Identification des systèmes
Compte-rendu des TD sous Matlab

Léa Yeromonahos

L3 Sciences pour l'ingénieur UFR Sciences et Techniques 12 mai 2022

Table des matières

Exercices d'introduction

Exercice 1

On dispose d'une série de données de N mesures $\{y_k\}_{1\leq k\leq N}$ d'une grandeur y en fonction du temps à intervalles réguliers. Les mesures sont espacées de 1 mn :

t_k	0	1	2	3	4	5	6	7	8	9
y_k	-1.04	1.21	3.36	4	5.43	8.45	10.67	13.61	15.12	16.11

TABLE 1 – Tableau des données

On affiche les y_k selon le temps :

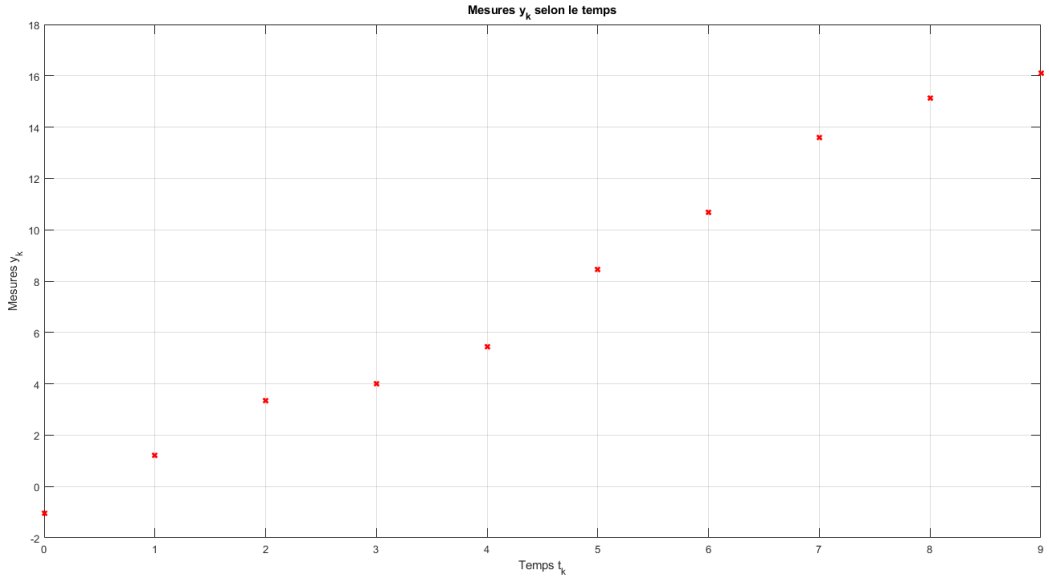

FIGURE 1 – Mesures y_k en fonction du temps

Pour calculer la droite qui approche au mieux au sens des Moindres Carrés Ordinaires (MCO), on choisit un modèle qui correspond au mieux aux données, ici une droite d'équation :

$$y = at_k + b \tag{1}$$

Que l'on peut aussi écrire :

$$y = M\theta \tag{2}$$

où M est la matrice contenant les (t_k + 1), de dimension $N \times 2$, et θ la matrice contenant les a et b, donc de dimension 2×1.
Pour estimer θ selon notre modèle y, on va calculer des a et b optimaux de façon à obtenir un critère minimal, c'est-à-dire le moins d'écart possible entre l'estimation et la réalité. Soit le calcul du critère J :

$$\begin{aligned} J &= e^T e \\ e &= y_k - y \end{aligned} \tag{3}$$

où y_k sont nos mesures et y le modèle. e est ce qu'on appelle l'erreur.
On crée donc notre matrice M, telle que :

```
N = 10;
t = (0:1:N-1)';

M = [t ones(N,1)]
```

M contient toutes les valeurs de t et 1, et est de dimension 10×2. Pour créer θ, on utilise M et nos y_k :

```
y = [-1.04 1.21 3.36 4 5.43 8.45 10.67 13.61 15.12 16.11]';
theta = [inv(M'*M)*M'*y]
```

On peut désormais calculer y estimé, et l'afficher :

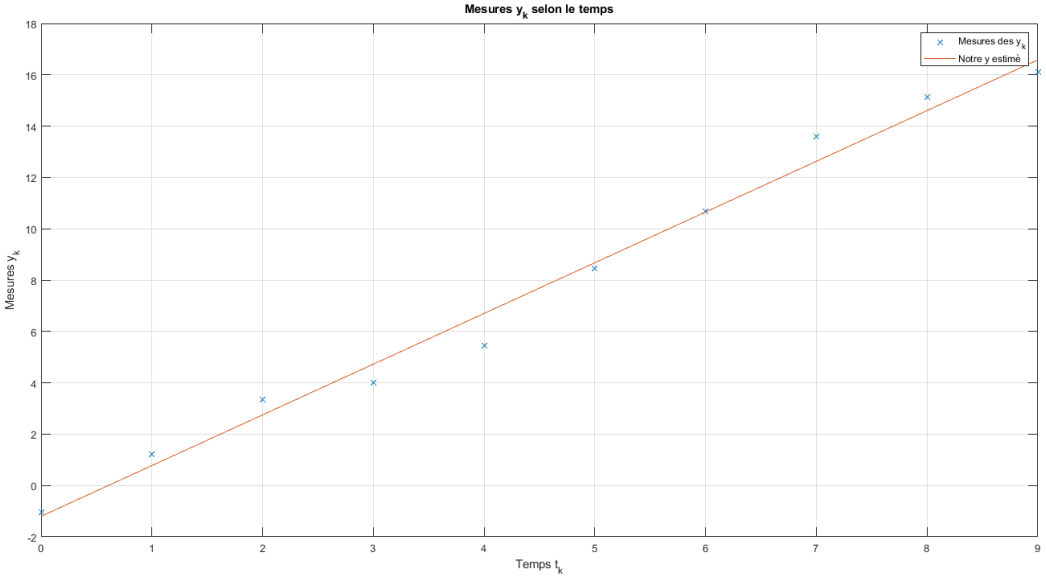

FIGURE 2 – Estimation de la droite y

La droite y estimée semble correspondre aux mesures y_k.
On affiche maintenant les erreurs calculées :

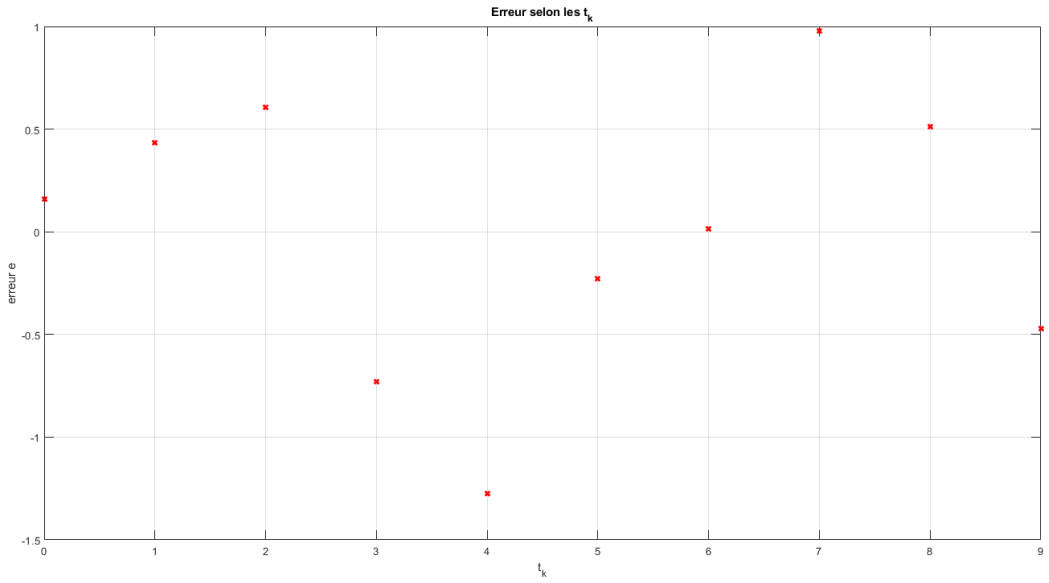

FIGURE 3 – Erreurs d'estimation de la droite y

Ces erreurs correspondent à la distance entre l'estimation et la mesure. On peut vérifier que le calcul est correct en regardant par exemple l'écart entre le 8ième point mesuré (à t = 7) et la droite estimée, il correspond bien à la plus grande valeur de e.
Le critère J calculé est de 4.2331.

On veut maintenant estimer y avec une parabole. On change donc de modèle :

$$y = at_k^2 + bt_k + c \tag{4}$$

On reprend la même méthode, sauf que M vaut cette fois :

```
M_p = [t.^2 t ones(N,1)];
```

et est de dimension $N \times 3$, ce qui impacte la dimension de θ, soit 3×1. On obtient la figure suivante :

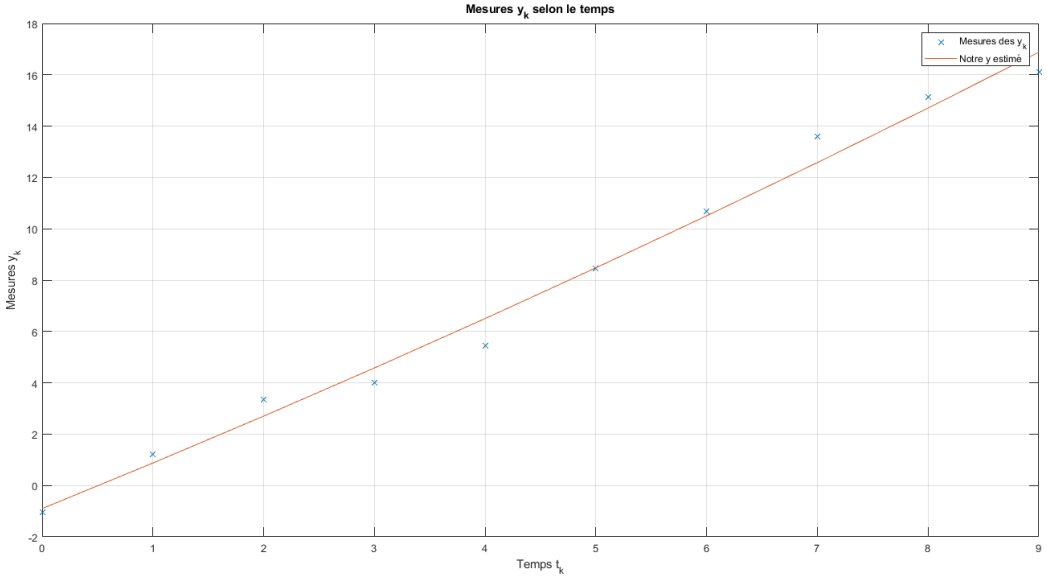

FIGURE 4 – Estimation de la parabole y

L'estimation semble meilleure, ce qui est vérifié par le calcul du critère soit J = 3.9051, meilleur (car inférieur) au précédent.

Pour un polynôme d'ordre 3, la matrice M est :

```
M_p3 = [t.^3 t.^2 t ones(N,1)];
```

Ce qui donne :

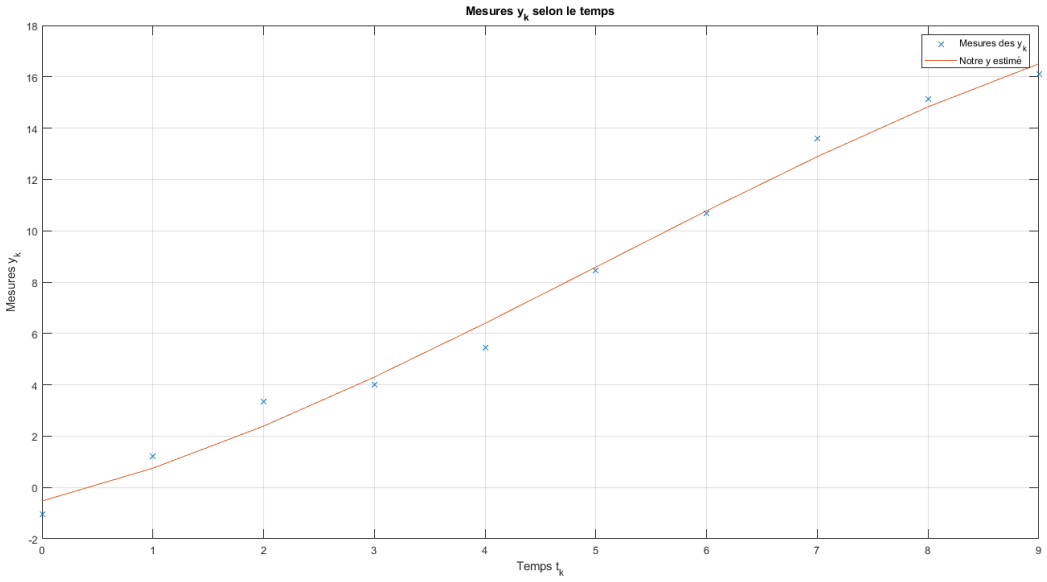

FIGURE 5 – Estimation avec un polynôme d'ordre 3 de y

L'estimation est encore meilleure, avec J = 3.2283. Plus le degré du polynôme augmente, plus il semble que les modèles se rapprochent des mesures.

On considère dorénavant que les mesures bruitées sont générées par le processus $y = at + b$ avec $a = 2$, $b = -1$ et un bruit gaussien centré d'écart-type $\sigma = 1.5$.
On génère 100 réalisations différentes de mesure :

```
N = 100;
t = (0:1:N-1)';
a = 2;
b = -1;
sigma = 1.5;
bruit = sigma*randn(N,1);
for k = 1:N
    y(k) = a*t(k)+b + bruit(k);
end
y = y';
```

On utilise les 3 MCO précédentes (droite, parabole et polynôme d'ordre 3) et on obtient visuellement la même chose pour les 3 :

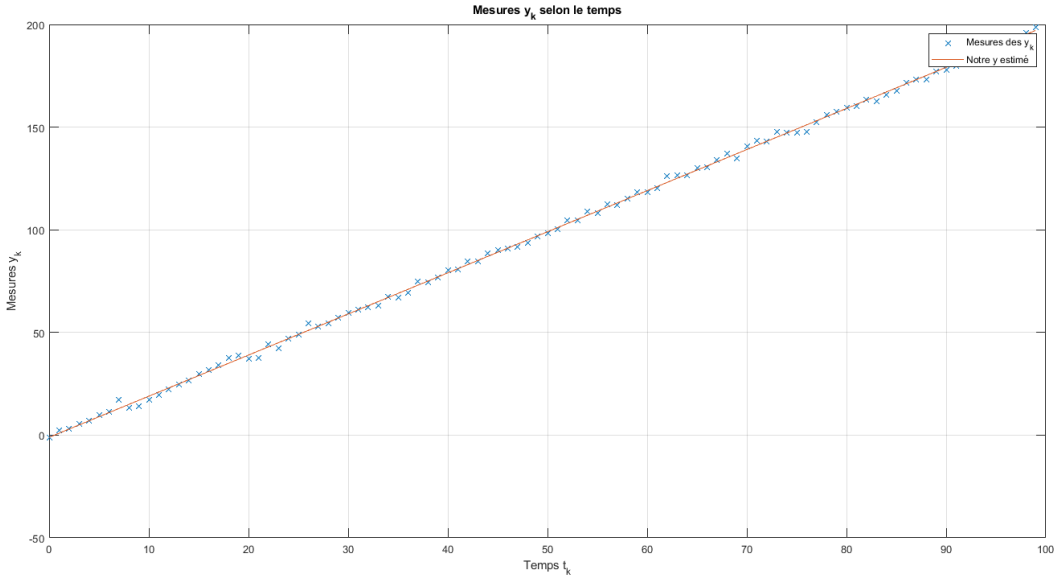

FIGURE 6 – Estimations des mesures bruitées

L'estimation dans les trois cas se rapproche d'une droite. Lorsque l'on regarde les erreurs :

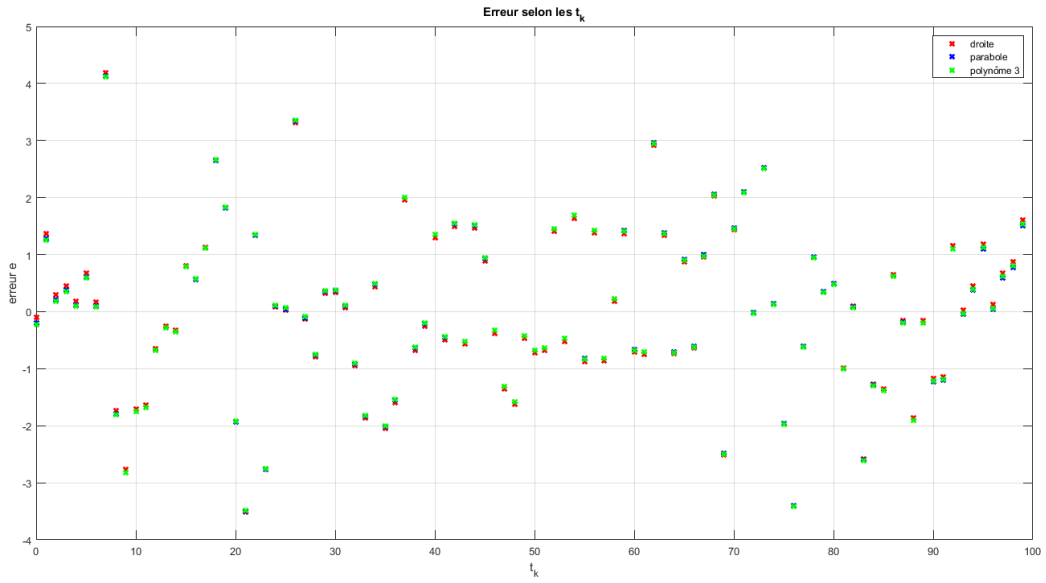

FIGURE 7 – Erreurs des 3 MCO pour les mesures bruitées

On observe que la parabole et le polynôme d'ordre 3 sont globalement moins bon que la droite. Lorsque l'on calcule les moyennes de chaque, on obtient e = 2.21e-14 pour la droite, e = -5.3e-14 pour la parabole et e = 2.11e-11 pour le polynôme d'ordre 3, ce qui confirme l'observation. L'estimation basée sur le modèle d'une droite possède l'erreur la plus petite dans ce cas.

Exercice 2

On dispose des données du recensement de la population d'une ville, tous les 10 ans, de 1900 à 1930 :

année	1900	1910	1920	1930
population	1000	1050	1134	1201

TABLE 2 – Population selon les années (1900 à 1930)

On souhaite modéliser cette progression de manière à prédire la population en 1940.
On commence par représenter la population selon les années, afin de déterminer un modèle :

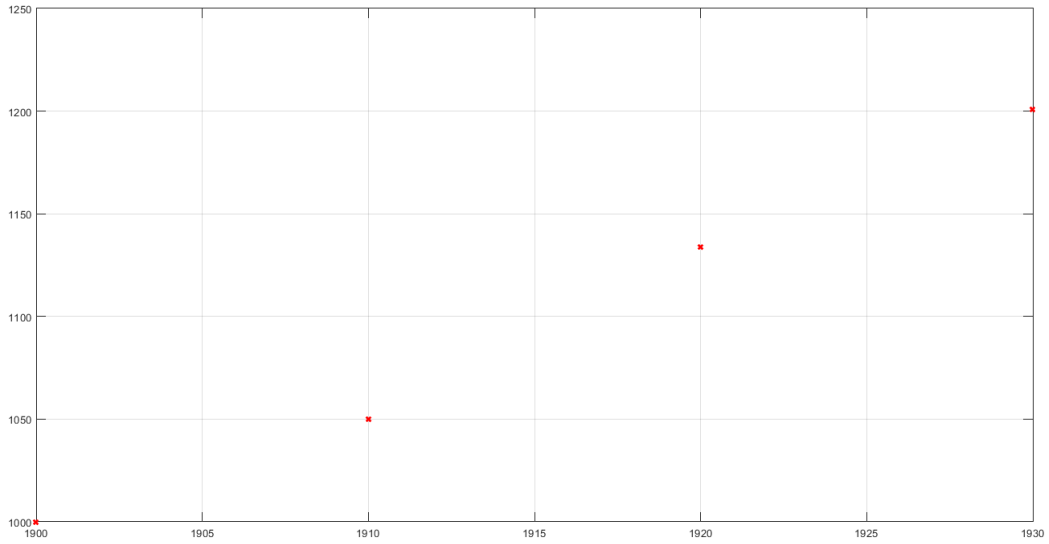

FIGURE 8 – Population selon les années

On prend donc comme modèle une droite, $y = at + b$ où t correspond ici aux années. Soit comme dans l'exercice précédent $y = M\theta$, avec :

```
a = [1900 1910 1920 1930]';
y = [1000 1050 1134 1201]';
N = length(a);
M = [a ones(N,1)];
theta = [inv(M'*M)*M'*y];
```

On crée donc notre y estimé (différent de la variable y du programme) et on l'affiche :

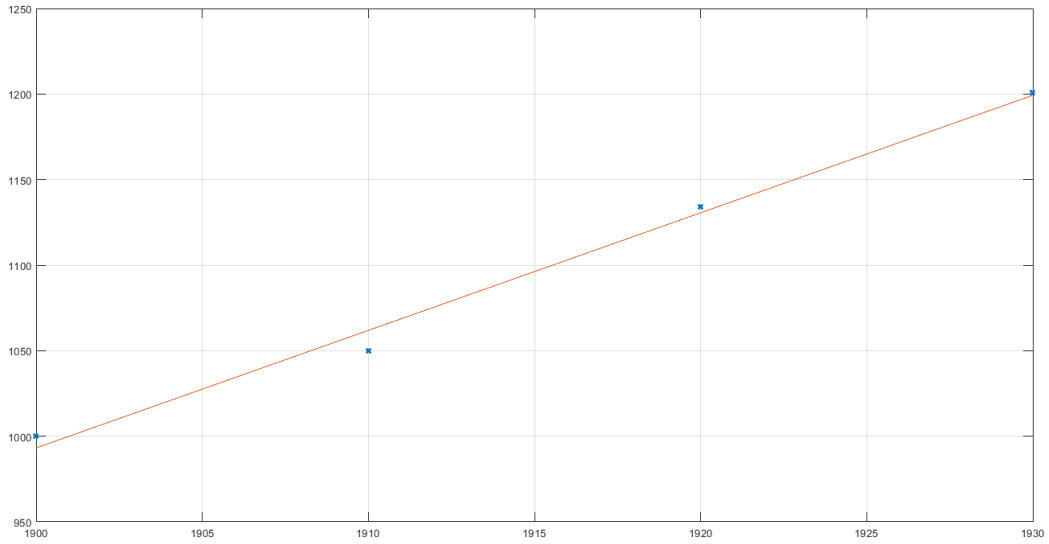

FIGURE 9 – Estimation avec un modèle linéaire

Maintenant que l'on a notre modèle, on peut essayer de prédire la variation de la population, c'est-à-dire estimer des

données pour 1940 et plus.
Pour cela, on crée un nouveau vecteur contenant le nombre d'années totale souhaité, puis une nouvelle matrice M contenant ces années. Enfin, on crée le nouveau y estimé :

```
a_p = [1900 1910 1920 1930 1940 1950]';
N_p = length(a_p);
M_p = [a_p ones(N_p,1)];
y_p = M_p*theta;
```

On affiche maintenant la prédiction :

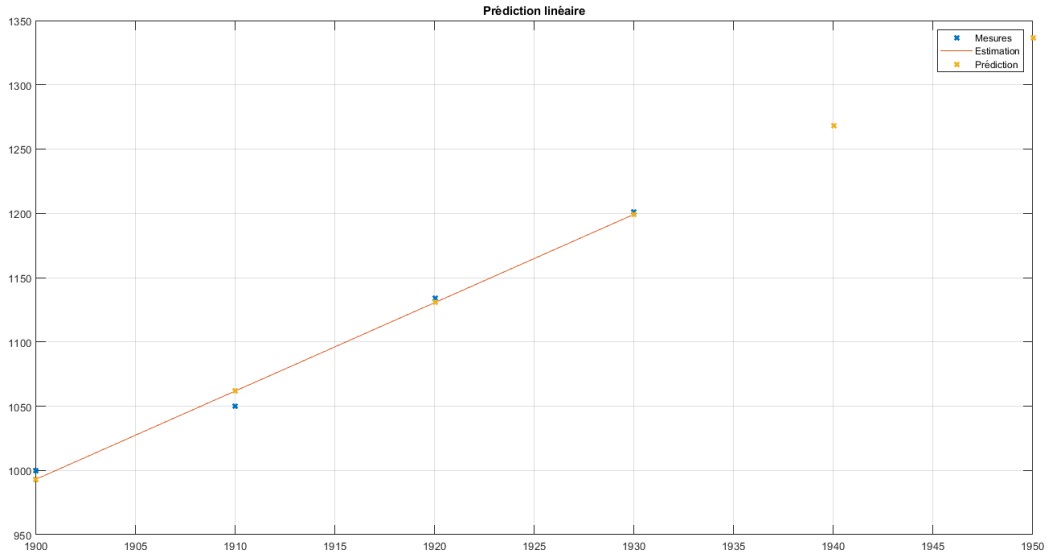

FIGURE 10 – Prédiction (en jaune) linéaire de la population

D'après cette prédiction, la population devrait atteindre les 1268 habitants en 1940.

