On a quatre modèle donc huit(8) courbes en tout. On peut utiliser la légende de la figure pour différencier les différentes réponses.

4. Comparaison des modèles

Notre comparaison est a priori biaisée entre ces modèles. En effet, nous n'avons pas une estimation de l'état initial pour la stimuler. Pour palier ce défaut, nous allons utiliser utiliser les critères sur l'identification et sur la validation. En regardant dans le tableau on constate que c'est le modèle ARX d'ordre 2 qui est le meilleur modèle. Nous avons confirmer cette analyse par une utilisation de la toolbox identification pour départager ces quatre modèles. Nous avons les mêmes coefficients à 3

chiffres après la virgule mais nous ne stimulons pas la sortie avec une estimation de l'état initial, ce que la toolbox sait faire.

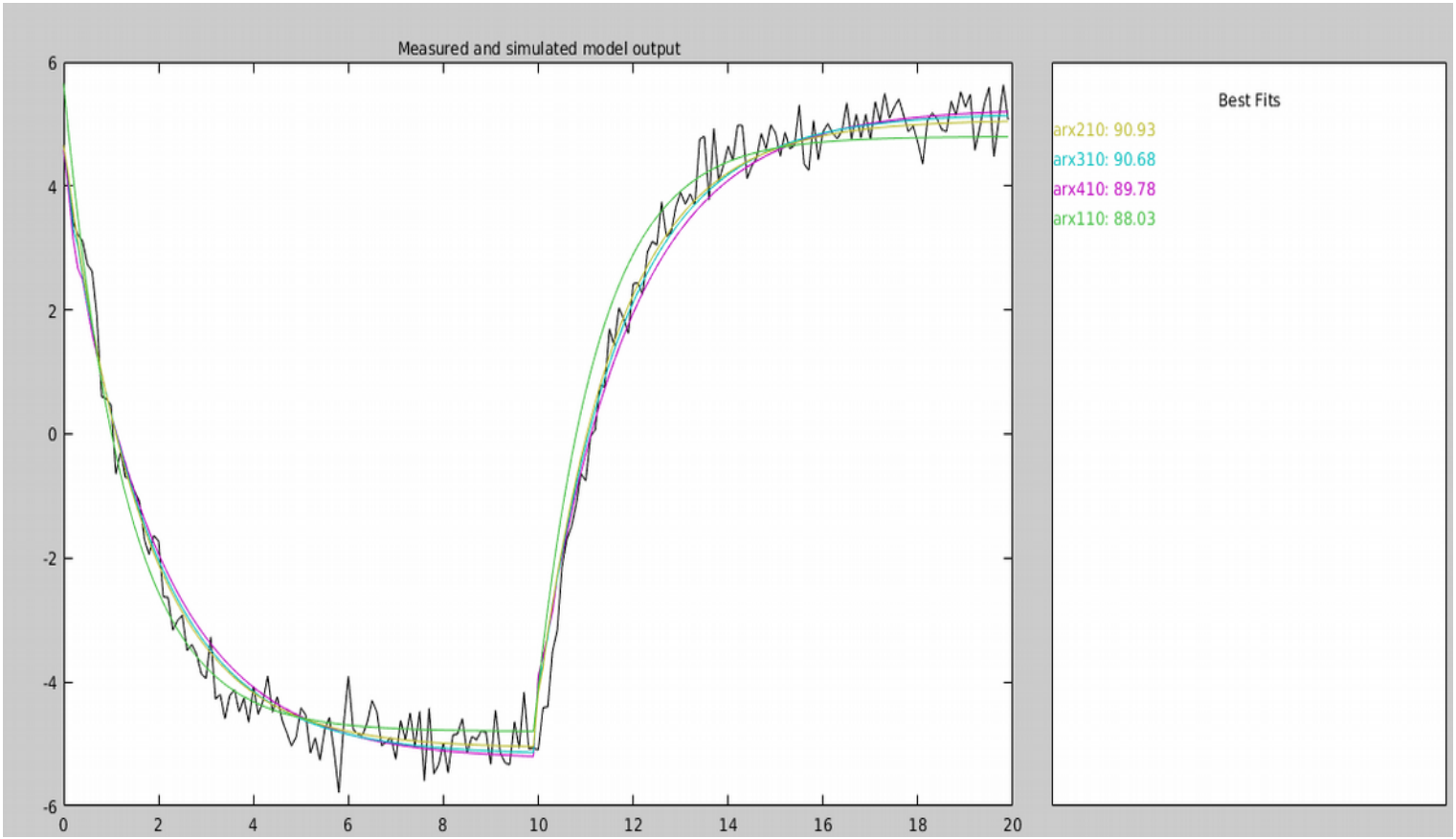


Figure 3: Best fits des modèles ARX avec de l'estimation de l'état initial.

