

TP Kalman

Compte Rendu

Yves ROBERT DU BOISLOUVEAU

Ngatam THIEBAUT

Table des matières

Mise en œuvre d'un filtre Kalman linéaire	3
Linéarisation autour d'un point d'équilibre:	3
Simulation du système linéarisé avec filtre de Kalman linéaire:.....	5
Étude de l'influence des paramètre du filtre sur les erreurs d'estimations	9
Conclusion sur le filtre de Kalman linéaire sur un modèle linéarisé	16
Simulation du système non linéaire avec filtre de Kalman linéaire:	17
Conclusion sur le filtre de Kalman linéaire sur un modèle non linéaire	19
Mise en œuvre d'un filtre de Kalman étendu	20
Simulation avec le filtre de Kalman étendu :	20
Étude de l'influence des paramètre du filtre sur les erreurs d'estimations	24
Conclusion sur le filtre de Kalman étendu	31
Évaluation des performances du filtre de Kalman étendu avec des simulations de Monte-Carlo	32
Simulations de Monte-Carlo pour le filtre de Kalman étendu	32
Étude de l'influence des paramètre du filtre sur les erreurs d'estimations avec les simulations de Monte-Carlo	37

Mise en œuvre d'un filtre Kalman linéaire

Linéarisation autour d'un point d'équilibre:

On considère les équations suivantes du système :

$$\begin{cases} \dot{\underline{x}}(t) = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} -x_2 * u + \beta \\ -\alpha * x_2^2 + u * x_2 \end{bmatrix} + \zeta(t) \\ y(t) = C \underline{x} = [1 \ 0] \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \omega(t) \end{cases}$$

On note la position d'équilibre $\underline{x}_e = [x_{e_1} \ x_{e_2}]^\tau$ pour une consigne \underline{u}_e . Cette position est donnée par le système suivant :

$$\begin{cases} \dot{x}_{e_1} = 0 \\ \dot{x}_{e_2} = 0 \end{cases} \Leftrightarrow \begin{cases} -x_2 * u + \beta = 0 \\ -\alpha * x_2^2 + u * x_2 = 0 \end{cases} \Leftrightarrow \{x_2 = \sqrt{\frac{\beta}{\alpha}}$$

On en déduit alors que la position d'équilibre est :

$$\underline{x}_e = \begin{bmatrix} x_{e_1} \\ x_{e_2} \sqrt{\frac{\beta}{\alpha}} \end{bmatrix}^\tau = \begin{bmatrix} x_{e_1} \\ \frac{1}{2} \end{bmatrix}^\tau \text{ et } \underline{u}_e = \sqrt{\beta * \alpha} = 2$$

Notons que, nos équations ne nous imposent pas de contraintes sur x_{e_1} , pour simplifier notre étude, nous allons dans un premier temps prendre $x_{e_1} = 0$.

L'on va maintenant considérer un léger déplacement depuis cette position, l'on notera :

$$\begin{cases} x_1 = x_{e_1} + \delta x_{e_1} \\ x_2 = x_{e_2} + \delta x_{e_2} \end{cases}$$

Le développement limité est donné par :

$$f(\varepsilon, \theta) = f(\varepsilon_0, \theta_0) + \left. \frac{\partial f}{\partial \varepsilon} \right|_{\varepsilon_0, \theta_0} (\varepsilon - \varepsilon_0) + \left. \frac{\partial f}{\partial \theta} \right|_{\varepsilon_0, \theta_0} (\theta - \theta_0) + t.o.s.$$

On note :

$$\begin{cases} f_1(\underline{x}, \underline{u}) = -x_2 * u + \beta + \xi \\ f_2(\underline{x}, \underline{u}) = -\alpha * x_2^2 * u + u * x_2 + \xi \end{cases}$$

On a alors :

$$f(\underline{x}, \underline{u}) = f(\underline{x}_e, \underline{u}_e) + \left. \frac{\partial f}{\partial \underline{x}} \right|_{x_e, u_e} (x - x_e) + \left. \frac{\partial f}{\partial \underline{u}} \right|_{x_e, u_e} (u - u_e)$$

Et :

$$\begin{cases} f_1(\underline{x}, \underline{u}) = f_1(\underline{x}_e, \underline{u}_e) + \left. \frac{\partial f_1}{\partial \underline{x}} \right|_{x_e, u_e} (x - x_e) + \left. \frac{\partial f_1}{\partial \underline{u}} \right|_{x_e, u_e} (u - u_e) \\ f_2(\underline{x}, \underline{u}) = f_2(\underline{x}_e, \underline{u}_e) + \left. \frac{\partial f_2}{\partial \underline{x}} \right|_{x_e, u_e} (x - x_e) + \left. \frac{\partial f_2}{\partial \underline{u}} \right|_{x_e, u_e} (u - u_e) \end{cases}$$

On va noter les matrices A et B telles que l'on ait :

$$\begin{cases} \dot{\underline{x}}_l = A\underline{x}_l + B\underline{u}_l + \zeta(t) \\ y_l = C\underline{x}_l + \omega(t) \end{cases} \quad (2)$$

avec $\underline{x}_l = \underline{x} - \underline{x}_e$, $\underline{u}_l = \underline{u} - \underline{u}_e$ et $y_l = y - x_{le}$.

On en déduit que :

$$A = \begin{bmatrix} 0 & -u_e \\ 0 & -u_e \end{bmatrix} = -\begin{bmatrix} 0 & \sqrt{\alpha * \beta} \\ 0 & \sqrt{\alpha * \beta} \end{bmatrix} = -2 \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix}$$

Et :

$$B = \begin{bmatrix} -x_{e_2} \\ x_{e_2} \end{bmatrix} = \begin{bmatrix} -\sqrt{\frac{\beta}{\alpha}} \\ \sqrt{\frac{\beta}{\alpha}} \end{bmatrix} = \begin{bmatrix} -0.50 \\ 0.50 \end{bmatrix}$$

Nous allons maintenant utiliser ces matrices pour l'implémentation dans MatLab.

Simulation du système linéarisé avec filtre de Kalman linéaire:

Après implémentation des matrices A et B dans le fichier Linearise, on a :

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  
% Linéarisation  
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  
  
function[A,B]=Linearise(X2e,Ue,Param)  
  
A= [0 -sqrt(Param.alpha*Param.beta);0 -sqrt(Param.alpha*Param.beta)];  
  
B= [-sqrt(Param.beta/Param.alpha);sqrt(Param.beta/Param.alpha)];
```

Puis, on met en place le filtre de Kalman linéaire dans le fichier LanceSimFK, ce code devient :

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  
% Simulations et Filtre de Kalman  
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  
  
%% Mise à zéro du workspace  
  
close all %ferme toutes les figures  
clear all %efface le "workspace"  
clc;  
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  
  
%% PARAMETRES DU SYSTEME REEL  
  
eval('ParametresReels'); %execute le fichier ParametresReels.m  
  
%temps de fin de simulation (en secondes)  
ParamReels.tfin=10;  
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  
  
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  
%%PARAMETRES DU FILTRE  
  
ParamFilt=ParamReels; %recopie des paramètres réels  
  
%Estimé initial  
ParamFilt.X0=ParamReels.X0;  
  
%Covariance de l'erreur d'estimation initiale  
ParamFilt.P00=diag([1e-3,1e-3]);  
  
%Covariance des bruits de dynamique  
ParamFilt.Q=diag([1e-5,1e-5]);  
%-----  
%Pour Discréteriser  
  
%Définition de la représentation d'état linéaire à temps continu  
SystCont=ss(ParamFilt.A,ParamFilt.B,ParamFilt.C,0);  
  
%discréterisation  
SystDiscr=c2d(SystCont,ParamFilt.Te);  
  
%Mise en forme pour utilisation plus pratique  
ParamFilt.F=SystDiscr.a;  
ParamFilt.G=SystDiscr.b;  
%-----  
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  
%% SIMULATION  
  
%exécution du fichier Simulink Linéaire : SimLin  
%sim('SimLin',ParamReels.tfin); si pas Simulink alors télécharger les données  
load DataLin; % Donées système linéarisé
```

```

%Execution du fichier Simulink Non linaire: SimNL
%sim('SimNL',ParamReels.tfin); % a utiliser si vous avez simulink
%load DataNLin; % Données système non-linéaire

%-----
NbMes=length(YReel); %Nombre de points de mesure
%-----
%Bruitage des mesures

%Génération d'un vecteur, de même taille que YReel,
%composé de de bruits gaussiens d'espérance nulle
% de covariance 1
BruitsCov1=randn(NbMes,1);

%Génération d'un vecteur, de même taille que YReel,
%composé de de bruits gaussiens d'espérance nulle
% de covariance ParamReels.R
Bruits=sqrt(ParamReels.R)*BruitsCov1;

%addition des bruits à YReel -> Y mesuré
Ymes = YReel + Bruits;
%-----
%%%%%%%%%%%%%
%%% FILTRAGE

%-----
% initialisation

%mise en forme
Xkk=ParamFilt.X00-[0;ParamFilt.Equilibre.X2e];
Pkk=ParamFilt.P00;
U=UReel-ParamFilt.Equilibre.Ue*ones(NbMes,1);

%Stockage
Resultat.HatX(1,:)=Xkk'; %Estimé
Resultat.DiagPkk(1,:)=diag(Pkk)'; %Composantes diag de Pkk
%-----

%-----
%Récurrence

for k=1:NbMes-1,
%......

%Prediction
Xkplus= ParamFilt.F*Xkk + ParamFilt.G*U(k,1);
Pkplus= ParamFilt.F*Pkk*ParamFilt.F' + ParamFilt.Q;
%Correction
K= Pkplus*ParamFilt.C'*inv(ParamFilt.C*Pkplus*ParamFilt.C+ParamFilt.R);
Xkk= Xkplus+K*(Ymes(k+1,1)-ParamFilt.C*Xkplus);
Pkk= (eye(2)-K*ParamFilt.C)*Pkplus;
%......



%......



%Stockage
% codé le Filtre de Kalman :
Resultat.HatX(k+1,:)=Xkk';
Resultat.DiagPkk(k+1,:)=diag(Pkk)';
Resultat.K(k+1,:)=K';
%.....
end;
%-----



%-----
%Prise en compte de la linéarisation : x2 = X2lin + X2e
% et construction de l'erreur d'estimation
% De-commentez les lignes ci-dessous lorsque vous aurez
% codé le Filtre de Kalman :

Resultat.HatX(:,2)=Resultat.HatX(:,2)+...
ParamFilt.Equilibre.X2e*ones(NbMes,1);

ErrEstim = XReel-Resultat.HatX;
%-----

```

```

%%%%%%%%%%%%%%%
%%%%% TRACES
numfig=0;
Texte={'X1','X2'};

%estimé et valeur réelle
for i=1:2,
    numfig=numfig+1;
    figure(numfig)
    xlabel('t')
    hold on
    a=plot(Tps,XReel(:,i),'b-');
    b=plot(Tps,Resultat.HatX(:,i),'r-');
    legend([a,b],'Reel','Estime')
    title(['Estimation de ',Texte{i}])
    grid on
end;

%erreur d'estimation
for i=1:2,
    numfig=numfig+1;
    figure(numfig)
    xlabel('t')
    hold on
    %plot(Tps,XReel(:,i)-Resultat.HatX(:,i),'m-');
    plot(Tps,ErrEstim(:,i),'m-');
    title(['Erreur d estimation de ',Texte{i}])
    grid on
end;

%composantes diagonales de Pkk
for i=1:2,
    numfig=numfig+1;
    figure(numfig)
    xlabel('t')
    hold on
    plot(Tps,Resultat.DiagPkk(:,i),'g-');
    title(['Composante diagonale de Pkk, numero ',num2str(i)])
    grid on
end;

%composantes de K
for i=1:2,
    numfig=numfig+1;
    figure(numfig)
    xlabel('t')
    hold on
    plot(Tps,Resultat.K(:,i),'k-');
    title(['Composante K, numero ',num2str(i)])
    grid on
end;
%%%%%%%%%%%%%%%

```

On n'oublie pas de finir de compléter le schéma-bloc Simulink afin d'avoir la variation sur la consigne :

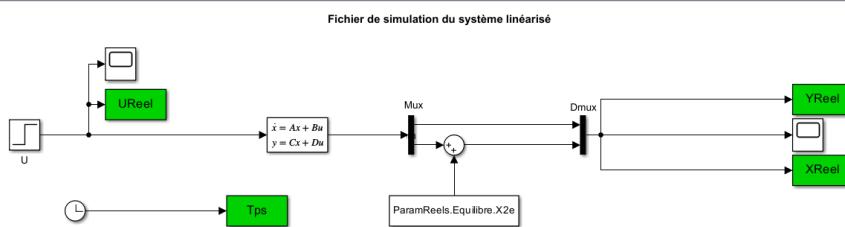
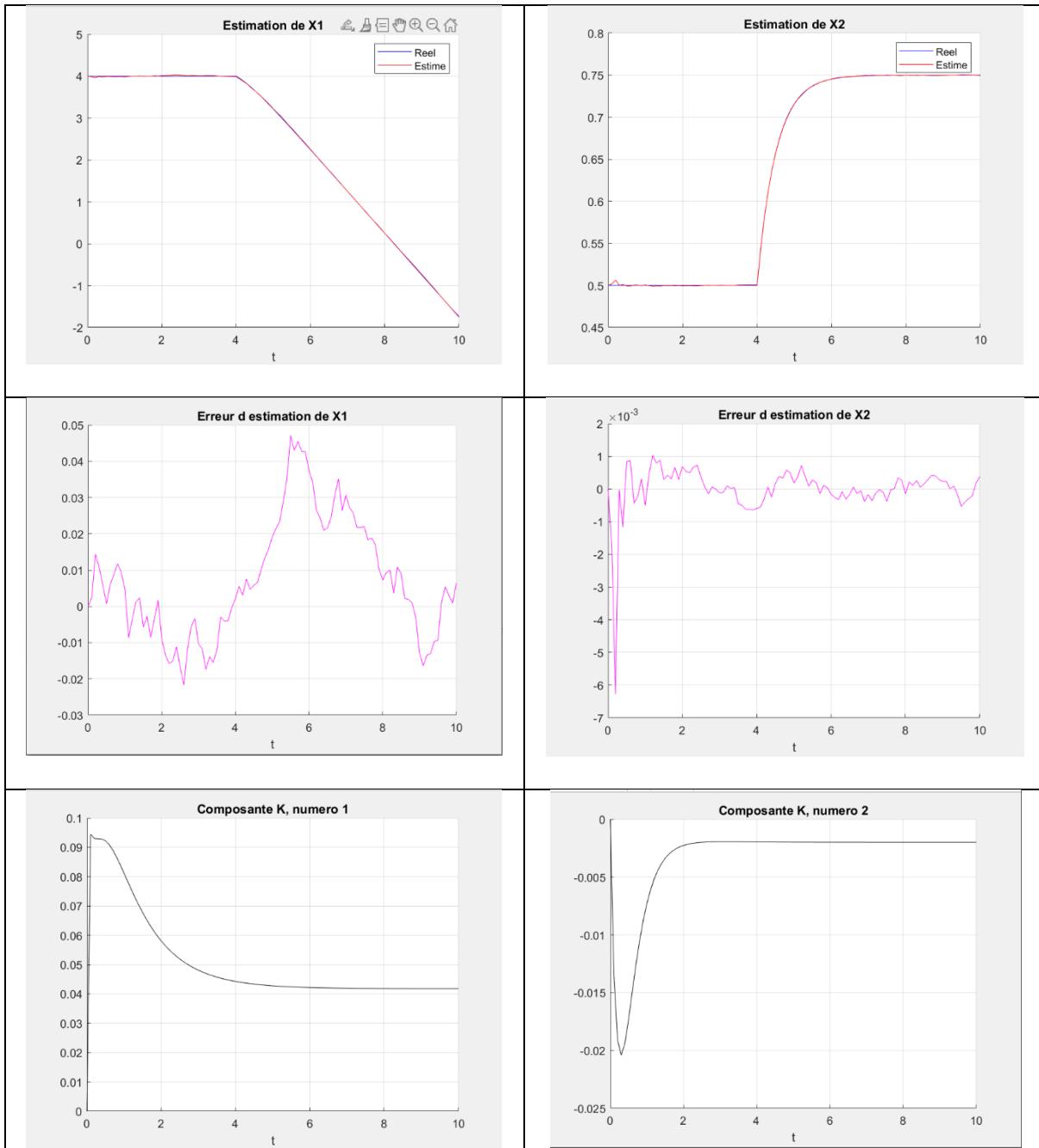


Figure 1: Schéma bloc de la simulation avec le filtre de Kalman linéaire

En lançant le fichier LanceSimFK avec la simulation linéaire du modèle, on a les courbes suivantes



On note que pour les estimations de x_1 et x_2 , l'erreur est faible mais fluctue.

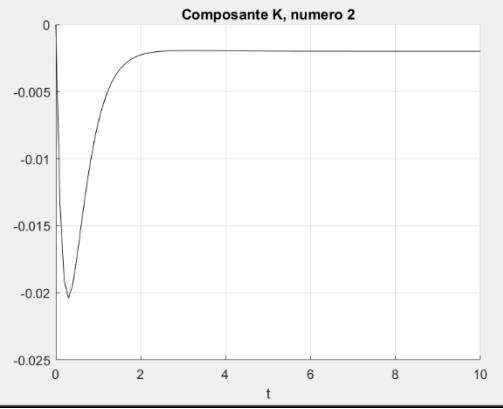
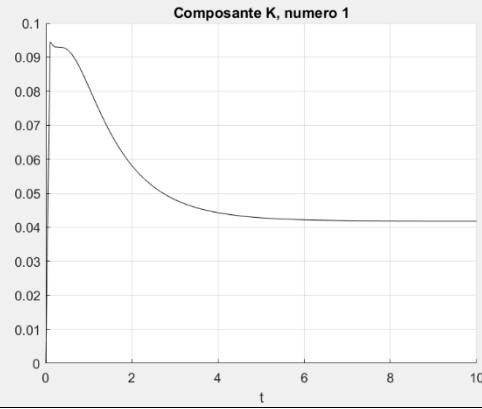
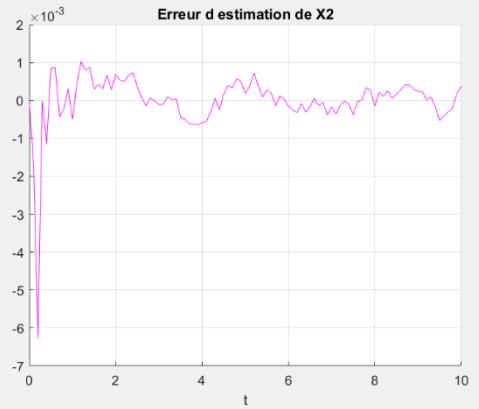
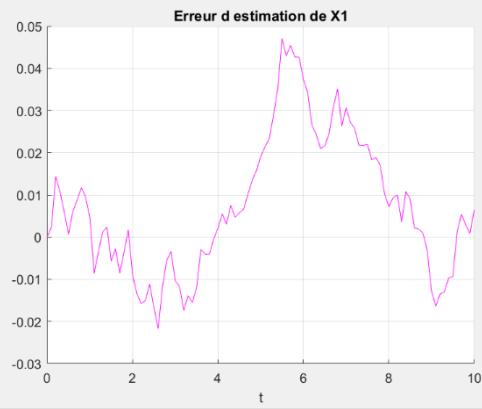
Les composantes du gain de Kalman varient entre 0 et 1, ce qui nous indique que notre filtre à tendance à donner plus d'importance à l'estimée de la mesure $x_{k+1|k}$ qu'à la correction calculée.

Dans un premier temps, l'on va donc changer la valeur de $x_{0/0}$ (plus précisément de x_{e_1} , car comme nous avons pu le voir dans la 1^{ère} partie sur la linéarisation de notre système autour d'un point d'équilibre, il n'y a pas de contraintes sur x_{e_1}), et voir si l'on arrive à réduire ces écarts et les variations de l'erreur d'estimation pour x_1 et x_2 et les faire tendre vers 0.