En utilisant un modèle quadratique (polynôme d'ordre 2), c'est-à-dire en changeant M et θ :

```
M_q = [a.^2 a ones(N,1)]; %estimation
M_qp = [a_p.^2 a_p ones(N_p,1)]; %prediction
theta_q = [inv(M_q'*M_q)*M_q'*y];
y_q = M_q*theta_q;
```

On obtient :

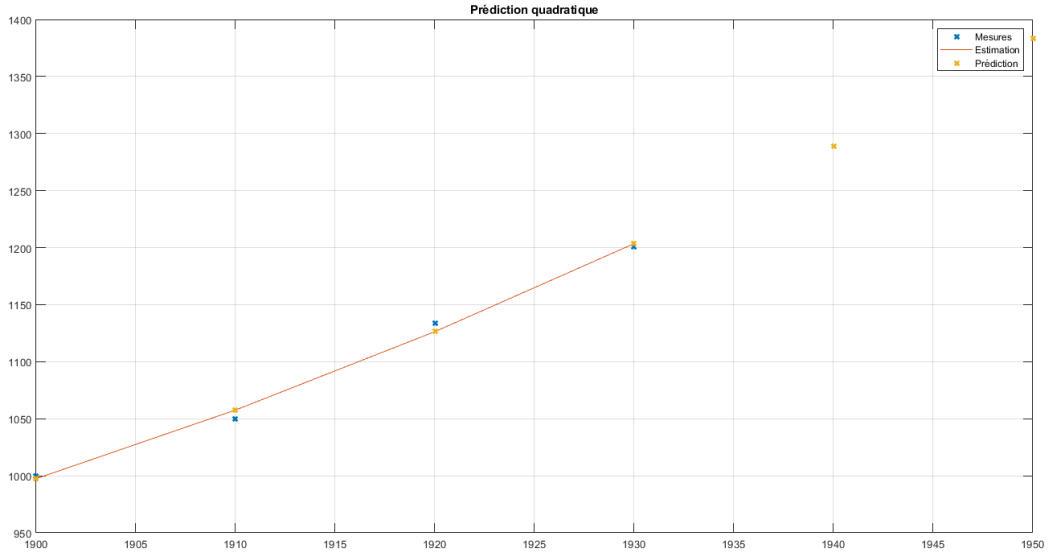

FIGURE 11 – Prédiction (en jaune) quadratique de la population

Selon ce modèle, on aurait 1289.25 habitants en 1940, soit légèrement plus qu'avec le modèle précédent.

Enfin, pour un polynôme d'ordre 3 :

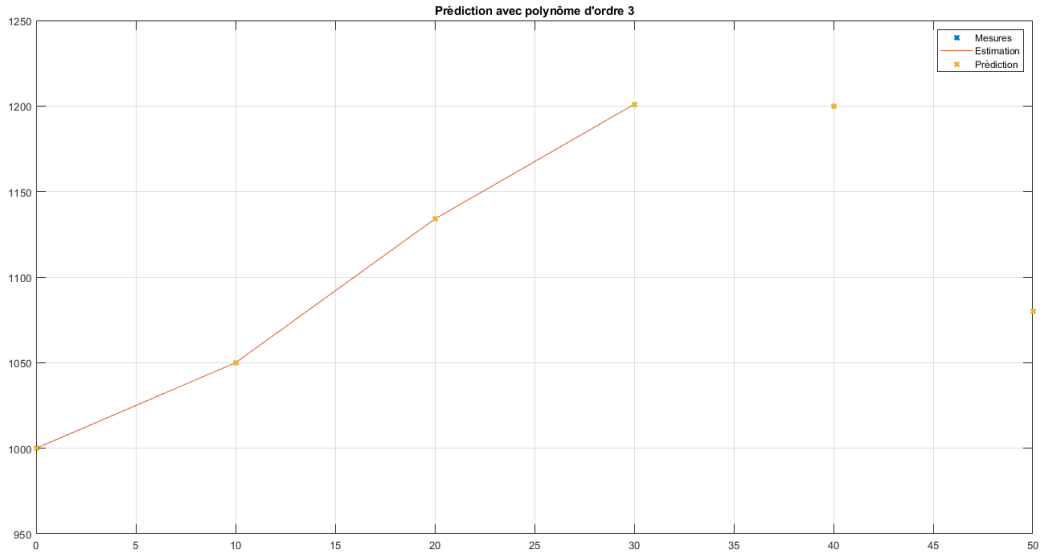

FIGURE 12 – Prédiction (en jaune) avec polynôme d'ordre 3 de la population

Afin de palier au problème de conditionnement des matrices, on a soustrait 1900 aux années.
Cette-fois, la population stagne en 1940 et diminue même drastiquement en 1950. Néanmoins, l'estimation se confond avec les mesures (resp. en rouge et bleu), contrairement aux modèles précédents.

Exercice 2 bis

On dispose des données du recensement de la population d'une ville, tous les 10 ans, de 1900 à 1980 :

Année	1900	1910	1920	1930	1940	1950	1960	1970	1980
Population	1000	1050	1104	1158	1212	1268	1323	1381	1400

TABLE 3 – Population selon les années (1900 à 1980)

En utilisant un modèle linéaire :

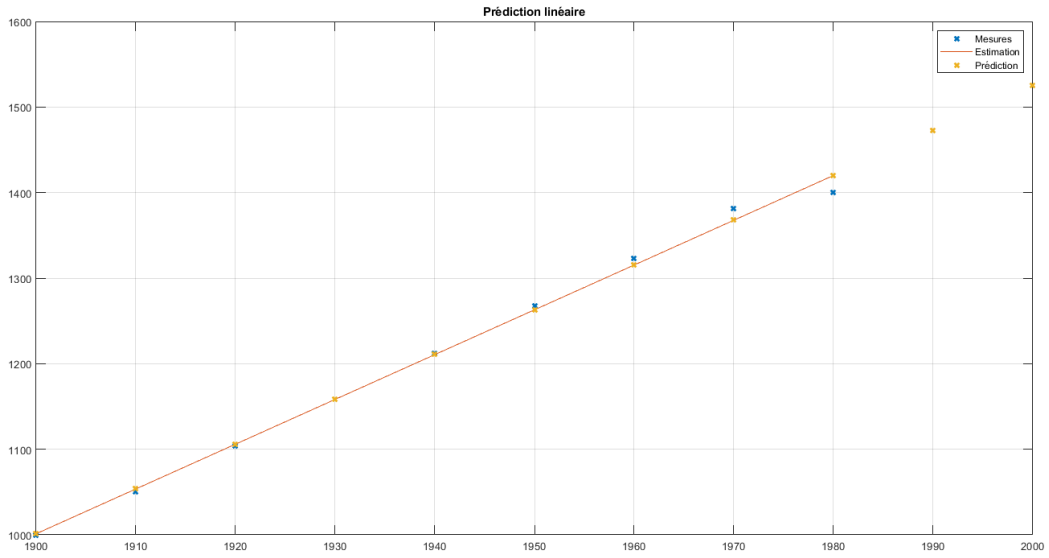

FIGURE 13 – Prédiction linéaire (en jaune) bis

La prédiction indique qu'en 2000 la population aura augmenté jusqu'à 1524 habitants.
Pour un modèle quadratique :

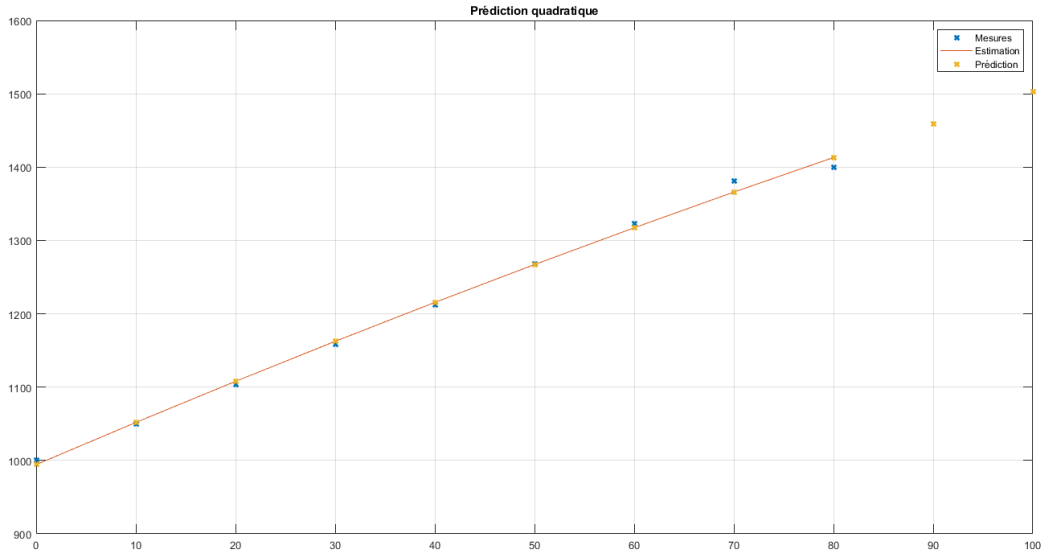

FIGURE 14 – Prédiction quadratique (en jaune) bis

Ici, la population en 2000 aura atteint les 1503 habitants, un peu inférieur au modèle linéaire.

Pour un polynôme d'ordre 3 :

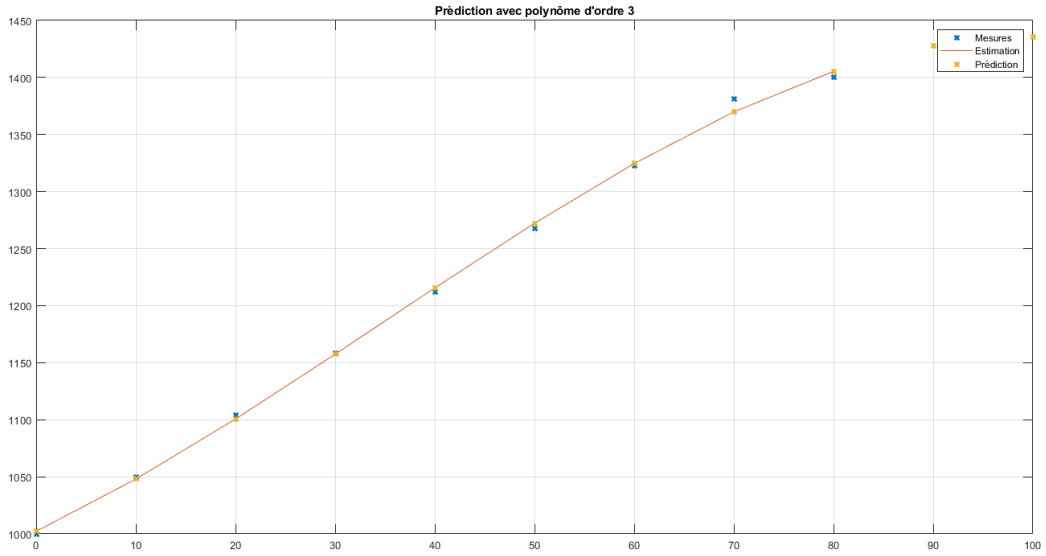

FIGURE 15 – Prédiction (en jaune) avec polynôme d'ordre 3

Cette fois-ci, la croissance de la population ralentit pour atteindre les 1435 habitants en 2000. On peut supposer qu'elle va ensuite diminuer.
Enfin, pour un polynôme d'ordre 7 :

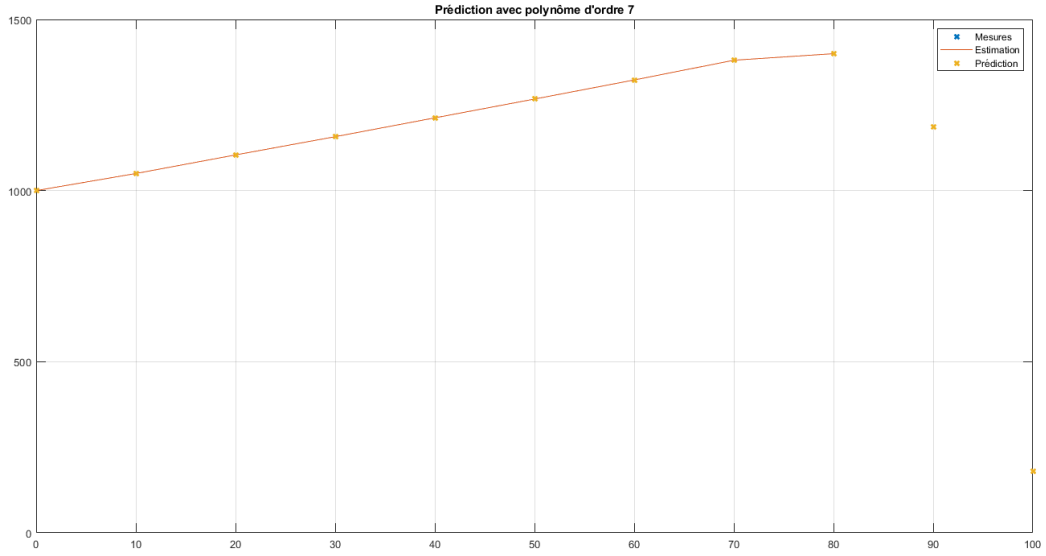

FIGURE 16 – Prédiction (en jaune) avec polynôme d'ordre 7

Avec ce modèle, la population atteint les 178 habitants en 2000, ce qui semble évidemment faux, en dehors de pandémie de peste et/ou de choléra, le nombre d'habitants ne devrait pas chuter aussi drastiquement.
Plus le degré du polynôme est élevé, plus les estimations sont proches des mesures, mais moins les prédictions sont réalistes.

Moindres Carrés Ordinaires (MCO)

Exercice 3

On mesure une tension sinusoïdale de fréquence f_0 connue et l'on veut en estimer l'amplitude et la phase à l'origine. Le signal mesuré est :

$$x_k = a\sin(2\pi f_0 t_k + \phi) + \varepsilon_k \tag{5}$$

où $0 \leq k \leq N-1$, ε_k est un bruit de mesure gaussien centré de variance σ^2, $t_k = k\Delta T$ et ΔT = 1 ms.

On peut écrire l'équation comme :

$$x_k = A_k^T \begin{bmatrix} a\cos(\phi) \\ a\sin(\phi) \end{bmatrix} \tag{6}$$

en utilisant la relation sin(a + b) = sin(a)cos(b) + cos(a)sin(b), et ainsi :

$$\begin{aligned} a\sin(2\pi f_0 t_k + \phi) &= a[\sin(2\pi f_0 t_k)\cos(\phi) + \cos(2\pi f_0 t_k)\sin(\phi)] \\ A_k^T &= \begin{bmatrix} \sin(2\pi f_0 t_k) \\ \cos(2\pi f_0 t_k) \end{bmatrix} \end{aligned} \tag{7}$$

En prenant N = 500, a = 3 V, $\phi = \frac{\pi}{5}$, σ = 1 V et $f_0 = 20$ Hz, on peut représenter le signal x_k (dit signal "vrai") et y_k (soit le signal mesuré) :

```
N = 500;
k = [1:N]';
dt = 0.001;
t = (k-1)*dt;
a = 3;
phi = pi/5;
f0 = 20;

sigma = 1;
e = sigma*randn(N,1);
x = a*sin(2*pi*f0*t + phi);
y = x + e;
```

On obtient donc :

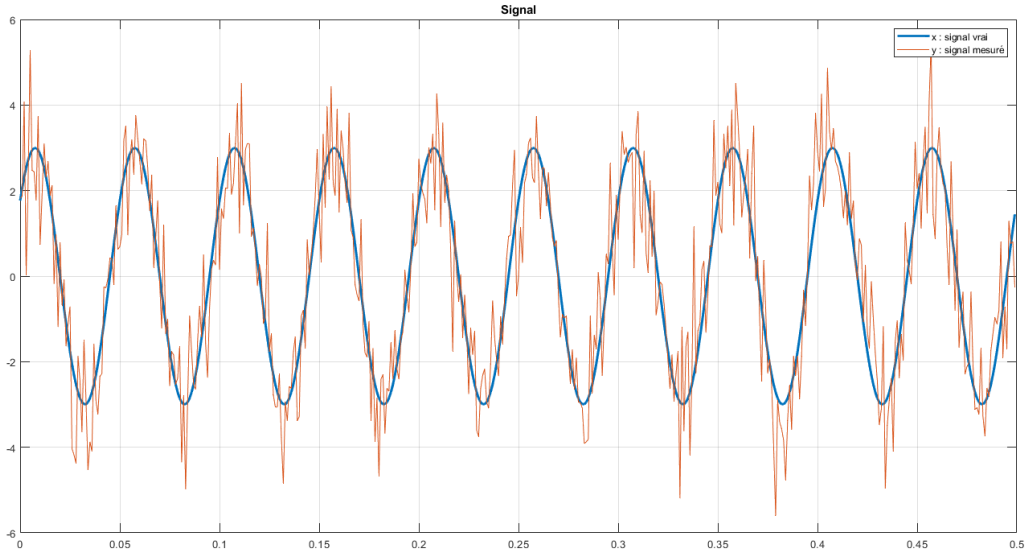

FIGURE 17 – Signaux "vrai" et mesuré

Maintenant que l'on sait à quoi ils ressemblent, on peut estimer θ au moyen des MCO. Pour cela, on crée notre matrice M, puis on calcule θ :

```
M = [sin(2*pi*f0*t) cos(2*pi*f0*t)];
theta = [inv(M'*M)*M'*y];
y_c = M*theta;
```

On obtient $\theta = \begin{bmatrix} 2.5231 \\ 1.7226 \end{bmatrix}$, et on peut vérifier l'estimation :

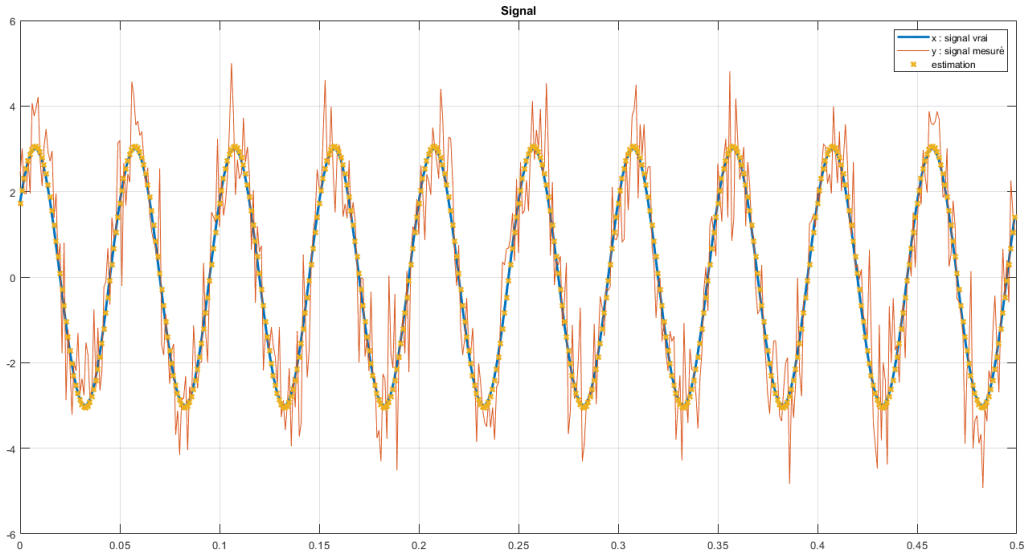

FIGURE 18 – Estimation (en jaune) des mesures

Notre θ semble correct. On peut maintenant calculer ses paramètres a et ϕ estimés, tels que :

```
phi_c = atan(theta(2)/theta(1))
a_c = sqrt(theta(1)^2 + theta(2)^2)
```

et on obtient $\phi = 0.60$ et a = 3.14, ils sont proches des valeurs originales (resp. 0.6283 et 3).

Si on augmente N, la précision augmente, comme par exemple avec N = 2000 et $\sigma = 1$, où on a $\phi = 0.6197$ et a = 3.0123.
Évidemment, sans bruit, ($\sigma = 0$), on retrouve exactement les mêmes valeurs.

Exercice 4

On mesure les N positions successives x_k d'un mobile se déplaçant à vitesse constante v sur l'axe Ox et on veut en estimer la vitesse et la position initiale v_0. Le signal mesuré est :

$$x_k = x_0 + vt_k + \varepsilon_k \tag{8}$$

où $0 \leq k \leq N-1$, ε_k est un bruit de mesure gaussien centré de variance σ^2, $t_k = k\Delta T$ et ΔT= 100 ms.
En prenant N = 100, x_0 = 5 m, v = 10 m/s et σ = 3 m, on peut représenter le signal x_k (dit signal "vrai") et y_k (soit le signal mesuré) :

```
N = 100;
k = [1:N]';
dt = 100*10^-3;
t = (k-1)*dt;
x0 = 5;
v = 10;

sigma = 3;
e = sigma*randn(N,1);
x = x0 + v*t;
y = x + e;
```

On obtient les signaux :

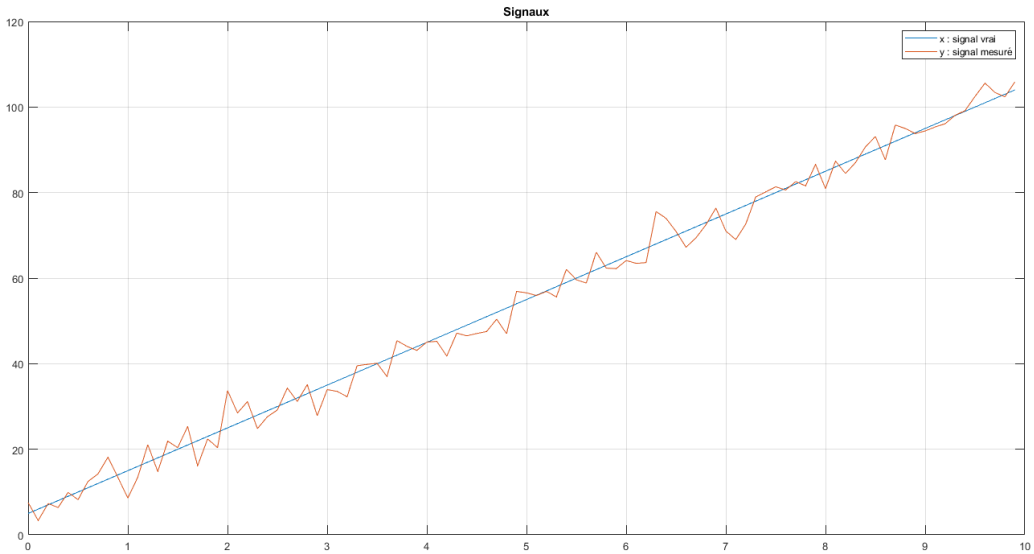

FIGURE 19 – Signaux vrai et mesuré

Le vecteur paramètre θ à estimer est $\theta = \begin{bmatrix} v \\ x_0 \end{bmatrix}$.
On peut maintenant utiliser la méthode des MCO, vue précédemment. Soit M notre matrice :

```
M = [t ones(N,1)];
theta_c = [inv(M'*M)*M'*y];
y_c = M*theta_c;
```

On représente notre estimation y_c :

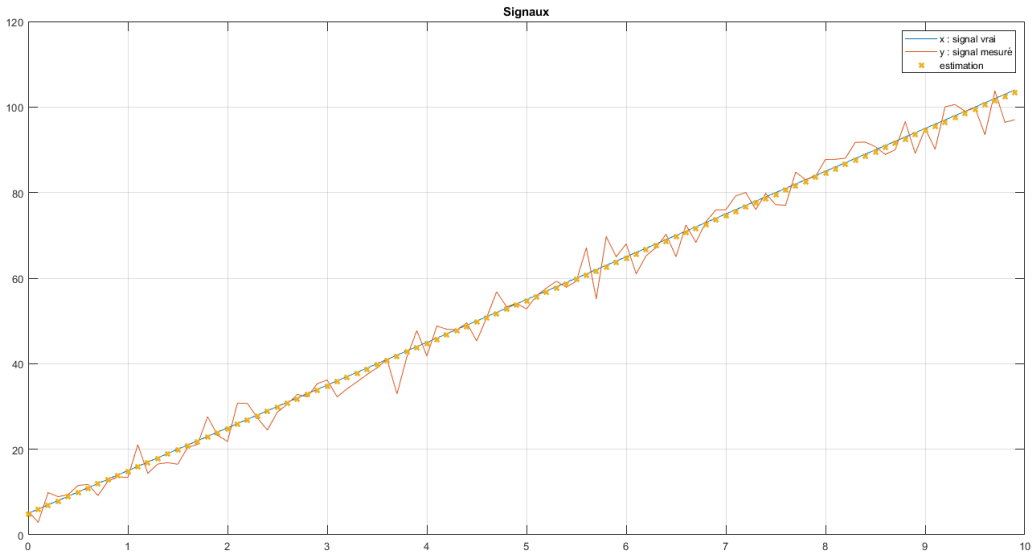

FIGURE 20 – Estimation des N positions du mobile

et l'on vérifie le vecteur $\theta_c = \begin{bmatrix} 9.95 \\ 4.88 \end{bmatrix}$, soit proche de l'original $\theta = \begin{bmatrix} 10 \\ 5 \end{bmatrix}$.

Exercice 3 bis

On mesure une tension sinusoïdale de fréquence f_0 connue et l'on veut en estimer l'amplitude et la phase à l'origine. Le signal mesuré est :

$$x_k = a\sin(2\pi f_0 t_k + \phi) + \varepsilon_k \tag{9}$$

où $0 \leq k \leq N-1$, ε_k est un bruit de mesure gaussien centré de variance σ^2, $t_k = k\Delta T$ et ΔT = 1 ms. On prendre N = 2000.
Pour $0 \leq k \leq 750$, a = 3.
Pour $1250 \leq k \leq N$, a = 5.
Enfin, pour $750 \leq k \leq 1250$, a varie linéairement.
La valeur de ϕ ne varie pas et vaut $\frac{\pi}{5}$, σ = 0.1 V et f_0 = 10 Hz.
Le modèle réel est donc :

$$x_k = a_k\sin(2\pi f_0 t_k + \phi) + \varepsilon_k \tag{10}$$

Pour tracer le signal "vrai" et le mesuré, on doit faire varier a :

```
N = 2000;
k = [1:N]';
```

```
t = (k-1)*0.001;
alpha = ((5-3)/(1250-750));
beta = 3-alpha*750;
x = zeros(N,1);
y = zeros(N,1);
for i = 1:N
    if k(i) <= 750
        a(i) = 3;
    elseif 1250 <= k(i)
        a(i) = 5;
    else
        a(i) = alpha*k(i) + beta;
    end
    x(i) = a(i)*sin(2*pi*f0*t(i) + phi);
    y(i) = x(i) + e(i);
end
```

On crée α et β qui sont les coefficients de notre droite pour a entre 750 et 1250.
On crée ensuite les vecteur x et y, puis on assigne les valeurs de a en fonction du cahier des charges. Pour la pente, on crée l'équation $a_k = \alpha k + \beta$.
On peut maintenant visualiser le signal vrai, le mesuré, et vérifier que notre a est bon :

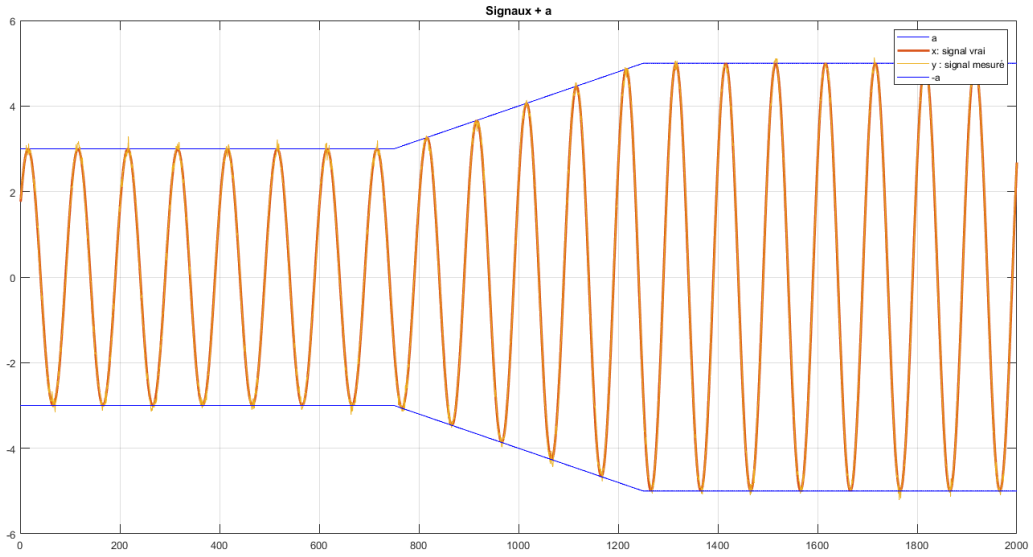

FIGURE 21 – Signaux vrai (rouge) et mesuré (jaune), et valeurs de a

Les signaux suivent bien les bornes de a. Comme le bruit est léger, on distingue peu la mesure.