Ayant les mêmes paramètres du systèmes réelles que la toolbox. Nous avons utilisé la toolbox pour mieux estimer l'état initial(`slim` n'arrivait pas à le faire correctement), on peut bien conclure que le modèle d'ordre 2(90,93% de fit contre 90,68 pour ARX3, 89,78 pour ARX4 et 88,03 pour ARX1) est le meilleur modèle. Nous constatons que contrairement à l'intuition, l'augmentation de l'ordre du modèle n'aboutit pas forcément au meilleur modèle!
Les modèles ARX donnent clairement de très bon modèles (robustesse au bruit notamment) par rapport au méthodes graphiques. Par contre, ils sont coûteux en calcul.

Dans cette partie, nous étions limité parce qu'on faisait tout à la main avant de passer à du code en Matlab. Nous étions limité en modèle et en estimation de l'état initial. Nous allons utilisé la toolbox SystemIdentification de Matlab: Avec cet outil, on aura la possibilité de tester plusieurs modèles(en introduisant des retards, du bruit, en changeant d'espace temps-fréquences,… et tout ceci très rapidement) et plusieurs options sur tester les modèles.

Dans tout le travail, on n'a considéré qu'on ne prend que la valeur de u(t) sans prendre les autres valeurs (u(t-i*Ts)) i.

On a considéré ensuite des modèles ARX plus général $y_k = \sum_{j=1}^{j=M} b_j u_{k-j} - \sum_{i=1}^{i=N} a_i y_{k-i}$ avec M=N.

On trouve un meilleur fit avec les données et on a pensé a simuler sur toutes les données et ensuite prendre la partie d'identification pour faire les critères

Modèle	Critère Identification	Critère Validation
ARX1	0,2675	0,3475
ARX2	0,1206	0,1304
ARX3	0,1502	0,1697
ARX4	0,2473	0,3130

Tableau 3: Critère pour le modèle tenant compte des entrées antérieures.

On voit clairement que le problème d'estimation de l'état initial est résolu en stimulant sur toutes les données et ensuite extraire les parties correspondantes à la validation pour calculer les critères.
Nous avons la figure ***Figure4***

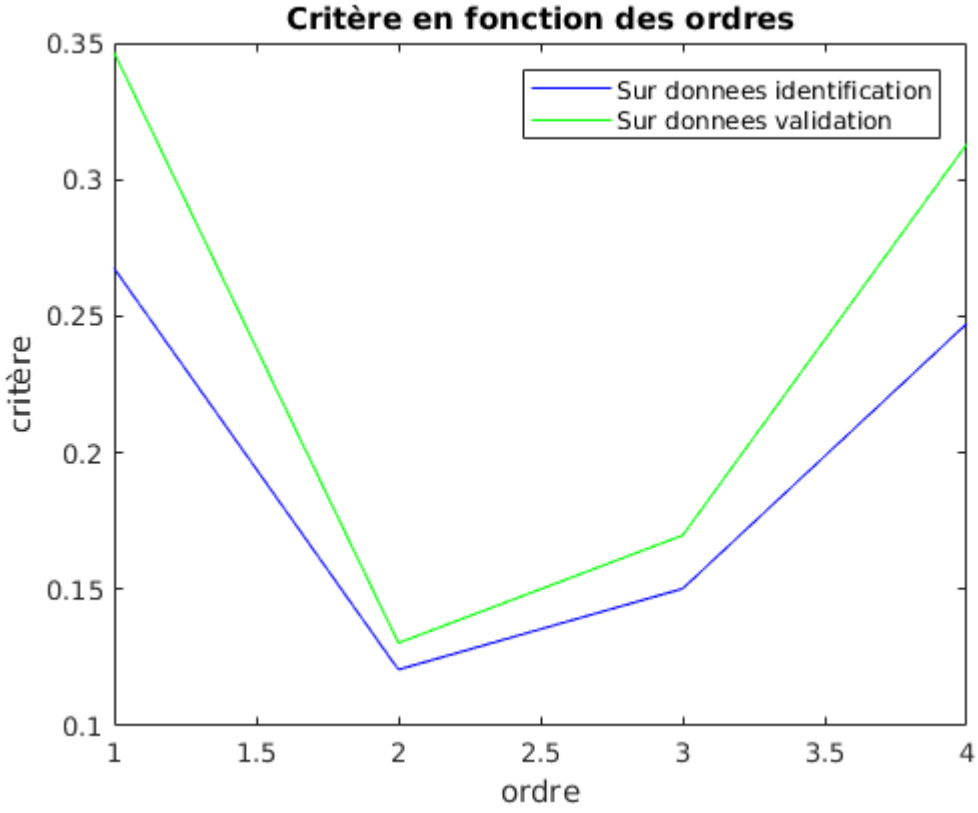


Figure4: Evolution du critère en fonction de l'ordre

On constate clairement que ARX d'ordre 2 est le meilleur au sens des moindres carrés ordinaires.

