Abderrahmane Aït-Simmou

Filtrage non-linéaire et expansion en chaos de Wiener

Mémoire
présenté
à la l'Université du Québec à Trois-Rivières
pour l'obtention
du grade de maître ès sciences (M.Sc.)

Département de mathématiques et d'informatique
UNIVERSITÉ DU QUÉBEC À TROIS-RIVIÈRES

Dédicace

Je dédie le présent mémoire à mes parents en témoignage de leur soutien et de leurs encouragements tout au long de mes études.

Remerciements

À travers cette page, j'aimerais remercier mon directeur de recherche M. Bruno Rémillard pour le support qu'il m'a apporté et la confiance qu'il m'a témoignée au cours des deux dernières années. Il m'a donné le goût de la recherche et il m'a permis par sa rigueur et ses connaissances d'améliorer mes aptitudes dans ce domaine.

Je tiens aussi à remercier le centre d'excellence MITACS (Mathematics of Information Technology and Complex Systems); ainsi que le groupe Lockheed Martin Canada pour leur soutien technique et financier.

Mes remerciements s'adressent aussi à Mhamed Mesfioui et Dominic Rochon qui ont accepté de corriger ce mémoire.

Table des matières

1 Introduction **8**

2 Modèles généraux de filtrage **12**

2.1 Description du problème de filtrage. 12

2.2 Quelques approches analytiques pour la résolution du problème de filtrage. 15

2.2.1 Filtre de Kalman. 15

2.2.2 Filtre de Kalman étendu (EKF) 18

2.2.3 Filtre IMM (Interacting Multiple models) 21

2.2.4 Conclusion 23

2.3 Solution générale du problème de filtrage. 24

2.3.1 Filtrage non linéaire 25

2.3.2 Filtrage linéaire 28

2.4 Estimation par la méthode du maximum de vraisemblance. . . 31

2.5 Conclusion 32

3 Approximation par l'expansion en chaos de Wiener 33

3.1 Approximation par la méthode de l'expansion en chaos de Wiener. 34

3.2 Développement de l'algorithme 42

3.3 Méthode numérique pour approximer la solution de l'équation de Fokker-Planck. 45

3.4 Conclusion 47

4 Quelques exemples d'applications 50

4.1 Principe de mesure d'un radar 51

4.2 Premier modèle (radar passif). 51

4.3 Deuxième modèle(radar actif) 54

4.4 Troisième modèle (modèle d'Ornstein-Ulhembeck) 55

4.5 Conclusion 57

5 Conclusion 58

6 Codes des programmes sur matlab **60**

6.1 Codes des programmes pour le premier modèle. 60

6.2 Codes des programmes pour le deuxième modèle. 70

6.3 Codes des programmes pour le troisième modèle. 79

A Processus de Markov et mouvement brownien **88**

A.1 Processus de Markov 88

A.2 Mouvement brownien 89

B Résultats graphiques. **91**

B.1 Résultats graphiques pour le premier modèle: 91

B.2 Résultats graphiques pour le deuxième modèle: 96

B.3 Résultats graphiques pour le troisième modèle: 98

Liste des tableaux

4.1 Erreurs quadratiques pour le premier modèle 53

4.2 Erreurs quadratiques pour le deuxième modèle 55

4.3 Erreurs quadratiques pour le troisième modèle 57

Liste des figures

3.1 Approximation de la densité gaussienne de moyenne 5 et d'écart type 2 par la base d'Hermite pour $\kappa = 10$ 48

3.2 Approximation de la densité gaussienne de moyenne 5 et d'écart type 2 par la base d'Hermite pour $\kappa = 40$ 49

B.1 Trajectoires réelle et prédite pour $\kappa = 10$. 91

B.2 Trajectoires réelle et prédite pour $\kappa = 15$. 92

B.3 Trajectoires réelle et prédite pour $\kappa = 30$. 92

B.4 Trajectoires réelle et prédite pour $\kappa = 50$. 93

B.5 Trajectoire réelle avec un nombre de points égal à 100. 93

B.6 Trajectoire réelle avec un nombre de points égal à 500. 94

B.7 Trajectoire réelle avec un nombre de points égal à 1000. 94

B.8 Trajectoire réelle et prédite de l'abscisse x. 95

B.9 Trajectoire réelle et prédite de l'ordonné y. 95

B.10 Trajectoire réelle et prédite pour $\kappa = 10$. 96

B.11 Trajectoire réelle et prédite pour $\kappa = 15$. 97

B.12 Trajectoire réelle et prédite pour $\kappa = 30$. 97

B.13 Trajectoire réelle pour un nombre de points égales à 1000. . . 98

B.14 Trajectoire réelle pour un nombre de points égales à 10000. . . 99

B.15 Trajectoire réelle et prédite pour $\kappa = 10$. 99

B.16 Trajectoire réelle et prédite pour $\kappa = 15$. 100

B.17 Trajectoire réelle et prédite pour $\kappa = 30$. 100

Chapitre 1

Introduction

Le filtrage est un outil de base pour le problème d'assimilation de données dans les modèles numériques. Cet outil est de type stochastique dans le cas où la dynamique du système étudié est chaotique. L'objectif du filtrage est de déterminer une bonne approximation de l'état d'un système dynamique au vu des données observées, ces dernières apparaissant comme les valeurs d'un processus lié à l'état du système et contaminées par un bruit d'observations. Le problème peut se présenter soit en temps discret soit en temps continu.
La recherche dans le domaine du filtrage est motivée par de nombreuses applications telles le pistage de cible ou l'évaluation du risque en finance. Le problème consiste essentiellement à estimer, à chaque instant d'observation, la valeur d'un processus aléatoire markovien $\{X_t\}_{t\geq 0}$, de transition connue, appelé signal, lié à un processus observation $\{Y_t\}_{t\geq 1}$. Le but est donc de calculer, de manière exacte ou approchée, à chaque instant t_k la distribution de probabilité conditionnelle de X_{t_k} étant donnée les observations jusqu'à

l'instant t_k. La suite de ces distributions constitue ce qu'on appelle le filtre optimal. Elle évolue dans le temps par une alternance d'étapes de prédiction et de correction. Dans le problème de filtrage on cherche à construire un algorithme récursif, qui pour calculer la nouvelle estimée de la loi du système, n'a besoin que de l'estimée à l'étape précédente et de la nouvelle observation. La condition nécessaire pour que le filtre optimal soit récursif est que le système soit markovien conditionnellement à l'observation [voir l'appendice pour la définition des processus de Markov].

En 1961 Kalman et Bucy (Andrew C. Harvey 1989) révolutionnent la théorie de l'estimation en fournissant le premier algorithme de filtrage récursif. Le filtre de Kalman-Bucy permet en effet le calcul exact et rapide du filtre optimal lorsque les modèles d'état et d'observation ne font intervenir que les fonctions linéaires et des bruits additifs gaussiens. On montre en fait que dans ce cas, la loi conditionnelle recherchée est gaussienne. Mais, en toute généralité, le problème du filtrage non linéaire n'admet pas de solution de dimension finie. En pratique on continue à utiliser, dans ce cas, des versions dérivées du filtre de Kalman-Bucy, comme le filtre de Kalman étendu, qui linéarise le modèle autour de l'estimée courante. Bien que cet algorithme soit depuis longtemps le filtre linéaire le plus utilisé, son efficacité est limitée à certains cas d'observations très précis. Cependant, lorsque le système est loin d'un modèle linéaire ou si le filtre est mal initialisé, cette méthode donne de mauvais résultats.

Parmi les méthodes de filtrage les plus utilisées actuellement, surtout dans le domaine de détection de trajectoires, on trouve les méthodes à modèles multiples MM (Multiple Model), en particulier la méthode à modèles multi-

ples interactifs (Interacting Multiple Model, IMM) proposée par Bar-Shalom, Chang et Blom(1982). Dans cette méthode on utilise plusieurs filtres associés aux divers modèles de dynamique de la cible. Ces filtres interagissent entre eux à chaque instant de mesure pour corriger leur estimé par rapport aux autres filtres.

D'autres approches basées sur les méthodes de Monte Carlo ont été récemment proposées pour résoudre le problème de filtrage. Leur avantage est qu'elles sont insensibles à la dimension de l'espace et aux non linéarités du système. Cependant les algorithmes développés jusqu'ici montrent un comportement instable en dehors de certaines hypothèses.

Dans ce mémoire on s'intéresse à une autre approche basée sur l'approximation de la densité conditionnelle par la résolution de l'équation de Fokker-Planck. En effet, si $f(x)$ est une fonction telle que $\mathbf{E}(X(t)) < \infty \ \forall t \geq 0$, alors sous certaines conditions la meilleure estimation au sens quadratique $\widehat{f}_t$ de $f(X(t))$ étant donné la trajectoire $\{Y(s), s \leq t\}$ est donnée par :

$$\widehat{f}_t = \frac{\int f(x)u(t,x)dx}{\int u(t,x)dx},$$

où $u(t,x)$ est une fonction aléatoire appelée densité de filtrage non normalisée (UFD) (e.g. Kushner 1997). L'estimation de $f(X(t))$ est complètement déterminée par le calcul de $u(t,x)$, solution de l'équation différentielle stochastique de Zakai. La résolution exacte de cette équation est presque impossible sauf dans certains cas. Ainsi, des méthodes numériques ont été suggérées pour son approximation.