On veut réaliser une estimation par fenêtre glissante de ϕ et a_k, c'est-à-dire que tous les Nf points (ici 100) précédent un instant t_k, on réalise une estimation.
Pour cela, on crée notre matrice M puis :

```
M = [sin(2*pi*f0*t) cos(2*pi*f0*t)];
Nf = 100;
u = 1:Nf;
for i = 1:(N-Nf+1)
    Mf = M(u,:);
    yf = y(u);
    theta_f(:,i) = inv(Mf'*Mf)*Mf'*yf;
```

```
    u = u + 1;
    a_f(i) = sqrt(theta_f(1,i)^2 + theta_f(2,i)^2);
    phi_f(i) = atan2(theta_f(2,i),theta_f(1,i));
end
```

On crée un vecteur u qui est notre fenêtre, puis pour i = 1 à N-Nf+1 soit 1901, on définit une nouvelle matrice Mf qui contient les lignes 1 à Nf et toutes les colonnes, puis un yf étant y sur 1 :Nf, et enfin le calcul du θ.
On incrémente notre vecteur u de 1, et ainsi on fait glisser notre fenêtre jusqu'au maximum de données disponibles pour l'opération.
Pour estimer les paramètres de θ, on calcule a selon la i-ème colonne, de même pour ϕ. On peut représenter l'évolution des paramètres selon le temps :

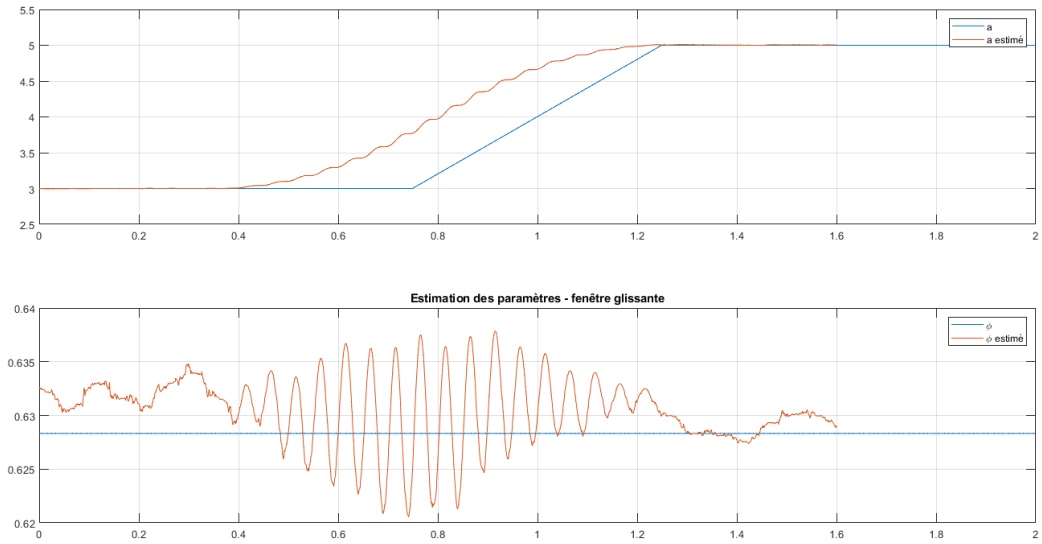

FIGURE 22 – Estimation des paramètres avec la fenêtre glissante

L'estimation de a est bonne mais décalée lors de la pente. Pour ϕ, l'estimation varie très peu.
On peut modifier le pas de la fenêtre Nf : si on l'augmente alors l'estimation est moins bonne, et inversement.

Moindres Carrés Récursifs (MCR)

Exercice 5

On mesure un signal constant dans le temps et l'on veut en estimer la valeur en temps réel. Le signal mesuré est :

$$y_k = a + \varepsilon_k \tag{11}$$

où $0 \leq k \leq N-1$, ε_k est un bruit de mesure gaussien centré de variance $\sigma^2 = 1$, $t_k = k\Delta T$ et $\Delta T = 1$ ms. On prendra N = 2000 et $\sigma = 10$ V.
Pour estimer l'évolution de θ au cours du temps, on utilise d'abord la méthode des MCO, telle que :

```
N = 2000;
a = 10;

sigma = 10;
e = sigma*randn(N,1);
y = a + e;

for i = 1:N
    M = ones(i,1);
    theta(i) = [inv(M'*M)*M'*y(1:i)];
end
```

On crée la matrice M de taille $i \times 1$, remplie de 1. On peut ainsi calculer θ pour chaque M, le tout N fois.
On obtient ainsi :

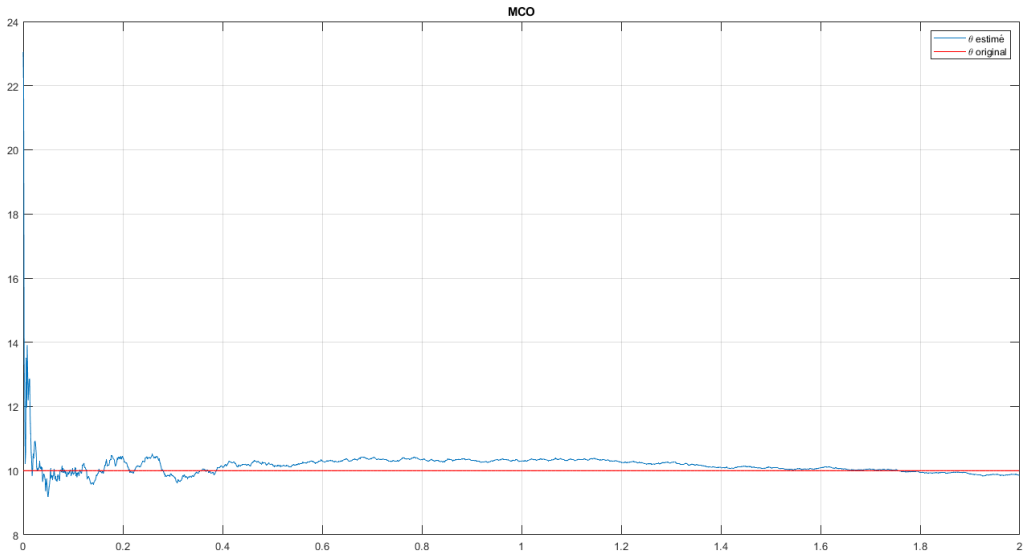

FIGURE 23 – Évolution de θ selon le temps, méthode des MCO

Le paramètre varie fortement sur les premiers temps puis finis par tendre vers la valeur de a, soit 10.

La méthode des Moindres Carrés Récursifs (MCR) implique plusieurs étapes de calcul :
Pour $0 \leq j \leq N$, on commence par calculer le gain de correction k, soit :

$$k_j = \frac{P_{j-1} m_j}{1 + m_j^T P_{j-1} m_j} \tag{12}$$

où les m_j sont les éléments de la matrice M, et P :

$$P_j = P_{j-1} - k_j m_j^T P_{j-1} \tag{13}$$

La matrice P est de dimension $p \times p$.
Vient ensuite le calcul des y_j estimés, tels que :

$$yest_j = m_j^T \theta_{j-1} \tag{14}$$

Et enfin le calcul du θ, actualisé à chaque itération :

$$\theta_j = \theta_{j-1} - k_j(y_j - yest_j) \tag{15}$$

Avant ces étapes, on initialise θ_0 et P_0 :

```
theta_mcr(1) = 0; %initialisation
P = 10^12;
for j = 1:N-1
    m = 1;
    k = (P*m)/(1+m'*P*m);
    P = P - k*m'*P;
    y_est = m'*theta_mcr(j);
    theta_mcr(j+1) = theta_mcr(j) + k*(y(j+1)-y_est);
end
```

Comme notre matrice M ne contient que des 1, de dimension $i \times 1$, on prend $m_j = 1$.
On obtient les résultats suivants :

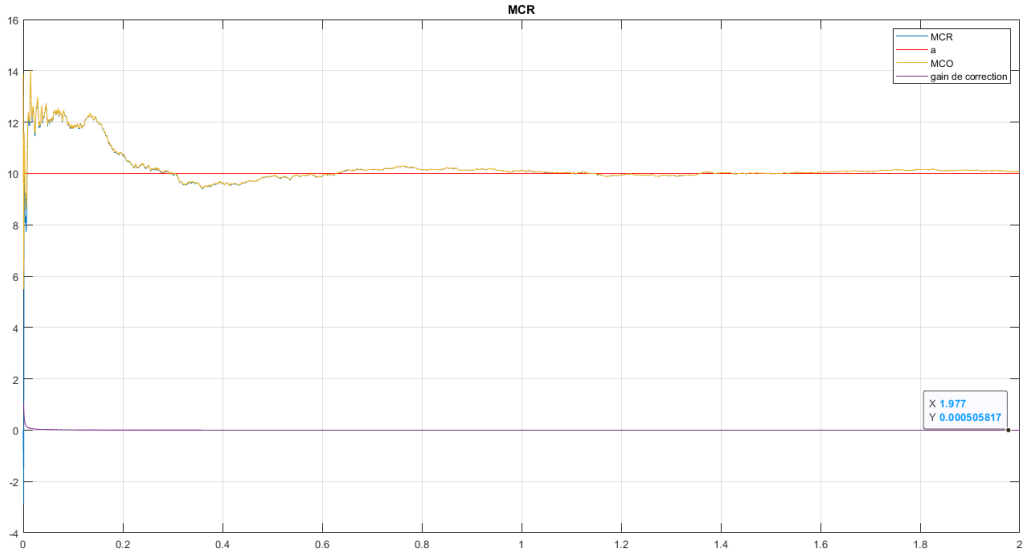

FIGURE 24 – Évolution de θ selon le temps, méthode des MCR

Les deux méthodes semblent avoir le même résultat, sauf au début où les MCR varient moins. Si on compare les temps d'exécution, alors on voit que les MCO prennent 0.0469 s pour leur calcul, contre 0 s (soit un nombre très petit) pour les MCR. Plus N augmente et plus le temps de calcul des MCO augmente, ainsi que les variations de θ, alors que les MCR restent constants.

L'évolution du gain de correction (en violet) nous montre que très vite l'estimation tend vers la réalité, car il est rapidement proche de 0.

Exercice 6

On reprend les données de l'exercice 3, en utilisant cette fois la méthode des MCR.
Soit $y_k = a\sin(2\pi f_0 t_k + \phi) + \varepsilon_k$. On a donc :

```
theta_mcr = zeros(p,N);
P = 10^12*eye(p,p);
theta_mcr(:,1) = [0;0];

for j = 2:N
    m = [sin(2*pi*f0*t(j)) cos(2*pi*f0*t(j))]';
    k = (P*m)/(1+m'*P*m);
    P = P - k*m'*P;
    y_est = m'*theta_mcr(:,j-1);
    theta_mcr(:,j) = theta_mcr(:,j-1) + k*(y(j)-y_est);
    phi_mcr(:,j) = atan2(theta_mcr(2,j),theta_mcr(1,j));
    a_mcr(:,j) = sqrt(theta_mcr(1,j).^2 + theta_mcr(2,j).^2);
end
```

On initialise donc notre θ estimé, de taille $p \times N$, où p est notre nombre de paramètres, ici 2. On initialise aussi P, une matrice $10^{12} \times$identité de dimension $p \times p$, puis la première colonne de notre θ à 0.
Afin de suivre l'évolution des θ, on les conserve aux indices $(:, j)$, soient toutes les lignes et la j-ième colonne.
On obtient ainsi :

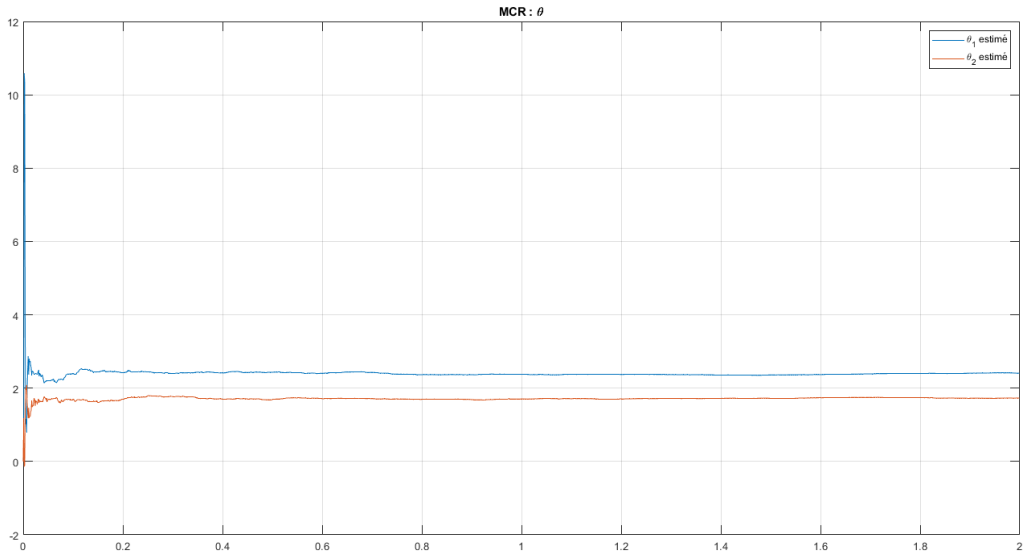

FIGURE 25 – Estimation de θ par MCR

Les valeurs tendent rapidement vers la stabilité. Le calcul et le suivi des paramètres a et ϕ nous permettent de vérifier notre méthode :

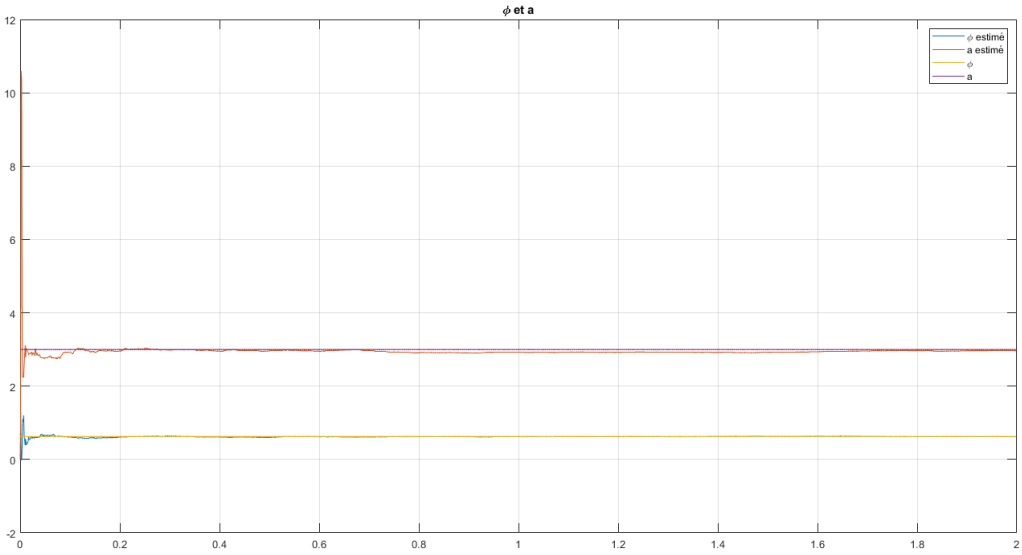

FIGURE 26 – Paramètres ϕ et a, méthode des MCR

On peut dire qu'au bout de 0.2 s, l'estimation se confond avec la réalité.

Exercice 4 bis

On reprend l'exercice 4 en utilisant la méthode des MCR. Soit x_0 et v nos paramètres à estimer, donc p = 2 :

```
theta_mcr = zeros(p,N);
P = 10^12*eye(p,p);
theta_mcr(:,1) = [0;0];

for j = 2:N
    m = [t(j) 1]';
    k = (P*m)/(1+m'*P*m);
    P = P - k*m'*P;
    y_est = m'*theta_mcr(:,j-1);
    theta_mcr(:,j) = theta_mcr(:,j-1) + k*(y(j)-y_est);
    x0_mcr(:,j) = theta_mcr(2,j);
    v_mcr(:,j) = theta_mcr(1,j);
end
```

On obtient, avec la méthode des MCR, pour N = 500 :

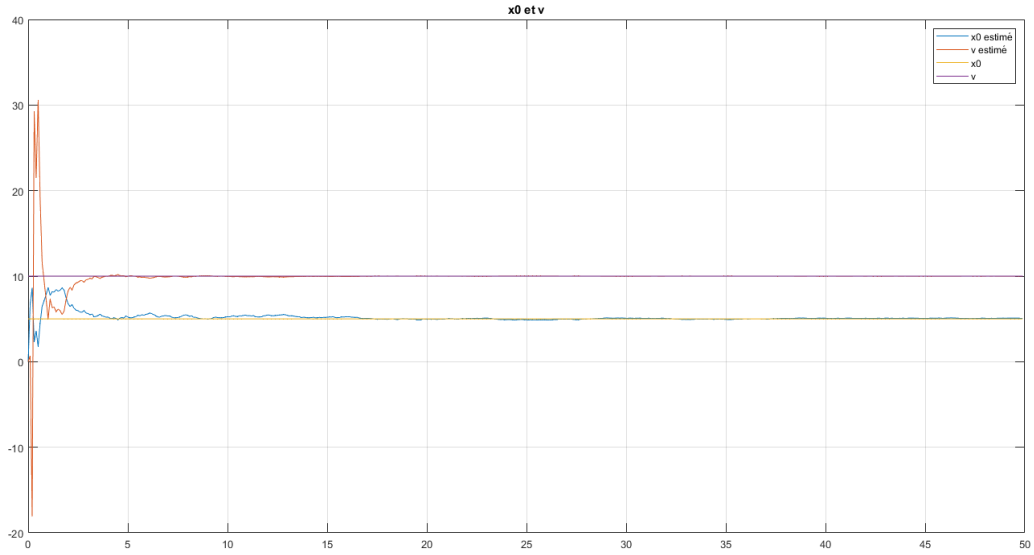

FIGURE 27 – Estimation des paramètres x_0 et v, pour N = 500, méthode des MCR

On observe de fortes variations la première seconde, puis à 5 s les paramètres estimés de la vitesse et de la position du mobile se confondent avec les originaux.
Pour N = 5000 :

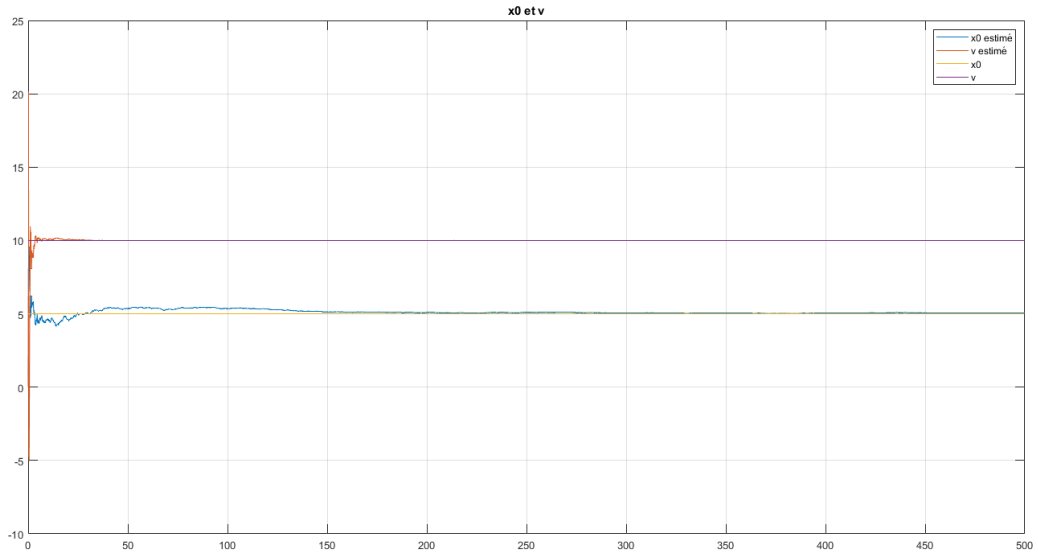

FIGURE 28 – Estimation des paramètres x_0 et v, pour N = 5000, méthode des MCR

Cette fois, vers 5 s v estimé est bon, pour 24 s pour x_0. Le temps d'exécution est de 0.0938 s alors qu'il est instantané pour N = 500.

Exercice 6 bis

On reprend l'exercice 3 avec les MCR et facteur d'oubli.
Le facteur d'oubli consiste à tenir compte des graduellement des estimations, c'est-à-dire que sur un nombre N d'échantillons, plus le nombre d'estimation tend vers N et plus les premières sont "oubliées". Ce facteur d'oubli, λ, est compris

entre 0.95 et 1, où $\lambda = 1$ donne le même résultat que s'il n'y avait pas de facteur d'oubli.
En application, on ajoute λ sur le gain de correction et P :

```
theta_mcr = zeros(p,N);
P = 10^12*eye(p,p);
theta_mcr(:,1) = [0;0];
lambda = 0.95;
for j = 2:N
    m = [sin(2*pi*f0*t(j)) cos(2*pi*f0*t(j))]';
    k = (P*m)/(lambda+m'*P*m);
    P = (1/lambda)*(P - k*m'*P);
    y_est(:,j) = m'*theta_mcr(:,j-1);
    theta_mcr(:,j) = theta_mcr(:,j-1) + k*(y(j)-y_est(:,j));
    a_mcr(:,j) = sqrt(theta_mcr(1,j).^2 + theta_mcr(2,j).^2);
    phi_mcr(:,j) = atan2(theta_mcr(2,j),theta_mcr(1,j));
end
```

Pour $\lambda = 0.95$, on a donc :

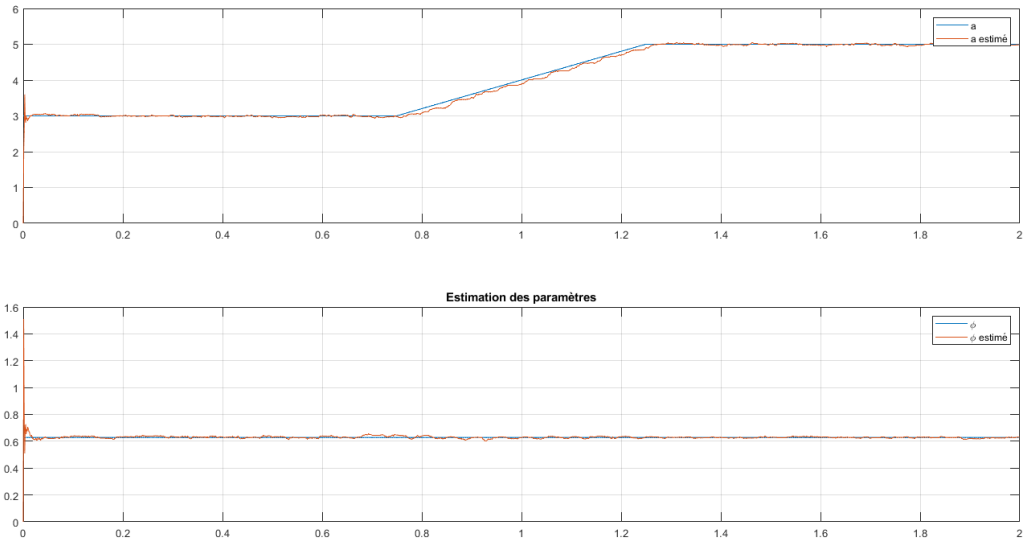

FIGURE 29 – Estimation des paramètres a et ϕ, méthode des MCR avec facteur d'oubli

L'estimation est plus fidèle et précise avec les MCR-λ que les MCO fenêtre glissante :

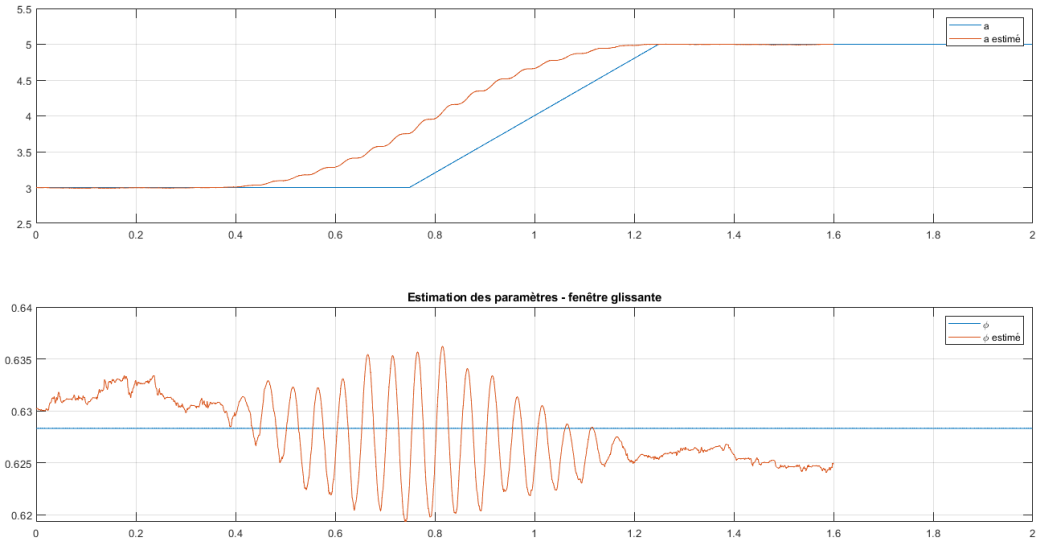

FIGURE 30 – Estimation des paramètres a et ϕ, méthode des MCO avec fenêtre glissante

Le temps de calcul est aussi beaucoup plus intéressant, avec 0.0469 s pour les MCR et 0.4219 s pour les MCO.