Comparaison avec les ARX de la toolbox: Les coefficients des polynômes ARX obtenues sont les mêmes que ceux obtenus avec la toolbox avec la méthode générique parce que nous avons utilisé la même structure et la même utilisation de l'entrée: On a utilisé directement u(t), u(t-Ts), u(t-2Ts),...

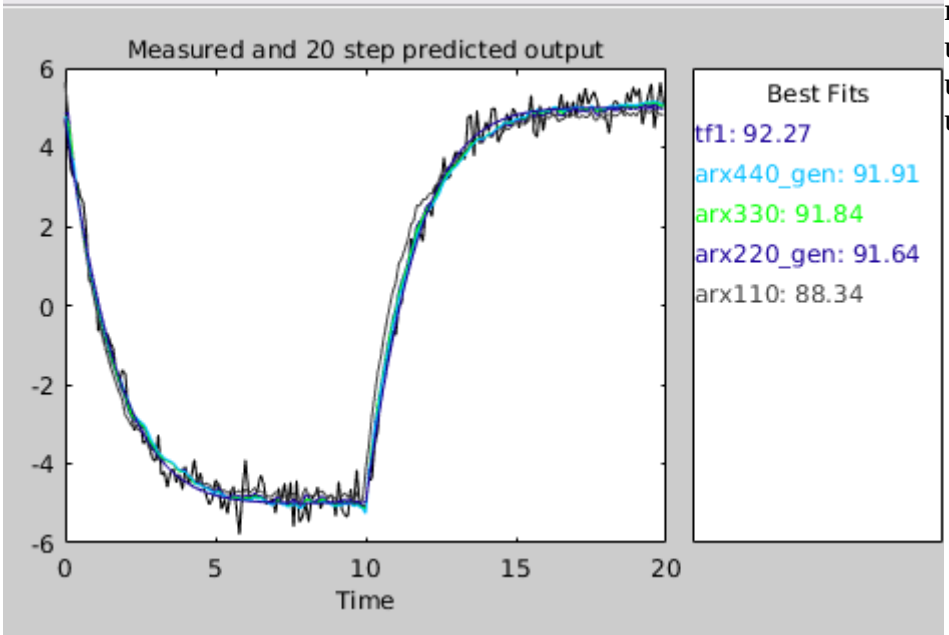


Figure5:Best fits sur les données validations

On constate que le modèle ARX3, ARX4 et le modèle ARX2 sont presque les mêmes en termes de fit. Bien que nous avons constaté que le modèle d'ordre 2 est meilleur, nous pensons qu'il l'est toujours. En effet c'est mieux d'avoir un système d'ordre plus faibles et si les fits sont presque les mêmes, c'est mieux de choisir celui qui a l'ordre le plus faible.

CONCLUSION:

Nous avons pu coder nous-même une famille de modèle avec les données. Nous avons pu faire la comparaison entre ces modèles, et ensuite avec les modèles que nous obtenons avec la toolbox. Nous optons pour le modèle d'ordre 2 en tenant compte des fits. Ceci nous a permis de bien appréhender les moindres carrés démontrer en cours.

DEUXIÈME PARTIE:Utilisation de la toolbox systemIdenfication de matlab

Nous allons utiliser par la suite, la toolbox systemIdenfication qui a cet atout de nous donner un grand nombre de modèles et une simplicité d'utilisation(à travers l'interface graphique et sans connaître les algorithmes qui tourne derrière).

Avec cette donnée, nous avons fait un prétraitement des données avant de faire l'identification car on a coupé les données de la fin: En effet il y avait des valeurs aberrantes vers la fin du fichier, le professeur de la séance nous a conseillé de le faire pour ne pas avoir des résultats biaisés.