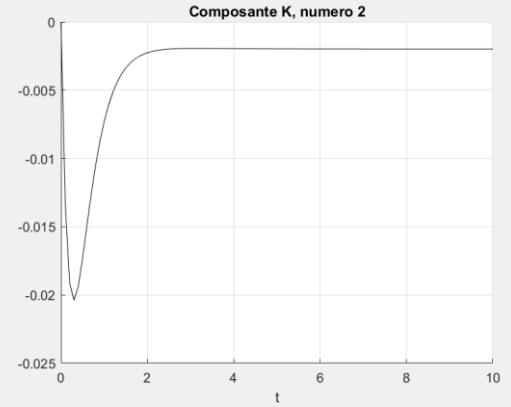
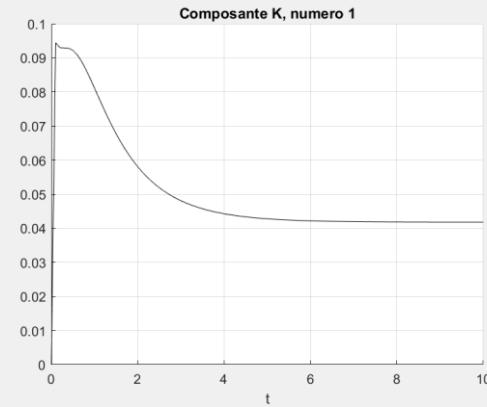
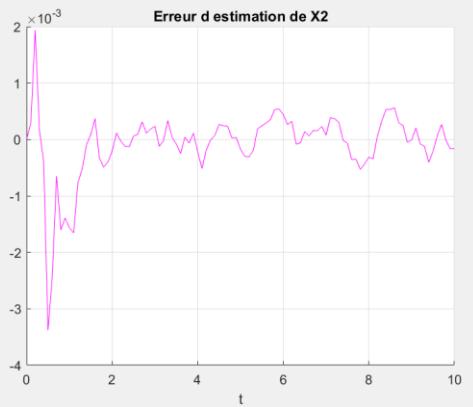
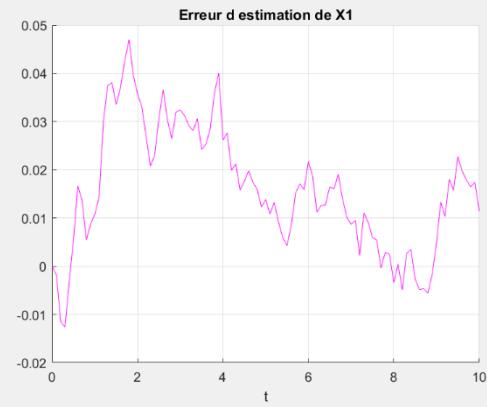
Étude de l'influence des paramètre du filtre sur les erreurs d'estimations

Après plusieurs simulation, l'on a :

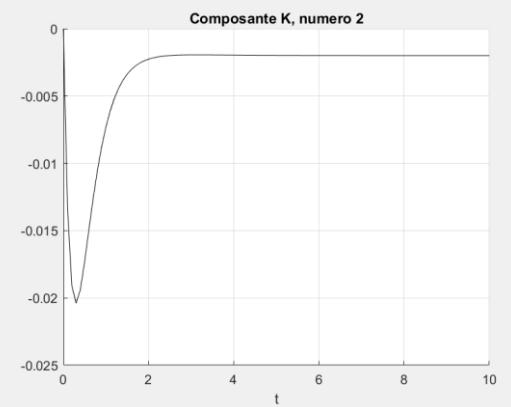
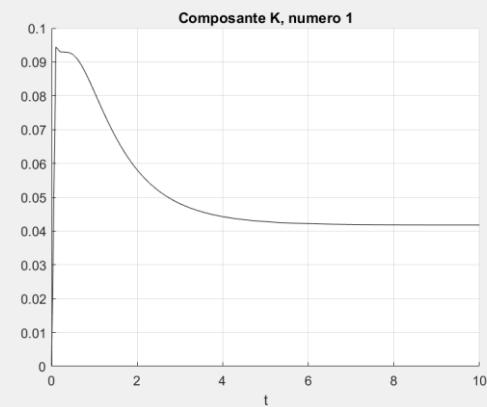
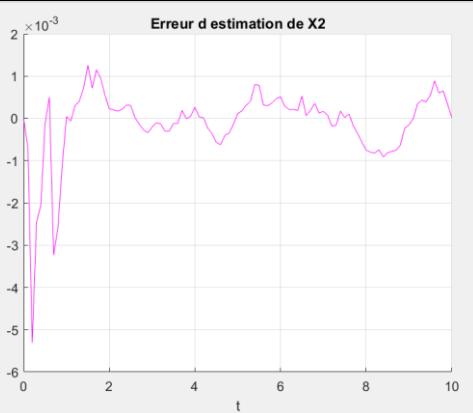
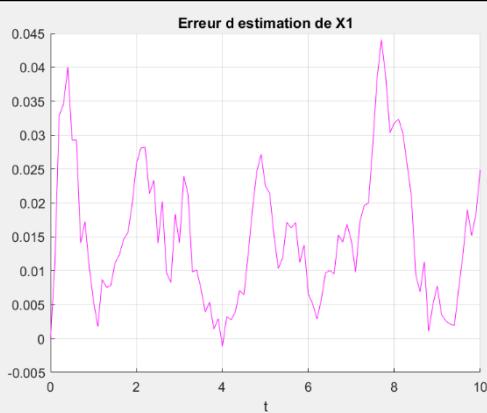
$$x_{0/0} = [10 \ x_{e_2}] \\ Q = diag([10^{-5}, 10^{-5}]) \\ R = 10^{-2} \\ P_{0|0} = diag([10^{-3}, 10^{-3}])$$



$$\begin{aligned}
x_{0/0} &= [100 \ x_{e_2}] \\
Q &= \text{diag}([10^{-5}, 10^{-5}]) \\
R &= 10^{-2} \\
P_{0|0} &= \text{diag}([10^{-3}, 10^{-3}])
\end{aligned}$$



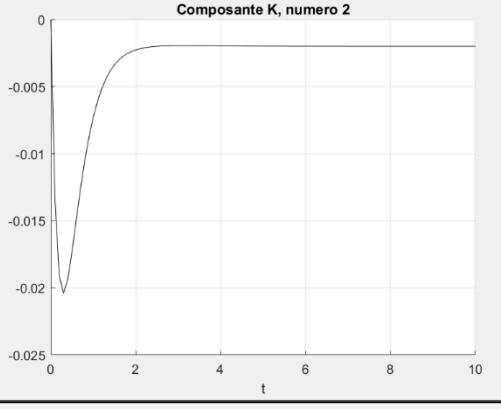
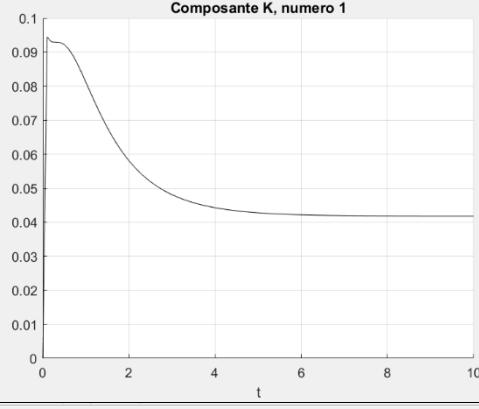
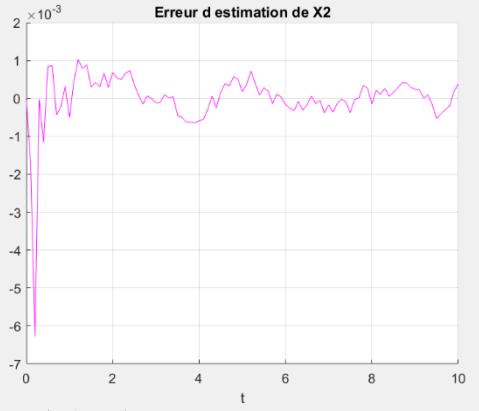
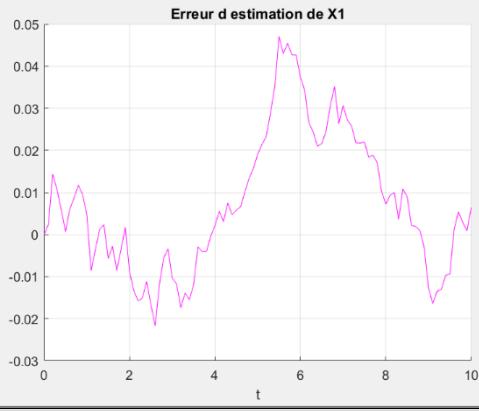
$$\begin{aligned}
x_{0/0} &= [0.1 \ x_{e_2}] \\
Q &= \text{diag}([10^{-5}, 10^{-5}]) \\
R &= 10^{-2} \\
P_{0|0} &= \text{diag}([10^{-3}, 10^{-3}])
\end{aligned}$$



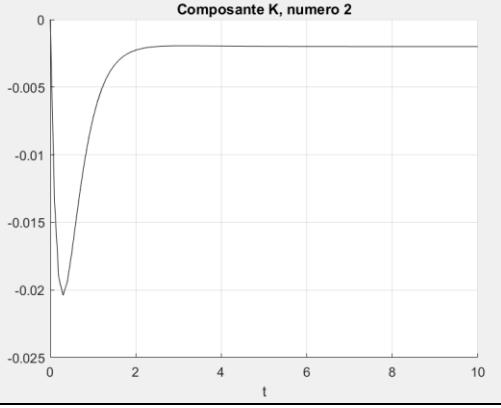
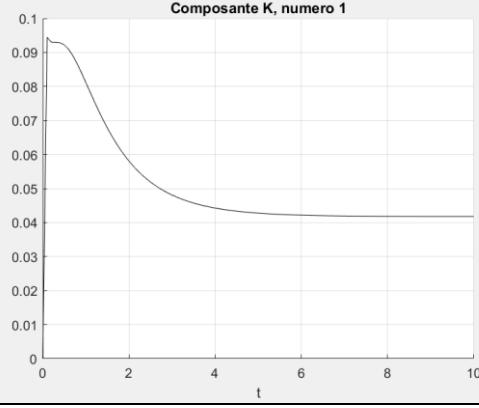
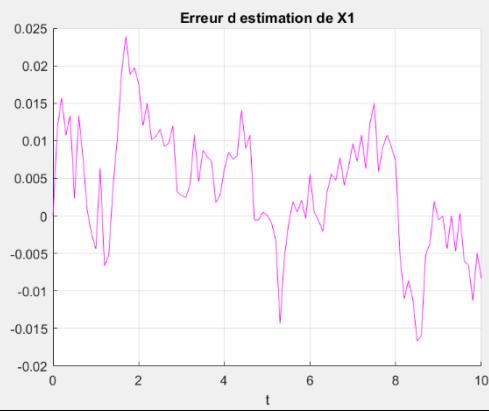
L'on remarque alors que les erreurs d'estimation sur x_1 et x_2 ne varient que très peu en plus d'être faibles. Les composantes du gain de Kalman varient entre 0 et 1, ce qui nous indique que notre filtre à tendance à donner plus d'importance à l'estimée de la mesure $x_{k+1|k}$ qu'à la correction calculée.

L'on commence en faisant varier Q :

$$\begin{aligned}x_{0/0} &= [10 \ x_{e_2}] \\Q &= \text{diag}([10^{-5}, 10^{-5}]) \\R &= 10^{-2} \\P_{0|0} &= \text{diag}([10^{-3}, 10^{-3}])\end{aligned}$$



$$\begin{aligned}x_{0/0} &= [10 \ x_{e_2}] \\Q &= \text{diag}([10^{-10}, 10^{-1}]) \\R &= 10^{-2} \\P_{0|0} &= \text{diag}([10^{-3}, 10^{-3}])\end{aligned}$$

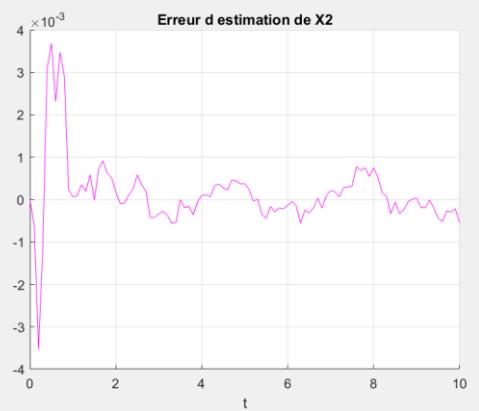
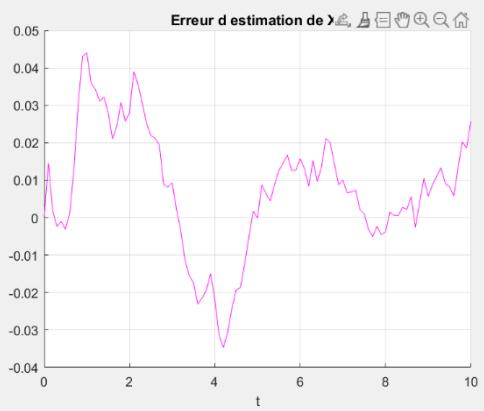


$$x_{0/0} = [10 \ x_{e_2}]$$

$$Q = \text{diag}([10^{-1}, 10^{-10}])$$

$$R = 10^{-2}$$

$$P_{0|0} = \text{diag}([10^{-3}, 10^{-3}])$$

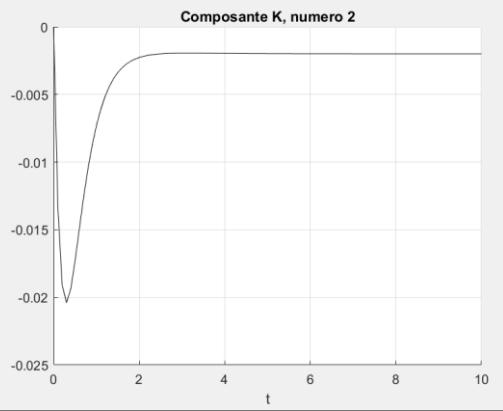
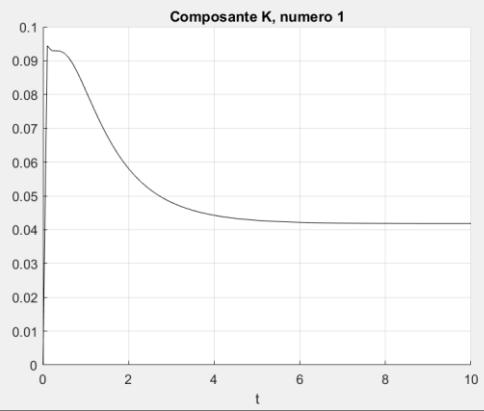
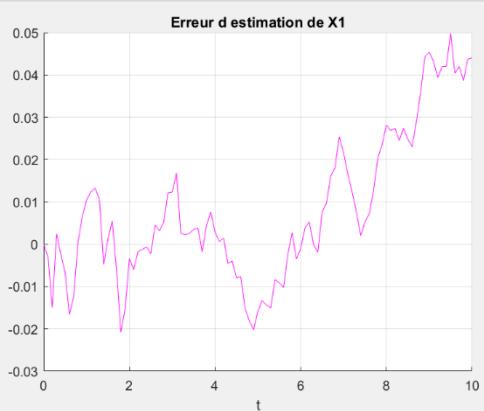


$$x_{0/0} = [10 \ x_{e_2}]$$

$$Q = \text{diag}([10^{-10}, 10^{-10}])$$

$$R = 10^{-2}$$

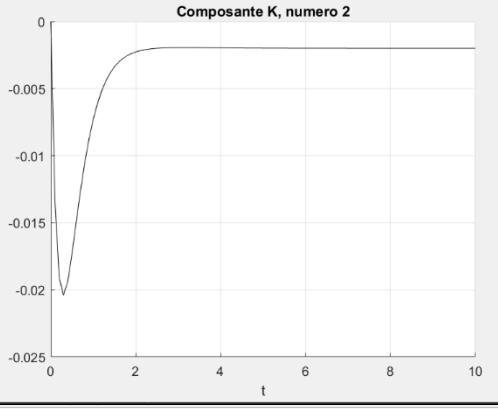
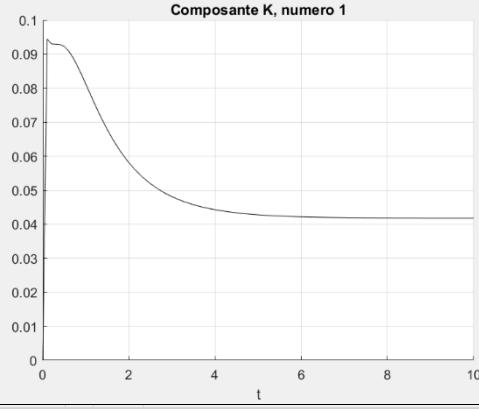
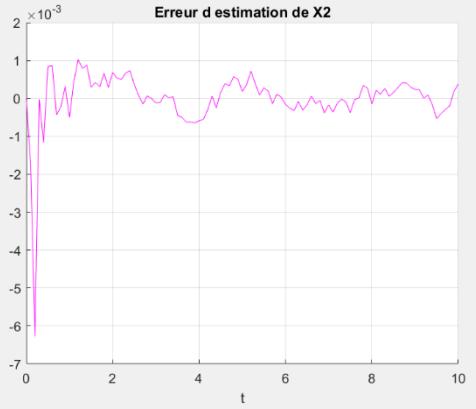
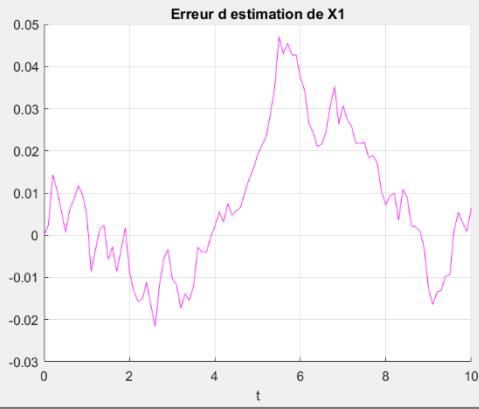
$$P_{0|0} = \text{diag}([10^{-3}, 10^{-3}])$$



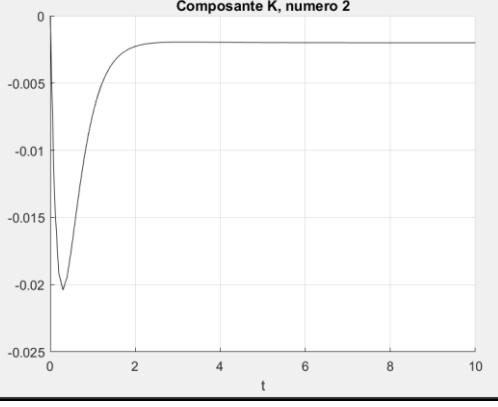
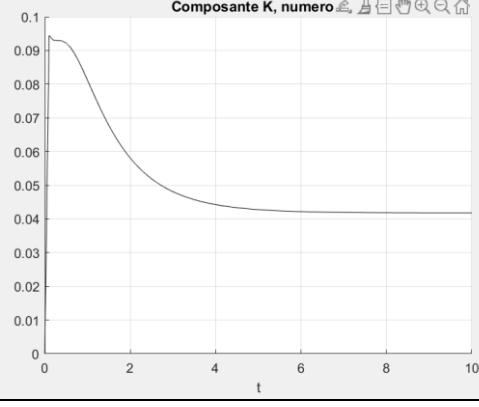
On remarque que faire varier Q influence légèrement sur les variations de l'erreur d'estimation de x_1 et x_2 .