Lorsque les paramètres du modèle sont connus à l'avance, une approche alternative consiste à séparer les composantes déterministes et stochastiques de l'équation de Zakai en utilisant la décomposition en chaos de Wiener. Cette

approche a été développé par Mikulevicius et Rozovskii (1993) . Le premier algorithme qui a utilisé cette méthode pour résoudre l'équation de Zakai a été suggéré par Lototsky , Mikulevicius et Rosovskii (1997).

Le but de ce mémoire est de reprendre la méthode numérique proposée par Sergey V. Lototsky et Boris L. Rozovskii (1998) pour le calcul de $\widehat{f}_t$. L'avantage de cette algorithme est dû à la simplicité de sa partie en temps réel(on-line) qui ne demande aucune resolution d'équation différentielle. Les calculs les plus compliqués liés à la résolution de l'équation de Fokker-Plank sont faits une seule fois et avant de recueillir les observations(off-line). Cette approche est basée sur la technique connue sous le nom de parametrisation de la densité de filtrage non normalisée (UFD) (Jazwinski 1970). Dans l'algorithme proposé (voir chapitre 3) nous suggérons une nouvelle méthode pour l'approximation des intégrales pour la résolution de l'équation de Fokker-Plank. On appliquera par la suite cette méthode de filtrage dans le domaine de poursuite de cible. En effet, on s'intéresse à prévoir les paramètres cinématiques d'un mobile (position, vitesse, accélération) au vu de mesures bruitées fournies par un capteur radar.

Le mémoire s'organisera de la manière suivante : dans le deuxième chapitre nous présenterons le problème général de filtrage. Les différentes approches adoptées pour traiter ce problème sont brièvement décrites. Le troisième chapitre a pour objet de décrire la méthode d'approximation proposées par Lototsky et Rozovskii (1998). Au quatrième chapitre nous appliquerons la méthode à trois modèles différents et nous discuterons les résultats. Nous terminerons notre par une conclusion générale.

Chapitre 2

Modèles généraux de filtrage

Dans la première section de ce chapitre, nous commençons par rappeler la vaste portée et la définition du problème de filtrage non linéaire. La seconde section rappelle les différentes approches adoptées pour résoudre ce problème. Dans la troisième, nous présenterons le filtre de Kalman, le filtre de Kalman étendu, et le filtre IMM . La quatrième section sera consacrée à décrire la solution générale du problème de filtrage non linéaire. Dans la dernière section on présentera la méthode du maximum de vraisemblance comme solution pour le problème de l'estimation des paramètres.

2.1 Description du problème de filtrage.

La recherche dans le domaine du filtrage est motivée par de nombreuses applications dans des domaines variés. Ce problème est en effet assez général

puisqu'il s'agit d'estimer l'état d'un système dynamique à partir d'observation bruitées.

En effet, les systèmes physiques ou économiques sont souvent sujets à des perturbations aléatoires. La modélisation de l'évolution dans le temps des états $\{X(t)\}_{t\geq 0}$ du système considéré s'écrit alors sous la forme d'un terme d'évolution déterministe et d'un terme aléatoire:

$$dX_t = b(X_t)dt + \sigma(X(t))dW_t, \tag{2.1}$$

où X_0 est de loi donnée , b est une fonction vectorielle connue caractérisant la partie déterministe de la dynamique, et σ est une fonction matricielle. $\{W(t)\}_{t\geq 0}$ est un mouvement brownien standard modélisant les perturbations aléatoires de la dynamique. Cette équation de la dynamique traduit la connaissance à priori que l'on a du système. Dans le but de déterminer l'état précis du système on élabore une procédure d'observation permettant de recueillir, à des instants discrets, des mesures des états du système. Ces observation sont, en général, entachées d'erreurs dû à l'imperfection de l'instrument de mesure. La suite d'observation $(Y_k)_{k\geq 1}$ est modélisée par l'équation suivante, reliant l'observation Y_k à l'état courant $X(t_k)$

$$Y_k = h(X(t_k)) + V_k, \tag{2.2}$$

où h est la fonction d'observation connue et $\{V_k\}_{k\geq 1}$ une suite de variables aléatoires de statistique connue modélisant l'imperfection des observations. Le problème consiste à estimer l'état du système, $X(t)$, à partir des observations bruitées (Y_1, Y_2,..., Y_k).

- Si $t > k$, on parle de lissage.
- Si $t = k$, on parle de filtrage.

- Si $t < k$, on parle de prédiction.

Si l'on choisit la distance moyenne quadratique, il est bien connu que l'estimée optimale de $X(t_k)$ au vu des observations $(Y_1, Y_2, ..., Y_k)$ est l'espérance conditionnelle $E(X(t_k)|(Y_1, Y_2, ..., Y_k))$. D'une manière générale, l'information la plus riche disponible étant donné les observations (Y_1, Y_2,..., Y_k) est contenue dans la lois conditionnelles de $X(t_k)$ sachant les observations passées.

Le problème de filtrage consistera donc à calculer cette mesure de probabilité conditionnelle. Les principales méthodes proposées dans ce domaine sont les méthodes analytiques, les méthodes de Monte-Carlo et les méthodes numériques.

Pour les méthodes analytiques, le filtre de Kalman, le filtre de Kalman-étendu et le filtre IMM (Interacting multiple model) sont les plus célèbres et représentent les outils les plus communément utilisés pour résoudre les problèmes de filtrage linéaire et non linéaire. Ces trois méthodes seront présentées en détails dans la prochaine section.

Les méthodes de Monte-Carlo, quant à elles, sont basées sur la simulation d'un grand nombres de variables aléatoires. Leur utilisation est justifiée par la loi des grands nombres qui permet d'approcher une mesure de probabilité lorsqu'on en connaît un échantillon. Ces méthodes ne seront pas traitées dans ce mémoire.

Dans notre travail, on s'intéresse plutôt aux méthodes numériques qui consiste à résoudre l'équation différentielle stochastique de Zakai de manière numérique. On peut se référer aux cours de Pardoux (1989) qui explique bien le lien entre filtrage non linéaire et EDP stochastique.

2.2 Quelques approches analytiques pour la résolution du problème de filtrage.

Parmi les méthodes analytiques les plus utilisées dans le domaine de filtrage on trouve le filtre de Kalman, Kalman-Étendu et IMM. dans cette section nous allons décrire ces trois filtres dans le cas discret.

2.2.1 Filtre de Kalman.

Le filtre de Kalman développé à l'origine par Kalman en 1960 pour le temps discret puis repris en 1961 par Kalman-Bucy pour le temps continu, permet de résoudre le problème de filtrage linéaire lorsque les bruits sont additifs et gaussiens. Une étude détaillée du filtre de Kalman est développée dans le livre de Harvey (1989). On considère ici le modèle linéaire gaussien, où le processus d'état X_k évolue suivant une équation linéaire avec un bruit additif gaussien et où l'observation Y_k est une fonction linéaire de l'état entachée d'un bruit additif gaussien. Plus précisément

$$X_k = c_k + F_k X_{K-1} + G_k W_k, \tag{2.3}$$

$$Y_k = d_k + H_k X_k + V_k. \tag{2.4}$$

On suppose de plus que :

- les coefficients F_k, c_k, G_k, H_k et d_k sont déterministes.

- Les bruits d'état et d'observation W_k et V_k sont des bruits blancs gaussiens de covariances respectives Q_k et R_k, indépendant entre eux ainsi qu'avec la condition initiale X_0.

- X_0 est gaussienne de moyenne p_0 et de covariance Q_0.

On montre dans ce cas que le processus $(X_k, Y_k)_{k\geq 1}$ est gaussien et que par conséquent les lois conditionnelles étant donné $Y_1, ..., Y_k$ sont aussi gaussiennes. Le problème est alors de dimension finie, il suffit de calculer l'espérance, notée $\widehat{X}_k$, et la matrice de covariance notée P_k. Ces deux quantités sont calculées recursivement et exactement par le filtre de Kalman-Bucy.
Les conditions initiales du filtre sont données par :

$$\widehat{X}_0 = p_0,$$

$$P_0 = Q_0.$$

$\widehat{X}_k$ et P_k sont calculés par les étapes de prédiction et correction.
Équations de prédiction :

$$\widehat{X}_{k|k-1} = F_k \widehat{X}_{k-1} + c_k, \tag{2.5}$$

$$P_{k|k-1} = F_k P_{k-1} + G_k Q_k {G_k}^T, \tag{2.6}$$

équations de correction :

$$\widehat{X}_k = \widehat{X}_{k|k-1} + K_k[Y_k - (H_k \widehat{X}_{k|k-1} + d_k)], \tag{2.7}$$

$$P_k = [I - K_k H_k] P_{k|k-1}, \tag{2.8}$$

$$K_k = P_{k|k-1} {H_k}^T [H_k P_{k|k-1} {H_k}^T + R_k]^{-1}, \tag{2.9}$$

où

$$\widehat{X}_k = E(X_k|Y_1, ..., Y_k),$$

$$\widehat{X}_{k|k-1} = E(X_k|Y_1, ..., Y_{k-1}),$$

$$P_k = E((X_k - \widehat{X}_k)(X_k - \widehat{X}_k)^T|Y_k, ..., Y_k),$$

et

$$P_{k|k-1} = E((X_k - \widehat{X}_k)(X_k - \widehat{X}_k)^T|Y_1, ..., Y_{k-1}).$$