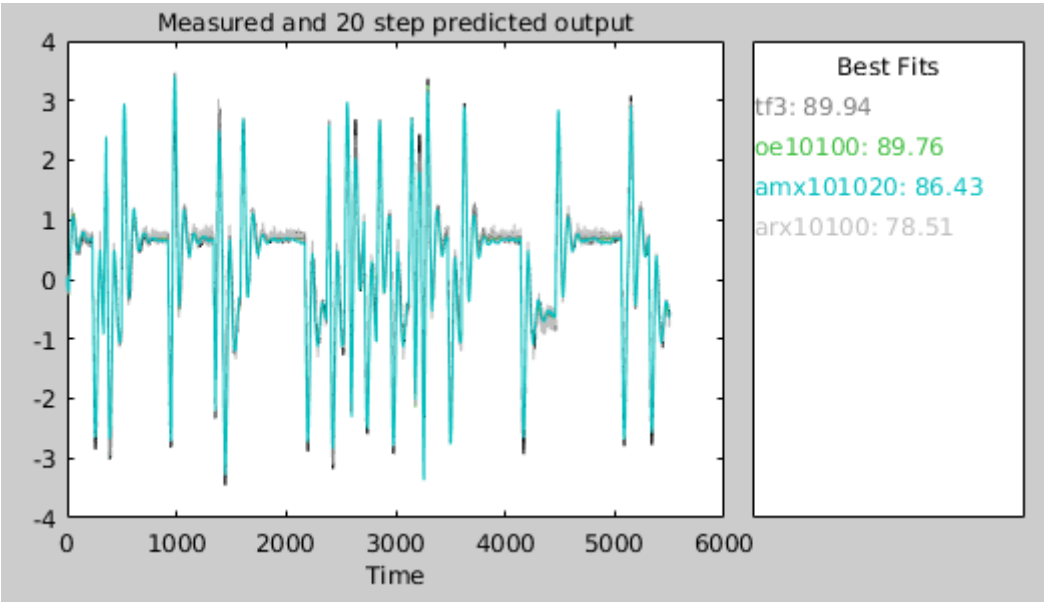


Figure6:Best filts pour données dat1 en simulations: Donc BO

En simulation(***Figure6***), nous constatons que le modèle OE d'ordre 10 a le meilleur fit. Mais l'ordre est quand même un peu trop élevé.

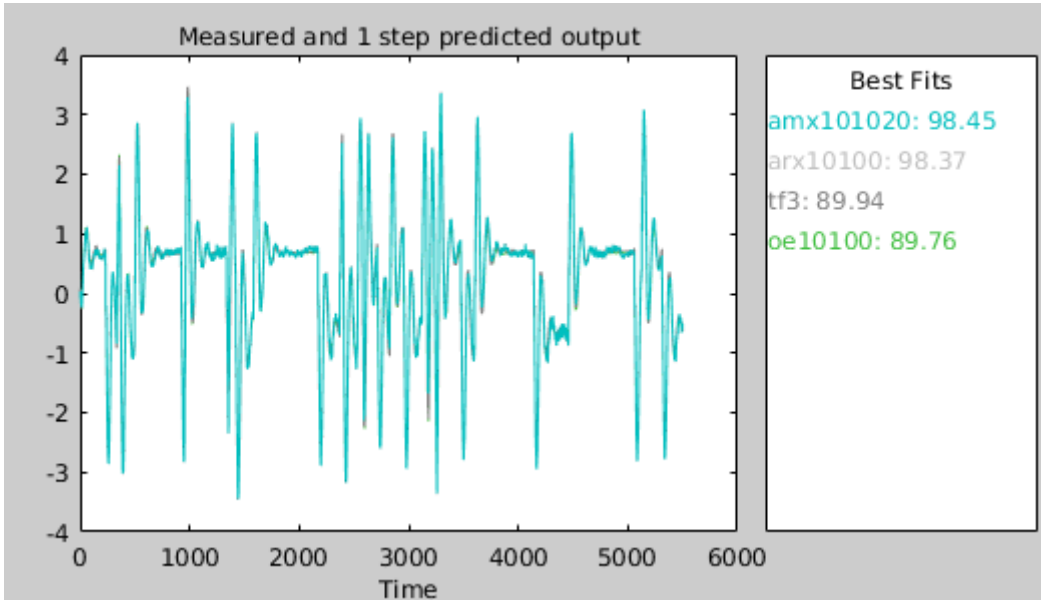


Figure7: Best filts pour données dat1 en précisions: Donc BF
En prédiction(***Figure7***), il vaut mieux utiliser le modèle ARIMAX d'ordre 10. Encore une fois, l'ordre est assez élevé.

CONCLUSION: Donc pour le choix des modèles, il faut tenir compte de ce qu'on veut faire : Simulation ou prédiction(observateur par exemple). Pour l'ordre, il faudra tenir compte de la technologie à utiliser. Donc c'est un compromis entre technologie et précision du modèle.