Exercice 6 ter

On reprend le même exercice, mais cette fois ϕ varie dans les mêmes conditions que a de $\frac{\pi}{5}$ à $\frac{2\pi}{5}$, et $\sigma = 1$:

```
alpha = ((5-3)/(1250-750));
beta = 3-alpha*750;
gamma = (((2*pi/5)-(pi/5))/(1250-750));
delta = (pi/5)-gamma*750;
x = zeros(N,1);
y = zeros(N,1);
for i = 1:N
    if k(i) <= 750
        a(i) = 3;
        phi(i) = pi/5;
    elseif 1250 <= k(i)
        a(i) = 5;
        phi(i) = 2*pi/5;
    else
        a(i) = alpha*k(i) + beta;
        phi(i) = gamma*k(i) + delta;
    end
    x(i) = a(i)*sin(2*pi*f0*t(i) + phi(i));
    y(i) = x(i) + e(i);
end
```

On obtient donc, pour les MCR-λ :

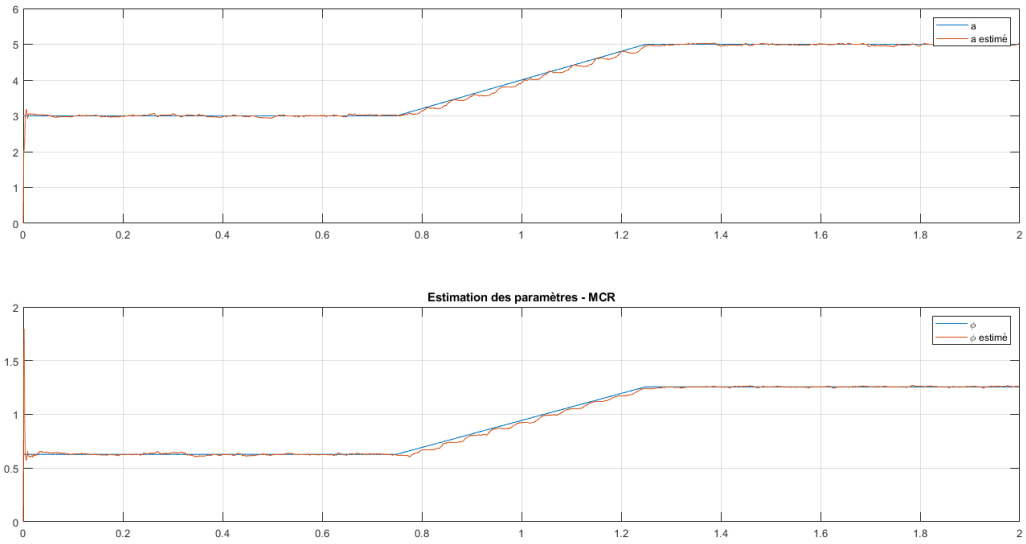

FIGURE 31 – Estimation des paramètres a et ϕ variants, méthode des MCR

et pour les MCO :

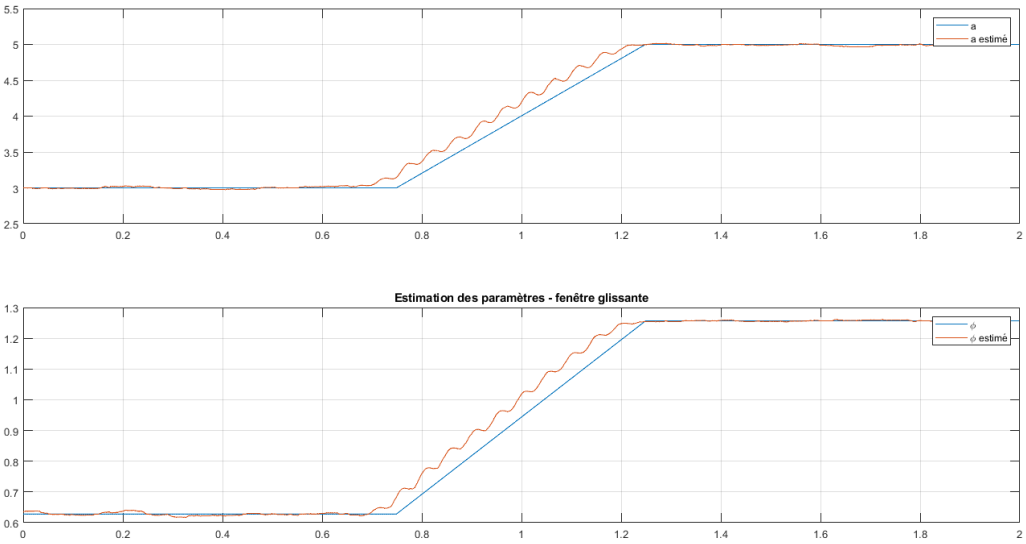

FIGURE 32 – Estimation des paramètres a et ϕ variants, méthode des MCO

Comme précédemment, l'estimation avec les MCR-λ est plus fidèle et surtout plus rapide : 0.013 s pour les MCR et 0.25 s pour les MCO !
Afin de visualiser les différences entre les deux méthodes, on a pris $\sigma = 0.1$. Pour la valeur demandée, on a plutôt quelque chose de ce type :

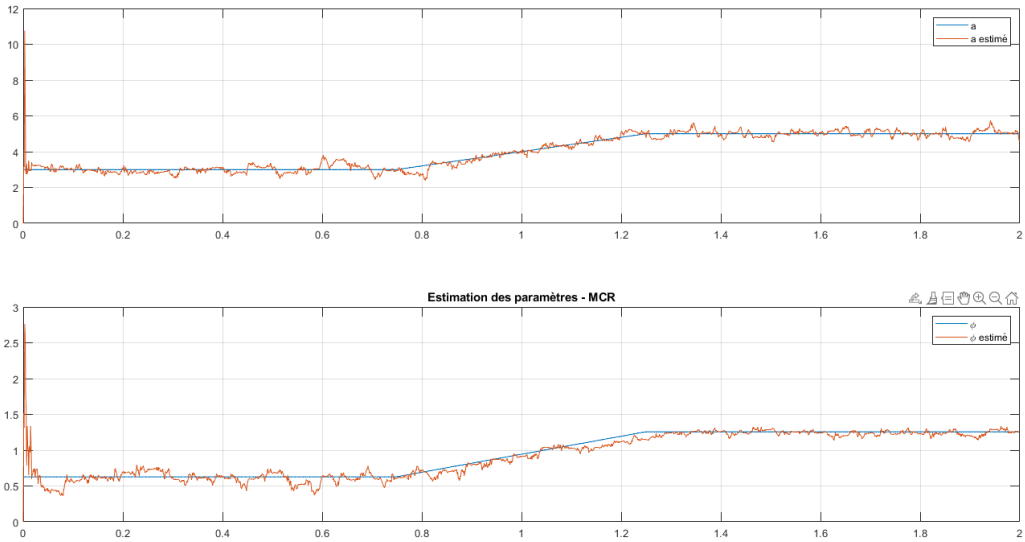

FIGURE 33 – Estimation des paramètres a et ϕ variants, méthode des MCR, $\sigma = 1$

Estimation de la réponse impulsionnelle (RI)

Exercice 8

Si on connait la réponse impulsionnelle $\{h_k\}$ où $0 \leq k \leq N-1$ d'un système physique et qu'on l'excite avec la séquence d'entrée $\{u_k\}$, alors on obtient la sortie $\{y_k\}$.
La RI que l'on suppose inconnue est $h_k = \sin(2\pi f_0 t_k)e^{-0.2t_k}$, avec $t_k = k\Delta T$, $\Delta T = 1$ s, $f_0 = 0.05$ Hz et N = 40.
On notera la séquence d'entrée $\{u_k\}$ avec $0 \leq k \leq L-1$, et $L = 32N$.
La sortie sera également de taille L, calculée avec *conv(u,h)*. On prendre enfin en compte un nombre de mesures $M = 8N$.

On se propose d'estimer $\{h_k\} = h$ sur la base d'essais entrée-sortie d'équation :

$$
\begin{aligned}
h &= \begin{bmatrix} h_0 \\ \vdots \\ h_{N-1} \end{bmatrix} = (U_J^T U_j)^{-1} U_J^T y_J \\
U_J &= \begin{bmatrix} u_{J+N-1} & u_{J+N-2} & \cdots & u_{J+0} \\ u_{J+N} & u_{J+N-1} & \cdots & u_{J+1} \\ \vdots & \vdots & \vdots & \vdots \\ u_{J+K-1} & u_{J+K-2} & \cdots & u_{J+K-N} \end{bmatrix}_{M \times N} \\
y_J &= \begin{bmatrix} y_{J+N-1} \\ y_{J+N} \\ \vdots \\ y_{J+K-1} \end{bmatrix}_{M \times 1}
\end{aligned}
\tag{16}
$$

où $K = N + M - 1$.
L'indice J correspond à l'origine des temps pris pour les calculs, soit 0 par défaut.
Pour $\{u_k\} = e^{-0.005t_k}$ et $\sigma = 0$:

```
f0 = 0.05; %Hz
N = 40;
dt = 1; %seconde
M = 8*N; %nbr de mesures
L = 32*N;
sigma = 0;
e = sigma*randn(L,1);

t = (0:L-1)'*dt;
y = conv(u, h);
y = y(1:L) + e;
J = 0;
indice = N+J:N+J+M-1;
U = u(indice);

for k = 1:N-1
    U = [U u(indice-k)];
end
h_est = inv(U'*U)*U'*y(indice);
y_est = conv(u, h_est);

epsilon = sqrt((abs(h_est-h)).^2);
```

Pour $J = 0$, on représente l'entrée, les réponses impulsionnelles réelle et estimée, et les sorties bruitée et non bruitée :

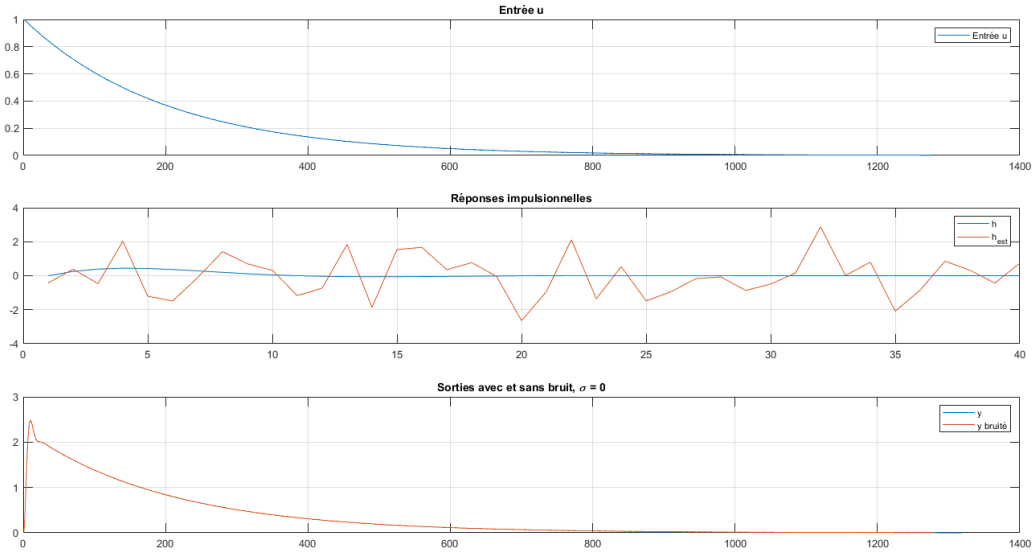

FIGURE 34 – Entrée, RI et sorties pour la première entrée, $\sigma = 0$ et J = 0

La sortie (bruitée ou non) correspond à l'entrée, mais la réponse impulsionnelle estimée ne suit pas vraiment l'originale. Pour J = 100 :

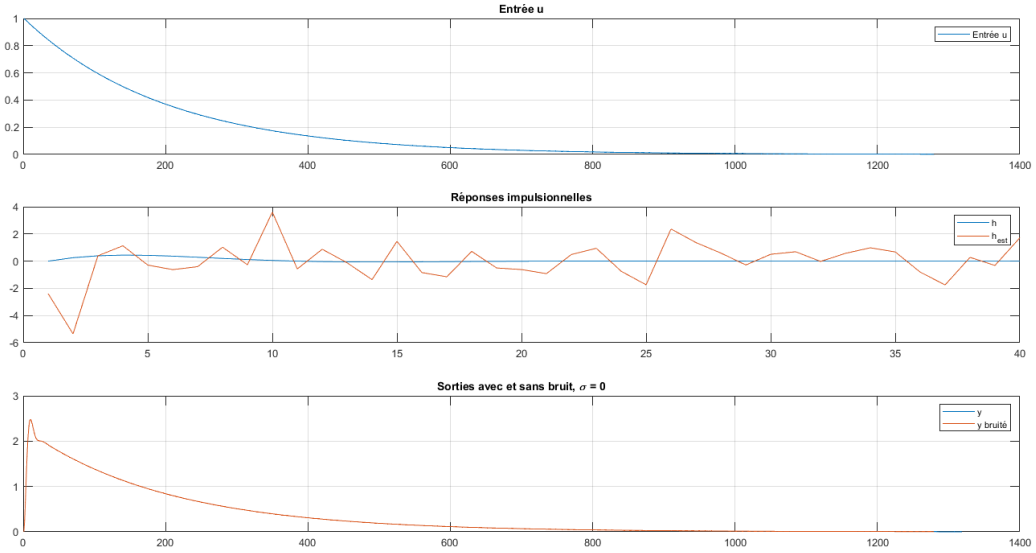

FIGURE 35 – Entrée, RI et sorties pour la première entrée, $\sigma = 0$ et J = 100

La RI estimée semble tendre vers la réelle.

Pour une entrée de type $\{u_k\} = e^{-0.005t_k}$ si $0 \leq k \leq \frac{M}{2}$, et $= e^{-0.005t_k} + 2e^{-0.001t_k}$ pour $\frac{M}{2} + 1 \leq k \leq L-1$, J = 0 :

```
t1 = t(1:M/2);
t2 = t(M/2+1:L);
u = [exp(-0.005*t1);exp(-0.005*t2)+2*exp(-0.001*t2)];
```

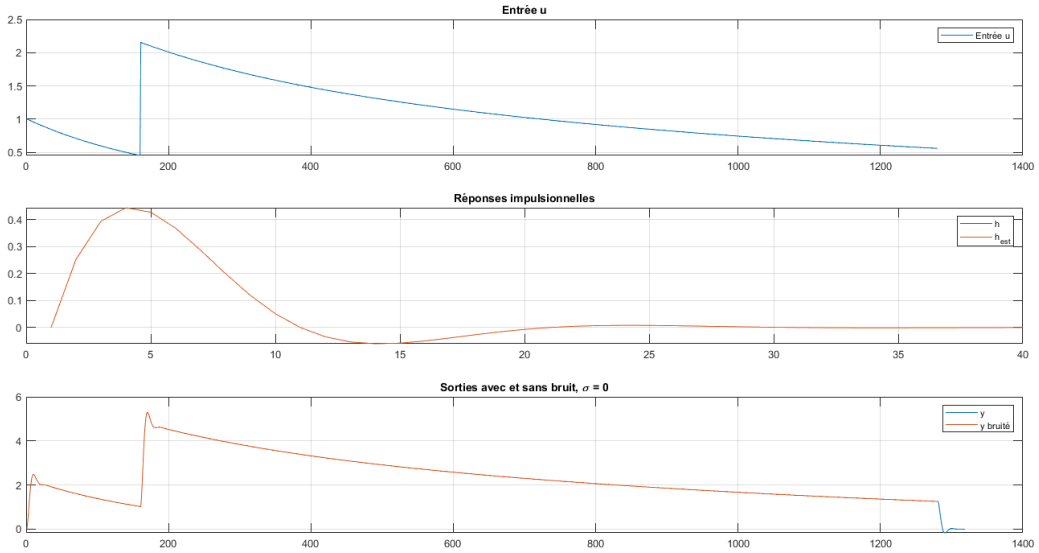

FIGURE 36 – Entrée, RI et sorties pour la deuxième entrée, $\sigma = 0$ et J = 0

Sans bruit, la RI estimée suit fidèlement l'originale. Pour $\sigma = 0.05$:

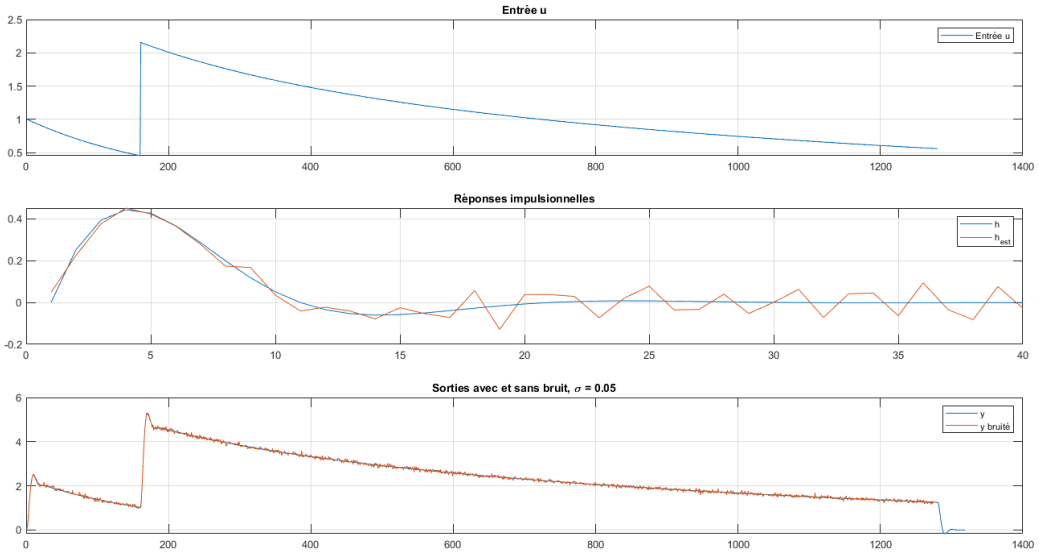

FIGURE 37 – Entrée, RI et sorties pour la deuxième entrée, $\sigma = 0.05$ et J = 0

Pour $\sigma = 0.2$ puis 0.5 :

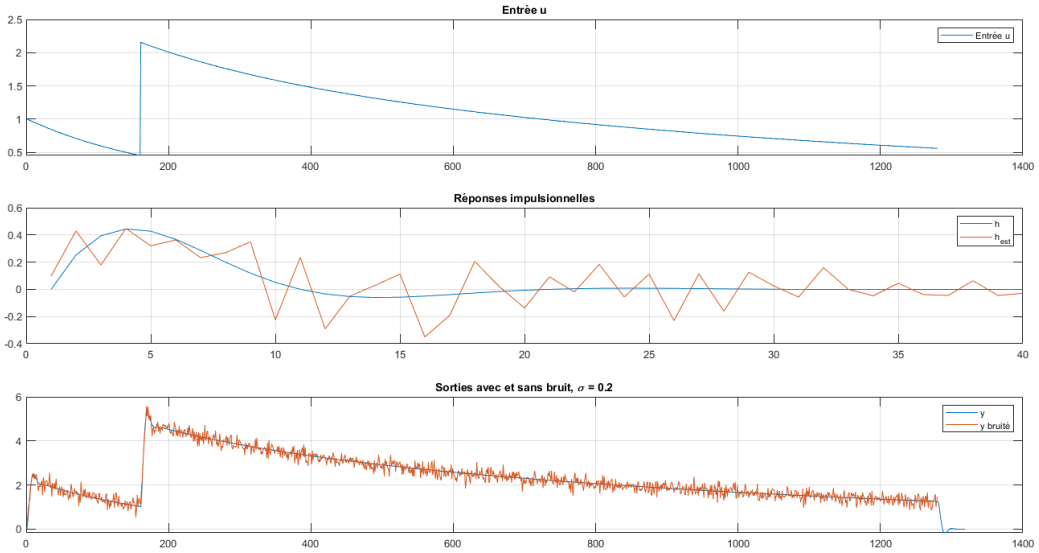

FIGURE 38 – Entrée, RI et sorties pour la deuxième entrée, $\sigma = 0.2$ et J = 0

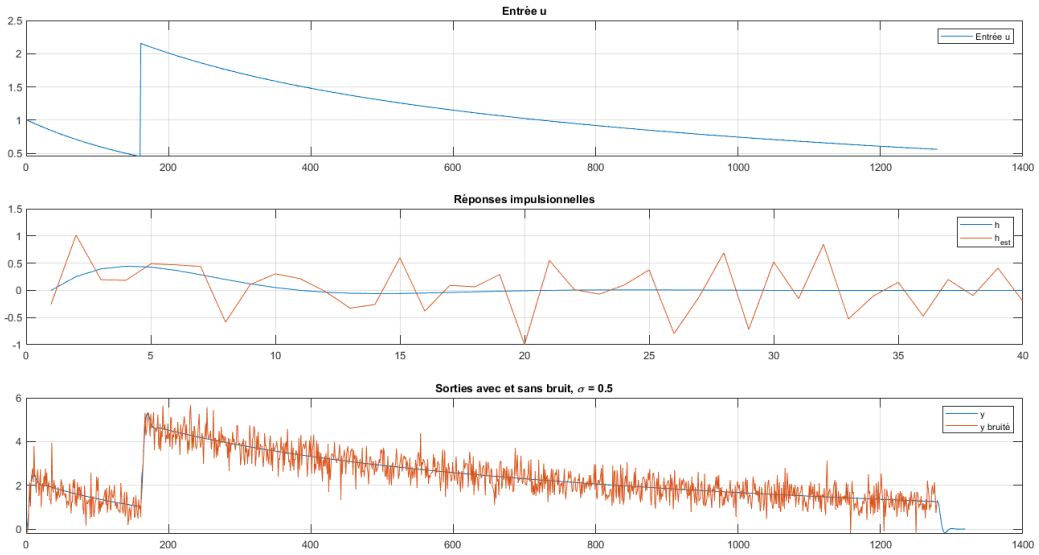

FIGURE 39 – Entrée, RI et sorties pour la deuxième entrée, $\sigma = 0.5$ et J = 0

Plus le bruit augmente, plus la RI estimée s'éloigne de l'originale, drastiquement pour $\sigma = 0.5$.

Pour une entrée de type $\{u_k\} = e^{-0.005t_k}$ si $0 \leq k \leq \frac{M}{4}$,
$e^{-0.005t_k} + 2e^{-\alpha t_k}$ si $\frac{M}{4} + 1 \leq k \leq \frac{M}{2}$,
$e^{-0.005t_k} + 2\sum_{z=1}^{2} ze^{-z\alpha t_k}$ si $\frac{M}{2} + 1 \leq k \leq \frac{3M}{4}$,
et $= e^{-0.005t_k} + 2\sum_{z=1}^{3} ze^{-z\alpha t_k}$ pour $\frac{3M}{4} + 1 \leq k \leq L-1$, avec $\alpha = 0.002$, $\sigma = 0.05$:

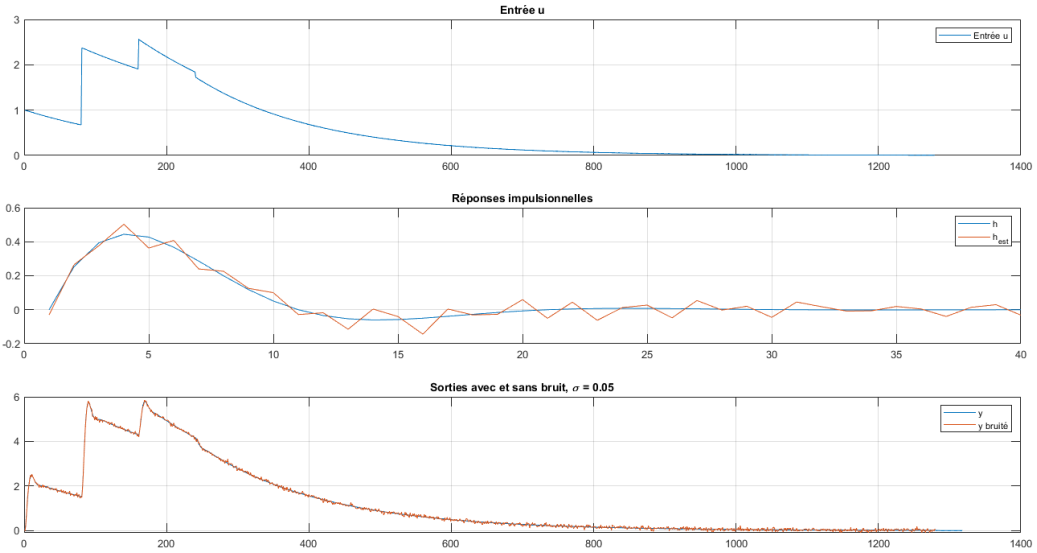

FIGURE 40 – Entrée, RI et sorties pour la troisième entrée, $\sigma = 0.05$ et J = 0

La RI estimée suit fidèlement l'originale sans bruit, avec $\sigma = 0.05$ c'est moins le cas.