K_k est appelé gain de Kalman. On note que les matrices de covariances $(P_k)_{k\geq 0}$ et $(P_{k|k-1})_{k\geq 0}$ (solutions de l'équation de Ricatti) et les gains $(K_k)_{k\geq 0}$ ne dépendent pas des observations $(Y_k)_{k\geq 1}$. Il est ainsi possible de les calculer d'avance et de diminuer la quantité de calculs à effectuer en temps réel.
On a trois approches conduisant aux équations du filtre de Kalman. Une première approche bayésienne consiste à calculer explicitement les deux étapes de prédiction et de correction. Une deuxième approche par projection, développée par Anderson et Moore qui consiste à montrer que les innovations $[Y_k - (H_k\widehat{X}_{k|k-1}+d_k)]$ constituent une suite de variables aléatoires indépendantes et de projeter le processus état sur la base orthonormale associée. Une troisième est de noter que le problème du filtrage linéaire gaussien est équivalent à un problème des moindres carrés récursifs. Il peut donc se résoudre en optimisant un critère (correspondant aux vraisemblances des observations) sous une contrainte (correspondant à la connaissance à priori de l'état).
Le filtre de Kalman, s'étend au cas conditionnellement gaussien, découvert par Lipster et Shiryayev (1978). Dans ce cas les coefficients $F_{k+1}, c_{k+1}, G_{k+1}, H_k$

et d_k dépendent des observations passées. Le couple $(X_k, Y_k)_{k\geq 1}$ n'est plus un processus gaussien mais la loi conditionnelle reste gaussienne et le filtre de Kalman (2.5 à 2.9) donne toujours la moyenne et la covariance.

2.2.2 Filtre de Kalman étendu (EKF)

On considère cette fois un système dynamique non linéaire, avec toujours des bruits blancs gaussiens additifs, indépendants entre eux et indépendants de la condition initiale dont la dynamique est donnée par

$$X_k = f(X_{k-1}) + GW_k, \tag{2.10}$$

$$Y_k = h(X_k) + V_k. \tag{2.11}$$

On suppose que les fonctions f et g sont dérivables et que W_k et V_k sont des bruits blancs de moyenne nulle et de matrice de covariance respective Q et R.

On notera par x_k^R la trajectoire de référence définie par

$$x_{k+1}^R = f(x_k^R). \tag{2.12}$$

Cette équation aux différences finies a pour condition initiale x_0^R. On notera par δ_k l'erreur entre X_k et x_k^R définie par :

$$\delta_k = X_k - x_k^R.$$

L'idée du filtre de Kalman étendu consiste à chercher un modèle linéaire qui décrit le processus dynamique δ_k. En effet si on remplace dans l'équation précédente X_k par sa valeur on obtient :

$$\delta_{k+1} = f(X_k) + GW_k - f(x_k^R).$$

Pour obtenir une approximation linéaire de cette équation, on procède par un développement de Taylor de la fonction $f(x)$ aux voisinage de x_k^R.

$$f(X_k) \approx f(x_k^R) + A_k(X_k - x_k^R),$$

où A_k est le jacobien de f, i.e la matrice carrée définie par

$$A_k = \begin{pmatrix} \frac{\partial f_1(x)}{\partial x_1} & \frac{\partial f_1(x)}{\partial x_2} & \dots & \frac{\partial f_1(x)}{\partial x_n} \\ \frac{\partial f_2(x)}{\partial x_1} & \frac{\partial f_2(x)}{\partial x_2} & \dots & \frac{\partial f_2(x)}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_n(x)}{\partial x_1} & \frac{\partial f_n(x)}{\partial x_2} & \dots & \frac{\partial f_n(x)}{\partial x_n} \end{pmatrix}_{X=x_k^R} = (\nabla_x f)(x_k^R).$$

Avec cette définition on obtient

$$\begin{aligned} \delta_{k+1} &\approx f(x_k^R) + A_k(X_k - x_k^R) + GW_k - f(x_k^R) \\ &= A_k\delta_k + GW_k \end{aligned} \tag{2.13}$$

On remarque donc que l'équation (2.13) est linéaire.

On procède de la même façon pour l'approximation de l'équation (2.11). Un développement en série de Taylor de la fonction $h(x)$ aux voisinage de x_k^R donne :

$$h(X_k) \approx h(x_k^R) + H_k(X_k - x_k^R) + V_k,$$

où

$$H_k = \begin{pmatrix} \frac{\partial h_1(x)}{\partial x_1} & \frac{\partial h_1(x)}{\partial x_2} & \dots & \frac{\partial h_1(x)}{\partial x_n} \\ \frac{\partial h_2(x)}{\partial x_1} & \frac{\partial h_2(x)}{\partial x_2} & \dots & \frac{\partial h_2(x)}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial h_n(x)}{\partial x_1} & \frac{\partial h_n(x)}{\partial x_2} & \dots & \frac{\partial h_n(x)}{\partial x_n} \end{pmatrix}_{X=x_k^R}.$$

On peut donc écrire :

$$Y_k \approx h(x_k^R) + H_k(X_k - x_k^R) + V_k.$$

On pose :

$$\begin{aligned} \nu_k &= Y_k - h(x_k^R) \\ &= h(X_k) + V_k - h(x_k^R) \\ &= H_k(X_k - x_k^R) + V_k \\ &= H_k\delta_k + V_k. \end{aligned} \tag{2.14}$$

Ainsi nous avons une relation linéaire entre ν_k et δ_k, les valeurs de ν_k sont calculées à partir des observations Y_k.

En partant de l'hypothèse que δ_k satisfait les équations (2.13) et (2.14) on peut appliquer le filtre de Kalman pour calculer $\widehat{\delta}_{k+1|k}$, $P_{k+1|k}$, $\widehat{\delta}_{k+1|k+1}$ et $P_{k+1|k+1}$.

Une fois $\widehat{\delta}_{k+1|k}$ et $\widehat{\delta}_{k+1|k+1}$ estimées, on calcule une estimation de X_{k+1} de la façon suivante :

$$\begin{aligned} \widehat{X}_{k+1|k} &= x_{k+1}^R + \widehat{\delta}_{k+1|k}, \\ \widehat{X}_{k+1|k+1} &= x_{k+1}^R + \widehat{\delta}_{k+1|k+1}. \end{aligned}$$

Pour la valeur initial de x_0^R, on peut la supposer égale à p_0, la moyenne de la variable X_0. On peut résumer le filtre de Kalman étendu par les deux étapes de prédiction et de correction suivantes:

1ère étape : équations de prédiction.

$$\begin{aligned} \widehat{X}_{k+1|k} &= f(\widehat{X}_{k|k}), \\ P_{k+1|k} &= A_k P_{k|k} A_k^T + GQG^T, \\ A_k &= \frac{\partial f(x,t)}{\partial x}|_{x=\widehat{X}_{k|k}} = \nabla_x f(\widehat{X}_{k|k}). \end{aligned}$$

2ème étape : équations de correction.

$$H_{k+1} = \frac{\partial h(x,t_k)}{\partial x}|_{x=\widehat{X}_{k+1|k}} = \nabla_x H(\widehat{X}_{k+1|k}),$$

$$
\begin{aligned}
K_{k+1} &= P_{k+1|k} H_{k+1}^T (H_{k+1} P_{k+1|k} H_{k+1}^T + R)^{-1}, \\
\widehat{X}_{k+1|k+1} &= \widehat{X}_{k+1|k} + k_{K+1}[y_{K+1} - h(\widehat{X}_{k+1|k}, k+1)], \\
P_{k+1|k+1} &= (I - K_{k+1} H_{k+1}) P_{k+1|k}.
\end{aligned}
$$

2.2.3 Filtre IMM (Interacting Multiple models)

Dans cette section, nous présentons une autre méthode de filtrage, le filtre IMM (Interacting multiple Modeles). Cette approche fait partie du groupe des filtres MM (modèles multiples) qui sont récemment très populaires surtout dans le domaine de poursuite de cible. La technique pour l'estimation du vecteur d'état est basée sur la combinaison d'un nombre fini de sous modèles, auxquels on applique simultanément le filtre de Kalman. La description de cette technique est couverte par plusieurs références [1,2] et nous allons simplement donner les différentes étapes de son algorithme.

Algorithme du filtre IMM:

On considère $x_k \in \mathbf{R}^n$ le vecteur état et $y_k \in \mathbf{R}^r$ le vecteur observation qui sont modélisés par les équation d'état et de mesure suivante :

$$
\begin{aligned}
x_{k+1} &= F(M_{k+1}) x_k + G(M_{k+1}) v_k, && (2.15) \\
y_k &= H x_k + w_k. && (2.16)
\end{aligned}
$$

$v_k \in \mathbf{R}^m$ et $w_k \in \mathbf{R}^r$ sont respectivement les bruits de l'équation d'état et de mesure, on suppose qu'ils sont des bruits blanc de moyenne nul et de matrice de covariance respectives Q_k et R_k . Le système, à un instant donnée k, est décris par un modèle parmi q modèles possible suivant la valeur du paramètre $M_k \in \{1, 2, ..q\}$. $M_k = i$ dénote que le système suit le $i$ème modèle à l'instant

k. Nous supposerons par la suite que le processus $(M_k)_{k\geq 1}$ est une chaîne de Markov de probabilités initiales $\mu_i = \mathbf{P}(M_0 = i)$ et de probabilités de transitions $p_{ij} = \mathbf{P}(M_k = j | M_{k-1} = i)$, pour $i, j = 1, 2, ..., q$.