On va continuer en faisant varier R .

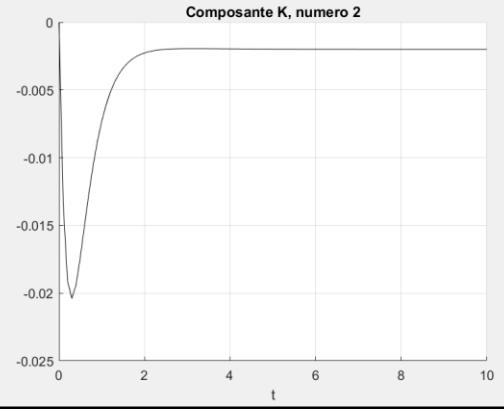
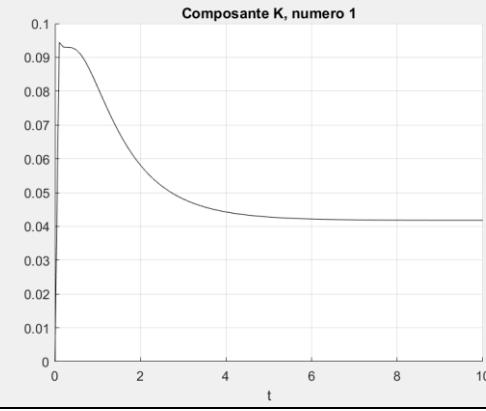
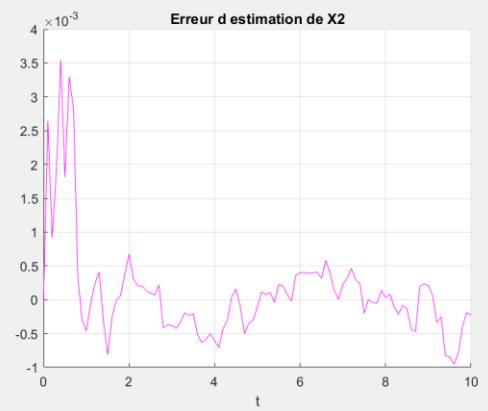
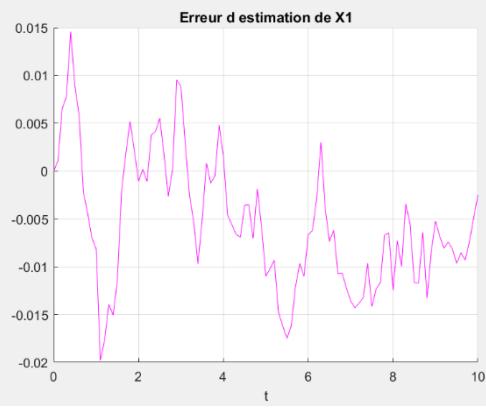
$$\begin{aligned}x_{0/0} &= [10 \ x_{e_2}] \\Q &= \text{diag}([10^{-5}, 10^{-5}]) \\R &= 10^{-2} \\P_{0|0} &= \text{diag}([10^{-3}, 10^{-3}])\end{aligned}$$



$$\begin{aligned}x_{0/0} &= [10 \ x_{e_2}] \\Q &= \text{diag}([10^{-5}, 10^{-5}]) \\R &= 10^{-5} \\P_{0|0} &= \text{diag}([10^{-3}, 10^{-3}])\end{aligned}$$



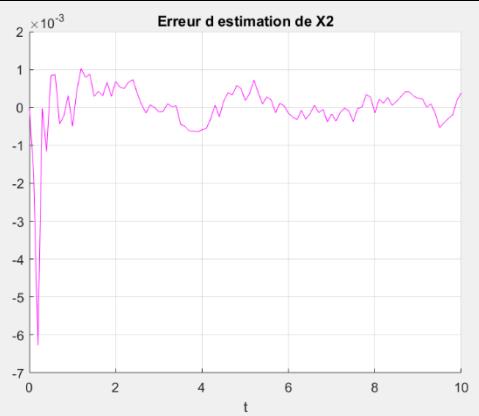
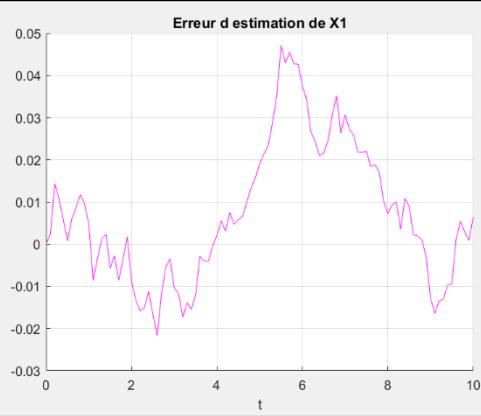
$$\begin{aligned}
x_{0/0} &= [10 \ x_{e_2}] \\
Q &= \text{diag}([10^{-5}, 10^{-5}]) \\
R &= 10^{-10} \\
P_{0|0} &= \text{diag}([10^{-3}, 10^{-3}])
\end{aligned}$$



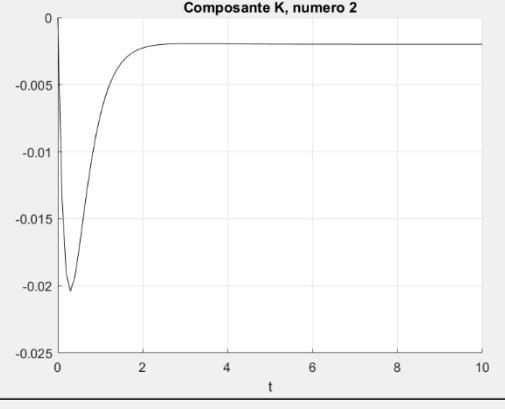
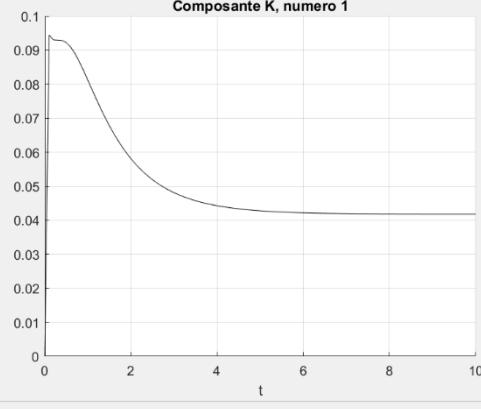
On note qu'en faisant varier R on influence les variations de l'erreur d'estimation sur x_1 et x_2 .

On va maintenant étudier l'influence de $P_{0|0}$.

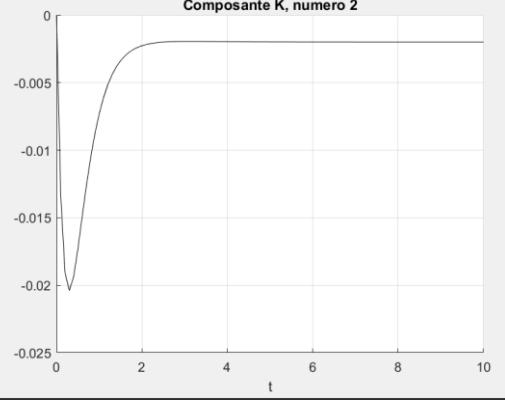
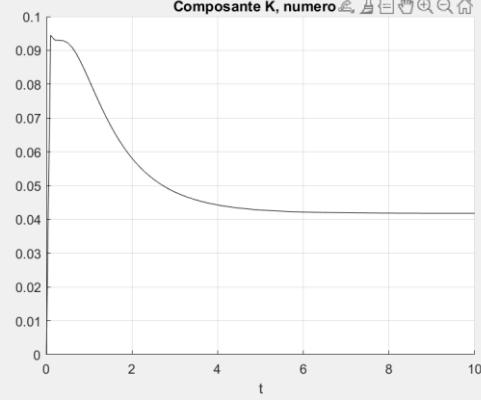
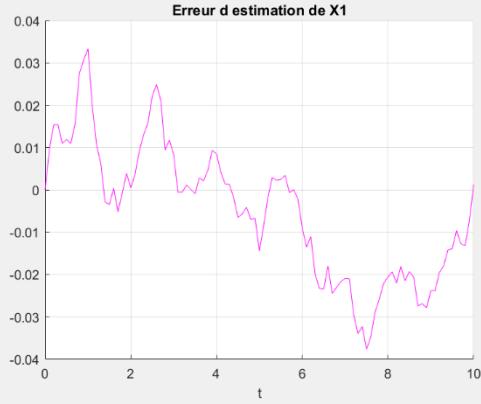
$$x_{0|0} = [10 \ x_{e_2}] \\ Q = \text{diag}([10^{-5}, 10^{-5}]) \\ R = 10^{-2} \\ P_{0|0} = \text{diag}([10^{-3}, 10^{-3}])$$



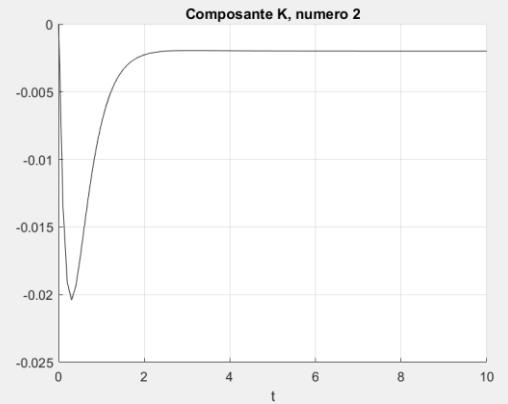
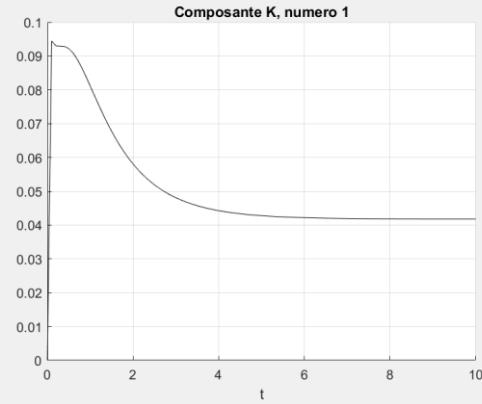
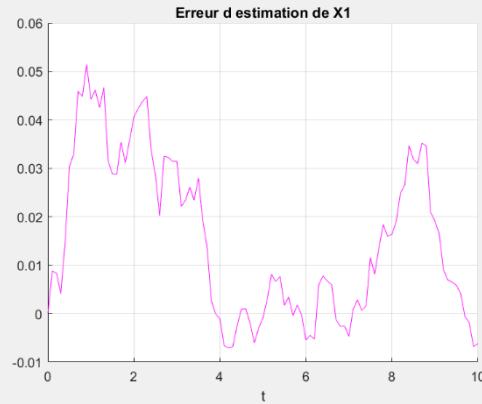
$$x_{0|0} = [10 \ x_{e_2}] \\ Q = \text{diag}([10^{-5}, 10^{-5}]) \\ R = 10^{-2} \\ P_{0|0} = \text{diag}([10^{-5}, 10^{-5}])$$



$$x_{0|0} = [10 \ x_{e_2}] \\ Q = \text{diag}([10^{-5}, 10^{-5}]) \\ R = 10^{-2} \\ P_{0|0} = \text{diag}([10^{-3}, 10^{-3}])$$



$$\begin{aligned}
x_{0|0} &= [10 \ x_{e_2}] \\
Q &= \text{diag}([10^{-5}, 10^{-5}]) \\
R &= 10^{-2} \\
P_{0|0} &= \text{diag}([10^{-10}, 10^{-10}])
\end{aligned}$$



On remarque alors que comme pour les autres paramètres, $P_{0|0}$, permet de changer l'amplitude de variation de l'erreur d'estimation de x_1 et x_2 .

Conclusion sur le filtre de Kalman linéaire sur un modèle linéarisé

Finalement, notre système linéarisé et plus simple et semble fonctionner avec le filtre de Kalman simple et l'on peut influencer l'amplitudes des variations de x_1 et x_2 avec les différents paramètres de l'algorithme ($x_{0|0}$, Q , R et $P_{0|0}$).

Nous allons maintenant utiliser le modèle non linéaire de notre système, et voir comment cela évolue.

Simulation du système non linéaire avec filtre de Kalman linéaire:

Pour passer au système non linéaire, il nous suffit de décommenter la ligne de code qui va utiliser le schéma-bloc non linéaire et de commenter la ligne qui va chercher le schéma-bloc linéaire de notre système :

```
%execution du fichier Simulink Linéaire : SimLin  
% sim('SimLin',ParamReels.tfin); % si pas Simulink alors télécharger les données  
% load DataLin; % Donées système linéarisé
```

```
%Execution du fichier Simulink Non linaire: SimNL  
sim('SimNL',ParamReels.tfin); % a utiliser si vous avez simulink  
%load DataNLin; % Données système non-linéaire
```

Les premières simulations avec les paramètres

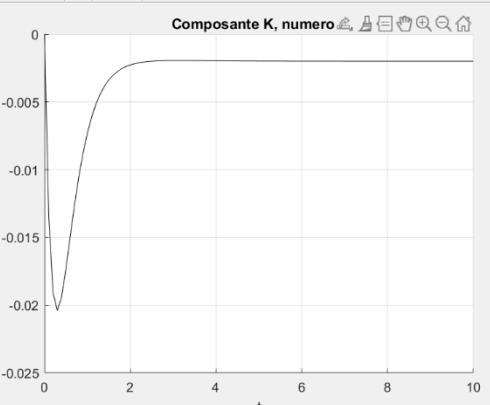
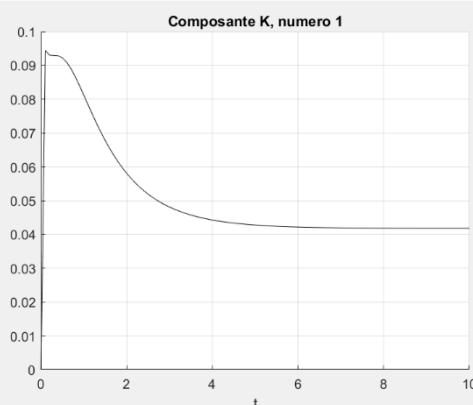
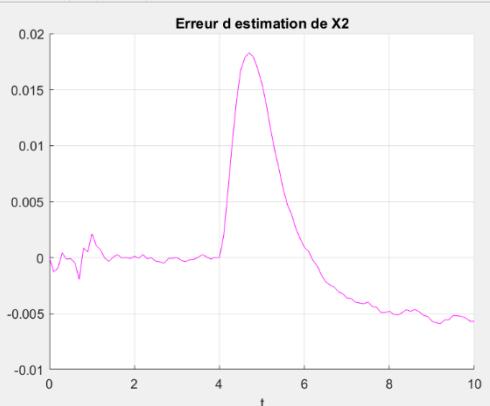
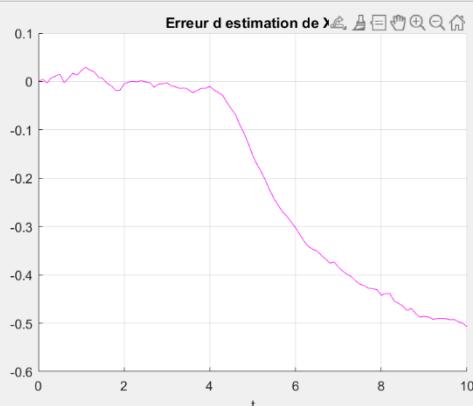
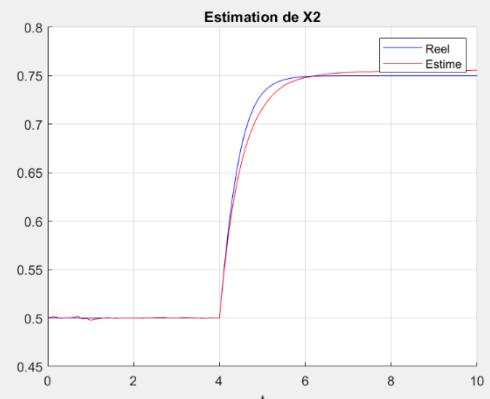
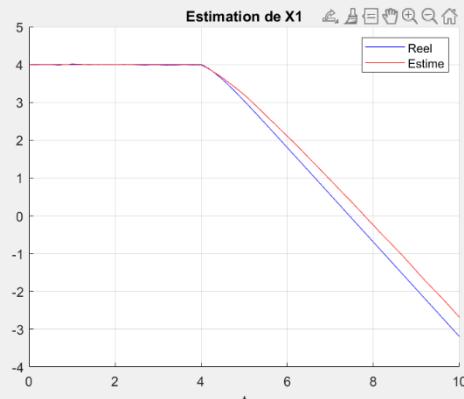
$$x_{0/0} = [10 \ x_{e_2}]$$

$$Q = diag([10^{-5}, 10^{-5}])$$

$$R = 10^{-2}$$

$$P_{0|0} = diag([10^{-3}, 10^{-3}])$$

donnent :



Conclusion sur le filtre de Kalman linéaire sur un modèle non linéaire

Cette fois-ci, on remarque que les erreurs d'estimation sur x_1 et x_2 sont plus grandes que précédemment, cependant les valeurs des composantes du gain de Kalman ne changent pas alors qu'elles devraient. Cela qui signifie que le filtre fait plus confiance à sa prédiction qu'à l'estimé de la mesure $x_{k+1|k}$, ce qui n'est plus le cas. **Nous devons donc passer à un filtre de Kalman plus adapté, le filtre de Kalman étendu.**

Mise en œuvre d'un filtre de Kalman étendu

L'intérêt du filtre de Kalman étendu est de pouvoir traiter des systèmes non linéaires, là où le filtre de Kalman classique n'est plus valable.