ANNEXES CODES MATLAB

<CODE METHODE GRAPHIQUES>

```
%% Travaux Pratiques d'identification des systÃ¨mes
%  Identification de systÃ¨mes linÃ©aired sous matlab
%  Professeur: DAAFOUZ Jamal
%  Etudiant(s):   GANOU Arouna

%%
close all

%% TP1 SEANCE  1
Donnees
uIdentification=u(1:200);
yIdentification=y(1:200);
N=200;

%% Modele d'ordre 1

% recherche de taux
K1=yIdentification(200)/uIdentification(200);
fe=10;% en Hz
te=1/fe;
taux=(130-100)*te;

%% Modele d'ordre 2

K2=yIdentification(200)/uIdentification(200);
dY=yIdentification(2:200)- yIdentification(1:199);
%plot(yModele,x)
ti=114;
si=0.63/max(yIdentification);
xi=0.05;
T2=(xi-1)/log(xi);
T1=T2/xi;
%% Validation
%donnees de validation
uValidation=u(201:400);
yValidation=y(201:400);
%systeme du premier ordre trouvÃ©
num1=K1;
dem1=[taux 1];
sys1erord1=tf(num1,dem1);
%systeme du deuxieme ordre trouvÃ©
num2=K2;
dem2=[T1*T2 (T1+T2) 1];
sys1erord2=tf(num2,dem2);
%temps
temps=te*(0:N-1)';

%criteres systeme d'ordre 1
yChapeau11=lsim(sys1erord1,uIdentification,temps);
yChapeau12=lsim(sys1erord1,uValidation,temps);
```

```
Critere11=(1/N)*norm((yIdentification-yChapeau11),2)^2;
Critere12=(1/N)*norm((yValidation-yChapeau12),2)^2;

%criteres systeme d'ordre 2
yChapeau21=lsim(sys1erord2,uIdentification,temps);
yChapeau22=lsim(sys1erord2,uValidation,temps);
Critere21=(1/N)*norm((yIdentification-yChapeau21),2)^2;
Critere22=(1/N)*norm((yValidation-yChapeau22),2)^2;

%% Stockage des critÃ¨res
% Tous les critÃ¨res sont dans ce tableau
%PremiÃ¨re ligne les critÃ¨res pour le premier ordre
%DeuxiÃ¨me ligne pour les critÃ¨red du deuxiÃ¨me ordre
%PremiÃ¨re colone pour la premiÃ¨re partie des donnees
%DeuxiÃ¨me colonne pour la deuxiÃ¨me partie des donnÃ©es
%Critere=zeros(size(Critere11),size(Critere12),size(Critere21),size(Critere22));
Critere=[Critere11 Critere12
        Critere21  Critere22];
%% Figures pour le tableau des validations croisÃ©es
figure()
pIdentenfication=plot(temps,yIdentification, 'g');
hold on
pValidation=plot(temps,yValidation,'y');
pOrdre11=plot(temps,yChapeau11, 'r*');
pOrdre12=plot(temps,yChapeau12, 'r-');
pOrdre21=plot(temps,yChapeau21, 'b*');
pOrdre22=plot(temps,yChapeau22, 'b');
legend([pIdentenfication, pValidation, pOrdre11, pOrdre12, pOrdre21,
pOrdre22],...
        'DonnÃ©e Identification',...
        'DonnÃ©e Validation',...
        'Ordre1 Identification','Ordre1 Validation',...
        'Ordre2 Identification','Ordre2 Validation')
xlabel('Temps')
ylabel('Reponses')
title('Les 4 reponses')
%% Simulation avec la totalitÃ© des donnÃ©es

[nTotEchan,~]=size(u);
tempsTotal=te*(0:nTotEchan-1);% pour commencer Ã  temps t=0
yChapeauOrdre1=lsim(sys1erord1,u,tempsTotal);
yChapeauOrdre2=lsim(sys1erord2,u,tempsTotal);

%% Figure pour la validation avec les toutes les donnÃ©es

figure();
pDonneeTotale=plot(tempsTotal,y,'g');
hold on
pOrdre1=plot(tempsTotal,yChapeauOrdre1,'r');
pOrdre2=plot(tempsTotal,yChapeauOrdre2,'b');
legend([pDonneeTotale,pOrdre1,pOrdre2],...
    'DonnÃ©es','Simulation ordre1','Simulation Ordre2');
xlabel('temps');
ylabel('Reponses aux simulations');
```