Pour une entrée de type signal carré de période 40 :

```
D = pi/40;
u = square(0:D:(L-1)*D)';
```

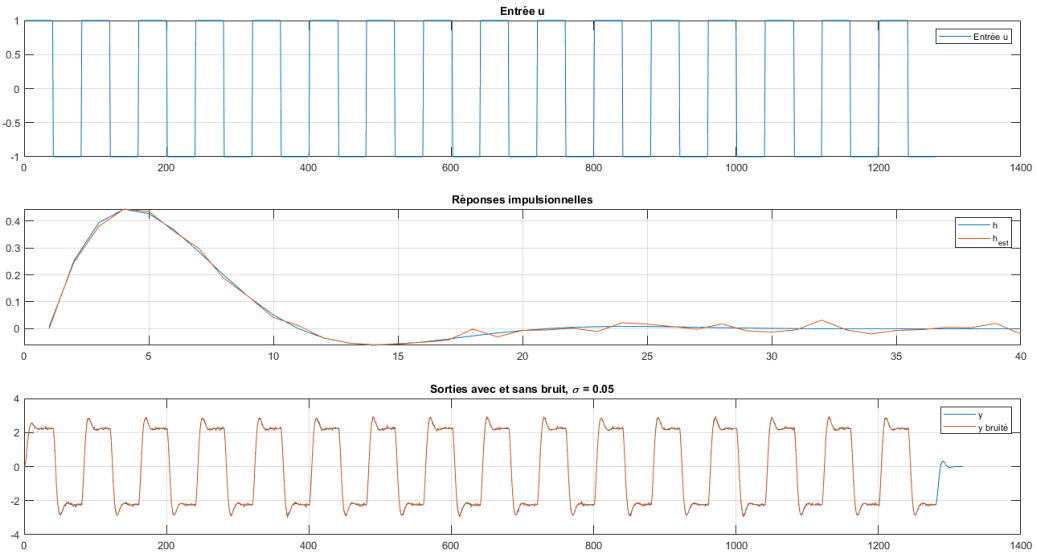

FIGURE 41 – Entrée, RI et sorties pour l'entrée carrée, $\sigma = 0.05$ et J = 0

La RI estimée suit assez bien l'originale avec $\sigma = 0.05$. Si on prend $\sigma = 0.5$:

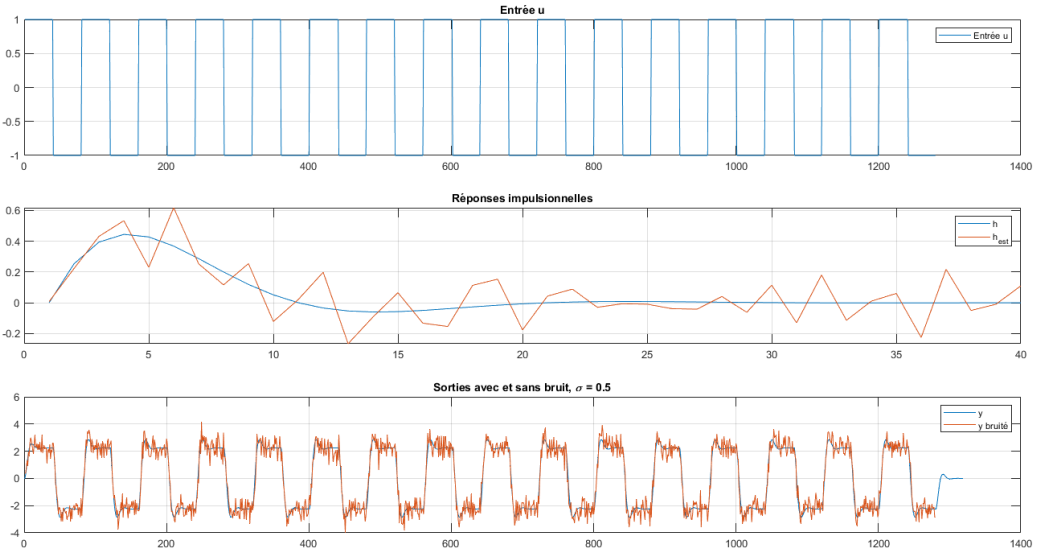

FIGURE 42 – Entrée, RI et sorties pour l'entrée carrée, $\sigma = 0.5$ et J = 0

C'est déjà un peu plus chaotique.

Enfin, pour une entrée de type sbpa (Séquence Binaire Pseudo Aléatoire), on a :

```
u = sbpa(dt, 8, 1, 1, 20);
u = u(1:L);
```

La période d'échantillonnage est de dt, pour 8 bits, et une période $T = 20$ s :

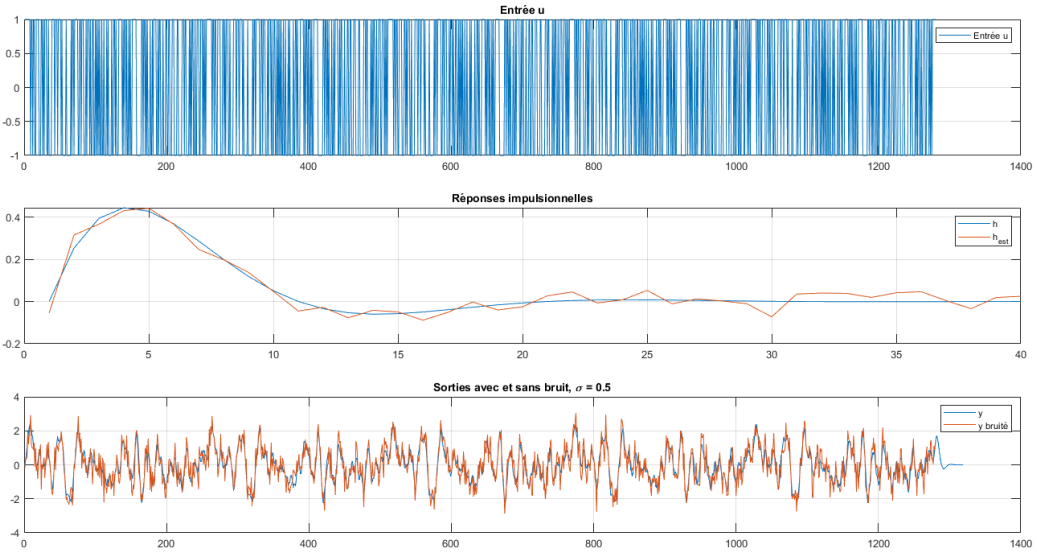

FIGURE 43 – Entrée, RI et sorties pour l'entrée sbpa, $\sigma = 0.5$ et J = 0

Même pour un bruit élevé, la RI estimée suit assez bien l'originale.
Le meilleur résultat obtenu est avec le sbpa.

Exercice 9

Dans cet exercice, on souhaite estimer la RI, supposée inconnue, d'un Système Linéaire.
Un signal binaire pseudo aléatoire (SPBA) de période $p = 7$ échantillons : $\{1,-1,1,1,1,-1,-1\}$ est utilisé pour exciter le système de fonction de transfert en z suivante :

$$H(z) = \frac{0.7z^{-1}}{1-0.3z^{-1}} \tag{17}$$

On veut calculer l'entrée $\{u_k\}$ sur $N_u = 30p$ échantillons :

```
spba = [1 -1 1 1 1 -1 -1]';
p = 7;
Nu = 30*p;

k = 30;
u = spba;
for i = 1:k-1
    u = [u; spba];
end
```

On crée notre séquence spba, puis dans un tableau de 30 lignes on ajoute les séquences.
Par TZ inverse, on calcule la RI théorique de notre système, soit :

$$h(k) = 0.3^{k-1}0.7 \tag{18}$$

On se limite à $Nh = 12$ échantillons et on représente h_k :

```
h = zeros(k,1);
h(1) = 0;
for n = 1:k
    h(n) = (0.3)^(n-1)*0.7;
end
Nh = 12;
h = [0;h];
```

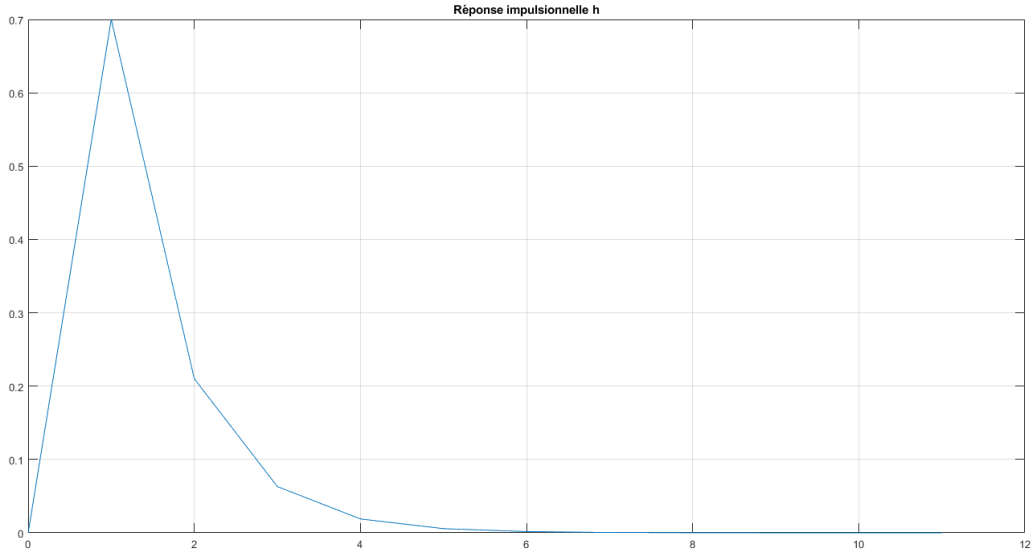

FIGURE 44 – Réponse impulsionnelle théorique

On calcule ensuite la sortie y_k :

```
y = conv(u, h);
sigma = 0.2;
e = sigma*randn(length(y),1);
y = y + e;
```

avec un $\sigma = 0.2$, on trace l'entrée et la sortie sur 80 échantillons :

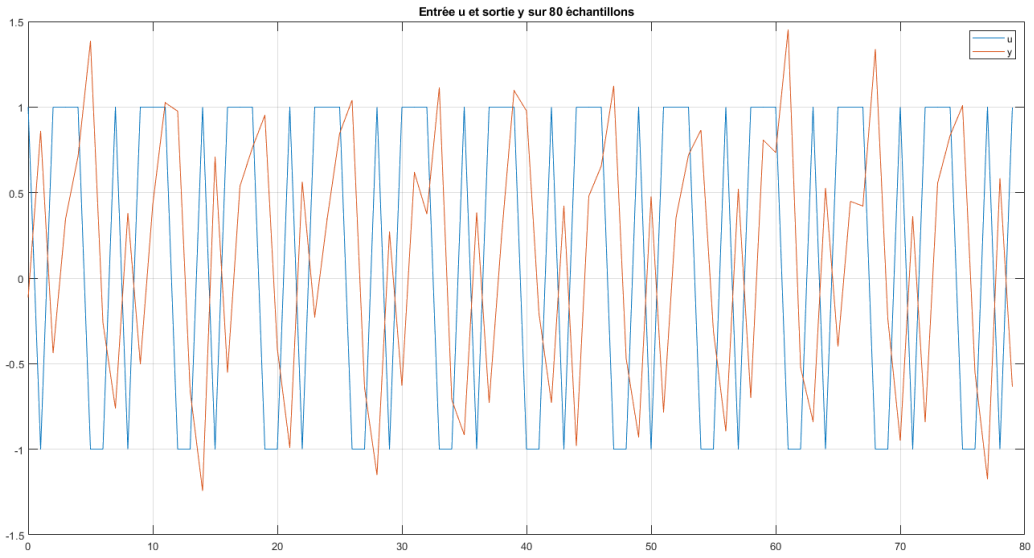

FIGURE 45 – Entrée et sortie sur 80 échantillons

On calcule maintenant h estimé avec $(U^T U)^{-1} U^T y$:

```
indice = N:N+M-1;
U = u(indice);
for l = 1:N-1
    U = [U u(indice-l)];
end
h_est = inv(U'*U)*U'*y(indice);
```

et on obtient, pour $N = p - 1$:

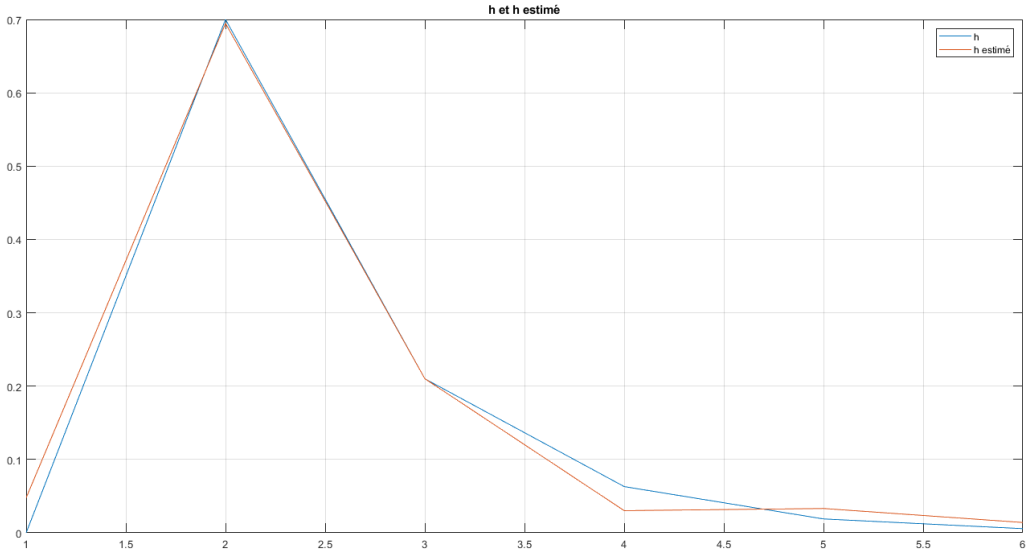

FIGURE 46 – RI estimée, pour N = p-1, p = 7

L'estimation suit à peu près la théorie. Si on prend maintenant $N = p$:

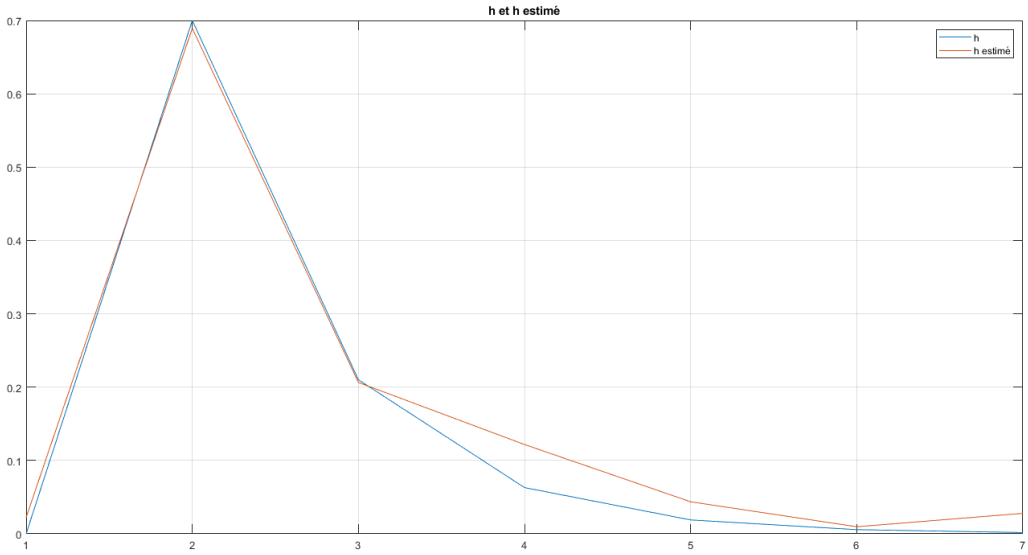

FIGURE 47 – RI estimée, pour N = p, p = 7

L'estimation suit un peu moins bien la théorie. Cette fois, pour $N = Nh - 1$:

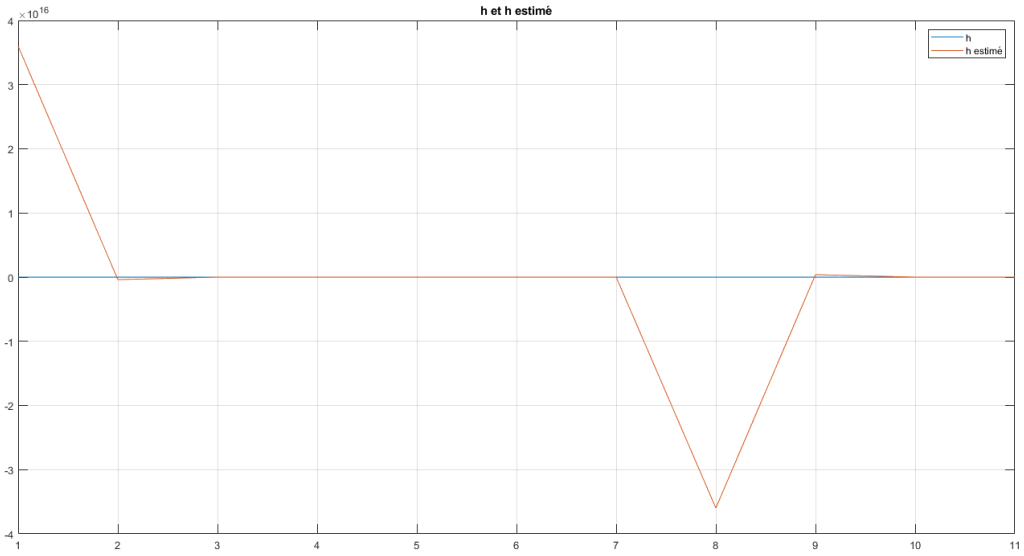

FIGURE 48 – RI estimée, pour N = Nh-1, p = 7

L'estimation est chaotique, elle ne suit pas la théorie. Il y a d'ailleurs un avertissement concernant le conditionnement de la matrice. En effet, si on prend $N = Nh - 1$, alors $N > p$, on ne peut pas calculer plus d'éléments qu'il n'en existe, d'où l'erreur. On aura le même problème pour $N = Nh$:

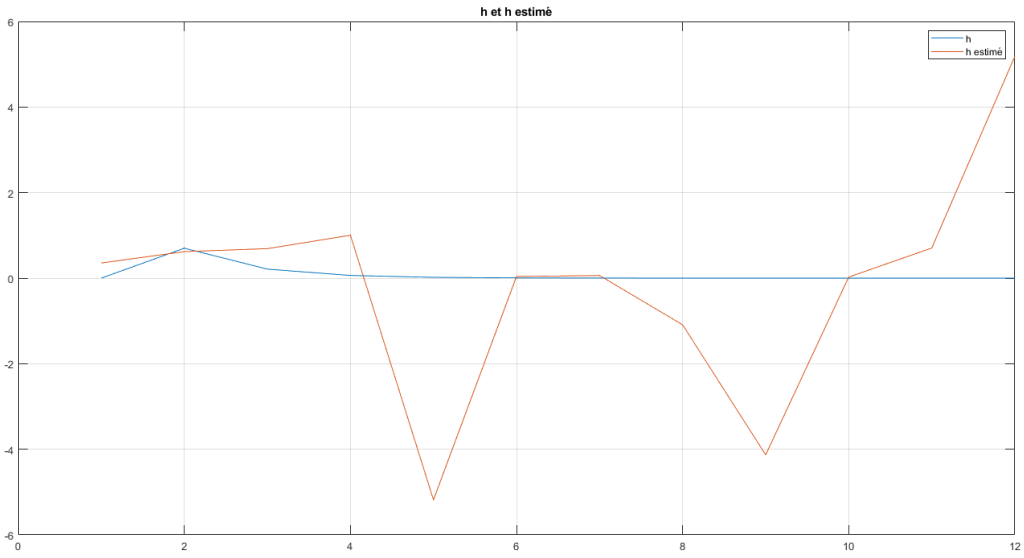

FIGURE 49 – RI estimée, pour N = Nh, p = 7

et $N = p + 1$:

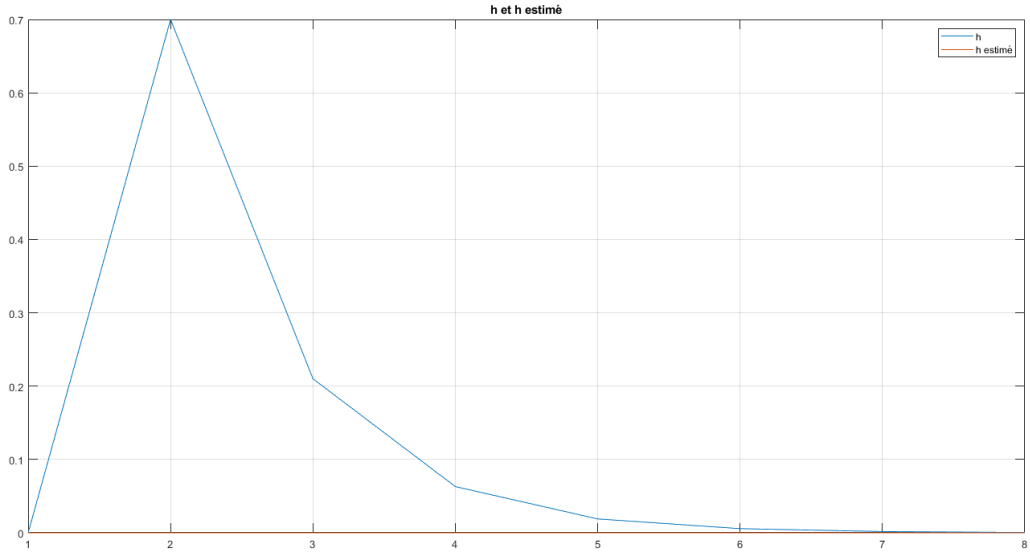

FIGURE 50 – RI estimée, pour N = p+1, p = 7

Pour une nouvelle séquence spba de période $p = 16$ $\{1,-1,1,1,1,-1,-1,1,-1,1,1,-1,-1,1,1,-1\}$, on a pour $N = p-1$:

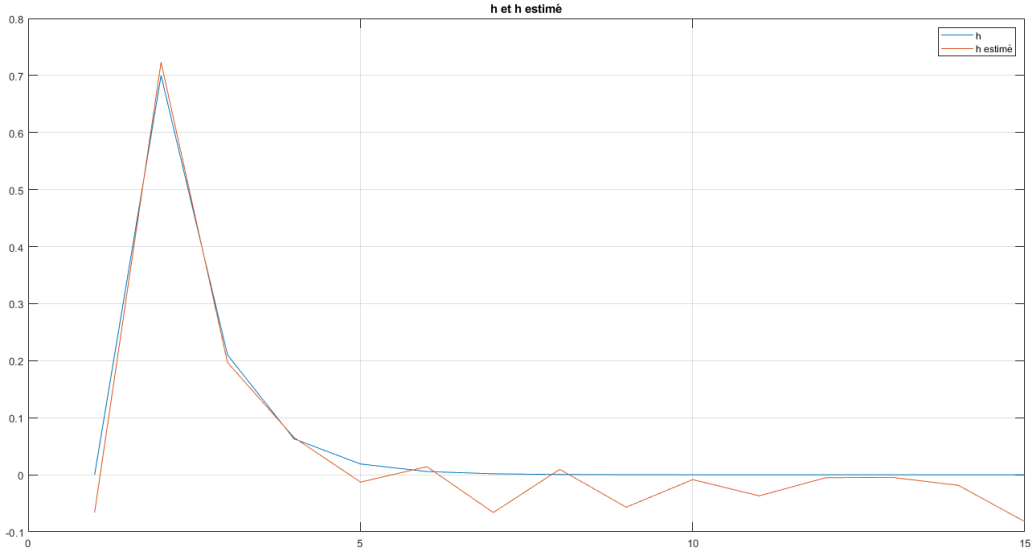

FIGURE 51 – RI estimée, pour N = p-1, p = 16

et $N = Nh$:

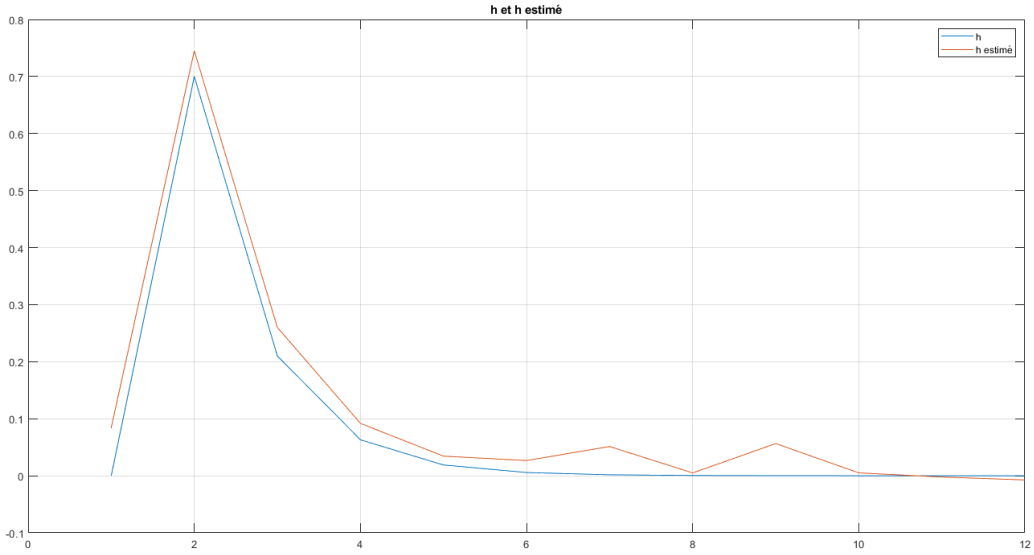

FIGURE 52 – RI estimée, pour N = Nh, p = 16

Ces figures confirment ce qui est montré plus haut : tant que N est inférieur à p, alors la RI estimée peut être calculée et est meilleure plus N est petit. Plus N tend vers p et moins la RI suit la théorie.