Simulation avec le filtre de Kalman étendu :

En implantant l'algorithme du filtre de Kalman étendu dans notre code MatLab, on a :

```
%%%%%%%%%%%%%%%
% Simulations et Filtre de Kalman Etendu
%%%%%%%%%%%%%%%
close all %ferme toutes les figures
clear all %efface le "workspace"
%%%%%%%%%%%%%%%
%PARAMETRES DU SYSTEME REEL
eval('ParametresReels'); %execute le fichier ParametresReels.m
%temps de fin de simulation (en secondes)
ParamReels.tfin=10;
%%%%%%%%%%%%%%%
%PARAMETRES DU FILTRE
ParamFilt=ParamReels; %recopie des paramètres réels
%Estimé initial
ParamFilt.X00=ParamReels.X0;
%Covariance de l'erreur d'estimation initiale
ParamFilt.P00=diag([1e-3,1e-3]);
%Covariance des bruits de dynamique
ParamFilt.Q=diag([1e-5,1e-5]);
%%%%%%%%%%%%%%%
%SIMULATION
%execution du fichier Simulink : SimNL
%sim('SimNL',ParamReels.tfin);
load DataNLin
NbMes=length(YReel); %Nombre de points de mesure
%-----
%Bruitage des mesures
%Génération d'un vecteur, de même taille que YReel,
%composé de de bruits gaussiens d'espérance nulle
% de covariance 1
BruitsCov1=randn(NbMes,1);
%Génération d'un vecteur, de même taille que YReel,
%composé de de bruits gaussiens d'espérance nulle
% de covariance ParamReels.R
Bruits=sqrt(ParamReels.R)*BruitsCov1;
%addition des bruits à YReel -> Y mesuré
Ymes = YReel + Bruits;
%-----
```

```

%FILTRAGE
%
%-----%
% initialisation

%mise en forme
Xkk=ParamFilt.X00;
Pkk=ParamFilt.P00;
U=UReel;

%Stockage
Resultat.HatX(1,:)=Xkk'; %Estimé
Resultat.DiagPkk(1,:)=diag(Pkk'); %Composantes diag de Pkk
%-----%

%
%-----%
%Récurrence

for k=1:NbMes-1,
    %
    %Prédiction et intégration numérique
    Xkplus=IntegEtat(Xkk,U(k,1),ParamFilt,ParamFilt.Te);

    %linéarisation
    ParamFilt.A=[0 -U(k,1);0 -2*ParamFilt.alpha*Xkk(2,1)+U(k,1)];
    ParamFilt.B=Xkk;

    %discrétisation
    ParamFilt.F= expm(ParamFilt.A*ParamFilt.Te);
    Pkplus=ParamFilt.F*Xkk*ParamFilt.F'+ParamFilt.Q;
    K= Pkplus*ParamFilt.C'*inv(ParamFilt.C'*Pkplus*ParamFilt.C+ParamFilt.R);
    Xkk= Xkplus+K*(Ymes(k+1,1)-ParamFilt.C*Xkplus);
    Pkk= (eye(2)-K*ParamFilt.C)*Pkplus;

    %Stockage
    Resultat.HatX(k+1,:)=Xkk';
    Resultat.DiagPkk(k+1,:)=diag(Pkk)';
    Resultat.K(k+1,:)=K';

end;
%-----%
%%%%%%%%%%%%%%%
%TRACES DES RESULTATS
numfig=0;
Texte={'X1','X2'};

% Tracé de l'état réel et de l'estimé sur une même figure
for i=1:2,
    numfig=numfig+1;
    figure(numfig)
    xlabel('t')
    hold on
    a=plot(Tps,XReel(:,i),'b-');
    b=plot(Tps,Resultat.HatX(:,i),'r-');
    legend([a,b],'Reel','Estime')
    title(['Estimation de ',Texte{i}])
    grid on
end;

%erreur d'estimation
for i=1:2,
    numfig=numfig+1;
    figure(numfig)
    xlabel('t')
    hold on
    a=plot(Tps,XReel(:,i)-Resultat.HatX(:,i),'m-');
    title(['Erreur d estimation de ',Texte{i}])
    grid on
end;

%composantes diagonales de Pkk
for i=1:2,
    numfig=numfig+1;

```

```

figure(numfig)
xlabel('t')
hold on
a=plot(Tps,Resultat.DiagPkk(:,i),'g-');
title(['Composante diagonale de Pkk, numero ',num2str(i)])
grid on
end;

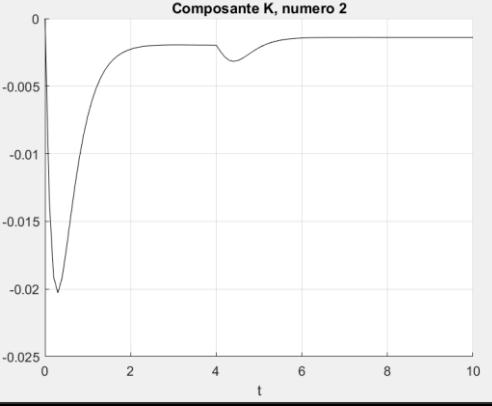
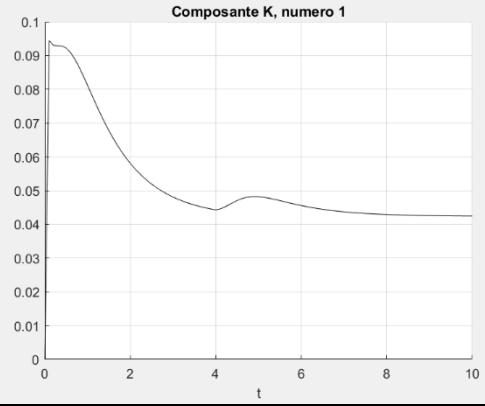
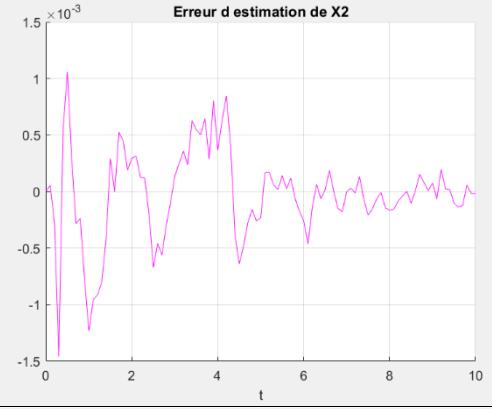
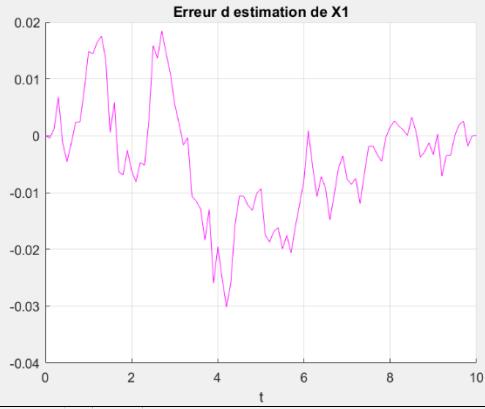
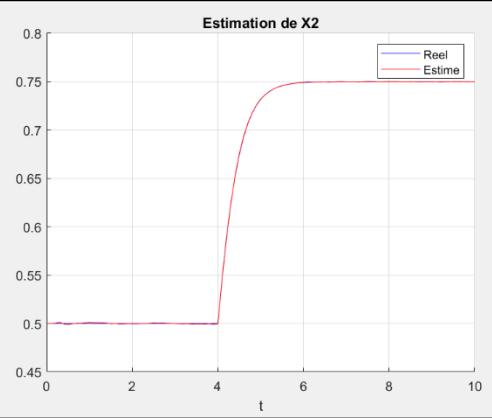
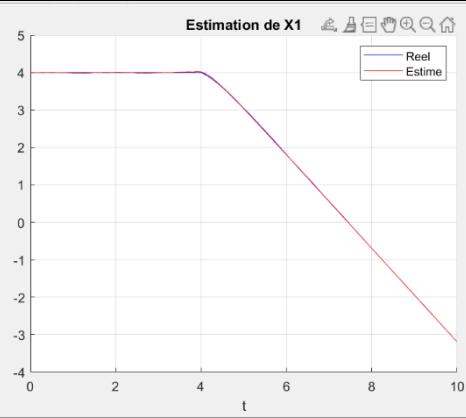
%composantes de K
for i=1:2,
    numfig=numfig+1;
    figure(numfig)
    xlabel('t')
    hold on
    a=plot(Tps,Resultat.K(:,i),'k-');
    title(['Composante K, numero ',num2str(i)])
    grid on
end;

%%%%%%%%%%%%%%%

```

En faisant une première simulation, on a :

$$\begin{aligned}x_{0/0} &= [10 \ x_{e_2}] \\Q &= \text{diag}([10^{-5}, 10^{-5}]) \\R &= 10^{-2} \\P_{0|0} &= \text{diag}([10^{-3}, 10^{-3}])\end{aligned}$$



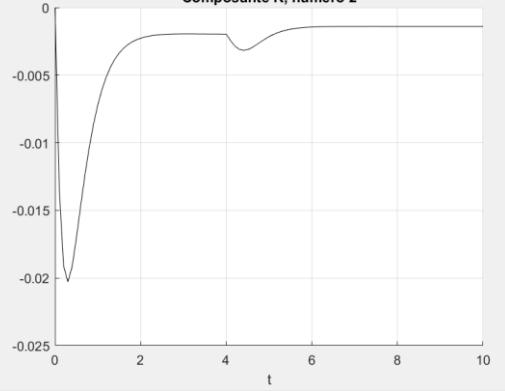
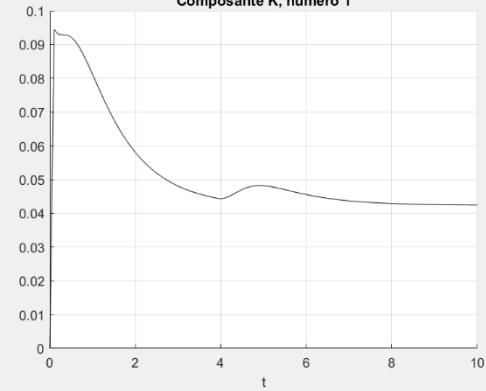
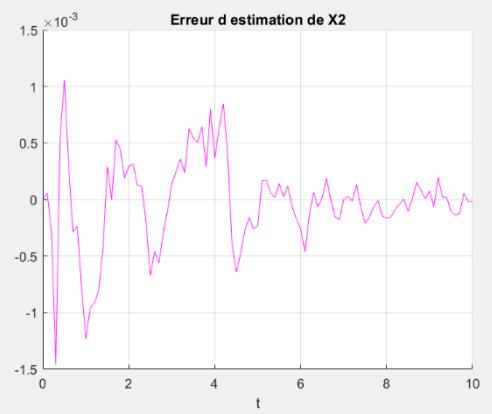
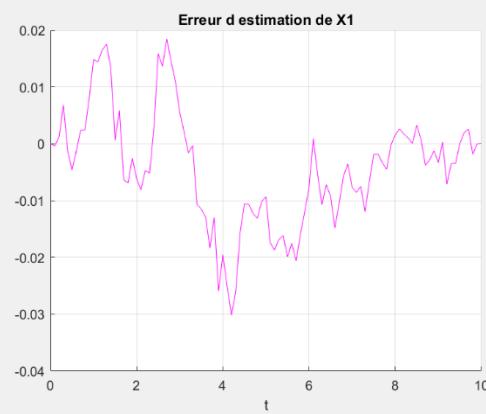
Cette fois-ci, on remarque que les erreurs d'estimation sur x_1 et x_2 sont bien plus faibles que lors de notre simulation sur le modèle non linéaire avec le filtre de Kalman linéaire, de plus ces variations semblent varier autour de 0.

On va, de manière similaire à la partie sur les simulations avec le modèle linéaire et le filtre de Kalman linéaire étudier les variations des différents paramètres.

Étude de l'influence des paramètres du filtre sur les erreurs d'estimations

On commence par faire varier $x_{0|0}$:

$$\begin{aligned}x_{0|0} &= [10 \ x_{e_2}] \\Q &= \text{diag}([10^{-5}, 10^{-5}]) \\R &= 10^{-2} \\P_{0|0} &= \text{diag}([10^{-3}, 10^{-3}])\end{aligned}$$

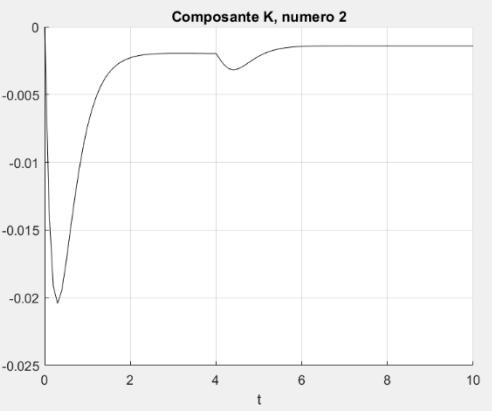
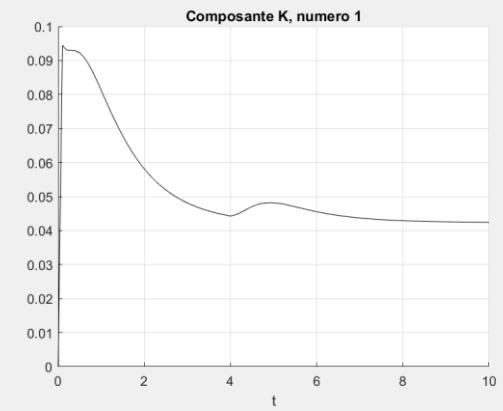
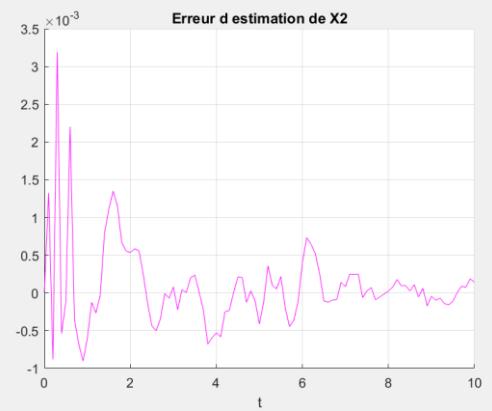
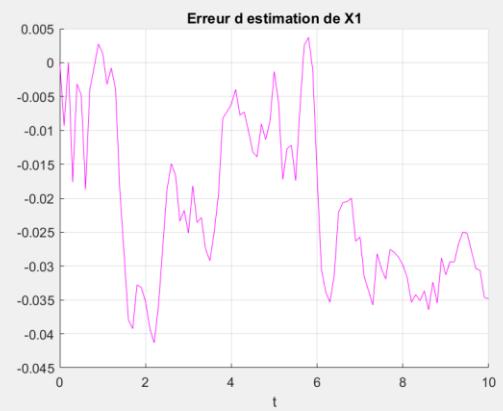


$$x_{0/0} = [100 \ x_{e_2}]$$

$$Q = \text{diag}([10^{-5}, 10^{-5}])$$

$$R = 10^{-2}$$

$$P_{0|0} = \text{diag}([10^{-3}, 10^{-3}])$$

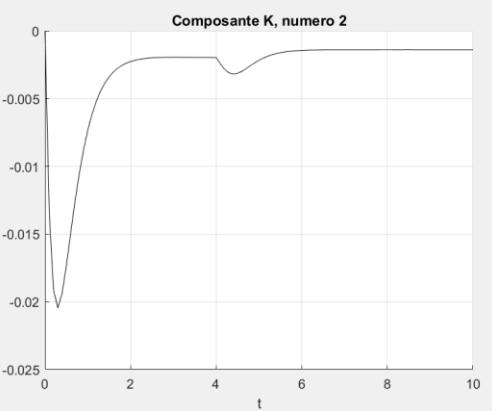
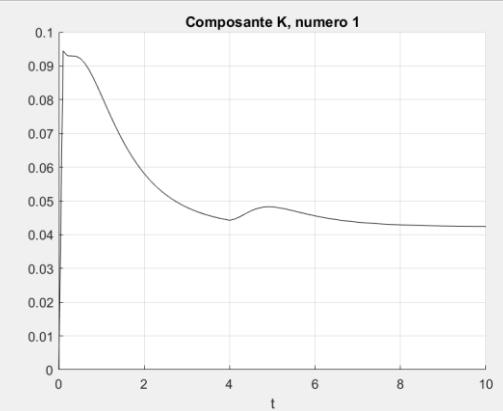
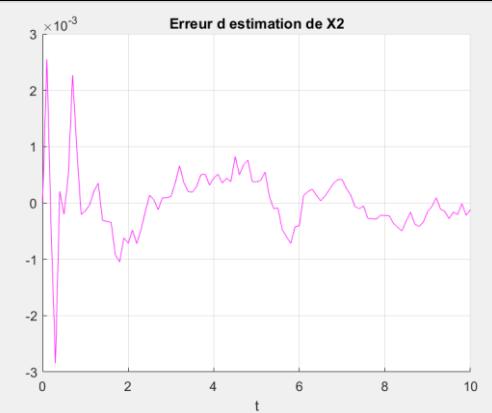


$$x_{0/0} = [0.1 \ x_{e_2}]$$

$$Q = \text{diag}([10^{-5}, 10^{-5}])$$

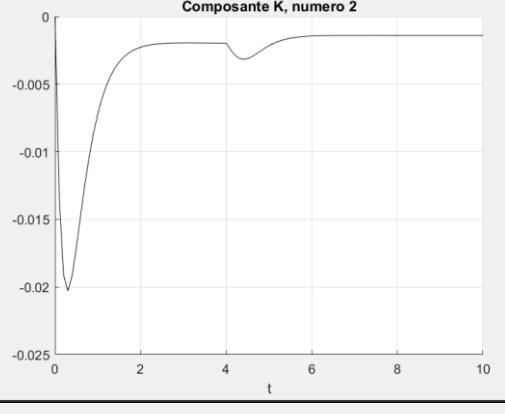
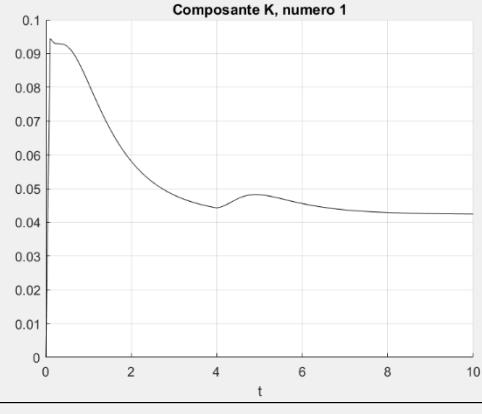
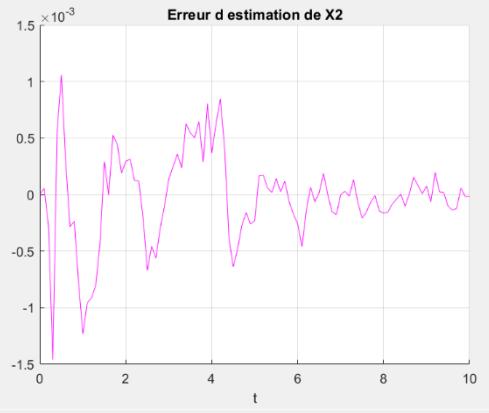
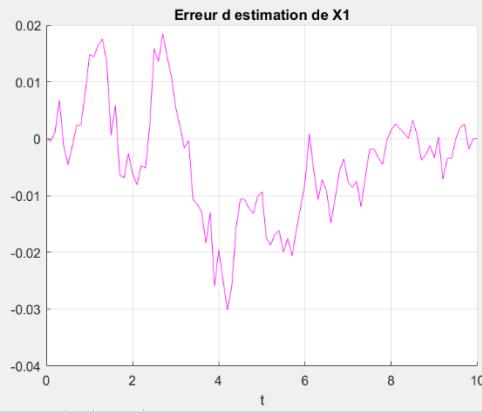
$$R = 10^{-2}$$

$$P_{0|0} = \text{diag}([10^{-3}, 10^{-3}])$$

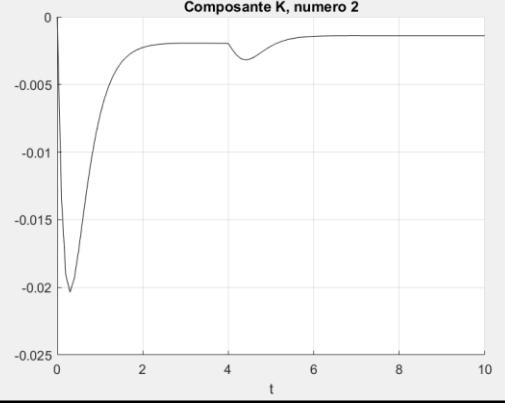
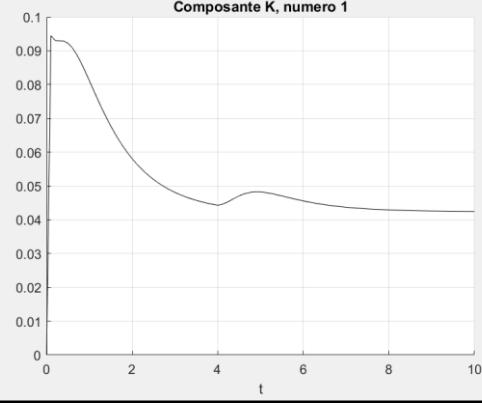
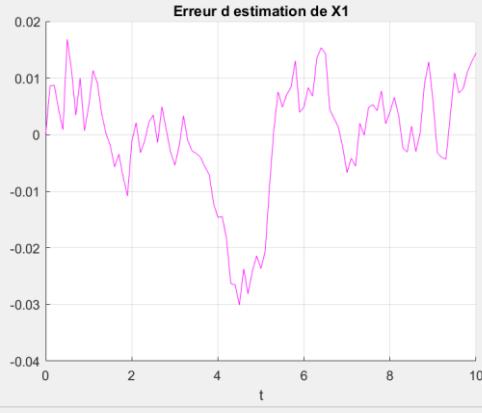