<CODE_METHODE ARX AVEC PERTE D'INFORMATION>

```
%% ENSEM Nancy, INPL
```

```
%% Travaux Pratiques d'identification des systÃ¨mes
%  Identification de systÃ¨mes linÃ©aires sous matlab
%  Professeur: DAAFOUZ Jamal
%  Etudiant(s):   GANOU Arouna


%%

close all

%% DonnÃ©es
Donnees
N=200;
%donnees d'identification
uIdentification=u(1:N);
yIdentification=y(1:N);
%donnees de validation
uValidation=u(N+1:end-1);
yValidation=y(N+1:end-1);

Hs=10; %en Hz
Ts=1/Hs;
tempsId=Ts*(0:N-1);% pour commencer Ã  temps t=0
tempsVal=Ts*(0:(max(size(u)-N-2)));
%% Ordre 1 Y(t+1)=-a1*Y(t)+b1*U(t) sans retard

p1=2;% Nombre de paramÃ¨tre Ã  identifier.
% Dimensionnement des matrices
%  theta=zeros(p,1);
%  Y=zeros(N,1);
%  Phi=zeros(N,p);
%Affectation de Y
Y1=yIdentification;
% Affectation de la matrice Phi
Phi1(:,1)=[0; -Y1(1:N-1)];
Phi1(:,2)=uIdentification(1:N);
% Calcul de theta
theta1=(Phi1'*Phi1)\Phi1'*Y1;


%% Ordre 2 Y(t)=-a1*Y(t-1)-a2*Y(t-2)+b1*U(t)

p2=3;% Nombre de paramÃ¨tre Ã  identifier.
% Dimensionnement des matrices
%  theta=zeros(p,1);
%  Y=zeros(N,1);
%  Phi=zeros(N,p);
%Affectation de Y
Y2=yIdentification;
% Affectation de la matrice Phi
Phi2(:,1)=[0;0;-Y2(1:N-2)];
Phi2(:,2)=[0;-Y2(1:N-1)];
Phi2(:,3)=uIdentification(1:N);
% Calcul de theta
theta2=(Phi2'*Phi2)\(Phi2'*Y2);

%% Ordre 3  Y(t)=-a1*Y(t-1)-a2*Y(t-2)-a3*Y(t-3)+b1*U(t)
p3=4;% Nombre de paramÃ¨tre Ã  identifier.
% Dimensionnement des matrices
```

```
%  theta=zeros(p,1);
%  Y=zeros(N,1);
%  Phi=zeros(N,p);
%Affectation de Y
Y3=yIdentification;
% Affectation de la matrice Phi
Phi3(:,1)=[0;0;0;-Y3(1:N-3)];
Phi3(:,2)=[0;0;-Y3(1:N-2)];
Phi3(:,3)=[0;-Y3(1:N-1)];
Phi3(:,4)=uIdentification(1:N);
% Calcul de theta
theta3=(Phi3'*Phi3)\Phi3'*Y3;

%% Ordre 3  Y(t)=-a1*Y(t-1)-a2*Y(t-2)-a3*Y(t-3)-a4*Y(t-4)+b1*U(t)
p4=5;% Nombre de paramÃ¨tre Ã  identifier.
% Dimensionnement des matrices
%  theta=zeros(p,1);
%  Y=zeros(N,1);
%  Phi=zeros(N,p);
%Affectation de Y
Y4=yIdentification;
% Affectation de la matrice Phi
Phi4(:,1)=[0;0;0;0;-Y4(1:N-4)];
Phi4(:,2)=[0;0;0;-Y4(1:N-3)];
Phi4(:,3)=[0;0;-Y4(1:N-2)];
Phi4(:,4)=[0;-Y4(1:N-1)];
Phi4(:,5)=uIdentification(1:N);
% Calcul de theta
theta4=(Phi4'*Phi4)\Phi4'*Y4;

%% SystÃ¨mes trouvÃ©s
sysARX1=tf(theta1(end),[1 theta1(1:end-1)'],Ts,'Variable','z^-1');
sysARX2=tf(theta2(end),[1 theta2(1:end-1)'],Ts,'Variable','z^-1');
sysARX3=tf(theta3(end),[1 theta3(1:end-1)'],Ts,'Variable','z^-1');
sysARX4=tf(theta4(end),[1 theta4(1:end-1)'],Ts,'Variable','z^-1');

%% Simulation avec les donnÃ©es

%ordre 1
yARX11=lsim(sysARX1,uIdentification,tempsId);%donnee identification
yARX12=lsim(sysARX1,uValidation,tempsVal);%donnee validation

%ordre 2
yARX21=lsim(sysARX2,uIdentification,tempsId);%donnee identification
yARX22=lsim(sysARX2,uValidation,tempsVal);%donnee validation

%ordre 3
yARX31=lsim(sysARX3,uIdentification,tempsId);%donnee identification
yARX32=lsim(sysARX3,uValidation,tempsVal);%donnee validation

%ordre 4
yARX41=lsim(sysARX4,uIdentification,tempsId);%donnee identification
yARX42=lsim(sysARX4,uValidation,tempsVal);%donnee validation

%% Calcul des critÃ¨res

%ordre 1
critArx11=(1/N)*norm((yIdentification-yARX11),2)^2;
critArx12=(1/N)*norm((yValidation-yARX12),2)^2;
```

```
%ordre 2
critArx21=(1/N)*norm((yIdentification-yARX21),2)^2;
critArx22=(1/N)*norm((yValidation-yARX22),2)^2;

%ordre 3
critArx31=(1/N)*norm((yIdentification-yARX31),2)^2;
critArx32=(1/N)*norm((yValidation-yARX32),2)^2;

%ordre 1
critArx41=(1/N)*norm((yIdentification-yARX41),2)^2;
critArx42=(1/N)*norm((yValidation-yARX42),2)^2;

%% Les profils des reponses
figure();hold on
fId=plot(tempsId,yIdentification,'c-');
fVal=plot(tempsVal,yValidation,'c-');
%ordre 1
f1Id=plot(tempsId,yARX11,'b:');
f1Val=plot(tempsVal,yARX12,'b-');

%ordre 2
f2Id=plot(tempsId,yARX21,'g:');
f2Val=plot(tempsVal,yARX22,'g-');

%ordre 3
f3Id=plot(tempsId,yARX31,'y:');
f3Val=plot(tempsVal,yARX32,'y-');

%ordre 4
f4Id=plot(tempsId,yARX41,'r:');
f4Val=plot(tempsVal,yARX42,'r-');
legend([fId,fVal,f1Id,f1Val,f2Id,f2Val,f3Id,f3Val,f4Id,f4Val],...
       'DonnÃ©es identification',...
       'DonnÃ©es validation',...
       'ordre 1 identification','Ordre 1 validation',...
       'ordre 2 identification','Ordre 2 validation',...
       'ordre 3 identification','Ordre 3 validation',...
       'ordre 4 identification','Ordre 4 validation');

xlabel('temps');
ylabel('Reponses');
title('Modele ARX sans decalage');
```