Dans un premier temps nous choisissons les états initiaux $\widehat{x}_0^i$, les matrices de covariances P_0^i, et les coefficients μ_0^i, $1 \leq i \leq m$. L'estimation de l'état et de la matrice de covariance du $j$ème modèle est calculé de la façon suivante :

$$P_{k-1|k-1}^{0j} = \sum_{i=1}^{m} \mu_{k-1}^{ij} (P_{k-1|k-1}^{i} + (\widehat{x}_{k-1|k-1}^{i} - \widehat{x}_{k-1|k-1}^{0j})(\widehat{x}_{k-1|k-1}^{i} - \widehat{x}_{k-1|k-1}^{0j})^T), \tag{2.17}$$

avec

$$\widehat{x}_{k-1|k-1}^{0j} = \sum_{i=1}^{m} \widehat{x}_{k-1|k-1}^{i} \mu_{k-1}^{ij}.$$

Les valeurs de μ^{ij} sont déterminées par l'équation suivante:

$$\mu_{k-1}^{ij} = \frac{p_{ij} \mu_{k-1}^{i}}{\overline{c}_{k-1}^{j}},$$

avec : $\overline{c}_{k-1}^{j} = \sum_{i=1}^{m} p_{ij} \mu_{k-1}^{i}$.

Après avoir choisi les valeurs initiales μ_0^i le calcul a posteriori des probabilités μ_k^i est donné par

$$\mu_k^i = \frac{\overline{c}_k^i \Lambda_k^i}{c_k},$$

avec

$$\begin{aligned} c_k &= \sum_{i=1}^{m} \Lambda_k^i \overline{c}_k^i, \\ \Lambda_k^i &\equiv \frac{e^{\frac{1}{2}(y_k - H\widehat{x}_{k|k-1}^i)^T (S_k^i)^{-1} (y_k - H\widehat{x}_{k|k-1}^i)}}{\sqrt{2\pi |S_k^i|}}, \\ S_k^i &= H P_{n|n-1}^i H_T + R. \end{aligned}$$

Les S_k sont les covariances des innovations du processus. Étant donné les valeurs précédentes on applique les équations du filtre de Kalman:

$$\begin{aligned} P^j_{k|k-1} &= F(j)P^{0j}_{k-1|k-1}F(j)^T + GQG^T, \\ K_k(j) &= P^{0j}_{k|k-1}H^T(HP^{0j}_{k|k-1}H^T + R)^{-1}, \\ \widehat{x}^j_{k|k} &= F(j)\widehat{x}^{0j}_{k-1|k-1} + K_k(j)(y_k - H\widehat{x}^{0j}_{k|k-1}), \\ P^j_{k|k} &= (I - K(j)_k H)P^{0j}_{k-1|k-1}. \end{aligned} \tag{2.18}$$

Finalement l'estimation du vecteur d'état ainsi que la matrice de covariance consiste à poser

$$\begin{aligned} \widehat{x}_{k|k} &= \sum_{i=1}^{m} \mu^i_k \widehat{x}^i_{k|k}, \\ P_{k|k} &= \sum_{i=1}^{m} \mu^i_k \{P^i_{k|k} + (\widehat{x}^i_{k|k} - \widehat{x}_{k|k})(\widehat{x}^i_{k|k} - \widehat{x}_{k|k})^T\}. \end{aligned}$$

2.2.4 Conclusion

Les filtres de Kalman, Kalman Étendu et IMM nous permettent donc d'estimer, d'une manière finie, le vecteur d'état $\widehat{X}_{t|t}$ ainsi que la matrice de covariance $P_{t|t}$. Bien que ces algorithmes soient les méthodes de filtrage les plus utilisées, leur efficacité est limitée à un certains cas d'observations très précises. Cependant lorsque le système s'éloigne trop d'un modèle linéaire ou si le filtre est mal initialisé, ces méthodes donnent de mauvais résultats.

2.3 Solution générale du problème de filtrage.

Dans le cas général, le problème de filtrage consiste à estimer le vecteur d'état $X = X_t$ au vu des observations $\{Y_\tau, t_0 \leq \tau \leq t\}$. Le vecteur d'état évolue soit en temps discret ou continu suivant une équation différentielle stochastique d'Itô.
Avant de donner la solution générale du filtrage, nous introduisons quelques notations que nous utiliserons par la suite.

Notations : *Nous notons par $\mathcal{Y}_k$ la σ-algèbre engendrée par les variables aléatoires $\{Y_\tau, t_0 \leq \tau \leq k\}$, i.e*

$$\mathcal{Y}_k = \sigma\{Y_\tau, t_0 \leq \tau \leq k\}.$$

Étant donné deux vecteurs aléatoires X et Y, nous désignons par $p_{X|Y}(\cdot|\cdot)$ la densité conditionnelle de la variable aléatoire X étant donnée Y définie par:

$$p_{X|Y}(x|y) = \mathbf{P}(X = x|Y = y).$$

Nous notons aussi par $p_{X|\mathcal{Y}_k}(\cdot)$ la densité conditionnelle de la variable X conditionnée par rapport à la tribu $\mathcal{Y}_k$.

Au sens du risque quadratique, la meilleur estimée de X_t étant donnée $\mathcal{Y}_\tau$ est donnée par la moyenne conditionnelle $E(X_t|\mathcal{Y}_\tau)$. Le calcul optimal de cette statistique ne peut se faire qu'en calculant la loi conditionnelle $p_{X_t|\mathcal{Y}_\tau}(\cdot)$. L'ensemble de ces densités conditionnelles constituent ce qu'on appelle filtre optimal. Le problème de filtrage est totalement résolu par le calcul de

ces densités. Dans le cas du filtrage linéaire, la densité $p_{X_t|\mathcal{Y}_\tau}(\cdot)$ est gaussienne, ainsi elle est caractérisée par sa moyenne et sa matrice de covariance et donc le problème de filtrage est résolu par le calcul de ces deux paramètres. Cependant, dans le cas du filtrage non linéaire, le problème est plus difficile. En général le problème est de dimension infinie, du fait qu'il n'existe pas un nombres finis de paramètres qui caractérisent la densité $p_{X_t|\mathcal{Y}_\tau}(\cdot)$. Dans cette section nous donnerons la solution générale du filtrage (non linéaire et linéaire) et nous allons limiter notre étude au cas où le vecteur d'état évolue dans le temps de manière continue et les observations sont discrètes.

2.3.1 Filtrage non linéaire

L'équation de la dynamique du système dans le cas continu/discret est l'équation différentielle stochastique (2.1), i.e.

$$dX_t = b(X_t)dt + \sigma(X_t)dW_t.$$

Le processus X_t est liée aux observations Y_k par l'équation (2.2), i.e.

$$Y_k = h(X_{t_k}) + v_k.$$

Nous supposerons par la suite que $(W_t)_{t\geq 0}$ est un mouvement brownien standard de dimension d_1 indépendant de la condition initiale X_{t_0}, les fonctions b, σ et h sont des fonctions vectorielles définies sur $\mathbf{R}^d$ à valeur respectivement dans $\mathbf{R}^d$, $\mathbf{R}^{d\times d_1}$ et $\mathbf{R}^r$. La suite $(V_k) \geq 1$ dans l'équation (2.2) est un bruit blanc indépendant de W_t et de X_0. La loi de probabilité $p(x)$ de la condition

initiale X_0 est supposée être connue. Étant donné que le processus X_t, solution de l'équation (2.1), est un processus de Markov (A.H. Jazwinski 1970, G. Kallianpur 1980), alors X_t est complètement déterminé par la connaissance de la probabilité conditionnelle $p_{X_t|X_{t'}}(.|.)$ et la densité de la probabilité de transition $p_{X_t|\mathcal{Y}_k}(.)$.

L'algorithme proposé pour résoudre le problème de filtrage doit satisfaire à des contraintes de temps réel. Pour cette raison on souhaite obtenir un algorithme récursif pour le calcul de $p_{x_{t_k}|\mathcal{Y}_k}(\cdot)$, i.e. tel que le calcul ne soit fonction que de la dernière observation Y_k et de la loi conditionnelle précédente $p_{X_{t_{k-1}}|\mathcal{Y}_{k-1}}(\cdot)$ sans avoir à utiliser les observations entre les instants $t = t_0$ et t_{k-1}. Le passage de $p_{X_{t_{k-1}}|\mathcal{Y}_{k-1}}(\cdot)$ à $p_{X_{t_k}|\mathcal{Y}_k}(.)$ fait intervenir la loi conditionnelle de transition $p_{X_{t_k}|\mathcal{Y}_{k-1}}(\cdot)$ que l'on peut calculer si l'on connaît $p_{X_t|X_{t_{k-1}}}(x_t, t|\xi)$ pour tout $t \in [t_{k-1}, t_k[$.