Puis l'on va faire varier Q :

$$\begin{aligned}x_{0/0} &= [10 \ x_{e_2}] \\Q &= \text{diag}([10^{-5}, 10^{-5}]) \\R &= 10^{-2} \\P_{0|0} &= \text{diag}([10^{-3}, 10^{-3}])\end{aligned}$$



$$\begin{aligned}x_{0/0} &= [10 \ x_{e_2}] \\Q &= \text{diag}([10^{-10}, 10^{-1}]) \\R &= 10^{-2} \\P_{0|0} &= \text{diag}([10^{-3}, 10^{-3}])\end{aligned}$$

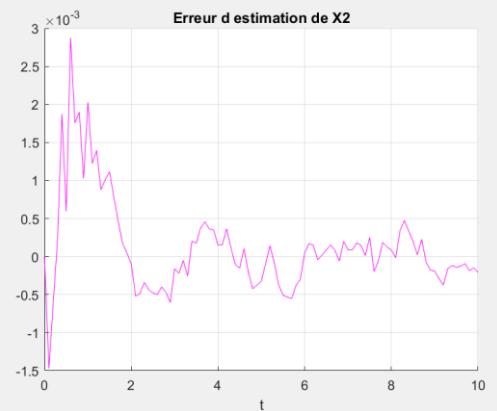
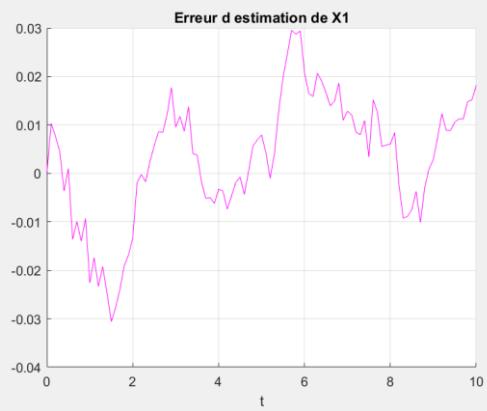


$$x_{0/0} = [10 \ x_{e_2}]$$

$$Q = \text{diag}([10^{-1}, 10^{-10}])$$

$$R = 10^{-2}$$

$$P_{0|0} = \text{diag}([10^{-3}, 10^{-3}])$$

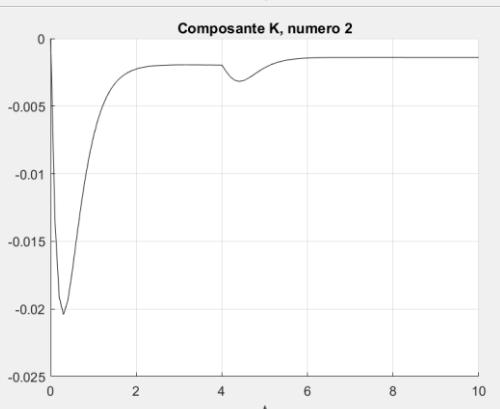
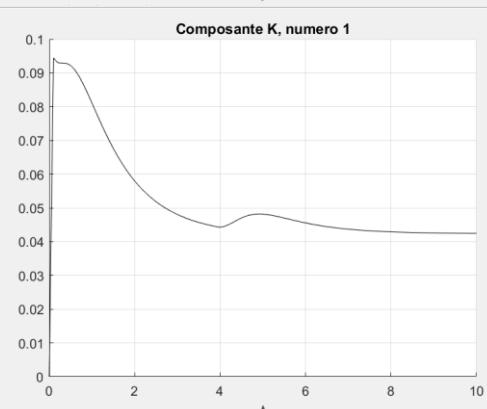
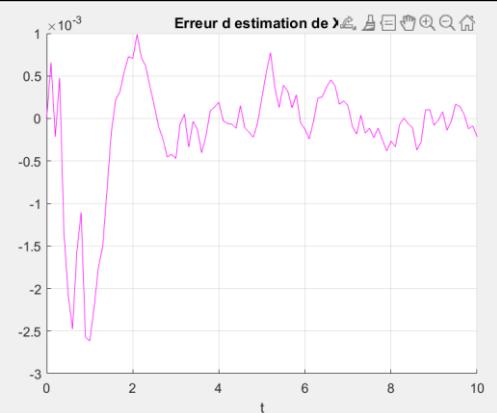
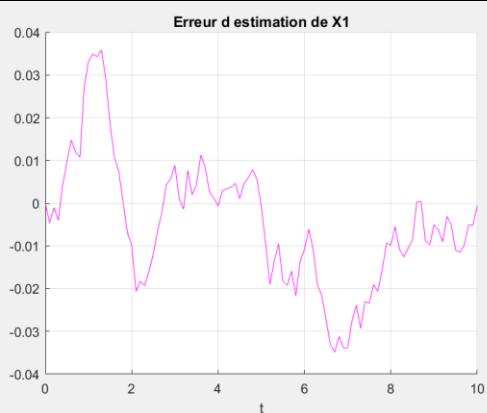
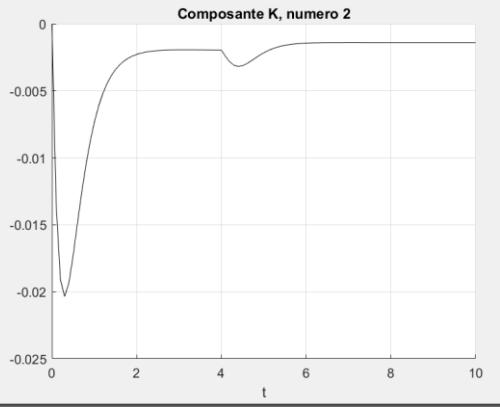
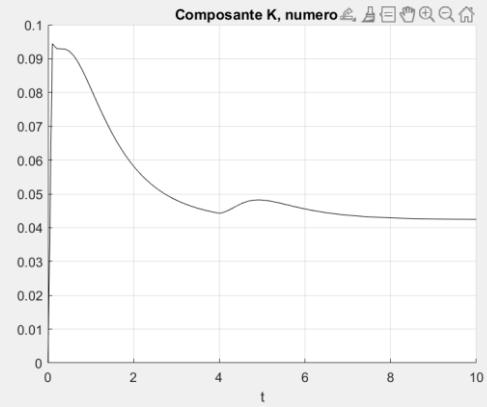


$$x_{0/0} = [10 \ x_{e_2}]$$

$$Q = \text{diag}([10^{-10}, 10^{-10}])$$

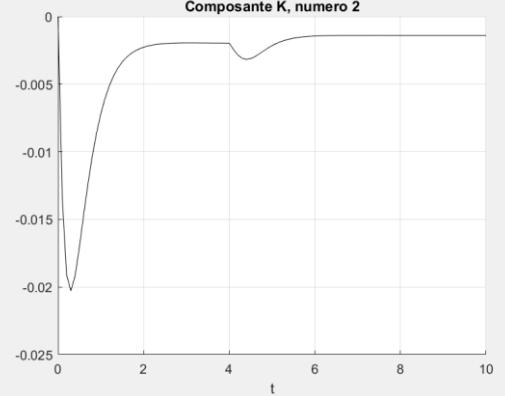
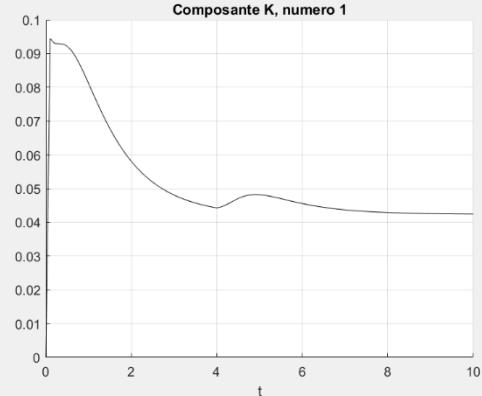
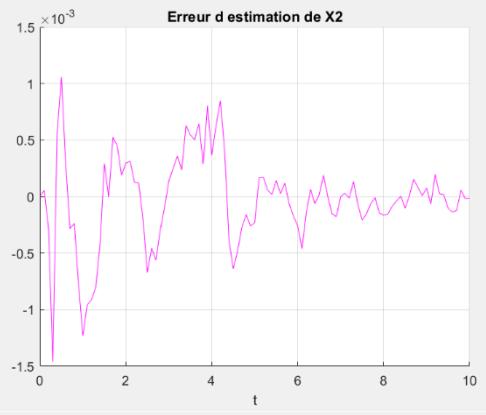
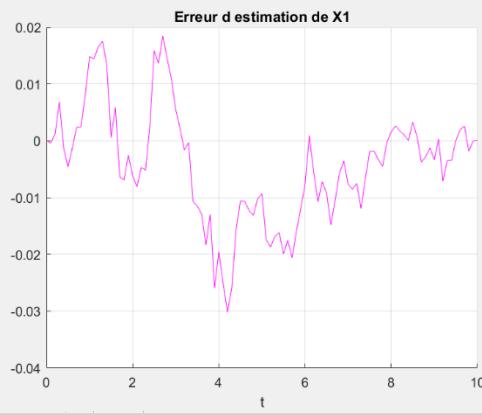
$$R = 10^{-2}$$

$$P_{0|0} = \text{diag}([10^{-3}, 10^{-3}])$$

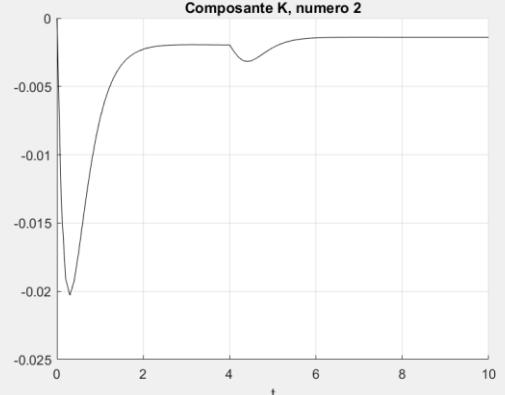
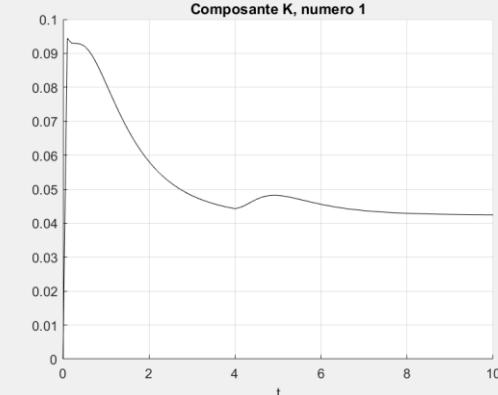
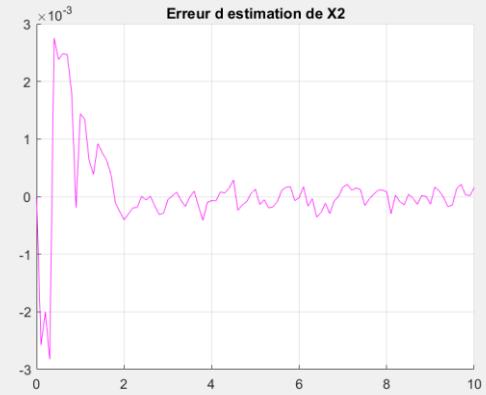
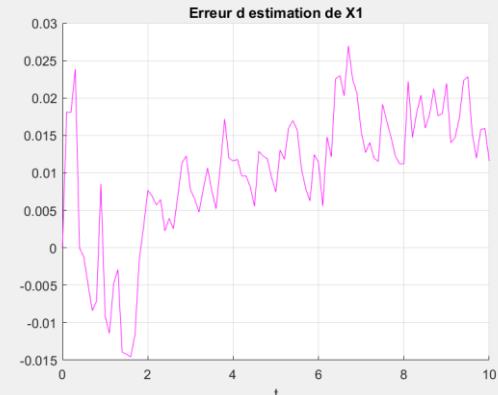


On va maintenant faire varier R :

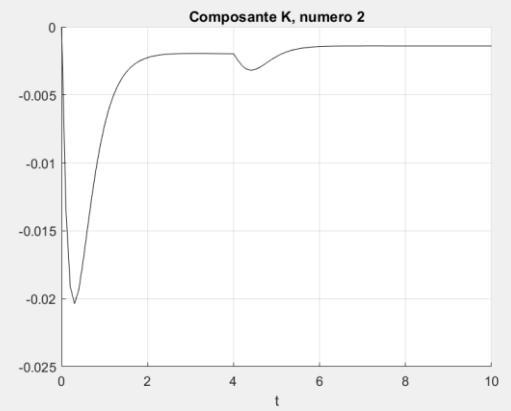
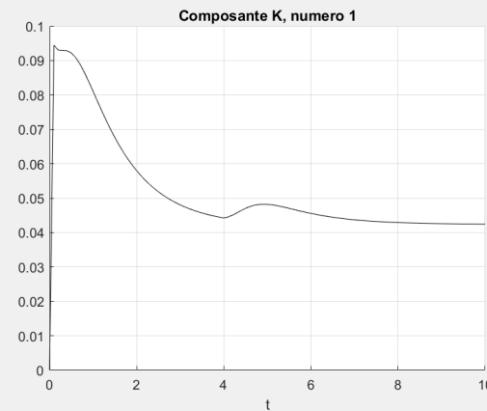
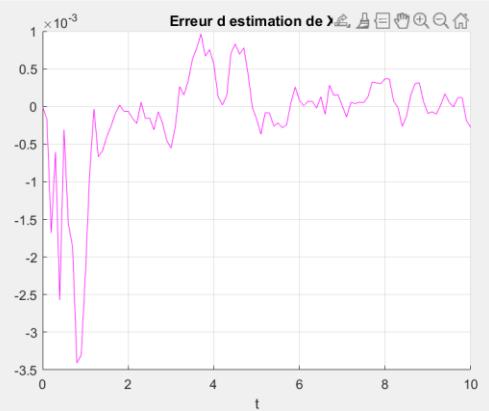
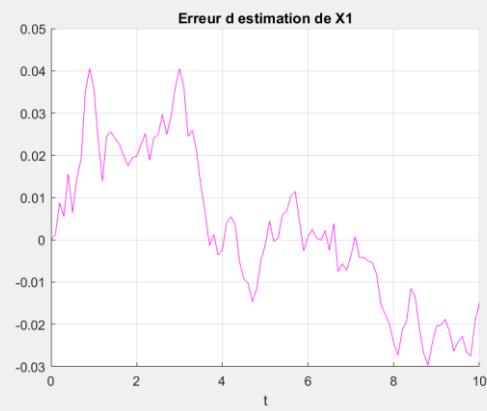
$$\begin{aligned}x_{0/0} &= [10 \ x_{e_2}] \\Q &= \text{diag}([10^{-5}, 10^{-5}]) \\R &= 10^{-2} \\P_{0|0} &= \text{diag}([10^{-3}, 10^{-3}])\end{aligned}$$



$$\begin{aligned}x_{0/0} &= [10 \ x_{e_2}] \\Q &= \text{diag}([10^{-5}, 10^{-5}]) \\R &= 10^{-5} \\P_{0|0} &= \text{diag}([10^{-3}, 10^{-3}])\end{aligned}$$

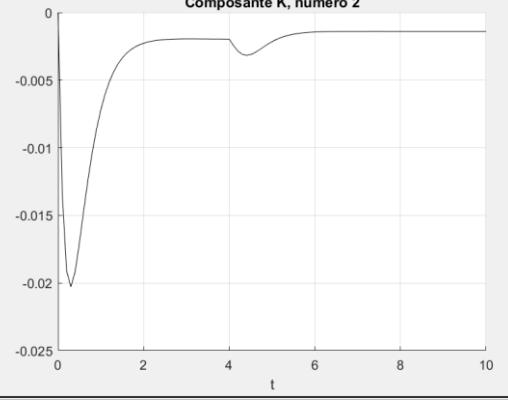
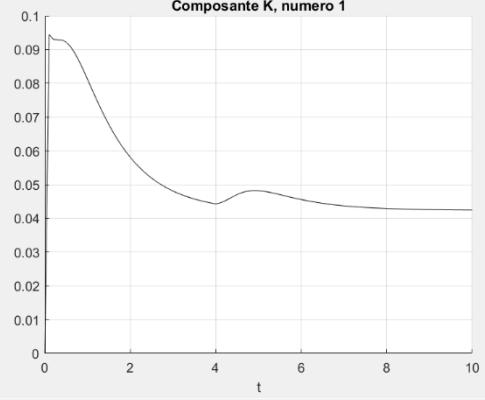
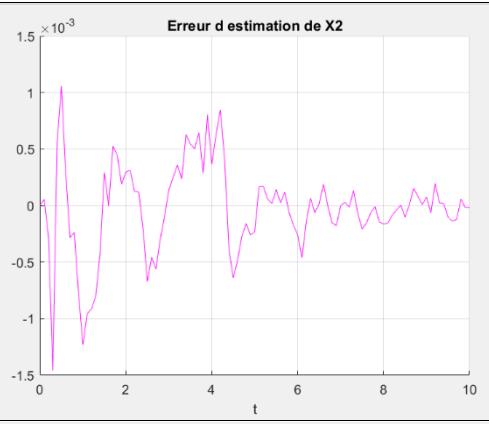
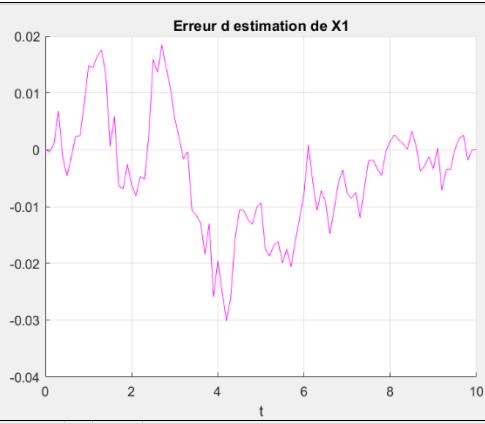


$$\begin{aligned}
x_{0/0} &= [10 \ x_{e_2}] \\
Q &= \text{diag}([10^{-5}, 10^{-5}]) \\
R &= 10^{-10} \\
P_{0|0} &= \text{diag}([10^{-3}, 10^{-3}])
\end{aligned}$$

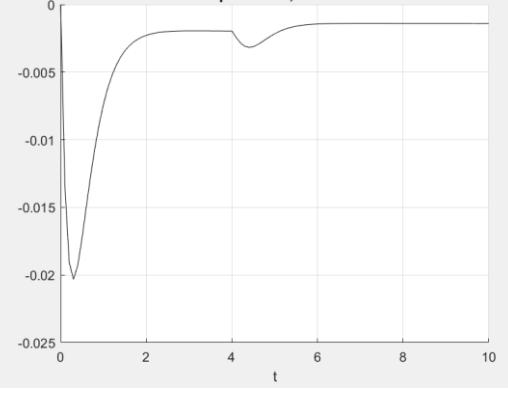
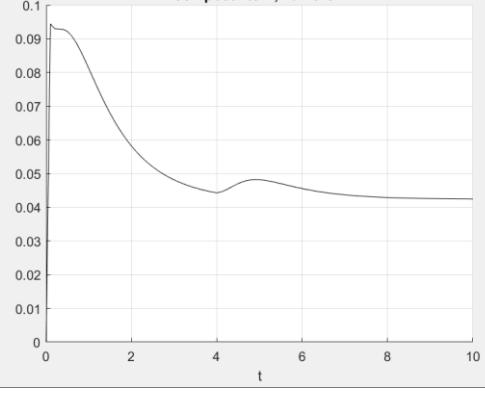
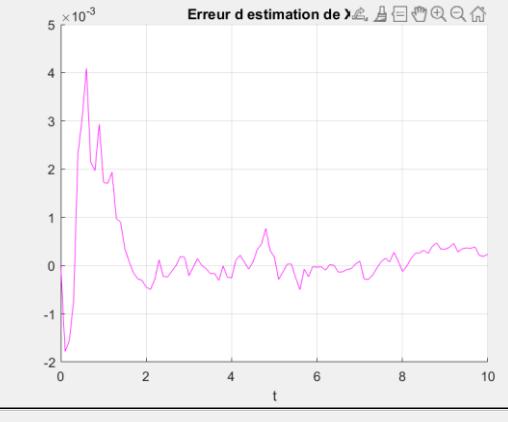
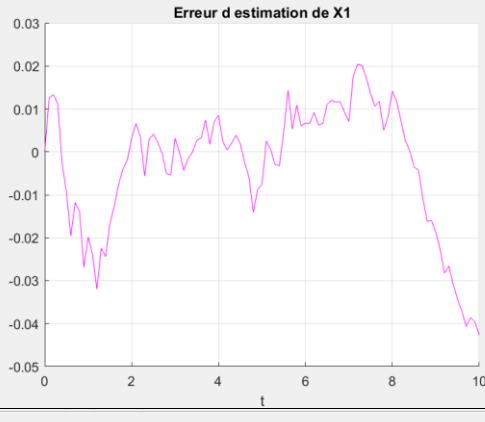


. On va maintenant étudier l'influence de $P_{0|0}$.