<CODE_METHODE ARX SANS PERTE D'INFORMATION>

```
%% ENSEM Nancy, INPL

%% Travaux Pratiques d'identification des systÃ¨mes
%  Identification de systÃ¨mes linÃ©aired sous matlab
%  Professeur:    DAAFOUZ Jamal
%  Etudiant(s):   GANOU Arouna
%%

close all
clear all

%% DonnÃ©es
```

```
Donnees;
N=200;
%donnees d'identification
uIdentification=u(1:N);
yIdentification=y(1:N);
%donnees de validation
uValidation=u(N+1:end-1);
yValidation=y(N+1:end-1);

Fs=10; %en Hz
Ts=1/Fs;
tempsId=Ts*(0:N-1);% pour commencer Ã  temps t=0
%% Ordre 1 Y(t+1)=-a1*Y(t)+b1*U(t) sans retard
n1d=1;%ordre du modÃ¨le
p1d=n1d+1;% Nombre de paramÃ¨tre Ã  identifier.
% Dimensionnement des matrices
%  theta=zeros(p,1);
%  Y=zeros(N,1);
%  Phi=zeros(N,p);
%Affectation de Y
Y1d=yIdentification(n1d+1:end);
% Affectation de la matrice Phi
Phi1d(:,1)=-yIdentification(1:end-n1d);
Phi1d(:,2)=uIdentification(n1d+1:end);
% Calcul de theta
theta1d=(Phi1d'*Phi1d)\Phi1d'*Y1d;


%% Ordre 2 Y(t)=-a1*Y(t-1)-a2*Y(t-2)+b1*U(t)
n2d=2;%ordre du modÃ¨le
p2d=n2d+1;% Nombre de paramÃ¨tre Ã  identifier.
% Dimensionnement des matrices
%  theta=zeros(p,1);
%  Y=zeros(N,1);
%  Phi=zeros(N,p);
%Affectation de Y
Y2d=yIdentification(n2d+1:end);
% Affectation de la matrice Phi
Phi2d(:,1)=-yIdentification(1:end-n2d);
Phi2d(:,2)=-yIdentification(2:end-n2d+1);
Phi2d(:,4)=uIdentification(n2d:end-1);
Phi2d(:,3)=uIdentification(n2d+1:end);

% Calcul de theta
theta2d=(Phi2d'*Phi2d)\(Phi2d'*Y2d);

%% Ordre 3  Y(t)=-a1*Y(t-1)-a2*Y(t-2)-a3*Y(t-3)+b1*U(t)
n3d=3; %ordre du modÃ¨le
p3d=n3d+1;% Nombre de paramÃ¨tre Ã  identifier.
% Dimensionnement des matrices
%  theta=zeros(p,1);
%  Y=zeros(N,1);
%  Phi=zeros(N,p);
%Affectation de Y
Y3d=yIdentification(n3d+1:end);
% Affectation de la matrice Phi
Phi3d(:,1)=-yIdentification(1:end-n3d);
Phi3d(:,2)=-yIdentification(2:end-n3d+1);
Phi3d(:,3)=-yIdentification(3:end-n3d+2);
Phi3d(:,4)=uIdentification(n3d-1:end-2);
```

```
Phi3d(:,5)=uIdentification(n3d:end-1);
Phi3d(:,6)=uIdentification(n3d+1:end);

% Calcul de theta
theta3d=(Phi3d'*Phi3d)\Phi3d'*Y3d;

%% Ordre 4 Y(t)=-a1*Y(t-1)-a2*Y(t-2)-a3*Y(t-3)-a4*Y(t-4)+b1*U(t)
n4d=4; %ordre du modÃ¨le
p4d=n4d+1;% Nombre de paramÃ¨tre Ã  identifier.
% Dimensionnement des matrices
%  theta=zeros(p,1);
%  Y=zeros(N,1);
%  Phi=zeros(N,p);
%Affectation de Y
Y4d=yIdentification(n4d+1:end);
% Affectation de la matrice Phi
Phi4d(:,1)=-yIdentification(1:end-n4d);
Phi4d(:,2)=-yIdentification(2:end-n4d+1);
Phi4d(:,3)=-yIdentification(3:end-n4d+2);