En effet on a pour tout $t \in [t_{k-1}, t_k[$, à cause de l'indépendance des sources de bruit,

$$p_{X_t|\mathcal{Y}_{k-1}}(x_t, t) = \int p_{X_t|X_{t_{k-1}}}(x_t, t|\xi) p_{X_{t_{k-1}}|\mathcal{Y}_{k-1}}(\xi) d\xi. \tag{2.19}$$

Étant donné que le processus X_t est généré par l'équation différentielle d'Itô (2.1), alors l'évolution de la densité $p_{X_t|X_{t_{k-1}}}$ est décrite par l'équation de Kolmogorov (e.g. Jazwinski, 1970, page 126-130) :

$$\frac{\partial p_{X_t|X_{t_{k-1}}}(x_t, t)}{\partial t} = L^*(p_{X_t|X_{t_{k-1}}}(x_t, t)), \tag{2.20}$$

où L est le générateur infinitésimal du processus de diffusion X_t, défini par

$$L = \sum_{i=1}^{d_1} b_i \frac{\partial}{\partial x_i} + \frac{1}{2} \sum_{i,j=1}^{d_1} a_{ij} \frac{\partial^2}{\partial x_i \partial x_j}. \tag{2.21}$$

avec $a = \sigma\sigma^T$. L^* est l'opérateur adjoint de L, i.e.

$$L^* f = -\sum_{i=1}^{d_1} \frac{\partial}{dx_i}(b_i f) + \frac{1}{2}\sum_{i,j} \frac{\partial^2}{\partial x i \partial x_j}(a_{i,j} f).$$

Par l'application de la formule de Bayes, on obtient l'équation qui donne la probabilité conditionnelle $p_{X_{t_k}|\mathcal{Y}_k}$.

$$p_{X_{t_k}|\mathcal{Y}_k}(x) = \frac{p_{X_{t_k}|\mathcal{Y}_{k-1}}(x, t_k) p_{Y_k|X_{t_k}}(y|x)}{P_{Y_k|\mathcal{Y}_{k-1}}(y)}, \tag{2.22}$$

avec

$$p_{Y_k|\mathcal{Y}_{k-1}}(y) = \int p_{X_{t_k}|\mathcal{Y}_{k-1}}(w, t_k) p_{Y_k|X_{t_k}}(y|w) dw. \tag{2.23}$$

Cette dernière équation est très importante car elle permet de calculer la fonction de vraisemblance de $Y_1, ...Y_k$ qui sert à estimer les paramètres.
Si V_k de l'équation (2.2) est un bruit blanc gaussien de matrice de covariance R_k alors

$$p_{Y_k|X_{t_k}}(y|x) = \frac{1}{(2\pi)^{\frac{r}{2}}|R_k|^{\frac{1}{2}}} \exp\{-\frac{1}{2}(y - h(x))^T R_k^{-1}(y - h(x))\}. \tag{2.24}$$

L'algorithme proposé pour la résolution générale du problème du filtrage non linéaire est composé de trois étapes.

Première étape: Initialisation de la densité $p_{X_0|\mathcal{Y}_0}$.

Deuxième étape: Calcul de la densité par la formule $p_{X_{t_k}|\mathcal{Y}_{k-1}}$.

$$p_{X_{t_k}|\mathcal{Y}_{k-1}}(x_{t_k}) = \int p_{X_{t_k}|X_{t_{k-1}}}(x_{t_k}, t_k|\xi) p_{X_{t_{k-1}}|\mathcal{Y}_{k-1}}(\xi) d\xi.$$

Troisième étape: À chaque instant k on détermine la densité $p_{X_{t_k}|\mathcal{Y}_k}$ à l'aide de l'équation

$$p_{X_{t_k}|\mathcal{Y}_k}(x) = \frac{p_{X_{t_k}|\mathcal{Y}_{k-1}}(x) p_{Y_k|X_{t_k}}(y|x)}{P_{Y_k|\mathcal{Y}_{k-1}}(y)}.$$

2.3.2 Filtrage linéaire

Le cas du filtrage linéaire est un cas particulier du problème du filtrage non linéaire. Le vecteur état dans le cas linéaire évolue suivant l'équation différentielle suivante :

$$dX_t = b(t)X_t dt + \sigma(t)dW_t, \tag{2.25}$$

où X_t est un vecteur de dimension d_1, b et σ sont des fonctions matricielle continues non aléatoires de dimension $d_1 \times d_1$ et $d_1 \times d$ respectivement, et $(W_t)_{t \geq t_0}$ est un mouvement brownien standard de dimension d. Les observations $(Y_k)_{k \geq 0}$ sont liées au vecteur état par l'équation de mesure linéaire suivante :

$$Y_k = H_k X_{t_k} + V_k, \tag{2.26}$$

où Y_k est vecteur de dimension r, H_k est une fonction matricielle de temps non aléatoire de dimension $r \times d_1$ et $\{V_k, k = 1, 2, ...\}$ est un bruit blanc gaussien de dimension r. On supposera que $V_k \sim N(0, R_k)$ avec R_k définie positive. Les distributions de X_{t_0}, $\{W_t\}$ et $\{V_k\}$ sont supposées indépendantes.
À chaque instant t_k, et à l'obtention de la mesure $Y_k = y_k$ on désire déterminer la densité conditionnelle $p_{X_{t_k}|\mathcal{Y}_k}$.
Or on sait que la loi de probabilité de X_{t_k} étant donnée les observations $\{Y_i, 0 \leq i \leq k\}$ est gaussienne, et donc caractérisée par sa moyenne et sa matrice de covariance.
Notons par $\widehat{X}_{t_k|t_k}$ sa moyenne et par $P_{t_k|t_k}$sa matrice de covariance. Le problème de filtrage consistera donc à calculer ces deux valeurs pour chaque instant t_k.

L'équation (2.21) nous donne l'expression de la densité $p_{X_{t_k}|\mathcal{Y}_k}$:

$$p_{X_{t_k}|\mathcal{Y}_k}(x) = \frac{p_{X_{t_k}|\mathcal{Y}_{k-1}}(x) p_{Y_k|X_{t_k}}(y|x)}{P_{Y_k|\mathcal{Y}_{k-1}}(y)}.$$

Dans le cas linéaire toutes les densités dans l'équation (2.21) sont gaussiennes. On pose

$$p_{X_{t_k}|\mathcal{Y}_{k-1}} \sim N(\widehat{X}_{t_k|t_{k-1}}, P_{t_k|t_{k-1}}).$$

D'après l'équation (2.22) la densité $p_{Y_k|X_{t_k}}$ est donnée par :

$$p_{Y_k|X_{t_k}}(y_k|x) = \frac{1}{(2\pi)^{\frac{r}{2}}|R_k|^{\frac{1}{2}}} \exp\{-\frac{1}{2}(y_k - H_k x)^T R_k^{-1}(y_k - H_k x)\}.$$

On montre aussi que :

$$\mathbf{E}(Y_k|\mathcal{Y}_{k-1}) = H_k \widehat{X}_{t_k|t_{k-1}},$$

et

$$\mathbf{E}\{(Y_k - \mathbf{E}(Y_k|\mathcal{Y}_{k-1}))(Y_k - \mathbf{E}(Y_k|\mathcal{Y}_{k-1}))^T|\mathcal{Y}_{k-1}\} = H_k P_{t_k|t_{k-1}} H_k^T + R_k.$$

De ce qui précède et par des calculs matriciels on en déduit l'expression de $\widehat{X}_{t_k|t_k}$ et de $P_{t_k|t_k}$:

$$\begin{aligned} \widehat{X}_{t_k|t_k} &= (H_k^T R_k^{-1} H_k + P_{t_k|t_{k-1}}^{-1})^{-1}(H_k^T R_k^{-1} y_k + P_{t_k|t_{k-1}}^{-1} \widehat{X}_{t_k|t_{k-1}}), \\ P_{t_k|t_k}^{-1} &= P_{t_k|t_{k-1}}^{-1} + H_k^T R_k^{-1} H_k. \end{aligned} \tag{2.27}$$

Pour simplifier l'écriture de l'équation (2.27) nous rappelons quelques résultats du calcul matriciel.

Soit P, R, et H des matrice d'ordre $d_1 \times d_1$, $r \times r$ et $r \times d_1$ respectivement. Si P est positive et R définie positive alors on a les résultats suivant :

$$(I + PH^T R^{-1} H)^{-1} = I - PH^T(HPH^T + R)^{-1}H, \tag{2.28}$$

$$(I + PH^T R^{-1} H)^{-1} P = P - PH^T(HPH^T + R)^{-1}HP, \tag{2.29}$$

$$(I + PH^T R^{-1} H)^{-1} PH^T R^{-1} = PH^T(HPH^T + R)^{-1}. \tag{2.30}$$

Si de plus P est définie positive alors

$$(I + PH^T R^{-1} H)^{-1} P = (P^{-1} + H^T R^{-1} H)^{-1}. \tag{2.31}$$

En utilisant la relation (2.31) dans (2.30) et (2.29) on obtient les relations suivantes

$$\begin{aligned} (P^{-1} + H^T R^{-1} H)^{-1} &= P - PH^T (HPH^T + R)^{-1} HP, \quad (2.32) \\ (P^{-1} + H^T R^{-1} H)^{-1} H^T R^{-1} &= PH^T (HPH^T + R)^{-1}. \quad (2.33) \end{aligned}$$

Or l'application des relations (2.32) et (2.33) dans l'équation (2.27) nous donne

$$\begin{aligned} \widehat{X}_{t_k|t_k} &= \widehat{X}_{t_k|t_{k-1}} + P_{t_k|t_{k-1}} H_k^T [H_k P_{t_k|t_{k-1}} H_k^T + R_k]^{-1} (y_k - H_k \widehat{X}_{t_k|t_{k-1}}). \\ P_{t_k|t_k} &= P_{t_k|t_{k-1}} - P_{t_k|t_{k-1}} H_k^T [H_k P_{t_k|t_{k-1}} H_k^T + R_k]^{-1} H_k P_{t_k|t_{k-1}}. \end{aligned} \tag{2.34}$$

Si on pose

$$K_k = P_{t_k|t_{k-1}} H_k^T [H_k P_{t_k|t_{k-1}} H_k^T + R_k]^{-1},$$

l'équation devient

$$\begin{aligned} \widehat{X}_{t_k|t_k} &= \widehat{X}_{t_k|t_{k-1}} + K_k (y_k - H_k \widehat{X}_{t_k|t_{k-1}}). \\ P_{t_k|t_k} &= P_{t_k|t_{k-1}} - K_k H_k P_{t_k|t_{k-1}} \end{aligned} \tag{2.35}$$

La dernière équation nous donne le célèbre filtre de Kalman-bucy, la matrice K_k représente le gain de Kalman.