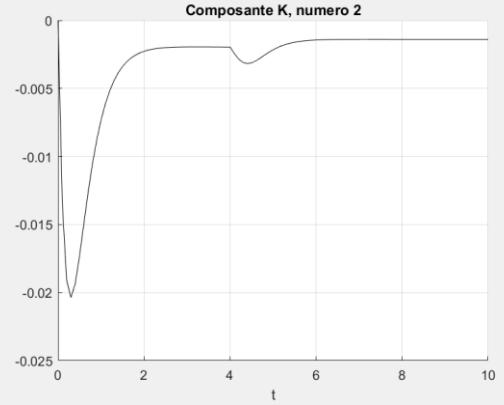
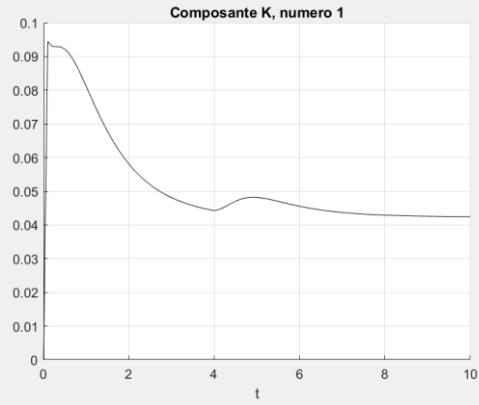
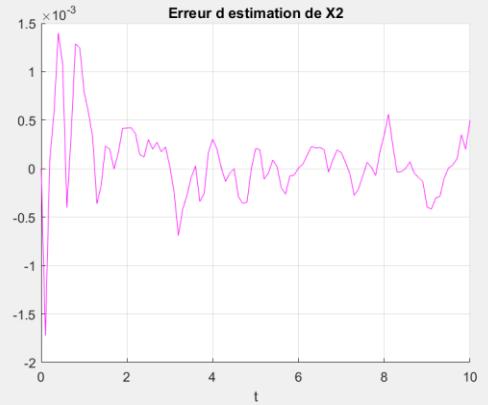
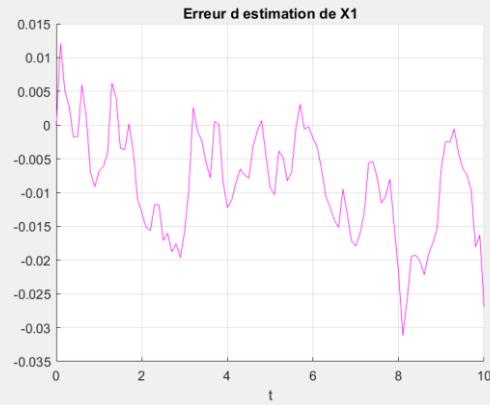
$$x_{0|0} = [10 \ x_{e_2}] \\ Q = \text{diag}([10^{-5}, 10^{-5}]) \\ R = 10^{-2} \\ P_{0|0} = \text{diag}([10^{-3}, 10^{-3}])$$



$$x_{0|0} = [10 \ x_{e_2}] \\ Q = \text{diag}([10^{-5}, 10^{-5}]) \\ R = 10^{-2} \\ P_{0|0} = \text{diag}([10^{-5}, 10^{-5}])$$



$$\begin{aligned}
x_{0|0} &= [10 \ x_{e_2}] \\
Q &= \text{diag}([10^{-5}, 10^{-5}]) \\
R &= 10^{-2} \\
P_{0|0} &= \text{diag}([10^{-10}, 10^{-10}])
\end{aligned}$$



Conclusion sur le filtre de Kalman étendu

Après avoir effectué toutes ces simulations en faisant varier les différents paramètres du filtre de Kalman étendu, nous pouvons en conclure que les paramètres de ce filtre ont le même rôle que ceux pour le filtre de Kalman linéaire.

La différence étant le type de système que l'on utilise, si le modèle est linéaire il est conseillé d'utiliser le filtre de Kalman linéaire qui est plus simple à mettre en place, à corriger et qui donne des résultats proches des résultats réelles.

Au contraire, si le modèle du système considéré n'est pas linéaire (ou si on ne peut le rendre linéaire autour d'un point d'équilibre), il est préférable d'utiliser le filtre de Kalman étendu qui est plus difficile à mettre en place et à corriger mais, qui donne des résultats plus proches des résultats réelles.

Analysons maintenant ces performances.

Évaluation des performances du filtre de Kalman étendu avec des simulations de Monte-Carlo

Simulations de Monte-Carlo pour le filtre de Kalman étendu

Afin d'évaluer les performances de notre filtre de Kalman étendu, nous allons utiliser les simulations de Monte-Carlo. Ces simulations vont nous permettre :

- **De tester notre filtre face à des incertitudes, évaluer la robustesse du filtre :** Chaque simulation va introduire une entrée initiale différentes et des bruits de mesure différents.
 - **D'estimer des données statistiques sur notre filtre :** Les moyennes des erreurs d'estimation, les données à ces extrema, sa covariance.
 - **De valider les modèles sur les bruits Q et R**

Le code des simulations de Monte-Carlo pour le filtre de Kalman étendu est le suivant :

```

%%%%%Simulations et de MonteCarlo pour Filtre de Kalman Etendu
%%%%%close all %ferme toutes les figures
clear all %efface le "workspace"
clc;
%%%%%%%%%%%%%%%
%PARAMETRES DU SYSTEME REEL

eval('ParametresReels'); %execute le fichier ParametresReels.m

%temps de fin de simulation (en secondes)
ParamReels.tfin=10;
%-----
%Etat initial Reel
ParamReelsX0Vrai=[4;ParamReels.Equilibre.X2e];
ParamReelsP00=diag([1e-3,1e-3]);
%%%%%%%%%%%%%%%
%PARAMETRES DU FILTRE

ParamFilt=ParamReels; %recopie des paramètres réels

%Estimé initial
ParamFilt.X00=ParamReels.X0;

%Covariance de l'erreur d'estimation initiale
ParamFilt.P00=diag([1e-3,1e-3]);

%Covariance des bruits de dynamique
ParamFilt.Q=diag([1e-5,1e-5]);

%Matrice de sortie
ParamFilt.C=[1,0];

%Nombre de simulations de MonteCarlo
ParamFiltNs=30;

%%%%%%%%%%%%%%%
%Execution du fichier Simulink Non linaire: SimNL
%sim('SimNL',ParamReels.tfin); % a utiliser si vous avez simulink
load DataNLin; % Données système non-linéaire

NbMes=length(YReel); %Nombre de points de mesure
%-----
%Bruitage des mesures

%Génération d'un vecteur, de même taille que YReel,
%composé de bruits gaussiens d'espérance nulle

```



```

%.....  

%PREDICTION  

Xkplus=IntegEtat(Xkk,u,ParamFilt,ParamFilt.Te);  

[A,B]=Linearise(Xkk(2,1),u,ParamFilt); %Linéarisation  

ParamFilt.F=expm(ParamFilt.Te*A); %discrétisation  

Pkplus=ParamFilt.F*Pkplus*ParamFilt.F'+ParamFilt.Q;  

%.....  

%.....  

%Correction  

y=Y(k+1,1);  

K=Pkplus*ParamFilt.C'*inv(ParamFilt.C*Pkplus*ParamFilt.C'+ParamFilt.R);  

Xkk=Xkplus+K*(y-ParamFilt.C*Xkplus);  

Pk=Pkplus-K*ParamFilt.C*Pkplus;  

%.....  

%.....  

%Stockage  

Resultat.HatX(k+1,:)=Xkk'; %Estimé  

Resultat.DiagPk(k+1,:)=diag(Pk);  

Resultat.K(k+1,:)=K';  

%.....  

end;  

%-----  

%Calcul de l'erreur d'estimation  

Resultat.Err=XReel-Resultat.HatX;  

%-----  

%Stockage pratique des résultats des simulations de MonteCarlo  

ResultatsMC.ErrX1(:,n)=Resultat.Err(:,1);  

ResultatsMC.ErrX2(:,n)=Resultat.Err(:,2);  

ResultatsMC.Diag11Pk(:,n)=Resultat.DiagPk(:,1);  

ResultatsMC.Diag22Pk(:,n)=Resultat.DiagPk(:,2);  

%-----  

end;  

%%%%%%%%%%%%%%  

%%%%%%%%%%%%%%  

%%%%%%%%%%%%%%  

%Traitemet des résultats des simulations de Monte-Carlo  

%-----  

%Calcul de la moyenne empirique de l'erreur d'estimation  

%Pour X1  

ResultatsMC.MeanX1=mean(ResultatsMC.ErrX1)';  

%Pour X2  

ResultatsMC.MeanX2=mean(ResultatsMC.ErrX2)';  

%Mise en forme pour faciliter les tracés  

ResultatsMC.MeanErr=[ResultatsMC.MeanX1,ResultatsMC.MeanX2];  

%-----  

%Calcul du max et du min de l'erreur d'estimation  

%Pour X1  

ResultatsMC.MaxErrX1=max(ResultatsMC.ErrX1)';  

ResultatsMC.MinErrX1=min(ResultatsMC.ErrX1)';  

%Pour X2  

ResultatsMC.MaxErrX2=max(ResultatsMC.ErrX2)';  

ResultatsMC.MinErrX2=min(ResultatsMC.ErrX2)';  

%Mise en forme pour faciliter les tracés  

ResultatsMC.MaxErr=[ResultatsMC.MaxErrX1,ResultatsMC.MaxErrX2];  

ResultatsMC.MinErr=[ResultatsMC.MinErrX1,ResultatsMC.MinErrX2];  

%-----  

%-----  

%Moyenne des composantes diagonales des covariances fournies par le filtre  

%Pour X1  

ResultatsMC.Mean11Pk=mean(ResultatsMC.Diag11Pk)';  


```

```

%Pour X2
ResultatsMC.Mean22Pkk=mean(ResultatsMC.Diag22Pkk');

%Mise en forme pour faciliter les tracés
ResultatsMC.MeanPkk=[ResultatsMC.Mean11Pkk,ResultatsMC.Mean22Pkk];
%-----

%-----
%Covariance empirique de l'erreur d'estimation

%Pour X1
ResultatsMC.CovErrX1=diag(cov(ResultatsMC.ErrX1'));
%Pour X2
ResultatsMC.CovErrX2=diag(cov(ResultatsMC.ErrX2'));

%Mise en forme pour faciliter les tracés
ResultatsMC.CovErr=[ResultatsMC.CovErrX1,ResultatsMC.CovErrX2];
%-----



%%%%%%%%%%%%%%%
%TRACES DES RESULTATS
Texte={'X1','X2'};
numfig=0;

% Moyenne empirique de l'erreur d'estimation, son min et son max
for i=1:2,
    numfig=numfig+1;
    figure(numfig)
    a=plot(Tps,ResultatsMC.MeanErr(:,i),'b-');
    hold on
    b=plot(Tps,ResultatsMC.MinErr(:,i),'g-');
    c=plot(Tps,ResultatsMC.MaxErr(:,i),'r-');
    legend([a,b,c],'Moyenne Empirique','Min','Max')
    title(['Erreur d estimation de ',Texte{i}]);
end;

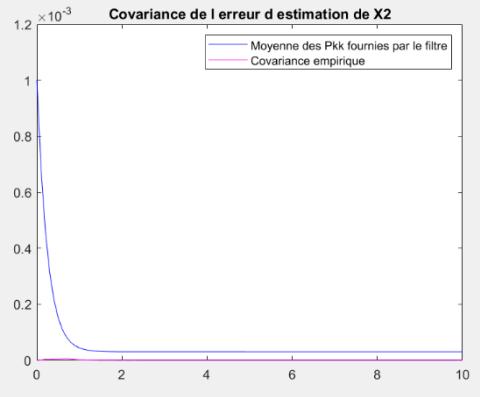
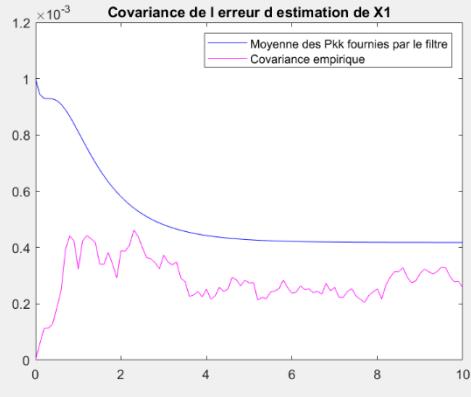
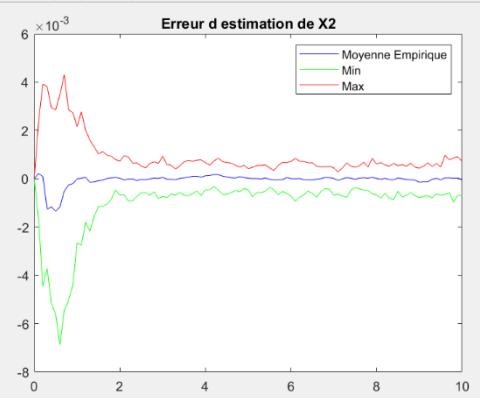
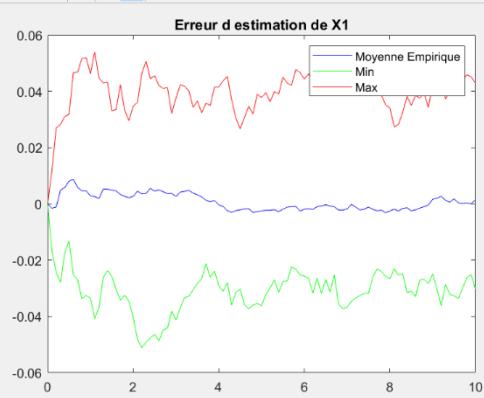
% Covariances
for i=1:2,
    numfig=numfig+1;
    figure(numfig)
    a=plot(Tps,ResultatsMC.MeanPkk(:,i),'b-');
    hold on
    b=plot(Tps,ResultatsMC.CovErr(:,i),'m-');
    legend([a,b],'Moyenne des Pkk fournies par le filtre','Covariance empirique')
    title(['Covariance de l erreur d estimation de ',Texte{i}]);
end;

%%%%%%%%%%%%%%

```

En lançant ce code on a :

$$\begin{aligned}x_{0/0} &= [4 \ x_{e_2}] \\Q &= \text{diag}([10^{-5}, 10^{-5}]) \\R &= 10^{-2} \\P_{0|0} &= \text{diag}([10^{-3}, 10^{-3}])\end{aligned}$$



Cela nous montre qu'au cours de la simulation (l'axe du temps est l'axe des abscisses), les moyennes empiriques sur les erreurs d'estimation sont proches de 0 (pour x_1 et x_2).

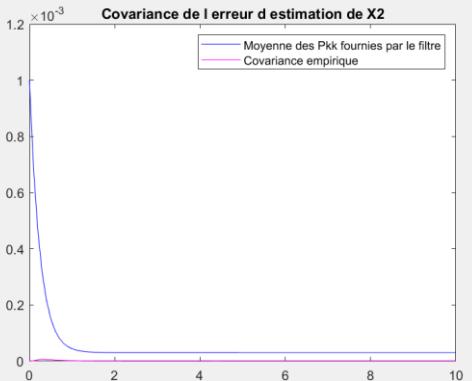
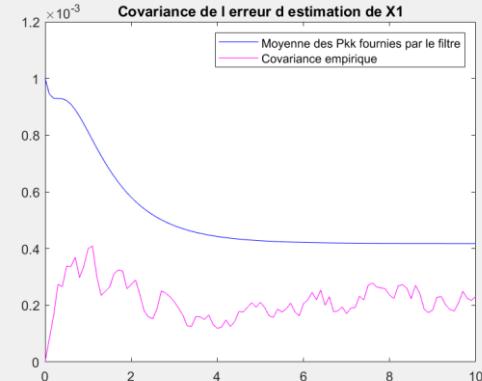
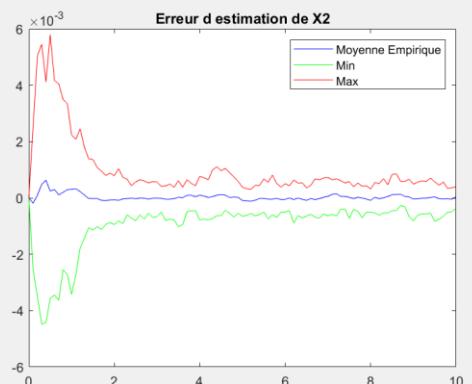
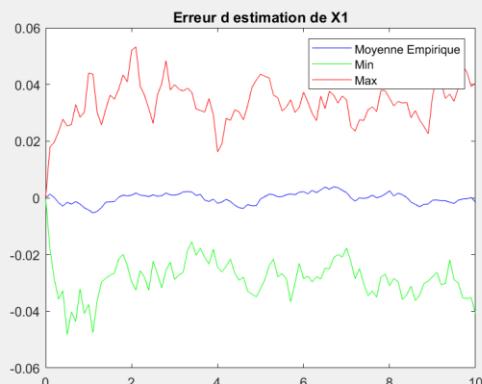
Mais cela nous permet aussi de voir que les covariances (la dépendance d'une variable par rapport à une autre). On remarque ici que pour les x_1 , on a les P_{kk} bien supérieurs aux covariances empiriques. On en déduit que, le filtre de Kalman étendu est prudent, son estimation correcte mais un peu protectrice. Pour les x_2 , on note que les P_{kk} sont proches des covariances réelles, on en déduit que l'estimation du filtre est très précise, le modèle est donc bien adapté.

Étude de l'influence des paramètres du filtre sur les erreurs d'estimations avec les simulations de Monte-Carlo

On va commencer par faire varier les valeurs de l'estimé initiale avec les valeurs que l'on a précédemment utilisées pour voir comment réagi notre filtre avec les simulations de Monte-Carlo.