Enfin, pour une spba de $p = 28$ gaussienne blanche centrée, soit :

```
spba = randn(28,1);
p = 28;
```

on a pour $N = Nh$:

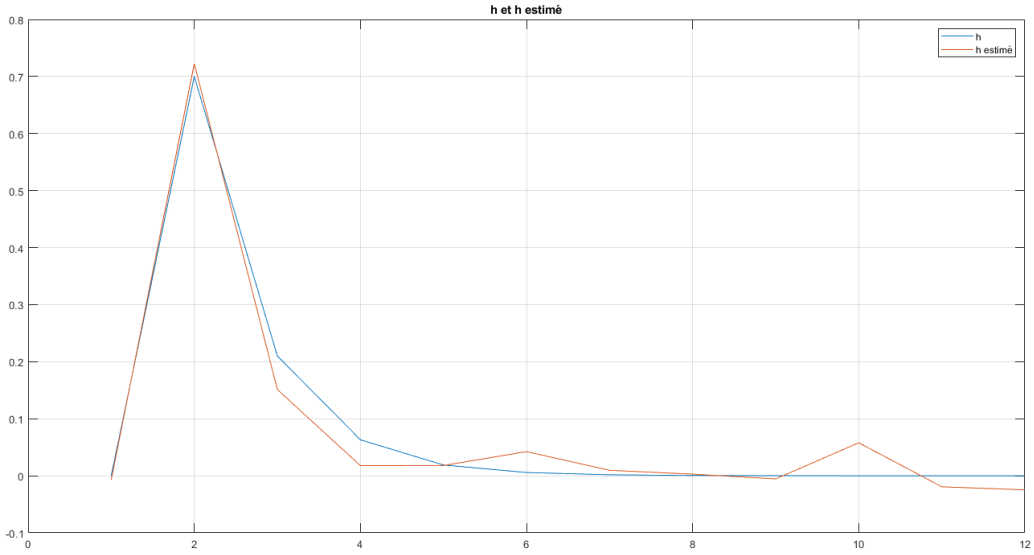

FIGURE 53 – RI estimée, pour N = Nh, p = 28

Pour $N = Nh$, c'est la meilleure estimation que l'on a faite de la RI. En effet, $Nh = 12$ et $p = 28$, donc N est bien inférieur à p.

Exercice 10

On considère un système dynamique du premier ordre :

$$y_k = -a_1 y_{k-1} + b_0 u_k + \varepsilon_k \tag{19}$$

avec $\theta = \begin{bmatrix} a_1 \\ b_0 \end{bmatrix} = \begin{bmatrix} -0.5 \\ 0.5 \end{bmatrix}$ les vrais paramètres du système, u_k l'entrée, y_k la sortie et ε_k un bruit blanc de moyenne nulle et d'écart-type $\sigma = 0.2$ ou 0.02.
On prendra pour u_k un signal périodique carré d'amplitude 1 et de période 150, et on traitera une séquence totale de longueur N = 900 échantillons.
L'objectif est d'estimer le vecteur de paramètres θ sur la base des séquences d'entrée et de sortie. On minimisera le critère quadratique basé sur l'erreur d'équation et un algorithme de MCR.
Dans le cas de θ stationnaire, on crée l'entrée selon les paramètres fixes, tel que :

```
N = 900;
k = 0:N-1;
dt = 1;
t = (k*dt)';
p = 2;
T = 150;

a1(k) = -0.5;
b0(k) = 0.5;
u = square(2*pi/T*t);
```

puis la sortie bruitée et non bruitée :

```
s = zeros(N,1);
y = zeros(N,1);
s(1) = b0(1)*u(1);
sigma = 0.02;
e = sigma*randn(N,1);
for i = 2:N
    s(i) = -a1(i)*s(i-1)+b0(i)*u(i);
    y(i) = s(i)+e(i);
end
```

Comme les indices en Matlab doivent être positifs strictement, et que notre équation prend en compte les sorties passées (y_{k-1}), on commence notre génération à $i = 2$, et on calcule en amont s(1) (notre sortie non bruitée).
Enfin, on applique la méthode des MCR-λ (plus efficace) :

```
theta_mcr = zeros(p,N);
P = 10^12*eye(p,p);
theta_mcr(:,1) = [0;0];

for j = 2:N
    m = [-y(j-1) u(j)]';
    k = (P*m)/(lambda+m'*P*m);
    P = (P - k*m'*P)/lambda;
    y_est = m'*theta_mcr(:,j-1);
    theta_mcr(:,j) = theta_mcr(:,j-1) + k*(y(j)-y_est);
end
```

La matrice M contient les $-y$ à l'indice $j-1$ et les u à l'indice j de notre équation.
Pour $\sigma = 0$, on obtient donc :

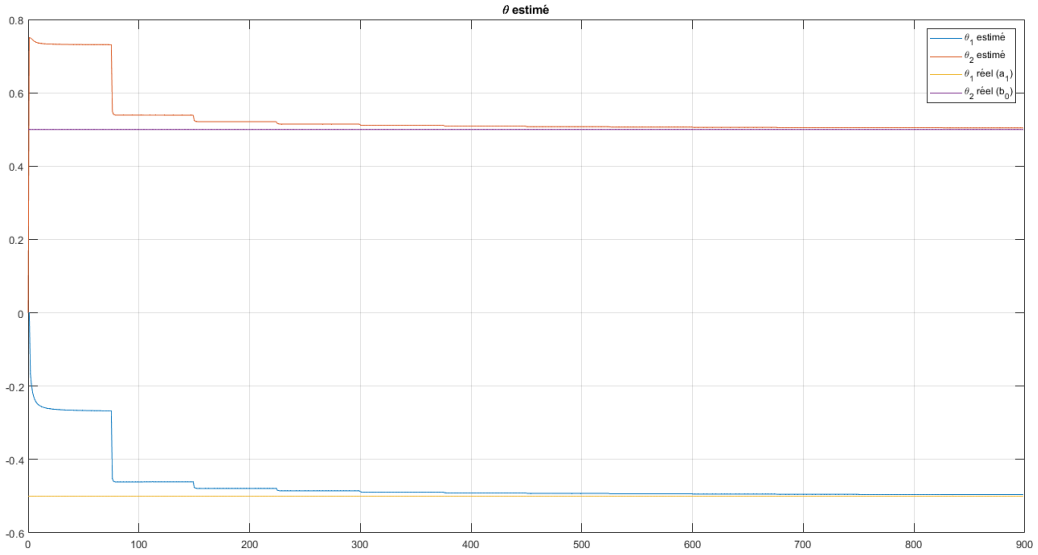

FIGURE 54 – Estimation de θ selon le temps, entrée carrée, paramètres stationnaires et $\sigma = 0$

Sans bruit, l'estimation tend vers la réalité vers $t = 400$ s. Si l'on rajoute du bruit, pour $\sigma = 0.02$:

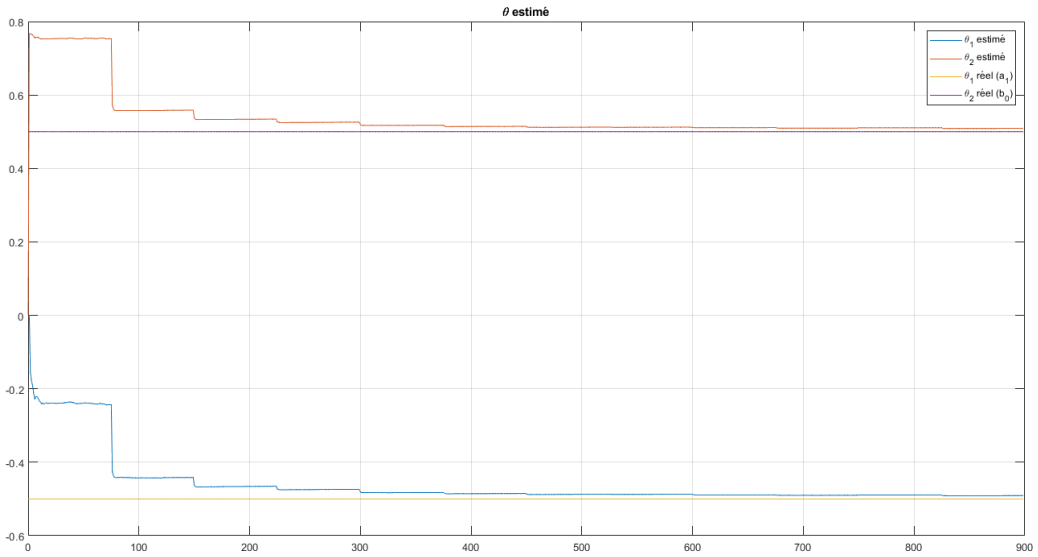

FIGURE 55 – Estimation de θ selon le temps, entrée carrée, paramètres stationnaires et $\sigma = 0.02$

Cette fois, l'estimation se rapproche de la réalité vers $t = 450$ s, mais sur les 900 échantillons elle atteint moins les vraies valeurs.
Pour $\sigma = 0.2$:

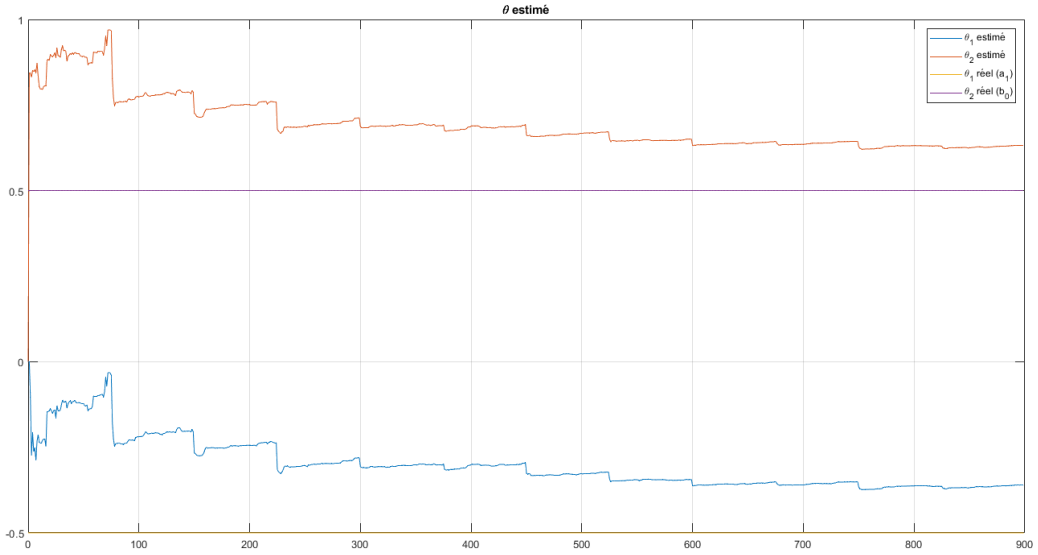

FIGURE 56 – Estimation de θ selon le temps, entrée carrée, paramètres stationnaires et $\sigma = 0.2$

L'estimation est de moins en moins bonne, elle met plus de temps à cesser ses variations et n'atteint jamais (sur les 900 échantillons mais on peut supposer sur un plus grand nombre encore) la réalité.
Enfin, pour $\sigma = 0.5$:

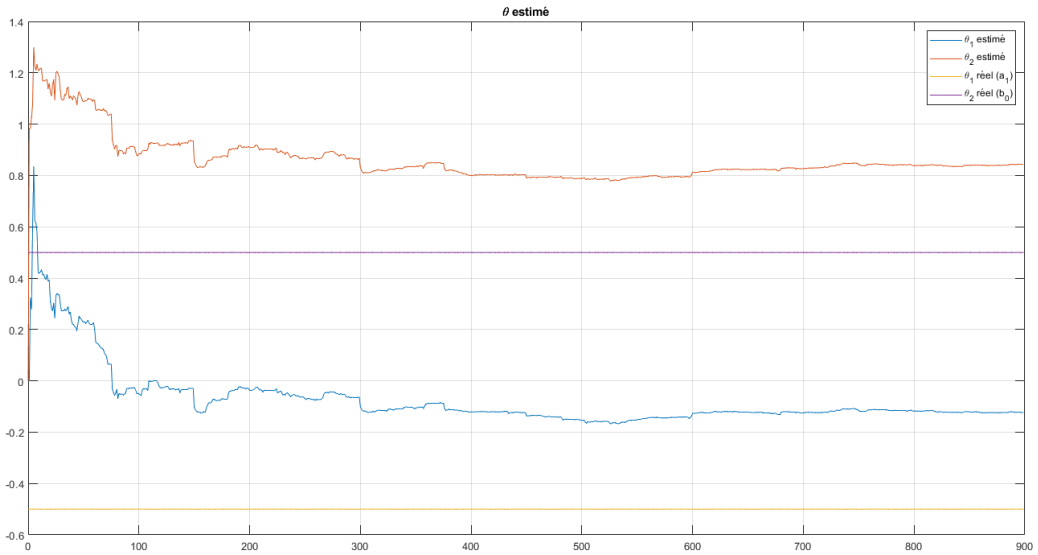

FIGURE 57 – Estimation de θ selon le temps, entrée carrée, paramètres stationnaires et $\sigma = 0.5$

L'estimation ne suit que très peu la réalité.

Si on change l'entrée en un sbpa :

```
u = randn(N,1);
```

Et on obtient, pour un $\sigma = 0$:

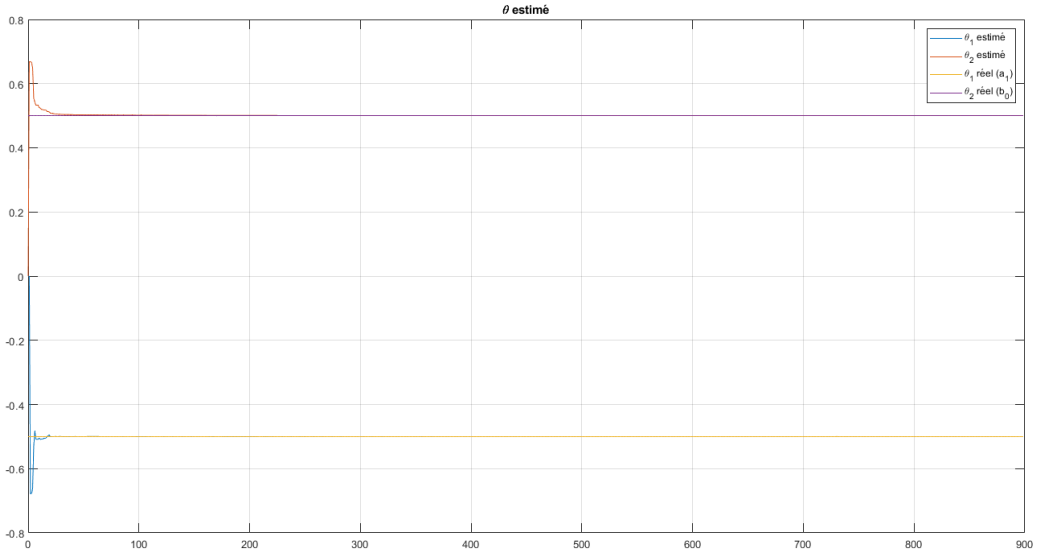

FIGURE 58 – Estimation de θ selon le temps, entrée spba, paramètres stationnaires et $\sigma = 0$

Sans bruit, l'estimation atteint exactement la réalité en $t = 28$ s, ce qui est bien meilleur que l'entrée carrée. Si on rajoute du bruit, par exemple $\sigma = 0.5$, alors on a :

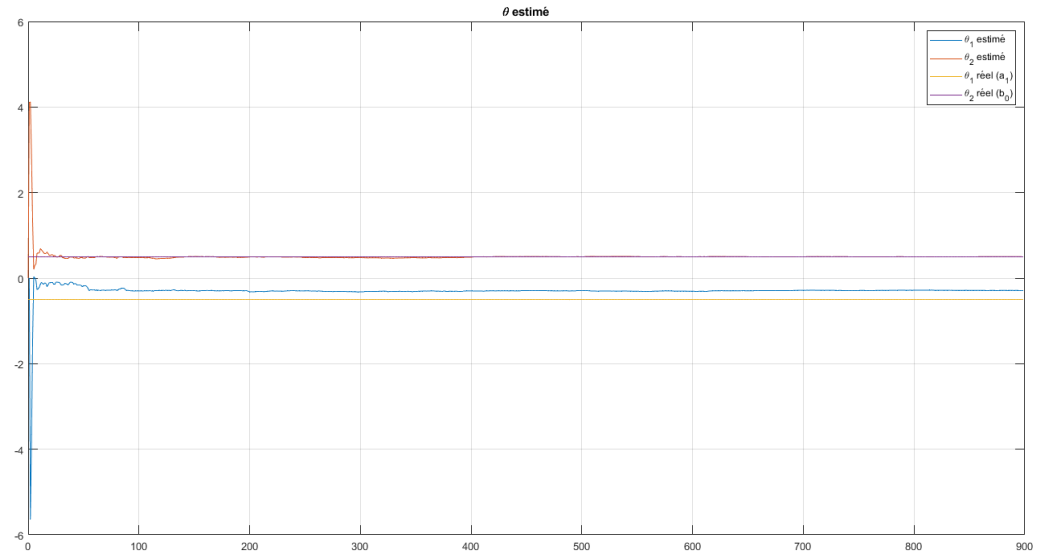

FIGURE 59 – Estimation de θ selon le temps, entrée spba, paramètres stationnaires et $\sigma = 0.5$

L'estimation de b_0 tend rapidement vers la réalité, alors qu'au contraire celle de a_1 n'atteint pas la valeur recherchée.

Maintenant, on veut estimer des paramètres non stationnaires, c'est-à-dire que pour $k \leq 350, \theta = \begin{bmatrix} -0.5 \\ 0.3 \end{bmatrix}$ et pour $k > 350, \theta = \begin{bmatrix} 0.5 \\ -0.3 \end{bmatrix}$:

```
for k = 1:N
    if k <= 350
        a1(k) = -0.5;
```

```
        b0(k) = 0.3;
    else
        a1(k) = 0.5;
        b0(k) = -0.3;
    end
end
```

On obtient donc, pour une entrée carrée et $\sigma = 0.5$:

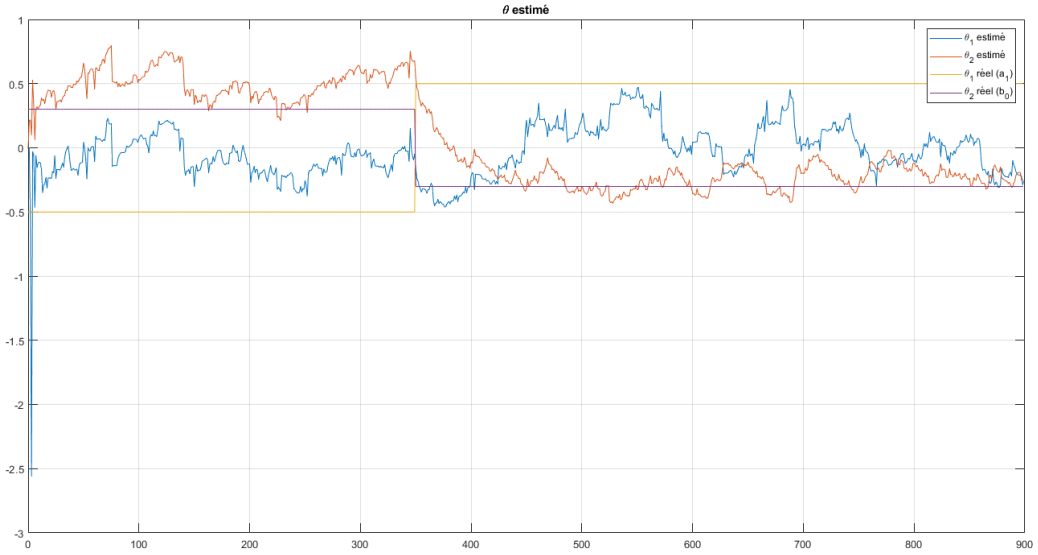

FIGURE 60 – Estimation de θ selon le temps, entrée carrée, paramètres non stationnaires et $\sigma = 0.5$

L'estimation ne suit que très peu la réalité, et avec une entrée spba :

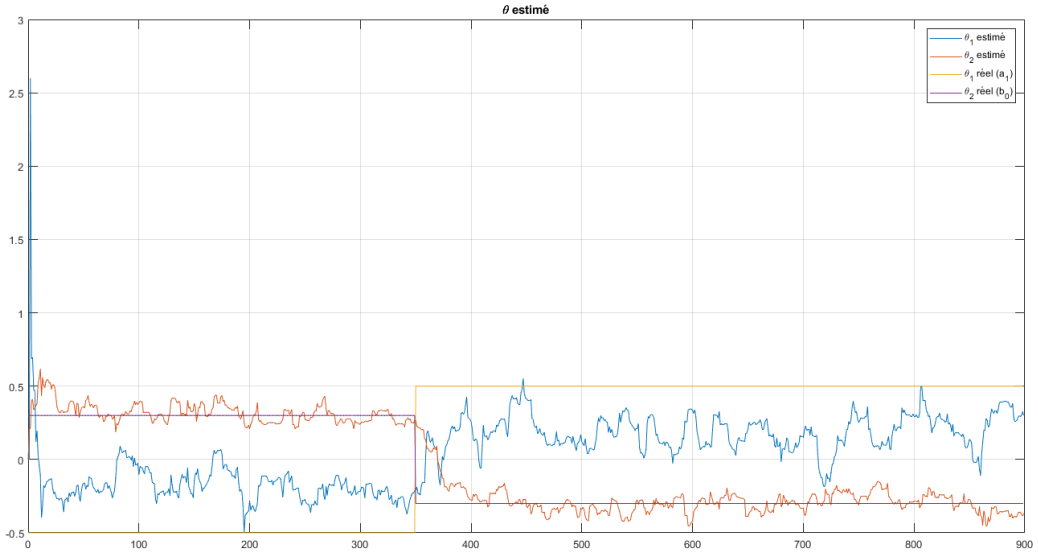

FIGURE 61 – Estimation de θ selon le temps, entrée spba, paramètres non stationnaires et $\sigma = 0.5$

Le résultat n'est qu'un peu meilleur. L'estimation est la meilleure avec une entrée spba et des paramètres stationnaires.

Exercice 11

On reprend le même exercice, cette fois avec $y_k = -a_1 y_{k-1} - a_2 y_{k-2} + b_0 u_k + b_1 u_{k-1} + \varepsilon_k$, et $\theta = \begin{bmatrix} a_1 \\ a_2 \\ b_0 \\ b_1 \end{bmatrix} = \begin{bmatrix} -0.5 \\ 0.3 \\ 0.5 \\ 1 \end{bmatrix}$,

et pour les non stationnaires :

Quand $k \leq 350, \theta = \begin{bmatrix} -0.5 \\ 0.3 \\ 0.5 \\ 1 \end{bmatrix}$, quand $350 \leq k \leq 600, \theta = \begin{bmatrix} 0.5 \\ -0.3 \\ 0.5 \\ 1 \end{bmatrix}$ et enfin, $k > 600, \theta = \begin{bmatrix} 0.5 \\ -0.3 \\ 0.9 \\ 1.6 \end{bmatrix}$:

```
for k = 1:N
    if k <= 350
        a1(k) = -0.5;
        a2(k) = 0.3;
        b0(k) = 0.5;
        b1(k) = 1;
    elseif k <= 600
        a1(k) = 0.5;
        a2(k) = -0.3;
        b0(k) = 0.5;
        b1(k) = 1;
    else
        a1(k) = 0.5;
        a2(k) = -0.3;
        b0(k) = 0.9;
        b1(k) = 1.6;
    end
end
```

La matrice m devient donc :

```
s = zeros(N,1);
y = zeros(N,1);
s(1) = b0(1)*u(1);
s(2) = -a1(2)*s(1)+b0(2)*u(2)+b1(2)*u(1);
e = sigma*randn(N,1);
for i = 3:N
    s(i) = -a1(i)*s(i-1)-a2(i)*s(i-2)+b0(i)*u(i)+b1(i)*u(i-1);
    y(i) = s(i)+e(i);
end

theta_mcr = zeros(p,N);
P = 10^12*eye(p,p);
theta_mcr(:,1) = [0 0 0 0];

for j = 3:N
    m = [-y(j-1) -y(j-2) u(j) u(j-1)]';
    k = (P*m)/(lambda+m'*P*m);
    P = (P - k*m'*P)/lambda;
    y_est = m'*theta_mcr(:,j-1);
    theta_mcr(:,j) = theta_mcr(:,j-1) + k*(y(j)-y_est);
end
```

On commence la boucle à $j = 3$, car on a du y_{k-2}, et donc on calcule en amont s(1) et s(2). On obtient, pour une entrée

carrée et $\sigma = 0$:

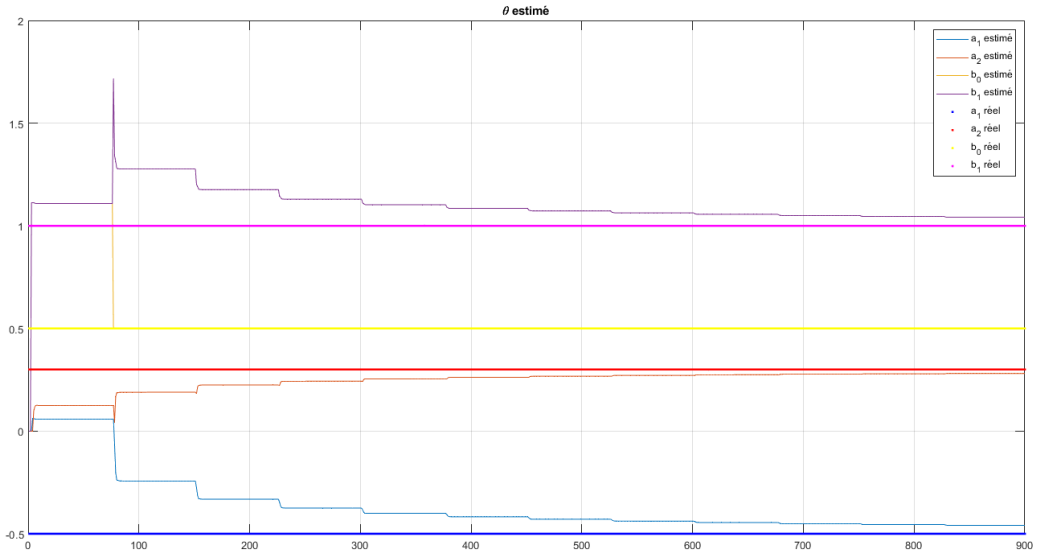

FIGURE 62 – Estimation de 4 paramètres stationnaires, entrée carrée et $\sigma = 0$

Les estimations n'atteignent pas exactement les valeurs recherchées à la fin des 900 échantillons (exceptée b_0), mais tendent tout de même vers elles.
Si on rajoute un $\sigma = 0.5$:

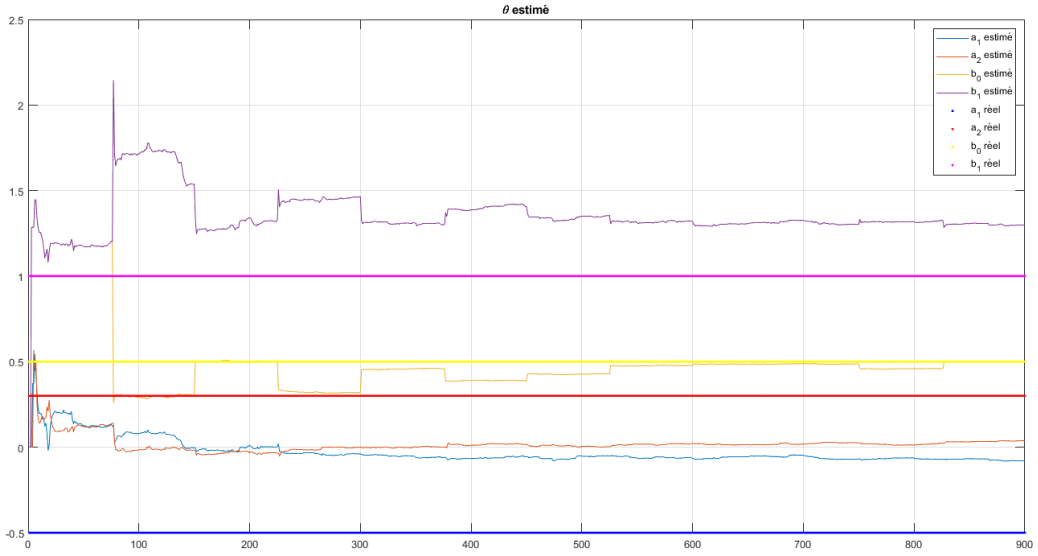

FIGURE 63 – Estimation de 4 paramètres stationnaires, entrée carrée et $\sigma = 0.5$

Comme attendu, les estimations n'approchent pas vraiment les réelles, sauf b_0 qui tend vers l'original.
Si on prend maintenant une entrée spba, pour $\sigma = 0$:

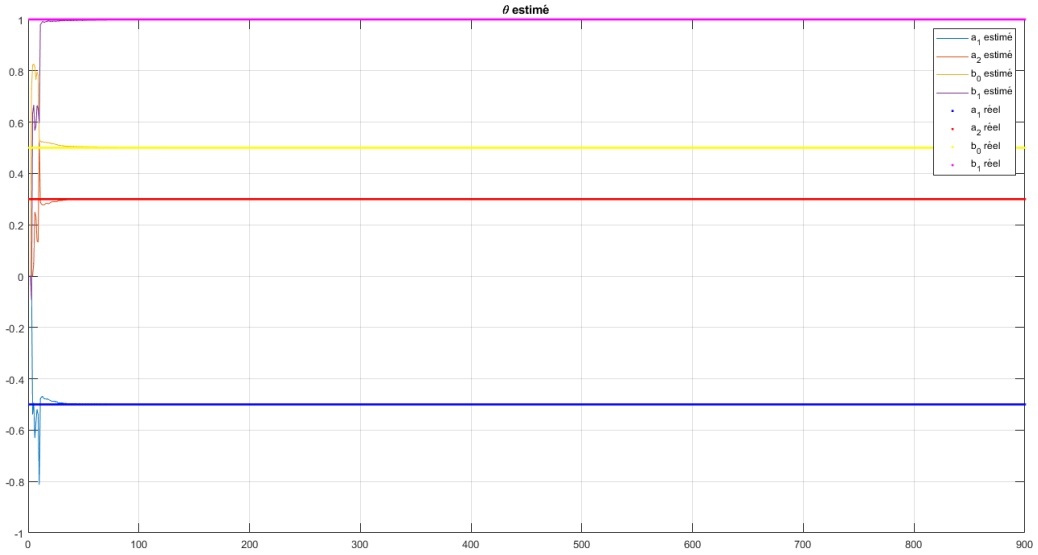

FIGURE 64 – Estimation de 4 paramètres stationnaires, entrée spba et $\sigma = 0$

Très vite ($t = 28$), l'estimation se confond avec la valeur recherchée. Si on rajoute du bruit ($\sigma = 0.5$) :

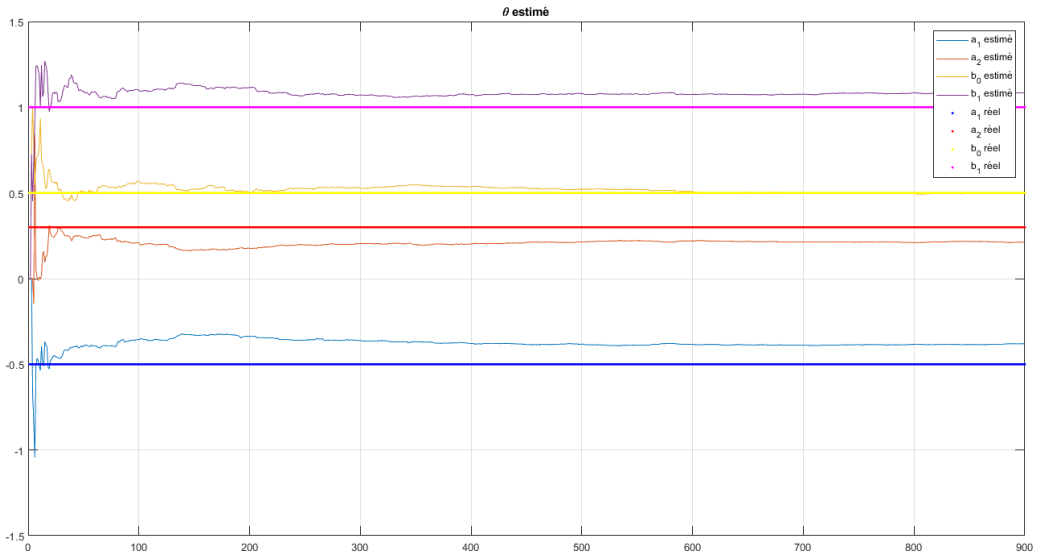

FIGURE 65 – Estimation de 4 paramètres stationnaires, entrée spba et $\sigma = 0.5$

Contrairement à l'entrée carrée, les estimations sont proches de la réalités, mais ne l'atteignent pas sur les 900 échantillons.

Si on prend maintenant les paramètres non-stationnaires, pour une entrée carrée et $\sigma = 0.5$:

FIGURE 66 – Estimation de 4 paramètres non-stationnaires, entrée carrée et $\sigma = 0.5$

Comme attendu, le bruit est trop élevé face au nombre de paramètres à estimer et la séquence d'entrée.
On peut vérifier avec une entrée spba :

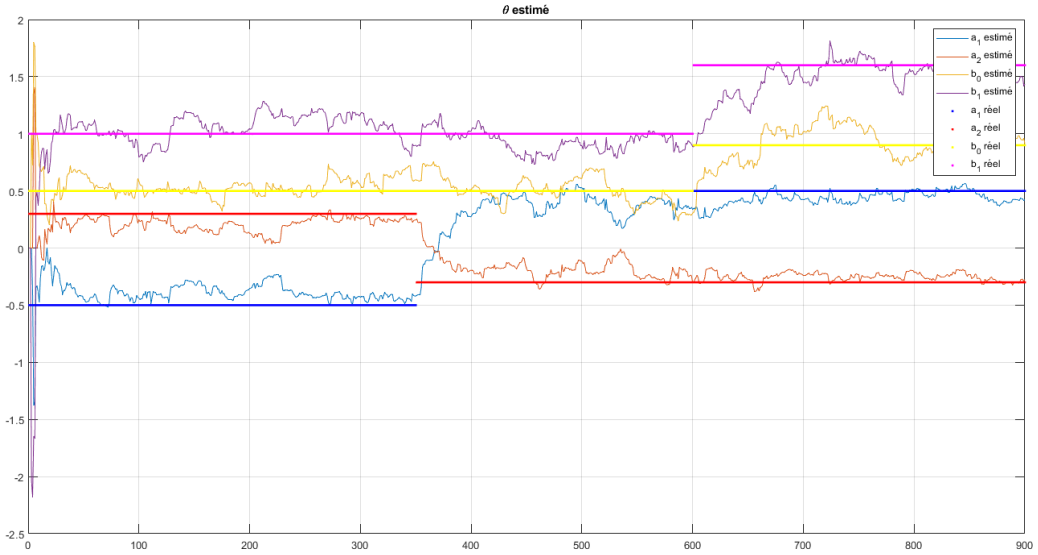

FIGURE 67 – Estimation de 4 paramètres non-stationnaires, entrée spba et $\sigma = 0.5$

L'estimation suit assez bien la réalité, malgré quelques variations.

Méthode des Moindres Carrés non-linéaires - Gradient

Exercice 12

On considère un système dynamique du premier ordre :

$$y_k = -a_1 y_{k-1} + b_0 u_k + \varepsilon_k = y_{M,k} + \varepsilon_k \tag{20}$$

où $k = 1,2,...N$, $y_0 = 0$, $u_k = 0$, $\forall k \leq 0$, et $\theta = \begin{bmatrix} a_1 \\ b_0 \end{bmatrix} = \begin{bmatrix} -0.5 \\ 0.5 \end{bmatrix}$, les vrais paramètres du système, u_k l'entrée, y_k la sortie et enfin ε_k un bruit de mesure blanc gaussien de moyenne nulle et d'écart-type σ. $\{y_{M,k}\}$ est la sortie du modèle. On pose $\theta = [\theta_1 \dots \theta_p]^T$.
On prendra pour u_k un signal périodique carré d'amplitude 1 et de période 150, et on traitera une séquence totale de longueur N = 900 échantillons.
L'objectif est d'estimer le vecteur de paramètres θ sur la base des séquences d'entrée et de sortie. On minimisera le critère quadratique basé sur l'erreur de sortie et un algorithme de descente de gradient à pas constant λ (différent du facteur d'oubli !).

On commence par écrire une fonction *reponse()* avec en arguments les paramètres, l'entrée u_k et en sortie de fonction la sortie $y_{M,k}$:

```
function s = reponse(a, b, u)
    N = length(u);
    L = length(a);
    P = length(b);
    s = zeros(N,1);
    for i = 1:L
        s(i) = 0;
        for k = 1:i-1
            s(i) = s(i) - a(k)*s(i-k);
        end
        for k = 0:i-1
            s(i) = s(i) + b(k+1)*u(i-k);
        end
    end
    for i = L+1:N
        s1 = 0;
        for k = 1:L
            s1 = s1 + a(k)*s(i-k);
        end
        s2 = 0;
        for k = 0:P-1
            s2 = s2 + b(k+1)*u(i-k);
        end
        s(i) = -s1 + s2;
    end
end
```

On commence par définir les variables N, L et P qui correspondent respectivement aux longueurs de l'entrée et des paramètres. On crée ensuite un tableau de taille N (de l'entrée donc) composé de 0.
On va calculer ensuite séparément la sortie : on commence donc du début (1) à la taille du premier paramètre (on suppose ici que $a > b$), où l'on calcule d'un côté les $a_1 y_{k-1}$, et de l'autre les $b_0 u_k$.
En deuxième partie, soit jusqu'à la fin des échantillons d'entrée, on refait les mêmes calculs.
On génère donc l'entrée, la sortie et la sortie bruitée :

```
T = 150;
N = 900;

a1 = -0.5;
b0 = 0.5;
sigma = 0.02;
e = sigma*randn(N, 1);
k = 0:N-1;
dt = 1;
t = (k*dt)';
theta = [a1 b0];
u = square(2*pi/T*t);


s = reponse(a1, b0, u);
y = s + e;
```

Pour $\sigma = 0.02$:

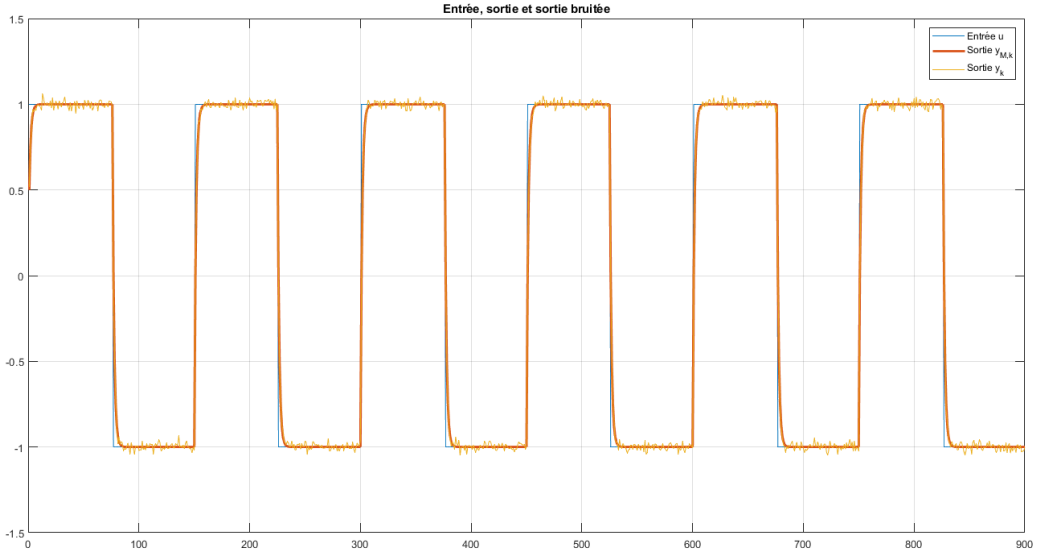

FIGURE 68 – Entrée, sortie et sortie bruitée

La fonction *reponse()* semble fonctionner : les sorties suivent l'entrée.
Afin de trouver θ, on va créer d'abord une fonction *critere()* afin de minimiser le critère $J = \frac{1}{N}\sum_{k=1}^{N}(y_k - y_{M,k})^2$, qui sera la sortie de la fonction, qui prendra en entrée les paramètres, l'entrée et la sortie. $y_{M,k}$ est la sortie de *reponse()* :

```
function J = critere(a, b, u, y)
    y_m = reponse(a, b, u);
    e = y_m - y;
    J = e'*e;
end
```

Maintenant que l'on a le critère, on a besoin d'une fonction *gradient()* permettant de réaliser le calcul du vecteur gradient de J. La fonction aura en entrée les paramètres, l'entrée et la sortie.

Le vecteur gradient ∇J se calcule en théorie en réalisant les dérivées partielles de J par rapport à θ, soit $\nabla J = \left[\frac{\partial J}{\partial \theta_1} \quad \dots \quad \frac{\partial J}{\partial \theta_p}\right]^T$.
En pratique, on prend la définition de la dérivée avec, pour remplacer la limite, une constante h très petite, soit (pour $p = 2$)

$\nabla J = \begin{bmatrix} \frac{J(\theta_1+h,\theta_2)-J(\theta_1,\theta_2)}{h} \\ \frac{J(\theta_1,\theta_2+h)-J(\theta_1,\theta_2)}{h} \end{bmatrix}_{\theta}^{T}$. On écrit donc la fonction *gradient()* :

```
function dJ = gradient(a, b, u, y)
    N = length(a);
    M = length(b);
    h = 10^-5;
    theta = [a b];
    for k = 1:length(theta)
        J = critere(a, b, u, y);
        for i = 1:N
            a_est = a;
            a_est(i) = a(i) + h;
            dJa(i) = (critere(a_est, b, u, y) - J)/h;
        end
        for i = 1:M
            b_est = b;
            b_est(i) = b(i) + h;
            dJb(i) = (critere(a, b_est, u, y) - J)/h;
        end
        dJ = [dJa;dJb];
    end
end
```

Soient N et m les longueurs des paramètres, $h = 10^{-5}$ et θ la matrice des paramètres.
On calcule le critère J, puis pour les premiers paramètres on réalise un premier calcul de gradient, de même pour les deuxièmes, et on réunit le tout dans la sortie gradient.

Maintenant, on crée une fonction *algoDescenteGradient()* qui va calculer les θ estimés de manière à minimiser le critère $J(\theta)$ sur la droite Δ passant par θ^i et de vecteur directeur $\nabla J(\theta^i)$.
On va pour cela calculer les $J(\theta^i - k\lambda\nabla J(\theta^i))$ jusqu'à obtenir la valeur minimale dans cette direction. On obtient alors une approximation du minimum de J sur l'intersection de la surface $J(\theta)$ et du plan vertical correspondant à la droite Δ.
On prendra $\lambda = 10^{-5}$:

```
function theta_est = algodescenteGradient(a, b, u, y, lambda)
    J = critere(a, b, u, y);
    theta = [a;b];
    flag = true;
    k = 1;
    while flag
        theta = theta - k*lambda*gradient(theta(1),theta(2),u,y);
        J_est = critere(theta(1),theta(2), u, y);
        if J_est >= J
            flag = false;
            theta_est = theta;
        end
        k = k+1;
        J = J_est;
    end
end
```

On calcule d'abord notre critère J, puis on crée une boucle *while* qui ne s'arrêtera que lorsque le critère sera minimal. Dans cette boucle, on réalise le calcul du θ, en incrémentant k de 1 à chaque nouvelle itération. L'ancien θ est remplacé par le nouveau, et le dernier J estimé devient le nouveau critère auquel se référer.

On peut maintenant réaliser l'estimation de θ par MC non-linéaire, en utilisant un algorithme récursif :

```
a_est = -0.35;
```

```
b_est = 0.4;
lambda = 10^-5;
flag = true;
theta_0 = [a_est;b_est];
Theta = theta_0;
J1 = critere(theta_0(1), theta_0(2), u, y);
k = 1;
while flag
    theta_tmp = algodescenteGradient(theta_0(1),theta_0(2),u,y,lambda);
    J_est = critere(theta_tmp(1),theta_tmp(2), u, y);

    if abs(J_est - J1) < 1e-5
        flag = false;
        theta_est = theta_0;
    end
    J1 = J_est;
    theta_0 = theta_tmp;
    k = k+1;
    Theta = [Theta theta_tmp];
end
```

On crée donc un θ_0 initial avec des paramètres faux, puis un deuxième qui prend la valeur de θ_0.
On réalise aussi un premier calcul de critère, J_1, puis on reprend le même principe que pour la fonction *algodescenteGradient()* : on calcule un θ avec la fonction, en prenant en paramètres les faux a et b, puis on en calcule le critère. A chaque nouvelle itération, le critère initial (de référence) devient le nouveau, θ_0 de même, on incrémente k, et on compare les critères. La boucle s'arrête lorsque la différence entre les deux est inférieure à la précision souhaitée (ici 10^{-5}).
On peut enfin tracer l'évolution des θ en fonction des i :

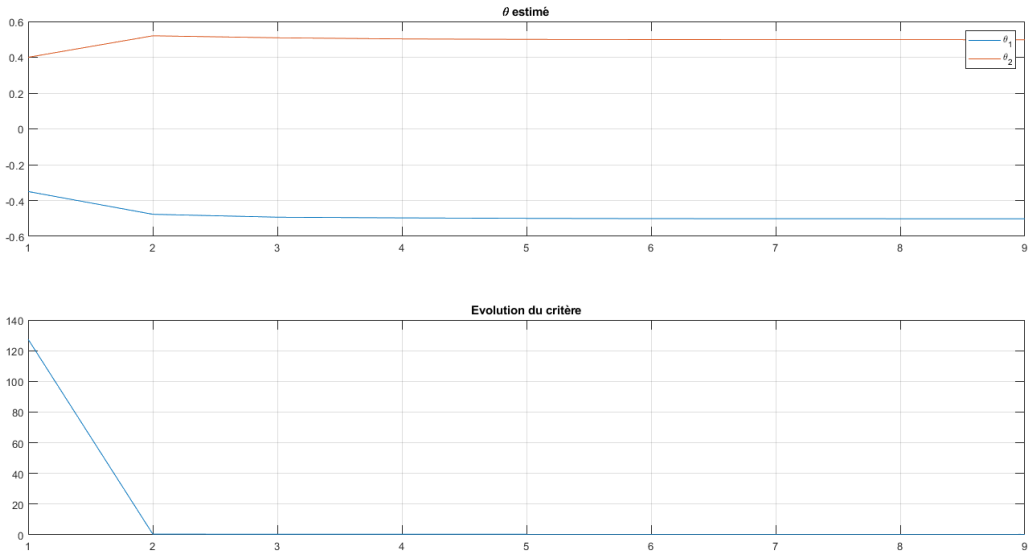

FIGURE 69 – Estimation des paramètres avec des MC non-linéaires, et suivi du critère

En 9 itérations, l'estimation a atteint la valeur recherchée exacte.
Le suivi du critère J nous permet de voir qu'à partir de la deuxième itération, l'estimation était très proche de la réalité.

Annexe - Codes

Exercice 1

```
clc; clear; close all

test = menu("Choix du y", "y connu", "y non connu");
if test == 1
    N = 10;
    t = (0:1:N-1)';
    y = [-1.04 1.21 3.36 4 5.43 8.45 10.67 13.61 15.12 16.11]';
else
    N = 100;
    t = (0:1:N-1)';
    a = 2;
    b = -1;
    sigma = 1.5;
    bruit = sigma*randn(N,1);
    for k = 1:N
        y(k) = a*t(k)+b + bruit(k);
    end
    y = y';
end

M = [t ones(N,1)];
theta = [inv(M'*M)*M'*y];

figure(1)
plot(t, y, 'x')
hold on
grid()
xlabel('Temps t_k')
ylabel('Mesures y_k')
y_est = M*theta; %modele
plot(t, y_est)
legend('Mesures des y_k', 'Notre y estime')
title("Mesures y_k selon le temps")


figure(2)
M_p = [t.^2 t ones(N,1)];
theta_p = [inv(M_p'*M_p)*M_p'*y];
plot(t, y, 'x')
hold on
y_p = M_p*theta_p;
plot(t, y_p)
grid()
legend('Mesures des y_k', 'Notre y estime')
title("Mesures y_k selon le temps")
xlabel('Temps t_k')
ylabel('Mesures y_k')
e_p = y - y_p; %erreur
J_p = e_p'*e_p; %critere

figure(3)
M_p3 = [t.^3 t.^2 t ones(N,1)];
theta_p3 = [inv(M_p3'*M_p3)*M_p3'*y];
plot(t, y, 'x')
hold on
y_p3 = M_p3*theta_p3;
plot(t, y_p3)
grid()
legend('Mesures des y_k', 'Notre y estime')
title("Mesures y_k selon le temps")
xlabel('Temps t_k')
```

```
ylabel('Mesures y_k')
e_p3 = y - y_p3; %erreur
J_p3 = e_p3'*e_p3; %critere

figure(4)
e = y - y_est; %erreur
J = e'*e; %critere
plot(t, e, 'xr', 'Linewidth', 2)
hold on
plot(t, e_p, 'xb', 'Linewidth', 2)
plot(t, e_p3, 'xg', 'Linewidth', 2)
grid()
title("Erreur selon les t_k")
xlabel("t_k")
ylabel("erreur e")
legend("droite", "parabole", "polynome 3")
```

Exercice 2

```
clc; clear; close all

a = [1900 1910 1920 1930]';
a_p = [1900 1910 1920 1930 1940 1950]';
y = [1000 1050 1134 1201]';
N = length(a);
N_p = length(a_p);

%lineaire
figure(1)
M = [a ones(N,1)];
M_p = [a_p ones(N_p,1)];
theta = [inv(M'*M)*M'*y];
plot(a, y, 'x', 'Linewidth', 2)
hold on
y_l = M*theta;
plot(a, y_l)
grid()
y_p = M_p*theta;
hold on
plot(a_p, y_p, 'x', 'Linewidth', 2)
title("Prediction lineaire")
legend("Mesures", "Estimation", "Prediction")

%quadratique
figure(2)
a = a - 1900;
a_p = a_p - 1900;
M_q = [a.^2 a ones(N,1)];
M_qp = [a_p.^2 a_p ones(N_p,1)];
theta_q = [inv(M_q'*M_q)*M_q'*y];
plot(a, y, 'x', 'Linewidth', 2)
hold on
y_q = M_q*theta_q;
plot(a, y_q)
grid()
y_pq = M_qp*theta_q;
hold on
plot(a_p, y_pq, 'x', 'Linewidth', 2)
title("Prediction quadratique")
legend("Mesures", "Estimation", "Prediction")

%p3
figure(3)
M_p3 = [a.^3 a.^2 a ones(N,1)];
```

```
M_q = [a_p.^3 a_p.^2 a_p ones(N_p,1)];
theta_p3 = [inv(M_p3'*M_p3)*M_p3'*y];
plot(a, y, 'x', 'Linewidth', 2)
hold on
y_p3 = M_p3*theta_p3;
plot(a, y_p3)
grid()
y_p = M_q*theta_p3;
hold on
plot(a_p, y_p, 'x', 'Linewidth', 2)
title("Prediction avec polynome d'ordre 3")
legend("Mesures", "Estimation", "Prediction")
```

Exercice 2 bis

```
clc; clear; close all

a = [1900:10:1980]';
a_p = [1900:10:2000]';
y = [1000 1050 1104 1158 1212 1268 1323 1381 1400]';
N = length(a);
N_p = length(a_p);

%lineaire
figure(1)
M = [a ones(N,1)];
M_p = [a_p ones(N_p,1)];
theta = [inv(M'*M)*M'*y];
plot(a, y, 'x', 'Linewidth', 2)
hold on
y_c = M*theta;
plot(a, y_c)
grid()
y_p = M_p*theta;
hold on
plot(a_p, y_p, 'x', 'Linewidth', 2)
title("Prediction lineaire")
legend("Mesures", "Estimation", "Prediction")

%quadratique
figure(2)
a = a - 1900;
a_p = a_p - 1900;
M_q = [a.^2 a ones(N,1)];
M_qp = [a_p.^2 a_p ones(N_p,1)];
theta_q = [inv(M_q'*M_q)*M_q'*y];
plot(a, y, 'x', 'Linewidth', 2)
hold on
y_q = M_q*theta_q;
plot(a, y_q)
grid()
y_pq = M_qp*theta_q;
hold on
plot(a_p, y_pq, 'x', 'Linewidth', 2)
title("Prediction quadratique")
legend("Mesures", "Estimation", "Prediction")

%p3
figure(3)
M_p3 = [a.^3 a.^2 a ones(N,1)];
M_q = [a_p.^3 a_p.^2 a_p ones(N_p,1)];
theta_p3 = [inv(M_p3'*M_p3)*M_p3'*y];
plot(a, y, 'x', 'Linewidth', 2)
hold on
```

```
y_p3 = M_p3*theta_p3;
plot(a, y_p3)
grid()
y_p = M_q*theta_p3;
hold on
plot(a_p, y_p, 'x', 'Linewidth', 2)
title("Prediction avec polynome d'ordre 3")
legend("Mesures", "Estimation", "Prediction")

%p7
figure(4)
M_p7 = [a.^7 a.^6 a.^5 a.^4 a.^3 a.^2 a ones(N,1)];
M_q = [a_p.^7 a_p.^6 a_p.^5 a_p.^4 a_p.^3 a_p.^2 a_p ones(N_p,1)];
theta_p7 = [inv(M_p7'*M_p7)*M_p7'*y];
plot(a, y, 'x', 'Linewidth', 2)
hold on
y_p7 = M_p7*theta_p7;
plot(a, y_p7)
grid()
y_p = M_q*theta_p7;
hold on
plot(a_p, y_p, 'x', 'Linewidth', 2)
title("Prediction avec polynome d'ordre 7")
legend("Mesures", "Estimation", "Prediction")
```