Phi4d(:,4)=-yIdentification(4:end-n4d+3);
Phi4d(:,5)=uIdentification(n4d-2:end-3);
Phi4d(:,6)=uIdentification(n4d-1:end-2);
Phi4d(:,7)=uIdentification(n4d:end-1);
Phi4d(:,8)=uIdentification(n4d+1:end);

% Calcul de theta
theta4d=(Phi4d'*Phi4d)\Phi4d'*Y4d;

%% SystÃ¨mes trouvÃ©s
sysARXD1=tf(theta1d(end)',[1 theta1d(1:end-1)'],Ts,'Variable','z^-1');
sysARXD2=tf(theta2d(3:end)',[1 theta2d(1:2)'],Ts,'Variable','z^-1');
sysARXD3=tf(theta3d(4:end)',[1 theta3d(1:3)'],Ts,'Variable','z^-1');
sysARXD4=tf(theta4d(5:end)',[1 theta4d(1:4)'],Ts,'Variable','z^-1');

%% Simulation avec les donnÃ©es
%ordre 1
yARXD11=lsim(sysARXD1,uIdentification,tempsId);%donnee identification
yARXD12=lsim(sysARXD1,uValidation,tempsId);%donnee validation

%ordre 2
yARXD21=lsim(sysARXD2,uIdentification,tempsId);%donnee identification
yARXD22=lsim(sysARXD2,uValidation,tempsId);%donnee validation

%ordre 3
yARXD31=lsim(sysARXD3,uIdentification,tempsId);%donnee identification
yARXD32=lsim(sysARXD3,uValidation,tempsId);%donnee validation

%ordre 4
yARXD41=lsim(sysARXD4,uIdentification,tempsId);%donnee identification
yARXD42=lsim(sysARXD4,uValidation,tempsId);%donnee validation

%% Les critÃ¨res

%ordre 1
critArxD11=(1/N)*norm((yIdentification-yARXD11),2)^2;
critArxD12=(1/N)*norm((yValidation-yARXD12),2)^2;

%ordre 2
critArxD21=(1/N)*norm((yIdentification-yARXD21),2)^2;
critArxD22=(1/N)*norm((yValidation-yARXD22),2)^2;
```

```
%ordre 3
critArxD31=(1/N)*norm((yIdentification-yARXD31),2)^2;
critArxD32=(1/N)*norm((yValidation-yARXD32),2)^2;

%ordre 1
critArxD41=(1/N)*norm((yIdentification-yARXD41),2)^2;
critArxD42=(1/N)*norm((yValidation-yARXD42),2)^2;
%% Les profils des reponses
figure();hold on
fId=plot(tempsId,yIdentification,'k-');
fVal=plot(tempsId,yValidation,'c-');
%ordre 1
f1Id=plot(tempsId,yARXD11,'b:');
f1Val=plot(tempsId,yARXD12,'b-');

%ordre 2
f2Id=plot(tempsId,yARXD21,'g:');
f2Val=plot(tempsId,yARXD22,'g-');

%ordre 3
f3Id=plot(tempsId,yARXD31,'y:');
f3Val=plot(tempsId,yARXD32,'y-');

%ordre 4
f4Id=plot(tempsId,yARXD41,'r:');
f4Val=plot(tempsId,yARXD42,'r-');
legend([fId,fVal,f1Id,f1Val,f2Id,f2Val,f3Id,f3Val,f4Id,f4Val],...
       'Sortie Reellle identification',...
       'Sortie Reelle validation',...
       'ordre 1 identification','Ordre 1 validation',...
       'ordre 2 identification','Ordre 2 validation',...
       'ordre 3 identification','Ordre 3 validation',...
       'ordre 4 identification','Ordre 4 validation');
xlabel('temps')
ylabel('Reponses');
title('Modele ARX decale')
```