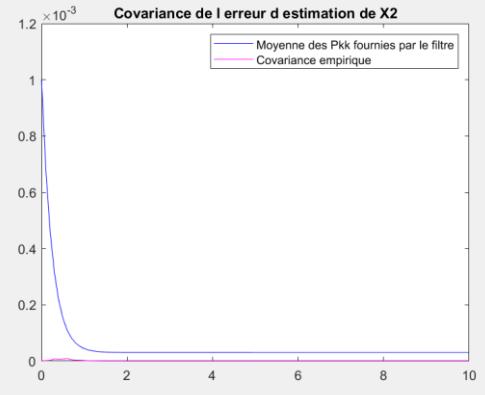
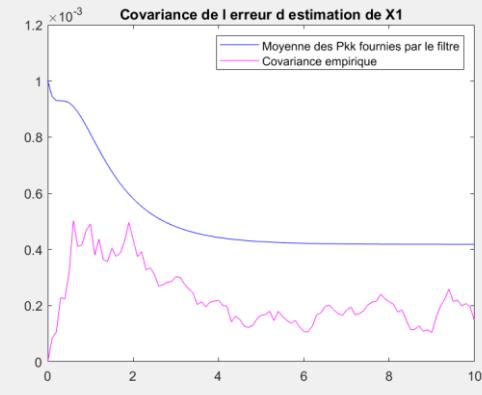
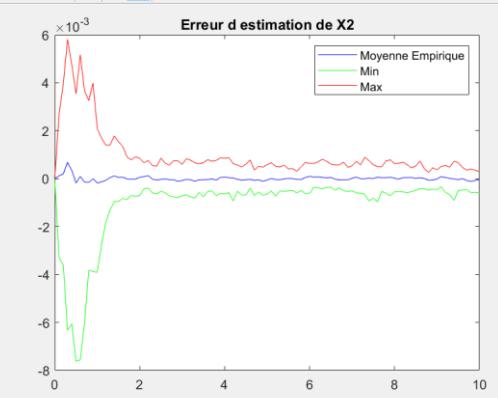
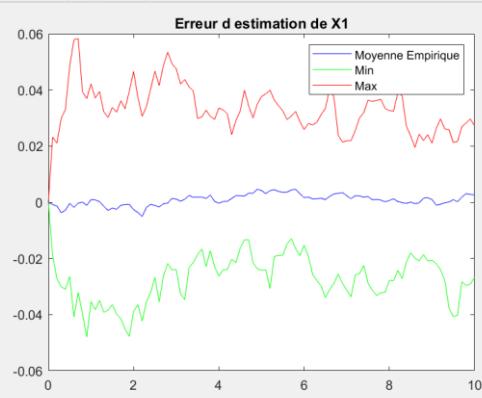
On a :

$$\begin{aligned}x_{0|0} &= [10 \ x_{e_2}] \\Q &= \text{diag}([10^{-5}, 10^{-5}]) \\R &= 10^{-2} \\P_{0|0} &= \text{diag}([10^{-3}, 10^{-3}])\end{aligned}$$

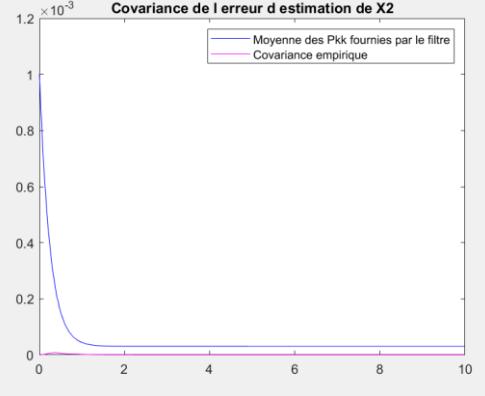
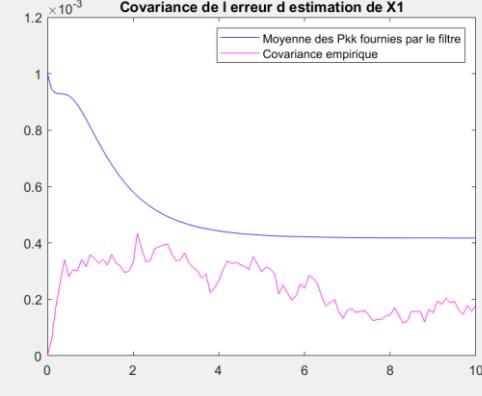
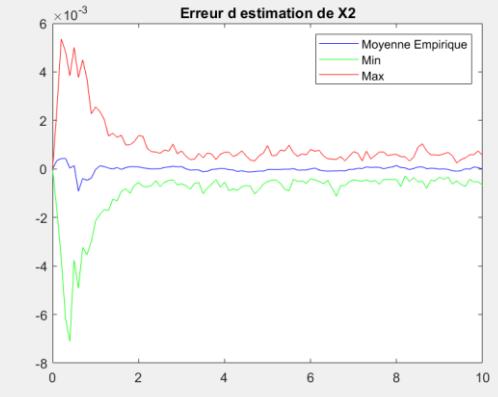
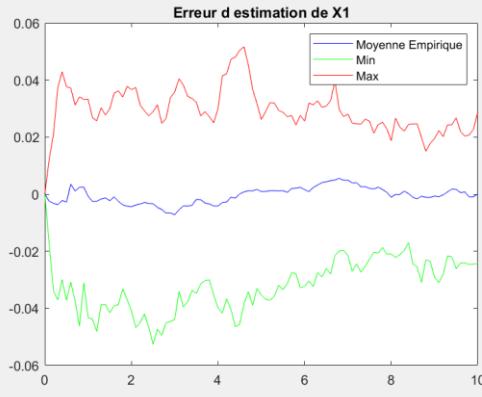


On commence par faire varier $x_{0|0}$:

$$\begin{aligned}x_{0|0} &= [100 \ x_{e_2}] \\Q &= \text{diag}([10^{-5}, 10^{-5}]) \\R &= 10^{-2} \\P_{0|0} &= \text{diag}([10^{-3}, 10^{-3}])\end{aligned}$$



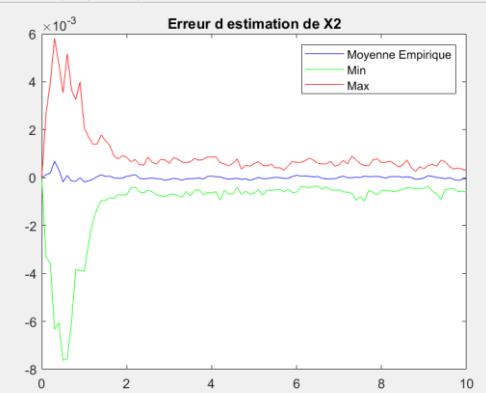
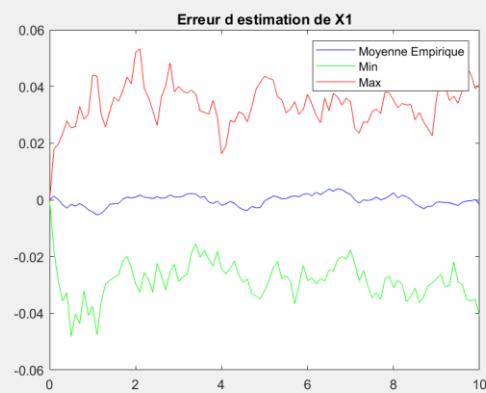
$$\begin{aligned}x_{0|0} &= [0.1 \ x_{e_2}] \\Q &= \text{diag}([10^{-5}, 10^{-5}]) \\R &= 10^{-2} \\P_{0|0} &= \text{diag}([10^{-3}, 10^{-3}])\end{aligned}$$



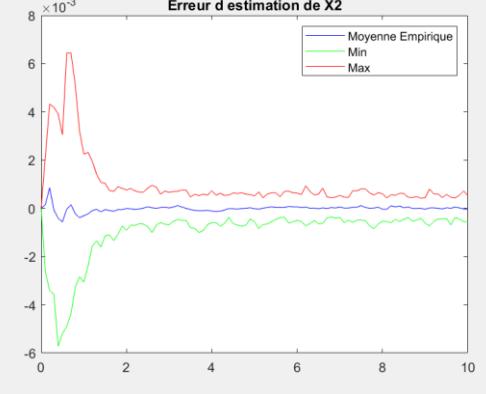
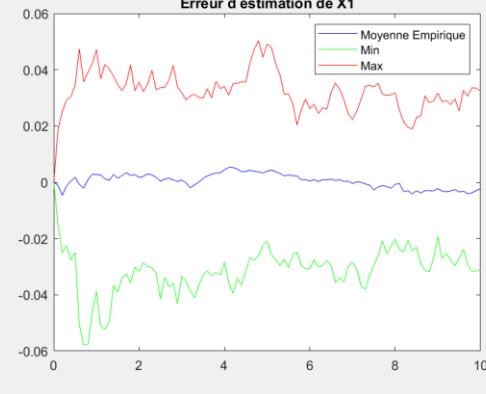
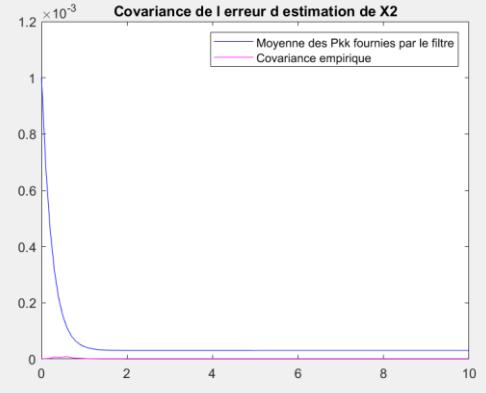
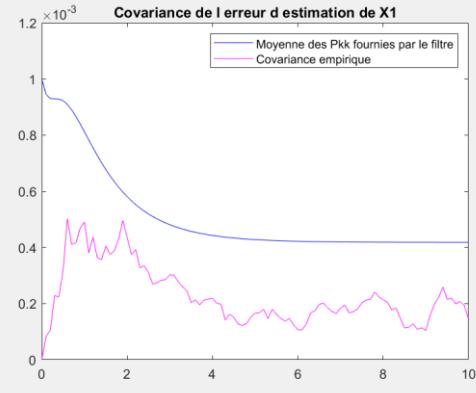
On remarque l'influence de $x_{0|0}$ est faible, les moyennes empiriques sont toujours autour de 0, et on a toujours les P_{kk} supérieur aux covariances empiriques pour x_1 et P_{kk} les P_{kk} biens supérieurs aux covariances empiriques pour x_1 et les P_{kk} plus proches des covariances empiriques pour les x_2 . Le filtre n'a pas changé de stratégie.

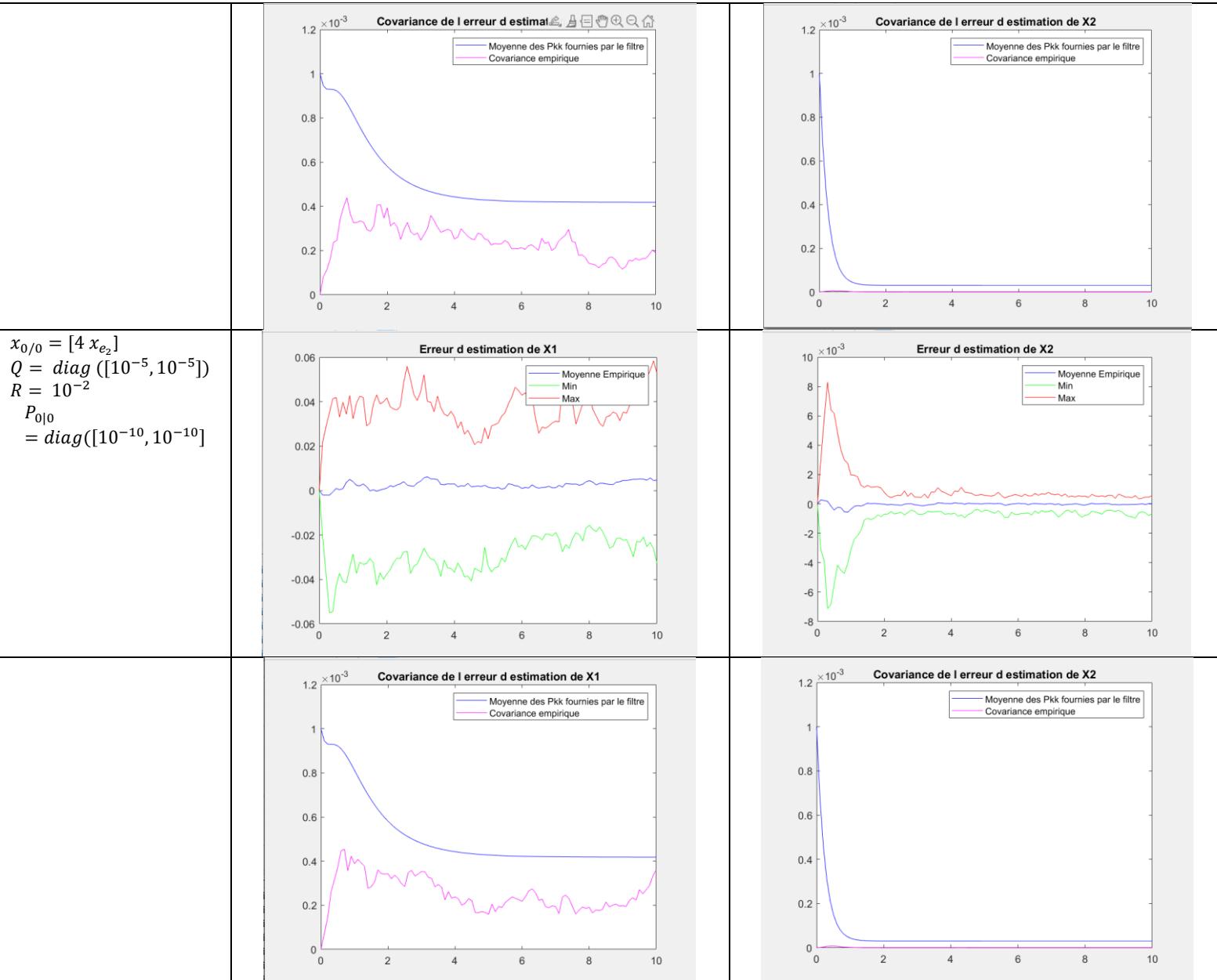
Passons au $P_{0|0}$:

$$x_{0|0} = [4 \ x_{e_2}] \\ Q = diag([10^{-5}, 10^{-5}]) \\ R = 10^{-2} \\ P_{0|0} = diag([10^{-3}, 10^{-3}])$$



$$x_{0|0} = [4 \ x_{e_2}] \\ Q = diag([10^{-5}, 10^{-5}]) \\ R = 10^{-2} \\ P_{0|0} = diag([10^{-5}, 10^{-5}])$$

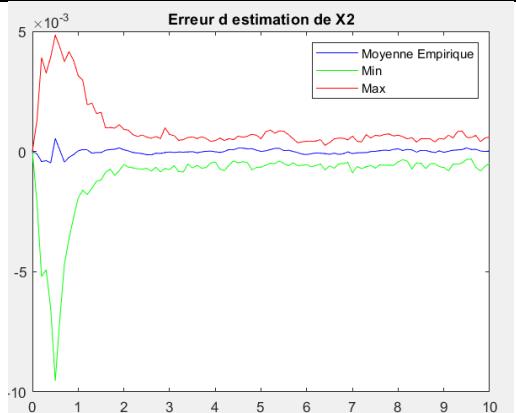
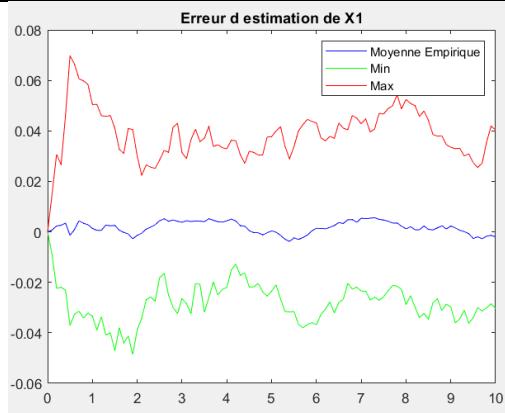




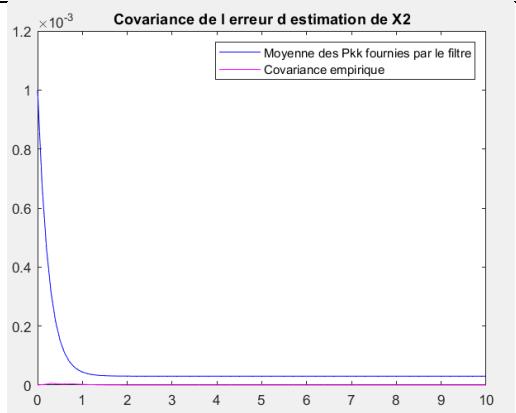
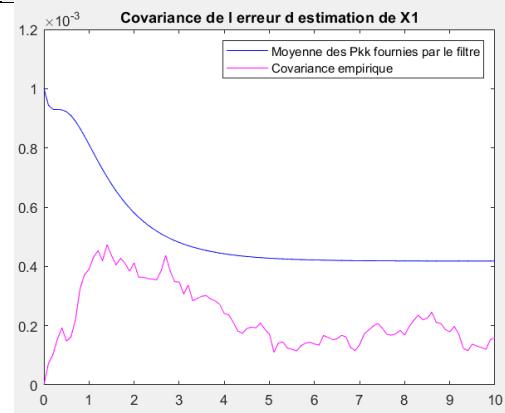
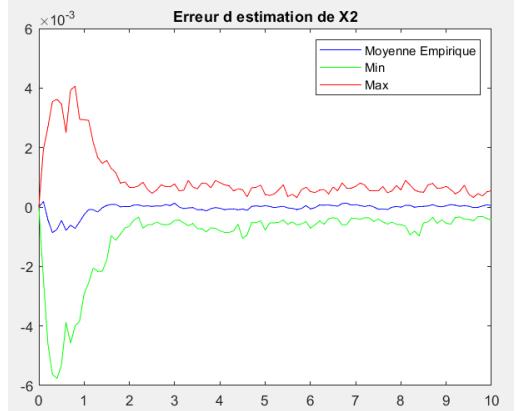
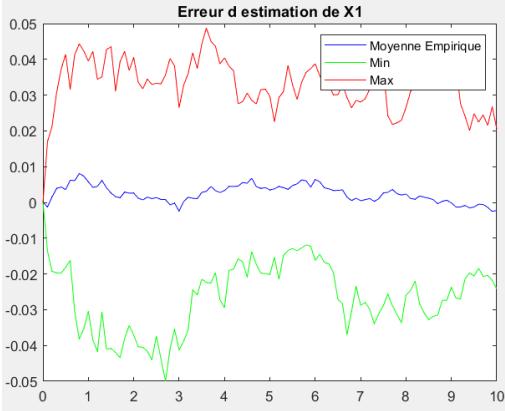
De la même manière, l'on remarque que le filtre ne change pas de stratégie. Puisque le filtre le change pas de stratégie peu importe les paramètres initiaux, l'on peut conclure que notre filtre de Kalman étendu est robuste par rapport à notre système et à son état initial.