2.4 Estimation par la méthode du maximum de vraisemblance.

Dans chacune des solutions du problème de filtrage, les paramètres du modèle doivent être connus à l'avance, ce qui n'est pas le cas dans la plupart des problèmes, alors on doit les estimer.
Parmi les méthodes d'estimation des paramètres les plus utilisées, on trouve la méthode du maximum de vraisemblance.

La théorie classique du maximum de vraisemblance est basée sur la densité jointe des observations $Y_1, Y_2, ..., Y_n$. Dans le cas où les observations sont indépendantes et uniformément distribués, la densité jointes est donnée par

$$L(y, \theta) = \prod_{i=1}^{n} p(y_i), \tag{2.36}$$

où $p(y_i)$ est la densité des variables observation et θ est le vecteurs des paramètres. La fonction $L(y, \theta)$ est interprétée comme la fonction de vraisemblance. Une maximisation de cette fonction par rapport à θ nous permet de trouver les valeurs des paramètres à estimer.
Dans la plupart des modèles les variables d'observations ne sont pas indépendantes, et donc la formule (2.35) n'est pas applicable. Dans ce cas on utilise les densités conditionnelles pour définir la densité jointe. Elle donnée par

$$L(y, \theta) = \prod_{k=1}^{N} p_{Y_k|\mathcal{Y}_{k-1}}(y_k), \tag{2.37}$$

où $p_{Y_k|\mathcal{Y}_{k-1}}$ est donnée par l'équation (2.23).

L'idée donc pour l'estimation des paramètres consiste à maximiser L par rapport à θ.

2.5 Conclusion

Tout au long de ce chapitre nous avons vu que la résolution du problème de filtrage (non linéaire et linéaire) revient à déterminer à chaque instant l'espérance conditionnelle de la variable d'état au vue des observations antérieures. Nous avons remarqué que cette statistique ne peut être calculée que par la détermination de la loi conditionnelle de la variable d'état dont l'évolution est décrite par l'équation différentielle stochastique de Fokker-Plank. L'algorithme proposé pour la résolution du problème de filtrage est en général difficile à réaliser en temps réel du fait qu'il n'existe pas un nombre fini de paramètres caractérisant la densité conditionnelle comme c'est le cas pour le filtrage linéaire. Une autre raison, c'est qu'à chaque étape de l'algorithme on doit résoudre l'équation de Fokker-Plank. Pour faire face à cette difficulté, on procède par des méthodes d'approximation de la densité conditionnelle.
Au prochain chapitre, nous allons présenter une méthode pour l'approximation de la densité conditionnelle, basée sur la séparation des paramètres déterministes et stochastiques dans l'équation de Fokker-Planck.

Chapitre 3

Approximation par l'expansion en chaos de Wiener

On considère le problème de filtrage dans lequel le vecteur d'état est un processus de Markov, continu par rapport au temps, est estimé à partir d'un processus d'observations bruité qui est mesuré d'une manière discrète. Notre objectif est de trouver un algorithme récursif qui nous permet d'obtenir la meilleur estimation au sens des moindres carrés du vecteur d'état. Dans le cas linéaire la solution est donné par le filtre de Kalman. Soient $\{X_t\}_{t\geq 0}$ le processus d'état et $y(k)$ les observations mesurées aux instant t_k, si l'on considère une fonction $f(x)$ tel que $\mathbf{E} \mid f(X_t) \mid^2 < \infty$ pour tout $t \geq 0$, alors il est bien connu (G. Kallianpur 1980, R.S. Lipster and A.N. Shiryayev 1978) que sous certaines conditions la meilleure estimée $\widehat{f}(k)$ de $f(X_{t_k})$ au sens quadratique étant donnée les observations $\{y(i), i \leq k\}$ est donnée par

l'équation suivante :

$$\widehat{f}(k) = \frac{\int f(x)p_k(x)dx}{\int p_k(x)dx},$$

La fonction $p_k(x)$, appelée densité de filtrage non normalisée(UFD). Le calcul de $\widehat{f}(k)$ est donc réduit à la détermination de $p_k(x)$. Dans la plupart des cas il est difficile de calculer $p_k(x)$ d'une manière exacte, on procède, alors par des méthodes numériques pour son approximation. Dans ce chapitre nous étudierons une méthode de filtrage non linéaire qui utilise l'expansion en chaos de Wiener pour approximer la densité de filtrage non normalisée.

3.1 Approximation par la méthode de l'expansion en chaos de Wiener.

Soit $(\Omega, \mathcal{F}, P)$ un espace de probabilité et $(W_t)_{t\geq 0}$ un mouvement brownien standard de dimension d_1 défini sur cette espace [voir l'appendice pour la définition du mouvement brownien]. On considère un processus d'état X_t évoluant suivant l'équation différentielle stochastique d'Itô :

$$dX_t = b(X_t)dt + \sigma(X(t))dW_t. \tag{3.1}$$

On désire estimer X_t étant donné les vecteurs observations y_k prises sur l'intervalle de temps [0,T] et modélisés par l'équation

$$y(k) = h(X(t_k)) + V_k, \tag{3.2}$$

où $(W_t)_{t\geq 0}$ est indépendant de la condition initiale X_0, les fonctions b, σ et h sont des fonctions vectorielles définies sur $\mathbf{R}^d$ à valeur respectivement dans

$\mathbf{R}^d$, $\mathbf{R}^{d\times d_1}$ et $\mathbf{R}^r$. $(V_k)_{k\geq 1}$ est une suite de vecteurs aléatoires gaussiennes indépendantes et identiquement distribuées de moyenne zéro et de matrice de covariance $(\frac{1}{\Delta})I$, I étant la matrice identité. On suppose par la suite les hypothèses de régularités suivantes:

1. $(V_k)_{k\geq 1}$ est indépendante de $\{W_t\}_{t\geq 0}$ et de X_0.
2. Les fonctions b, σ, et h sont de classe C^∞ dont toutes les dérivées sont bornées.
3. Le vecteur aléatoire X_0 a pour densité $p = p(x)$ où p est une fonction de classe C^∞.

Notons par T_t^* l'opérateur solution de l'équation de Fokker-Planck liée au processus d'état X_t. En effet $u(t,x) = T_t^*\varphi(x)$ si

$$\frac{\partial u(t,x)}{\partial t} = L^* u(t,x), \tag{3.3}$$

où L est le générateur infinitésimal du processus de diffusion X_t, défini par

$$L = \sum_{i=1}^{d_1} b_i \frac{\partial}{\partial x_i} + \frac{1}{2}\sum_{i,j=1}^{d_1} a_{ij}\frac{\partial^2}{\partial x_i \partial x_j},$$

avec $a = \sigma\sigma^*$. L^* est l'opérateur adjoint de L.
Soit $0 = t_0 < t_1 < ... < t_M = T$ une partition de l'interval de temps $[0,T]$ avec $\Delta = t_i - t_{i-1}$ pour tout $i = 1, ..., M$. La densité conditionnelle $p_{X_{t_k}|\mathcal{Y}_k}$ est alors définie de la façon suivante :

$$p_{X_{t_k}|\mathcal{Y}_k}(x) = \frac{p_k(x)}{\int p_k(\xi)d\xi},$$

avec

$$\begin{aligned} p_0(x) &= p(x) \\ p_k(x) &= \exp(\Delta \sum_{l=1}^{r} h_l(x) y_l(k) - \frac{\Delta}{2} \sum_{l=1}^{r} h_l^2(x)) T_\Delta^* p_{k-1}(x). \end{aligned} \tag{3.4}$$

En effet pour tout $k \geq 1$ les équations (2.22) et (2.23) donnent

$$p_{X_{t_k}|\mathcal{Y}_k}(x) = \frac{p_{Y_k|X_{t_k}}(y(k)|x) p_{X_{t_k}|\mathcal{Y}_{k-1}}(x, t_k)}{\int p_{Y_k|X_{t_k}}(y(k)|\xi) p_{X_{t_k}|\mathcal{Y}_{k-1}}(\xi, t_k) d\xi}. \tag{3.5}$$

Du fait que $V_k \sim N(0, \frac{1}{\Delta} I)$ pour tout $k \geq 1$ et d'après l'équation (2.24) on a

$$\begin{aligned} p_{Y_k|X_{t_k}}(y(k)|x) &= \frac{1}{(2\pi)^{\frac{r}{2}} |\frac{1}{\Delta} I|^{\frac{1}{2}}} \exp\{-\frac{1}{2}(y(k) - h(x))^T (\frac{1}{\Delta} I)^{-1} (y(k) - h(x))\} \\ &= \frac{1}{(2\pi)^{\frac{r}{2}} |\frac{1}{\Delta} I|^{\frac{1}{2}}} \exp\{-\frac{\Delta}{2}(y(k) - h(x))^T (y(k) - h(x))\} \\ &= \frac{1}{(2\pi)^{\frac{r}{2}} |\frac{1}{\Delta} I|^{\frac{1}{2}}} \exp\{-\frac{\Delta}{2} \sum_{l=1}^{r} (y_l(k) - h_l(x))^2\}. \end{aligned}$$