Exercice 3

```
clc; clear; close all

test = menu("Choix de N", "N = 500", "N = 2000", "N = 5000");
if test == 1
    N = 500;
    test2 = menu("Choix du sigma", "sigma = 0", "sigma = 1", "sigma = 2");
    if test2 == 1
        sigma = 0;
    elseif test2 == 2
        sigma = 1;
    else
        sigma = 2;
    end
elseif test == 2
    N = 2000;
    test2 = menu("Choix du sigma", "sigma = 0", "sigma = 1", "sigma = 2");
    if test2 == 1
        sigma = 0;
    elseif test2 == 2
        sigma = 1;
    else
        sigma = 2;
    end
else
    N = 5000;
    test2 = menu("Choix du sigma", "sigma = 0", "sigma = 1", "sigma = 2");
    if test2 == 1
        sigma = 0;
    elseif test2 == 2
        sigma = 1;
    else
        sigma = 2;
    end
end
k = [1:N]';
t = (k-1)*0.001;
a = 3;
```

```
phi = pi/5;
f0 = 20;

e = sigma*randn(N,1);
x = a*sin(2*pi*f0*t + phi);
y = x + e;

figure(1)
plot(t, x, 'Linewidth', 2)
hold on
plot(t, y)
grid()
M = [sin(2*pi*f0*t) cos(2*pi*f0*t)];
theta = [inv(M'*M)*M'*y];
y_c = M*theta;
plot(t, y_c, 'x', 'Linewidth', 2)
phi_c = atan(theta(2)/theta(1))
a_c = sqrt(theta(1)^2 + theta(2)^2)
title("Signal")
legend("x : signal vrai", "y : signal mesure", "estimation")
```

Exercice 4

```
clc; clear; close all

N = 100;
k = [1:N]';
dt = 100*10^-3;
t = (k-1)*dt;
x0 = 5;
v = 10;

sigma = 3;
e = sigma*randn(N,1);
x = x0 + v*t;
y = x + e;

figure(1)
plot(t, x)
hold on
plot(t, y)
grid()
M = [t ones(N,1)];
theta_c = [inv(M'*M)*M'*y];
y_c = M*theta_c;
plot(t, y_c, 'x', 'Linewidth', 2)
title("Signaux")
legend("x : signal vrai", "y : signal mesure", "estimation")
```

Exercice 3 bis

```
clc; clear; close all

N = 2000;
k = [1:N]';
t = (k-1)*0.001;
p = 2;

phi = pi/5;
f0 = 10;
```

```
sigma = 0.1;
e = sigma*randn(N,1);


%a, x, y
alpha = ((5-3)/(1250-750));
beta = 3-alpha*750;
x = zeros(N,1);
y = zeros(N,1);
for i = 1:N
    if k(i) <= 750
        a(i) = 3;
    elseif 1250 <= k(i)
        a(i) = 5;
    else
        a(i) = alpha*k(i) + beta;
    end
    x(i) = a(i)*sin(2*pi*f0*t(i) + phi);
    y(i) = x(i) + e(i);
end

%mcr lambda
theta_mcr = zeros(p,N); %initialisation
P = 10^12*eye(p,p);
theta_mcr(:,1) = [0;0];
lambda = 0.95;
for j = 2:N
    m = [sin(2*pi*f0*t(j)) cos(2*pi*f0*t(j))]';
    k = (P*m)/(lambda+m'*P*m);
    P = (1/lambda)*(P - k*m'*P);
    y_est(:,j) = m'*theta_mcr(:,j-1);
    theta_mcr(:,j) = theta_mcr(:,j-1) + k*(y(j)-y_est(:,j));
    a_mcr(:,j) = sqrt(theta_mcr(1,j).^2 + theta_mcr(2,j).^2);
    phi_mcr(:,j) = atan2(theta_mcr(2,j),theta_mcr(1,j));
end

%fenetre glissante
M = [sin(2*pi*f0*t) cos(2*pi*f0*t)];
Nf = 400;
u = 1:Nf;
for i = 1:(N-Nf+1)
    Mf = M(u,:); %1 a nf, toutes colonnes
    yf = y(u);
    theta_f(:,i) = inv(Mf'*Mf)*Mf'*yf;
    u = u + 1;
    a_f(i) = sqrt(theta_f(1,i)^2 + theta_f(2,i)^2);
    phi_f(i) = atan2(theta_f(2,i),theta_f(1,i));
end

figure(1)
plot(a, 'b')
grid()
hold on
plot(x, 'Linewidth', 2)
plot(y)
plot(-a, 'b')
% plot(y_est, 'x')
legend("a", "x: signal vrai","y : signal mesure", "-a", "y estime")
title("Signaux + a")

figure(2)
subplot(2,1,1)
plot(a)
hold on
plot(a_mcr)
grid()
```

```
legend("a", "a estime")
subplot(2,1,2)
plot(phi*ones(N,1))
hold on
plot(phi)
grid()
legend("phi", "phi estime")

figure(3)
subplot(2,1,1)
plot(t(1:length(a)), a)
hold on
plot(t(1:length(a_f)), a_f)
grid()
legend("a", "a estime")

subplot(2,1,2)
plot(t, phi*ones(N,1))
hold on
plot(t(1:length(phi_f)), phi_f)
grid()
legend("\phi", "\phi estime")

title("Estimation des parametres - fenetre glissante")
```

Exercice 5

```
clc; clear; close all

N = 2000;
a = 10;
dt = 0.001;
k = [1:N]';
t = (k-1)*dt;

sigma = 10;
e = sigma*randn(N,1);
y = a + e;

%mco
t1 = cputime;
for i = 1:N
    M = ones(i,1);
    theta(i) = [inv(M'*M)*M'*y(1:i)];
end
temps_MCO = cputime - t1

%mcr
t2 = cputime;
theta_mcr(1) = 0; %initialisation
P = 10^12;
for j = 1:N-1
    m = 1;
    k(j+1) = (P(j)*m)/(1+m'*P(j)*m);
    P(j+1) = P(j) - k(j+1)*m'*P(j);
    y_est = m'*theta_mcr(j);
    theta_mcr(j+1) = theta_mcr(j) + k(j+1)*(y(j+1)-y_est);
end
temps_MCR = cputime - t2

figure(1)
plot(t, y)
grid()
title("Donnees")
```

```
figure(2)
plot(t,theta)
hold on
plot(t,a*ones(N,1), 'r')
grid()
legend("\theta estime", "\theta original")
title("MCO")

figure(3)
plot(t,theta_mcr)
grid()
hold on
plot(t,a*ones(N,1), 'r')
plot(t,theta)
plot(t, k)
title("MCR")
legend("MCR", "a", "MCO", "gain de correction")
```

Exercice 6

```
clc; clear; close all

N = 2000;
k = [1:N]';
dt = 0.001;
t = (k-1)*dt;
a = 3;
phi = pi/5;
f0 = 20;
p = 2;

sigma = 1;
e = sigma*randn(N,1);
x = a*sin(2*pi*f0*t + phi);
y = x + e;

figure(1)
for i = 1:N
    M = [sin(2*pi*f0*t(i)) cos(2*pi*f0*t(i))]';
    theta(:,i) = [inv(M'*M)*M'*y(i)];
end
plot(t, theta')
grid()
title("MCO")

theta_mcr = zeros(p,N); %initialisation
P = 10^12*eye(p,p);
theta_mcr(:,1) = [0;0];

for j = 2:N
    m = [sin(2*pi*f0*t(j)) cos(2*pi*f0*t(j))]';
    k = (P*m)/(1+m'*P*m);
    P = P - k*m'*P;
    y_est = m'*theta_mcr(:,j-1);
    theta_mcr(:,j) = theta_mcr(:,j-1) + k*(y(j)-y_est);
    phi_mcr(:,j) = atan2(theta_mcr(2,j),theta_mcr(1,j));
    a_mcr(:,j) = sqrt(theta_mcr(1,j).^2 + theta_mcr(2,j).^2);
end

figure(2)
plot(t, theta_mcr')
grid()
title("MCR : \theta")
```

```
legend("\theta_1 estime", "\theta_2 estime")

figure(3)
plot(t,phi_mcr)
hold on
plot(t,a_mcr)
plot(t,phi*ones(N,1))
plot(t,a*ones(N,1))
grid()
title("\phi et a")
legend("\phi estime", "a estime", "\phi", "a")
```

Exercice 4 bis

```
clc; clear; close all

test = menu("Choix du N", "N = 500", "N = 5000");
if test == 1
    N = 500;
else
    N = 5000;
end
k = [1:N]';
dt = 100*10^-3;
t = (k-1)*dt;
x0 = 5;
v = 10;
p = 2;

sigma = 3;
e = sigma*randn(N,1);
x = x0 + v*t;
y = x + e;

theta_mcr = zeros(p,N); %initialisation
P = 10^12*eye(p,p);
theta_mcr(:,1) = [0;0];

t1 = cputime;
for j = 2:N
    m = [t(j) 1]';
    k = (P*m)/(1+m'*P*m);
    P = P - k*m'*P;
    y_est = m'*theta_mcr(:,j-1);
    theta_mcr(:,j) = theta_mcr(:,j-1) + k*(y(j)-y_est);
    x0_mcr(:,j) = theta_mcr(2,j);
    v_mcr(:,j) = theta_mcr(1,j);
end
t1 = cputime-t1;
figure(1)
plot(t,theta_mcr')
grid()
title("\theta")
legend("\theta_1", "\theta_2")

figure(2)
plot(t,x0_mcr)
hold on
plot(t,v_mcr)
plot(t, x0*ones(N,1))
plot(t, v*ones(N,1))
grid()
legend("x0 estime", "v estime", "x0", "v")
title("x0 et v")
```

Exercice 6 ter

```
clc; clear; close all

N = 2000;
k = [1:N]';
dt = 0.001;
t = (k-1)*dt;
p = 2;
f0 = 10;

sigma = 1;
e = sigma*randn(N,1);


%a, x, y
alpha = ((5-3)/(1250-750));
beta = 3-alpha*750;
gamma = (((2*pi/5)-(pi/5))/(1250-750));
delta = (pi/5)-gamma*750;
x = zeros(N,1);
y = zeros(N,1);

for i = 1:N
    if k(i) <= 750
        a(i) = 3;
        phi(i) = pi/5;
    elseif 1250 <= k(i)
        a(i) = 5;
        phi(i) = 2*pi/5;
    else
        a(i) = alpha*k(i) + beta;
        phi(i) = gamma*k(i) + delta;
    end
    x(i) = a(i)*sin(2*pi*f0*t(i) + phi(i));
    y(i) = x(i) + e(i);
end

%mcr lambda
theta_mcr = zeros(p,N); %initialisation
P = 10^12*eye(p,p);
theta_mcr(:,1) = [0;0];
lambda = 0.95;
t1 = cputime;
for j = 2:N
    m = [sin(2*pi*f0*t(j)) cos(2*pi*f0*t(j))]';
    k = (P*m)/(lambda+m'*P*m);
    P = (1/lambda)*(P - k*m'*P);
    y_est(:,j) = m'*theta_mcr(:,j-1);
    theta_mcr(:,j) = theta_mcr(:,j-1) + k*(y(j)-y_est(:,j));
    a_mcr(:,j) = sqrt(theta_mcr(1,j).^2 + theta_mcr(2,j).^2);
    phi_mcr(:,j) = atan2(theta_mcr(2,j),theta_mcr(1,j));
end
t1 = cputime-t1;

%fenetre glissante
M = [sin(2*pi*f0*t) cos(2*pi*f0*t)];
Nf = 100;
u = 1:Nf;
t2 = cputime;
for i = 1:(N-Nf+1)
    Mf = M(u,:); %1 a nf, toutes colonnes
```

```
    yf = y(u);
    theta_f(:,i) = inv(Mf'*Mf)*Mf'*yf;
    u = u + 1;
    a_f(i) = sqrt(theta_f(1,i)^2 + theta_f(2,i)^2);
    phi_f(i) = atan2(theta_f(2,i),theta_f(1,i));
end
t2 = cputime-t2;
tf = [1:N-Nf+1]'*dt;

figure(1)
plot(a)
grid()
hold on
plot(y)
plot(-a)
plot(y_est, 'x')
legend("a", "y : signal mesure", "-a", "y estime")

figure(2)
subplot(2,1,1)
plot(t,a)
hold on
plot(t,a_mcr)
grid()
legend("a", "a estime")
subplot(2,1,2)
plot(t, phi)
hold on
plot(t, phi_mcr)
grid()
legend("\phi", "\phi estime")
title("Estimation des parametres - MCR")

figure(3)
subplot(2,1,1)
plot(t, a)
hold on
plot(tf, a_f)
grid()
legend("a", "a estime")

subplot(2,1,2)
plot(t, phi)
hold on
plot(tf, phi_f)
grid()
legend("\phi", "\phi estime")
title("Estimation des parametres - fenetre glissante")
```

Exercice 8

```
clc; clear; close all

f0 = 0.05; %Hz
N = 40;
dt = 1; %seconde
M = 8*N; %nbr de mesures
L = 32*N;
sigma = 0;
e = sigma*randn(L,1);

th = (0:N-1)'*dt;
h = sin(2*pi*f0*th).*exp(-0.2*th); %a trouver
```

```
t = (0:L-1)'*dt;

alpha = 0.002;
test = menu("Choix du type d'entree", 'exp(-0.005*t)', 'quest 2', 'quest 3', 'carree'
    , 'spba');
if test == 1
    u = exp(-0.005*t);
elseif test == 2
    t1 = t(1:M/2);
    t2 = t(M/2+1:L);
    u = [exp(-0.005*t1);exp(-0.005*t2)+2*exp(-0.001*t2)];
elseif test == 3
    t1 = t(1:M/4);
    t2 = t((M/4)+1:M/2);
    t3 = t((M/2)+1:(3*M/4));
    t4 = t(((3*M/4)+1):L-1);
    for z = 1:2
        s1 = z*exp(-z*alpha*t3);
    end
    for z = 1:3
        s2 = z*exp(-z*alpha*t4);
    end
    u = [exp(-0.005*t1);exp(-0.005*t2)+2*exp(-alpha*t2);exp(-0.005*t3)+2*s1;exp
        (-0.005*t4)+2*s2];
elseif test == 4
    D = pi/40;
    u = square(0:D:(L-1)*D)';
elseif test == 5
    u = sbpa(dt, 8, 1, 1, 20);
    u = u(1:L);
end
test2 = menu("Choix du J", "J = 0", "J = 100");
if test2 == 1
    J = 0;
else
    J = 100;
end
y = conv(u, h);
y_b = y(1:L) + e;
indice = N+J:N+J+M-1;
U = u(indice);

for k = 1:N-1
    U = [U u(indice-k)];
end
h_est = inv(U'*U)*U'*y_b(indice);
y_est = conv(u, h_est);

epsilon = sqrt((abs(h_est-h)).^2); %erreur quadratique moyenne
figure(1)
subplot(3,1,1)
plot(u)
grid()
legend("Entree u")
title("Entree u")

subplot(3,1,2)
plot(h)
hold on
plot(h_est)
title("h et h_{est}")
grid()
legend("h", "h_{est}")
title("Reponses impulsionnelles")

subplot(3,1,3)
```

```
plot(y)
hold on
plot(y_b)
grid()
legend("y", "y bruite")
title("Sorties avec et sans bruit, \sigma = 0.5")
```

Exercice 9

```
clc; clear; close all

choix_spba = menu("Choix de la sequence", "spba : p = 7", "spba : p = 16", "spba :
    bruit gaussien blanc, p = 28");
if choix_spba == 1
    spba = [1 -1 1 1 1 -1 -1]';
    p = 7;
elseif choix_spba == 2
    spba = [1 -1 1 1 1 -1 -1 1 -1 1 1 -1 -1 1 1 -1]';
    p = 16;
elseif choix_spba == 3
    spba = randn(28,1);
    p = 28;
end
Nu = 30*p;

k = 30;
u = spba;
for i = 1:k-1
    u = [u; spba];
end

figure(1)
plot(u)
grid()
title("Entree u")

h = zeros(k,1);
h(1) = 0;
for n = 1:k
    h(n) = (0.3)^(n-1)*0.7;
end
Nh = 12;
h = [0;h];
t = [0:Nh-1]';

figure(2)
plot(t, h(1:Nh))
grid()
title("Reponse impulsionnelle h")

y = conv(u, h);
sigma = 0.2;
e = sigma*randn(length(y),1);
y = y + e;
figure(3)
plot(y)
grid()
title("Sortie y bruitee de \sigma = 0.2")

figure(4)
M = 80; %nbr echantillons
v = [0:M-1]';
plot(v, u(1:M))
hold on
```

```
plot(v, y(1:M))
grid()
legend("u", "y")
title("Entree u et sortie y sur 80 echantillons")

test = menu("Choix du N", "p-1", "p", "Nh-1", "Nh", "p+1");
if test == 1
    N = p-1;
elseif test == 2
    N = p;
elseif test == 3
    N = Nh-1;
elseif test == 4
    N = Nh;
else
    N = p+1;
end

indice = N:N+M-1;
U = u(indice);
for l = 1:N-1
    U = [U u(indice-l)];
end
h_est = inv(U'*U)*U'*y(indice);

figure(5)
plot(h(1:length(h_est)))
hold on
plot(h_est)
grid()
legend("h", "h estime")
title("h et h estime")
```

Exercice 10

```
clc; clear; close all

N = 900;
k = 0:N-1;
dt = 1;
t = (k*dt)';
p = 2;
T = 150;
test = menu("Choix du theta", "theta stationnaire", "theta non stationnaire");
if test == 1
    lambda = 1;
    for k = 1:N
        a1(k) = -0.5;
        b0(k) = 0.5;
    end
else
    lambda = 0.95;
    for k = 1:N
        if k <= 350
            a1(k) = -0.5;
            b0(k) = 0.3;
        else
            a1(k) = 0.5;
            b0(k) = -0.3;
        end
    end
end

test = menu("Choix de la sequence u", "signal carre", "spba");
```

```
if test == 1
    u = square(2*pi/T*t);
else
    u = randn(N,1);
end

s = zeros(N,1);
y = zeros(N,1);
s(1) = b0(1)*u(1);
sigma = 0.02;
e = sigma*randn(N,1);
for i = 2:N
    s(i) = -a1(i)*s(i-1)+b0(i)*u(i);
    y(i) = s(i)+e(i);
end

k = 800;
C = [-y(1:k) u(2:k+1)];
theta_est = inv(C'*C)*C'*y(2:k+1);

theta_mcr = zeros(p,N); %initialisation
P = 10^12*eye(p,p);
theta_mcr(:,1) = [0;0];

for j = 2:N
    m = [-y(j-1) u(j)]';
    k = (P*m)/(lambda+m'*P*m);
    P = (P - k*m'*P)/lambda;
    y_est = m'*theta_mcr(:,j-1);
    theta_mcr(:,j) = theta_mcr(:,j-1) + k*(y(j)-y_est);
end

figure(1)
plot(s)
hold on
plot(y)
grid()
title("Sortie y bruitee et non bruitee")
legend("non bruitee", "bruitee")

figure(2)
plot(theta_mcr')
grid()
hold on
plot(a1)
plot(b0)
title("\theta estime")
legend("\theta_1 estime", "\theta_2 estime", "\theta_1 reel", "\theta_2 reel")
```

Exercice 11

```
clc; clear; close all

N = 900;
k = 0:N-1;
dt = 1;
t = (k*dt)';
p = 4;
T = 150;
test = menu("Choix du theta", "theta stationnaire", "theta non stationnaire");
if test == 1
    lambda = 1;
    for k = 1:N
        a1(k) = -0.5;
```

```
        a2(k) = 0.3;
        b0(k) = 0.5;
        b1(k) = 1;
    end
else
    lambda = 0.95;
    for k = 1:N
        if k <= 350
            a1(k) = -0.5;
            a2(k) = 0.3;
            b0(k) = 0.5;
            b1(k) = 1;
        elseif k <= 600
            a1(k) = 0.5;
            a2(k) = -0.3;
            b0(k) = 0.5;
            b1(k) = 1;
        else
            a1(k) = 0.5;
            a2(k) = -0.3;
            b0(k) = 0.9;
            b1(k) = 1.6;
        end
    end
end

test = menu("Choix de la sequence u", "signal carre", "spba");
if test == 1
    u = square(2*pi/T*t);
else
    u = randn(900,1);
end

s = zeros(N,1);
y = zeros(N,1);
s(1) = b0(1)*u(1);
s(2) = -a1(2)*s(1)+b0(2)*u(2)+b1(2)*u(1);
test = menu("Choix du sigma", "0", "0.02", "0.2", "0.5");
if test == 1
    sigma = 0;
elseif test == 2
    sigma = 0.02;
elseif test == 3
    sigma = 0.2;
else
    sigma = 0.5;
end
e = sigma*randn(N,1);
for i = 3:N
    s(i) = -a1(i)*s(i-1)-a2(i)*s(i-2)+b0(i)*u(i)+b1(i)*u(i-1);
    y(i) = s(i)+e(i);
end

k = 800;
C = [-y(2:k) -y(1:k-1) u(3:k+1) u(2:k)];
theta_est = inv(C'*C)*C'*y(2:k);

theta_mcr = zeros(p,N); %initialisation
P = 10^12*eye(p,p);
theta_mcr(:,1) = [0 0 0 0];

for j = 3:N
    m = [-y(j-1) -y(j-2) u(j) u(j-1)]';
    k = (P*m)/(lambda+m'*P*m);
    P = (P - k*m'*P)/lambda;
    y_est = m'*theta_mcr(:,j-1);
```

```
    theta_mcr(:,j) = theta_mcr(:,j-1) + k*(y(j)-y_est);
end

figure(1)
plot(s)
hold on
plot(y)
grid()
title("Sortie y bruitee et non bruitee")
legend("non bruitee", "bruitee")

figure(2)
plot(theta_mcr')
grid()
hold on
plot(a1, 'b.')
plot(a2, 'r.')
plot(b0, 'y.')
plot(b1, 'm.')
title("\theta estime")
legend("a_1 estime", "a_2 estime", "b_0 estime", "b_1 estime","a_1 reel", "a_2 reel",
    "b_0 reel", "b_1 reel")
```

Exercice 12

reponse

```
function s = reponse(a, b, u)
    N = length(u);
    L = length(a);
    P = length(b);
    s = zeros(N,1);
    for i = 1:L
        s(i) = 0;
        for k = 1:i-1
            s(i) = s(i) - a(k)*s(i-k);
        end
        for k = 0:i-1
            s(i) = s(i) + b(k+1)*u(i-k);
        end
    end
    for i = L+1:N
        s1 = 0;
        for k = 1:L
            s1 = s1 + a(k)*s(i-k);
        end
        s2 = 0;
        for k = 0:P-1
            s2 = s2 + b(k+1)*u(i-k);
        end
        s(i) = -s1 + s2;
    end
end
```

critere

```
function J = critere(a, b, u, y)
    y_m = reponse(a, b, u);
    e = y_m - y;
    J = e'*e;
end
```

gradient

```
function dJ = gradient(a, b, u, y)
    N = length(a);
    M = length(b);
    h = 10^-5;
    theta = [a b];
    for k = 1:length(theta)
        J = critere(a, b, u, y);
        for i = 1:N
            a_est = a;
            a_est(i) = a(i) + h;
            dJa(i) = (critere(a_est, b, u, y) - J)/h;
        end
        for i = 1:M
            b_est = b;
            b_est(i) = b(i) + h;
            dJb(i) = (critere(a, b_est, u, y) - J)/h;
        end
        dJ = [dJa;dJb];
    end
end
```

algodescenteGradient

```
function theta_est = algodescenteGradient(a, b, u, y, lambda)
    J = critere(a, b, u, y);
    theta = [a;b];
    flag = true;
    k = 1;
    while flag
        theta = theta - k*lambda*gradient(theta(1),theta(2),u,y);
        J_est = critere(theta(1),theta(2), u, y);
        if J_est >= J
            flag = false;
            theta_est = theta;
        end
        k = k+1;
        J = J_est;
    end
end
```

Programme principal

```
clc; clear; close all

T = 150;
N = 900;

a1 = -0.5;
b0 = 0.5;
a_est = -0.35;
b_est = 0.4;
sigma = 0.02;
e = sigma*randn(N, 1);
k = 0:N-1;
dt = 1;
t = (k*dt)';
theta = [a1 b0];
u = square(2*pi/T*t);
```

```
s = reponse(a1, b0, u);
y = s + e;

figure(1)
% J = critere(a_est, b_est,u,y);
% dJ = gradient(a_est, b_est, u, y);
plot(u)
hold on
plot(s, "LineWidth", 2)
plot(y)
grid()
legend("Entree u", "Sortie y_{M,k}", "Sortie y_k")
title("Entree, sortie et sortie bruitee")

lambda = 10^-5;
flag = true;
theta_0 = [a_est;b_est];
Theta = theta_0;
J1 = critere(theta_0(1), theta_0(2), u, y);
k = 1;
J_p = J1;
while flag
    theta_tmp = algodescenteGradient(theta_0(1),theta_0(2),u,y,lambda);
    J_est = critere(theta_tmp(1),theta_tmp(2), u, y);

    if abs(J_est - J1) < 1e-5
        flag = false;
        theta_est = theta_0;
    end
    J1 = J_est;
    theta_0 = theta_tmp;
    k = k+1;
    Theta = [Theta theta_tmp];
    J_p = [J_p J_est];
end

figure(2)
subplot(2,1,1)
plot(Theta')
grid()
legend("\theta_1", "\theta_2")
title("\theta estime")
subplot(2,1,2)
plot(J_p)
grid()
title("Evolution du critere")
```