Puis l'on va faire varier Q :

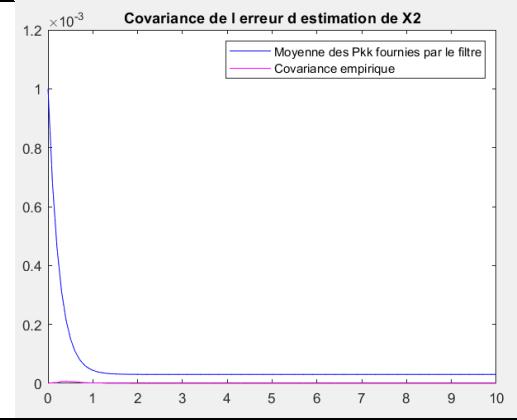
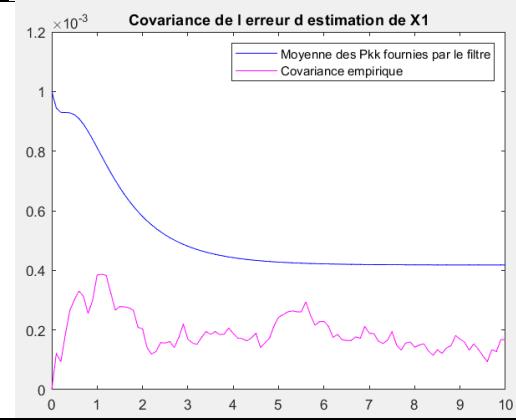
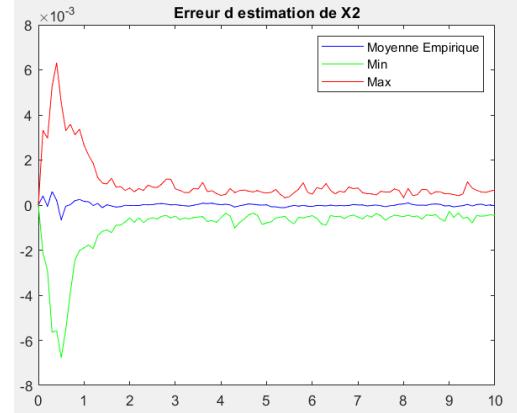
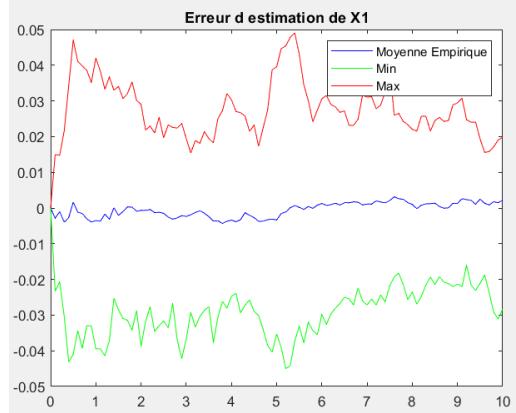
$$\begin{aligned}x_{0/0} &= [10 \ x_{e_2}] \\Q &= \text{diag}([10^{-10}, 10^{-1}]) \\R &= 10^{-2} \\P_{0|0} &= \text{diag}([10^{-3}, 10^{-3}])\end{aligned}$$



$$\begin{aligned}x_{0/0} &= [10 \ x_{e_2}] \\Q &= \text{diag}([10^{-1}, 10^{-10}]) \\R &= 10^{-2} \\P_{0|0} &= \text{diag}([10^{-3}, 10^{-3}])\end{aligned}$$

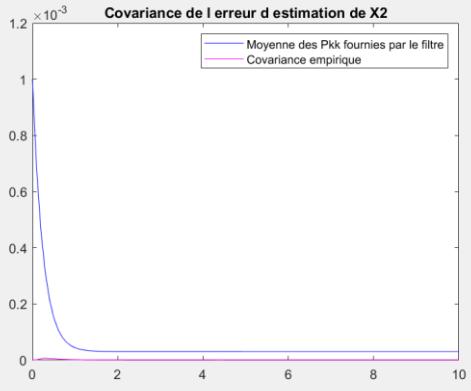
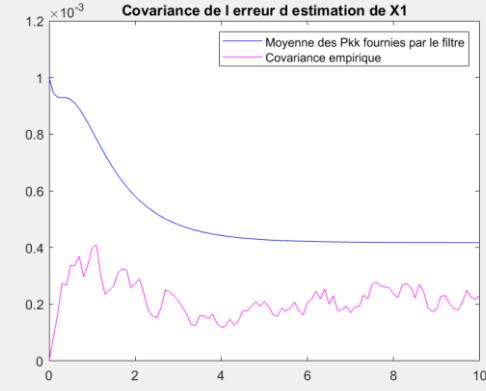
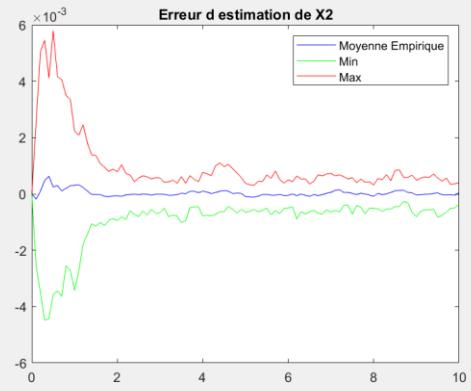
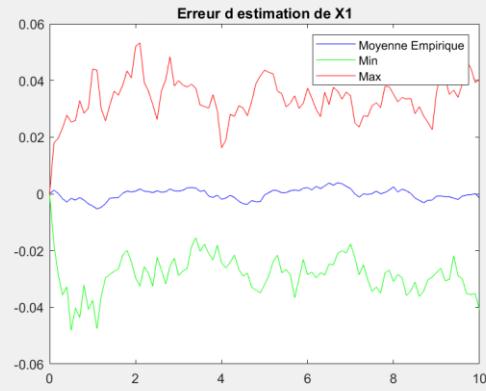


$$\begin{aligned}
x_{0|0} &= [10 \ x_{e_2}] \\
Q &= \text{diag}([10^{-10}, 10^{-10}]) \\
R &= 10^{-2} \\
P_{0|0} &= \text{diag}([10^{-3}, 10^{-3}])
\end{aligned}$$

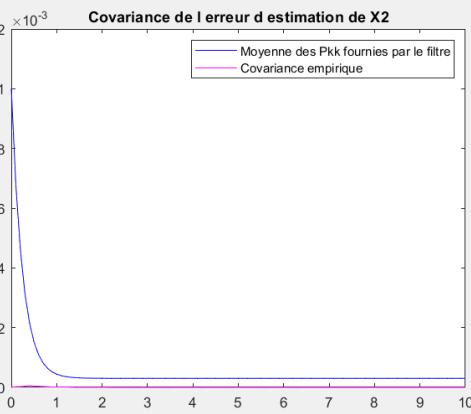
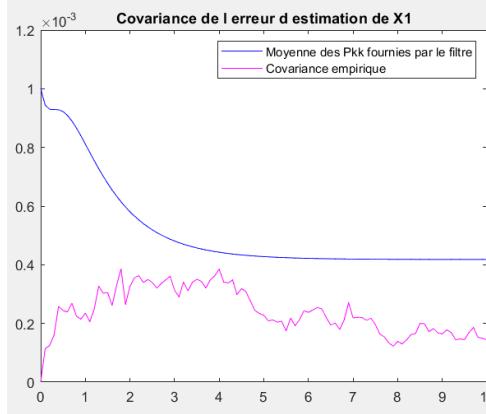
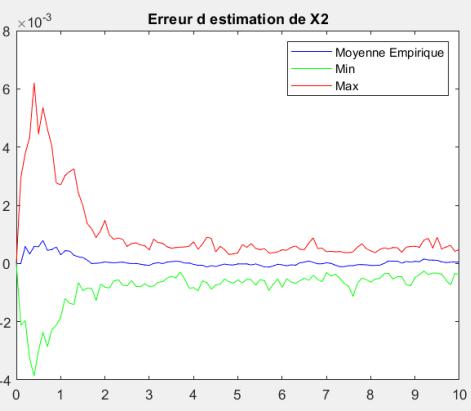
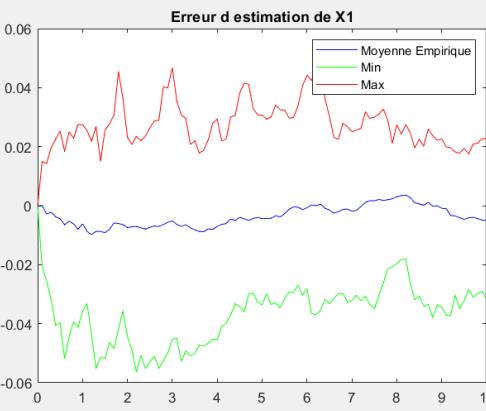


Et enfin on fait varier R

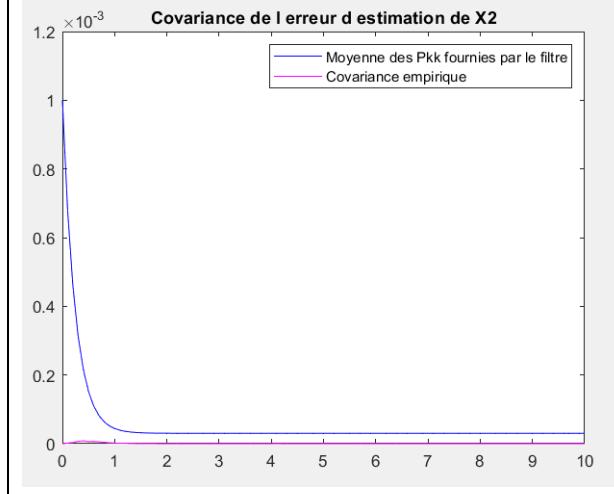
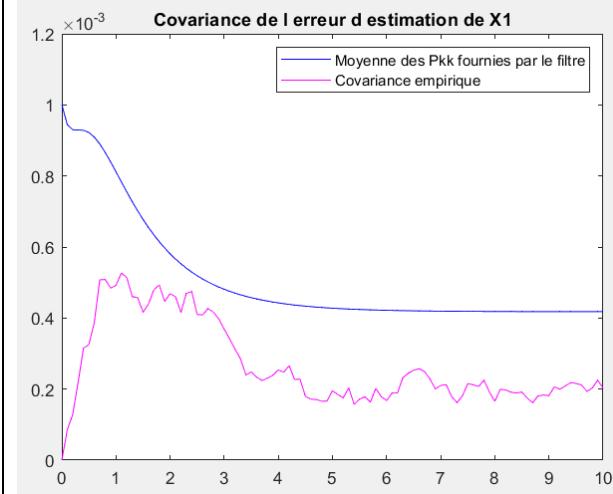
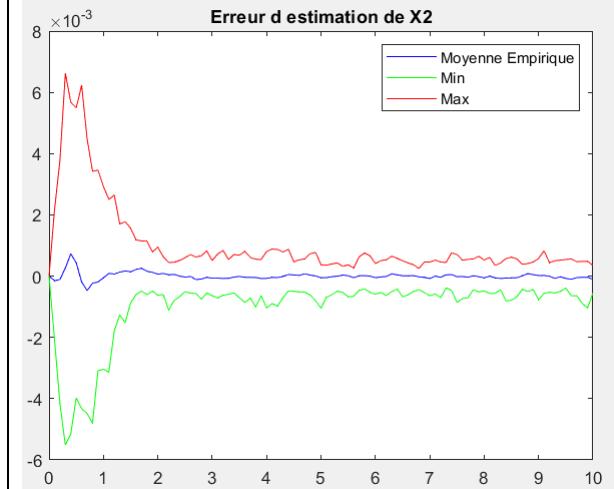
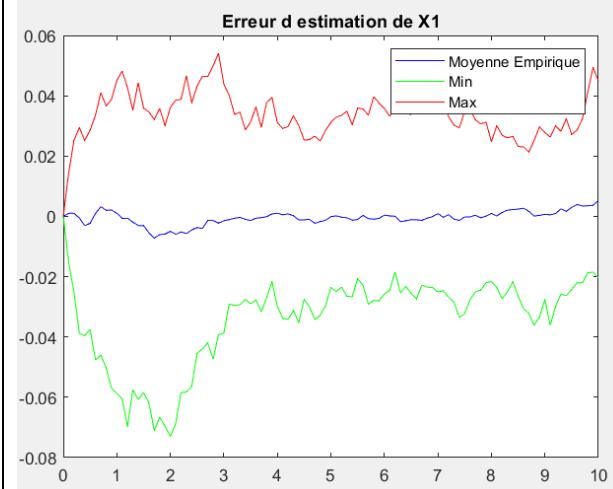
$$x_{0/0} = [10 \ x_{e_2}] \\ Q = \text{diag}([10^{-5}, 10^{-5}]) \\ R = 10^{-2} \\ P_{0|0} = \text{diag}([10^{-3}, 10^{-3}])$$



$$x_{0/0} = [10 \ x_{e_2}] \\ Q = \text{diag}([10^{-5}, 10^{-5}]) \\ R = 10^{-5} \\ P_{0|0} = \text{diag}([10^{-3}, 10^{-3}])$$



$$\begin{aligned}
x_{0/0} &= [10 \ x_{e_2}] \\
Q &= \text{diag}([10^{-5}, 10^{-5}]) \\
R &= 10^{-10} \\
P_{0|0} &= \text{diag}([10^{-3}, 10^{-3}])
\end{aligned}$$



On remarque que à chaque fois le filtre ne réagit pas trop aux bruits (variations de R).

On notera aussi que on a les P_{kk} de x_1 et des x_2 qui sont supérieurs aux covariances empiriques, il est d'ailleurs souvent bien supérieur aux covariances empiriques pour les x_1 , on peut donc faire les coefficients de Q afin que les P_{kk} et les covariances empiriques des x_1 soient le plus proches possible.

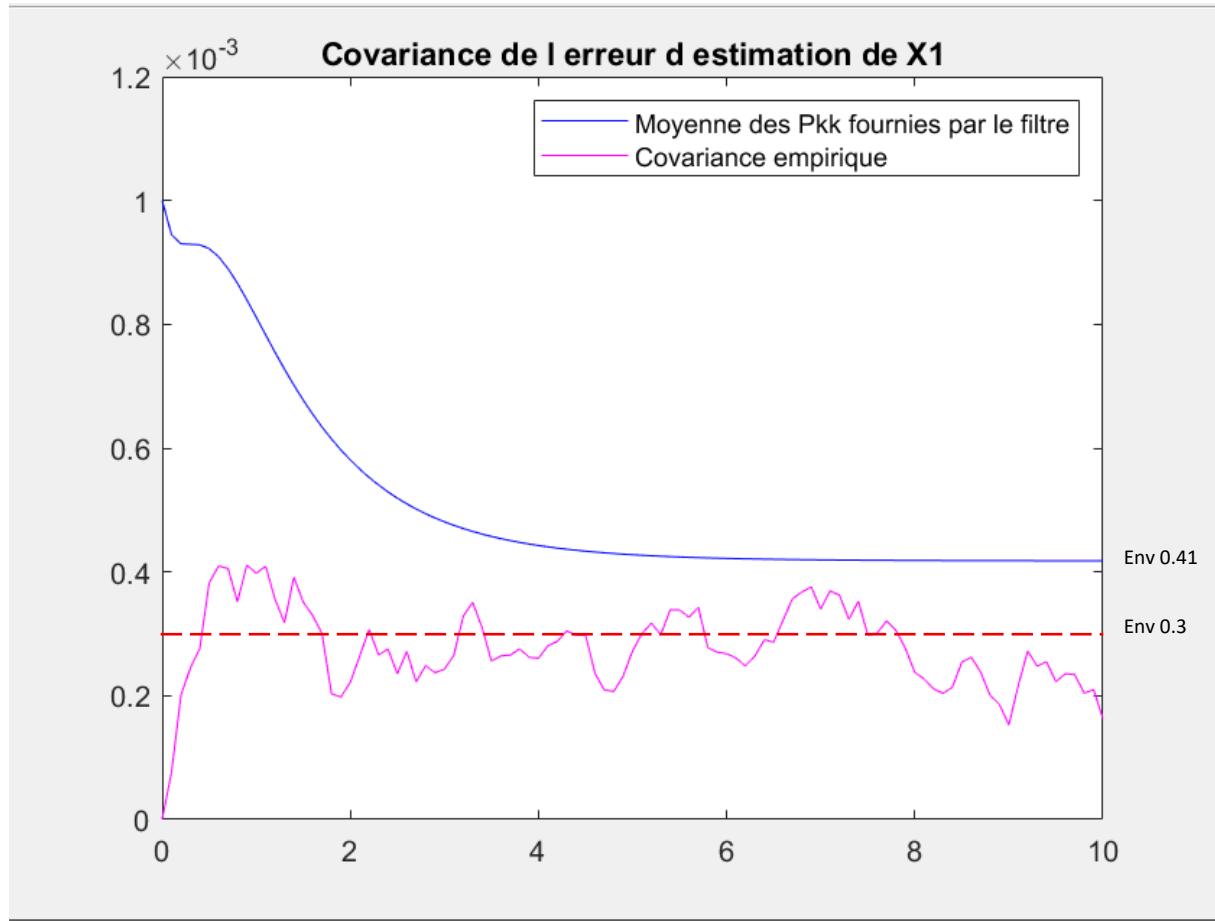
On a, avec :

$$x_{0|0} = [10 \ x_{e_2}]$$

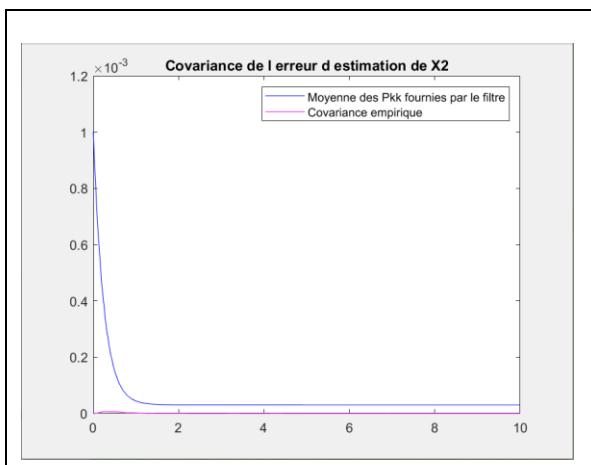
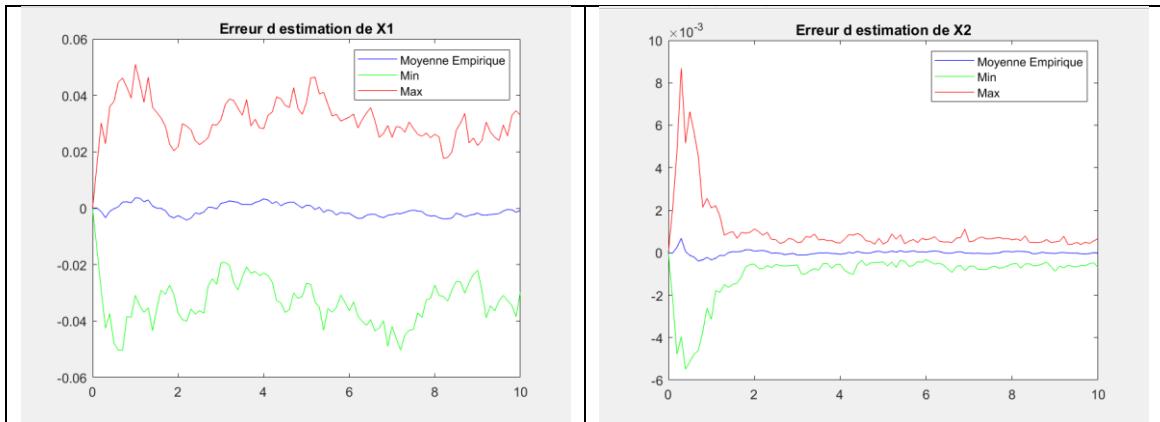
$$Q = \text{diag}([10^{-10}, 10^{-1}])$$

$$R = 10^{-2}$$

$$P_{0|0} = \text{diag}([10^{-5}, 10^{-5}])$$



On a donc une différence en moyenne de 0.11 ce qui semble raisonnable. De plus on a :



Conclusion

Le Filtre de Kalman linéaire fonctionne bien sur un modèle linéaire ou un modèle non linéaire qui a été linéarisé autour d'un point d'équilibre. Cependant, dès que le modèle n'est plus linéaire, le filtre ne donne plus des valeurs proches du système réel, il faut alors passer sur le filtre de Kalman étendu.

Le Filtre de Kalman étendu fonctionne bien sur un modèle non linéaire, bien que plus compliqué à mettre en place et à corriger (il faudra sûrement passer par des simulations de Monte-Carlo afin d'évaluer la robustesse du filtre, l'impact des différents bruits dessus et son compromis prudence/confiance).

Pour notre système, nous recommandons le paramétrage suivant sur un filtre de Kalman étendu :

$$x_{0|0} = [10 \ x_{e_2}]$$

$$Q = \text{diag}([10^{-10}, 10^{-1}])$$

$$R = 10^{-2}$$

$$P_{0|0} = \text{diag}([10^{-5}, 10^{-5}])$$

Car il nous donne un filtre peu sensible aux bruits, un peu prudent et dont les moyennes empiriques des erreurs sont proches de 0.