En remplaçant $p_{Y_k|X_{t_k}}(y(k)|x)$ par sa valeur dans l'équation (3.5) on obtient

$$p_{X_{t_k}|\mathcal{Y}_k}(x) = \frac{\exp\{\Delta \sum_{l=1}^{r} y_l(k) h_l(x) - \frac{\Delta}{2} \sum_{l=1}^{r} h_l^2(x)\} p_{X_{t_k}|\mathcal{Y}_{k-1}}(x, t_k)}{\int \exp\{\Delta \sum_{l=1}^{r} y_l(k) h_l(\xi) - \frac{\Delta}{2} \sum_{l=1}^{r} h_l^2(\xi)\} p_{X_{t_k}|\mathcal{Y}_{k-1}}(\xi, t_k) d\xi}. \tag{3.6}$$

Or $\forall t \in [t_{k-1}, t_k[$ on a :

$$\frac{\partial p_{X_{t_k}|\mathcal{Y}_{k-1}}(x, t)}{\partial t} = L^*(p_{X_{t_k}|\mathcal{Y}_{k-1}}(x, t)),$$

avec la condition initiale

$$p_{X_{t_k}|\mathcal{Y}_{k-1}}(x, t_k) = p_{X_{t_{k-1}}|\mathcal{Y}_{k-1}}(x).$$

Remarques 3.1.1 *Si u est solution de l'équation différentielle suivante :*

$$\begin{aligned} \frac{\partial u(t,x)}{\partial t} &= L^* u(t,x) \ \forall t \in]t_{k-1}, t_k[, \\ u(t_{k-1}, x) &= \varphi(x), \end{aligned}$$

alors on a pour tout $t \in [t_{k-1}, t_k]$

$$u(t,x) = T^*_{t-t_{k-1}} \varphi(x).$$

En effet si on pose $v(t,x) = u(t_{k-1} + t, x)$ alors on a

$$\begin{aligned} \frac{\partial v(t,x)}{\partial t} &= L^* v(t,x) \ \forall t \in]0, \Delta[, \\ u(0,x) &= \varphi(x), \end{aligned}$$

on en déduit que

$$v(t,x) = T^*_t \varphi(x).$$

Ainsi, pour tout $t \in [t_{k-1}, t_{t_k}]$,

$$u(t,x) = v(t - t_{k-1}, x) = T^*_{t-t_{k-1}} \varphi(x).$$

En particulier on a pour tout $t \in [t_{k-1}, t_k]$,

$$p_{X_{t_k}|\mathcal{Y}_{k-1}}(x, t_k) = T^*_\Delta p_{X_{t_{k-1}}|\mathcal{Y}_{k-1}}(x).$$

Par substitution dans l'équation (3.6) on obtient

$$p_{X_{t_k}|\mathcal{Y}_k}(x) = \frac{\exp\{\Delta \sum_{l=1}^{r} y_l(k) h_l(x) - \frac{\Delta}{2} \sum_{l=1}^{r} h_l^2(x)\} T^*_\Delta p_{X_{t_{k-1}}|\mathcal{Y}_{k-1}}(x)}{\int \exp\{\Delta \sum_{l=1}^{r} y_l(k) h_l(\xi) - \frac{\Delta}{2} \sum_{l=1}^{r} h_l^2(\xi)\} T^*_\Delta p_{X_{t_{k-1}}|\mathcal{Y}_{k-1}}(\xi) d\xi}.$$

Pour $k = 1$ on a :

$$\begin{aligned} p_{X_{t_1}|\mathcal{Y}_1}(x) &= \frac{\exp\{\Delta \sum_{l=1}^{r} y_l(1)h_l(x) - \frac{\Delta}{2}\sum_{l=1}^{r} h_l^2(x)\}T_\Delta^* p_{X_{t_0}|\mathcal{Y}_0}(x)}{\int \exp\{\Delta \sum_{l=1}^{r} y_l(1)h_l(\xi) - \frac{\Delta}{2}\sum_{l=1}^{r} h_l^2(\xi)\}T_\Delta^* p_{X_{t_0}|\mathcal{Y}_0}(\xi)d\xi} \\ &= \frac{\exp\{\Delta \sum_{l=1}^{r} y_l(1)h_l(x) - \frac{\Delta}{2}\sum_{l=1}^{r} h_l^2(x)\}T_\Delta^* p_0(x)}{\int \exp\{\Delta \sum_{l=1}^{r} y_l(1)h_l(\xi) - \frac{\Delta}{2}\sum_{l=1}^{r} h_l^2(\xi)\}T_\Delta^* p_0(\xi)d\xi} \\ &= \frac{p_1(x)}{\int p_1(\xi)d\xi}. \end{aligned}$$

Supposons que

$$p_{X_{t_{k-1}}|\mathcal{Y}_{k-1}}(x) = \frac{p_{k-1}(x)}{\int p_{k-1}(\xi)d\xi}.$$

Or on a

$$\begin{aligned} p_{X_{t_k}|\mathcal{Y}_k}(x) &= \frac{\exp\{\Delta \sum_{l=1}^{r} y_l(k)h_l(x) - \frac{\Delta}{2}\sum_{l=1}^{r} h_l^2(x)\}T_\Delta^* p_{X_{t_{k-1}}|\mathcal{Y}_{k-1}}(x)}{\int \exp\{\Delta \sum_{l=1}^{r} y_l(k)h_l(\xi) - \frac{\Delta}{2}\sum_{l=1}^{r} h_l^2(\xi)\}T_\Delta^* p_{X_{t_{k-1}}|\mathcal{Y}_{k-1}}(\xi)d\xi} \\ &= \frac{\exp\{\Delta \sum_{l=1}^{r} y_l(k)h_l(x) - \frac{\Delta}{2}\sum_{l=1}^{r} h_l^2(x)\}T_\Delta^* \frac{p_{k-1}(x)}{\int p_{k-1}(\xi)d\xi}}{\int \exp\{\Delta \sum_{l=1}^{r} y_l(k)h_l(\xi) - \frac{\Delta}{2}\sum_{l=1}^{r} h_l^2(\xi)\}T_\Delta^* \frac{p_{k-1}(\xi)}{\int p_{k-1}(\xi)d\xi}d\xi} \\ &= \frac{\exp\{\Delta \sum_{l=1}^{r} y_l(k)h_l(x) - \frac{\Delta}{2}\sum_{l=1}^{r} h_l^2(x)\}T_\Delta^* p_{k-1}(x)}{\int \exp\{\Delta \sum_{l=1}^{r} y_l(k)h_l(\xi) - \frac{\Delta}{2}\sum_{l=1}^{r} h_l^2(\xi)\}T_\Delta^* p_{k-1}(\xi)d\xi} \\ &= \frac{p_k(x)}{\int p_k(\xi)d\xi}. \end{aligned}$$

Donc pour tout $k \geq 1$ on a

$$p_{X_{t_k}|\mathcal{Y}_k}(x) = \frac{p_k(x)}{\int p_k(\xi)d\xi}.$$

$p_k(x)$ est appelé densité de filtrage non normalisée. Soit $f = f(x)$, $x \in \mathbf{R}^d$ une fonction mesurable tel que $\mathbf{E} \mid f(X_t) \mid^2 < \infty$ pour tout $t \geq 0$. La meilleure estimation, au sens des moindres carrés, de $f(X_{t_k})$ étant donnée les observations $y(m)$, $m = 1, ..., k$ est donnée par

$$\widehat{f}(k) = \frac{\int f(x) p_k(x) dx}{\int p_k(x) dx},$$

on pose par la suite

$$\phi_k[f] = \int f(x) p_k(x) dx$$

et

$$\phi_k[1] = \int p_k(x) dx,$$

et donc

$$\widehat{f}(k) = \frac{\phi_k[f]}{\phi_k[1]}.$$

Notons par Γ l'ensemble des multi-indices $\gamma = \{\gamma_1, ..., \gamma_d\}$ où les γ_i sont des entiers positifs. Soit $\{e_l\}_{l \in \Gamma}$ la base d'Hermite de $\mathbf{L}_2(\mathbf{R}^d)$ définie par

$$e_l(x_1, ..., x_2) = \prod_{i=1}^{d} e_{l_i}(x_i), \tag{3.7}$$

où

$$e_n(t) = \frac{1}{\sqrt{2^n \pi^{\frac{1}{2}} n!}} e^{\frac{-t^2}{2}} H_n(t),$$

et

$$H_n(t) = (-1)^n e^{t^2} \frac{d^n}{dt^n} e^{-t^2}, \ n \geq 0.$$

$H_n(t)$ est le *n-ième* polynôme d'Hermite donné par l'équation suivante

$$H_n(t) = \sum_{j=0}^{[\frac{n}{2}]} (-1)^j \frac{n!}{j!(n-2j)!} (2t)^{n-2j},$$

où $[\cdot]$ désigne la partie entière. Chaque fonction g de l'espace $\mathbf{L}_2$

$$\mathbf{L}_2 = \mathbf{L}_2(\mathbf{R}^d, e^{-|x|^2}dx),$$

admet une expansion dans la base d'Hermite, de la forme

$$g(x) = \sum_{l\in\Gamma} g_l e_l(x),$$

où

$$g_l = \int g(x)e_l(x)dx \ \forall \ l \in \Gamma.$$

Or pour tout $k \geq 1$ $p_k \in \mathbf{L}_2$ et donc il existe une suite de coefficients $\{\Psi_l(k)\}_{l\in\Gamma}$ tel que

$$p_k(x) = \sum_{l\in\Gamma} \Psi_l(k)e_l(x), \tag{3.8}$$

où

$$\Psi_l(k) = \int p_k(x)e_l(x).$$

Pour $l \in \Gamma$ on définit $\mid l \mid = \sum_{i=1}^{d} l_i$ et pour un entier positif κ on dira que $l \leq \kappa$ ssi $\mid l \mid \leq \kappa$. Pour un κ on définit l'ensemble Γ_κ par $\Gamma_\kappa = \{l \in \Gamma :\mid l \mid \leq \kappa\}$, le nombre d'éléments de Γ_κ est $\frac{(\kappa+1)(\kappa+2)}{2}$.

La méthode d'approximation par l'expansion en chaos de Wiener de la densité non normalisée $p_k(x)$ consiste à procéder par une truncation de la somme dans l'équation(3.8). En effet pour un nombre entier positif κ que l'on choisi on approxime p_k par $\overline{p}_k$ avec

$$\overline{p}_k(x) = \sum_{l\leq\kappa} \Psi_l(k)e_l(x). \tag{3.9}$$

On en déduit les approximations $\overline{\phi}_k[f]$ de $\phi_k[f]$ et $\overline{\phi}_k[1]$ de $\phi_k[1]$

$$\overline{\phi}_k[f] = \sum_{l \leq \kappa} \Psi_l(k) \int f(x) e_l(x) dx,$$

et

$$\overline{\phi}_k[1] = \sum_{l \leq \kappa} \Psi_l(k) \int e_l(x) dx.$$

On suppose par la suite que la fonction f satisfait la condition suivante :

$$\mid f(x) \mid \leq C_f(1 + \mid x \mid^{\alpha}),\ x \in \mathbf{R}^d, \tag{3.10}$$

avec α et C_f des constantes réels strictement positives. Le théorème suivant dont on ne donnera pas la preuve assure la convergence des deux approximations.

Théorème 3.1.1 *: Pour tous $M > 0$ il existe des constantes positives A_p, B_p, C_p, $A_\phi(C_f, \alpha)$, $B_\phi(C_f, \alpha)$ et $C_\phi(\alpha)$ tel que:*

$$max_{1 \leq k \leq M} \mathbf{E} \parallel p_k - \overline{p}_k \parallel_{L_2(\mathbf{R}^d)} \leq A_p \Delta + \frac{B_p}{\sqrt{\Delta}} \exp(-C_p (\ln N)^2), \tag{3.11}$$

et

$$max_{1 \leq k \leq M} \mathbf{E} \mid \phi_k[f] - \overline{\phi}_k[f] \mid \leq A_\phi(C_f, \alpha)\Delta + \frac{B_\phi(C_f, \alpha)}{\sqrt{\Delta}} \exp(-C_\phi(\alpha)(\ln N)^2). \tag{3.12}$$

Toutes les constantes définies ci-dessus dépendent de $T = M\Delta$ et des paramètres du modèle.

Pour la preuve de ce théorème voir Lototsky et Rozovskii (1998). Les inégalités 3.11 et 3.12 montrent que les deux approximations convergent en moyenne lorsque Δ et κ tendent respectivement ver 0 et $+\infty$, pour M fixé.

3.2 Développement de l'algorithme

On a

$$\begin{aligned} p_k(x) &= \exp(\Delta \sum_{i=1}^{r} h_i(x) y_i(k) - \frac{\Delta}{2} \sum_{i=1}^{r} h_i^2(x)) T_\Delta^* p_{k-1}(x) \\ &= \exp(-\frac{\Delta}{2} \sum_{i=1}^{r} h_i^2(x)) \exp(\Delta \sum_{i=1}^{r} h_i(x) y_i(k)) T_\Delta^* p_{k-1}(x) \\ &= \exp(-\frac{\Delta}{2} \sum_{i=1}^{r} h_i^2(x)) \prod_{i=1}^{r} \exp(\Delta h_i(x) y_i(k)) T_\Delta^* p_{k-1}(x). \end{aligned}$$

Un développement de Taylor de p_k à l'ordre deux donne

$$\begin{aligned} p_k(x) &\approx \exp(-\frac{\Delta}{2} \sum_{i=1}^{r} h_i^2(x))(\prod_{i=1}^{r}(1 + \Delta h_i(x) y_i(k) + \frac{\Delta^2}{2} h_i^2(x) y_i^2(k))) T_\Delta^* p_{k-1}(x) \\ &\approx \exp(-\frac{\Delta}{2} \sum_{i=1}^{r} h_i^2(x))(1 + \Delta \sum_{i=1}^{r} h_i(x) y_i(k) + \\ &\quad \frac{\Delta^2}{2} \sum_{i=1}^{r} \sum_{j=1}^{r} h_i(x) h_j(x) y_i(k) y_j(k)) T_\Delta^* p_{k-1}(x). \end{aligned}$$

Pour alléger l'écriture on posera par la suite

$$\begin{aligned} \Phi_k(x) &= \exp(-\frac{\Delta}{2} \sum_{i=1}^{r} h_i^2(x))(1 + \Delta \sum_{i=1}^{r} h_i(x) y_i(k) + \\ &\quad \frac{\Delta^2}{2} \sum_{i=1}^{r} \sum_{j=1}^{r} h_i(x) h_j(x) y_i(k) y_j(k)). \end{aligned}$$

Ainsi

$$p_k(x) \approx \Phi_k(x) T_\Delta^* p_{k-1}(x).$$

On en déduit que

$$\overline{p}_k(x) \approx \Phi_k(x) T_\Delta^* \overline{p}_{k-1}(x). \tag{3.13}$$

Or pour tout k tel que $0 \leq k \leq M$ on a :

$$\overline{p}_k(x) = \sum_{l \leq \kappa} \Psi_l(k) e_l(x),$$

et par substitution dans l'équation 3.13 on obtient

$$\sum_{l \leq \kappa} \Psi_l(k+1) e_l(x) = \sum_{l \leq \kappa} \Phi_{k+1}(x) T_\Delta^* e_l(x) \Psi_l(k).$$

Or

$$\Psi_l(k+1) = \int e_l(x) \sum_{n \leq \kappa} \Psi_n(k+1) e_n(x) dx,$$

et donc

$$\begin{aligned} \Psi_l(k+1) &= \int e_l(x) \sum_{n \leq \kappa} \Phi_{k+1}(x) T_\Delta^* e_n(x) \Psi_n(k) dx \\ &= \sum_{n \leq \kappa} (\int e_l(x) \Phi_{k+1}(x) T_\Delta^* e_n(x) dx) \Psi_n(k). \end{aligned} \tag{3.14}$$

Du fait que l'on connais $\Psi_l(0)$ pour tous $l \leq \kappa$ alors l'équation 3.14 nous permet d'avoir un algorithme récursif pour calculer les $\Psi_l(k)$. De plus, pour tout $l, n \leq \kappa$, on a

$$\begin{aligned} &\int e_l(x) \Phi_{k+1}(x) T_\Delta^* e_n(x) dx \\ &= \int e_l(x) G(x) T_\Delta^* e_n(x) dx \\ &\quad + \Delta \sum_{i=1}^{r} (\int e_l(x) G(x) h_i(x) T_\Delta^* e_n(x) dx) y_i(k) \\ &\quad + \frac{\Delta^2}{2} \sum_{i=1}^{r} \sum_{j=1}^{r} (\int e_l(x) G(x) h_i(x) h_j(x) T_\Delta^* e_n(x) dx) y_i(k) y_j(k), \end{aligned}$$

où

$$G(x) = \exp(-\frac{\Delta}{2} \sum_{i=1}^{r} h_i^2(x)).$$

Pour alléger les notations on pose pour tout $i,j=1,...,r$ et tout $l,n\in\Gamma$ tel que $l,n\leq\kappa$

$$\begin{aligned} q_{ln} &= \int e_l(x)G(x)T_\Delta^* e_n(x)dx, \\ q_{ln}^i &= \int e_l(x)G(x)h_i(x)T_\Delta^* e_n(x)dx, \\ q_{ln}^{ij} &= \int e_l(x)G(x)h_i(x)h_j(x)T_\Delta^* e_n(x)dx, \\ f_l &= \int f(x)e_l(x)dx,\ \Psi_l(0)=\int p(x)e_l(x)dx. \end{aligned} \tag{3.15}$$

on en déduit l'algorithme suivant:

1. Les calculs indépendants des mesures (calculs off-line):

   - on calcule pour tout $i,j=1,...,r$ et $l,n\leq\kappa$, les q_{ln}, $q_{ln}^i, q_{ln}^{ij}, f_l$ et $\Psi_l(0)$;
   - puis on pose $\overline{p}_0(x)=p(x)$ et $\overline{\Phi}_0[f]=\sum_{l\leq\kappa}\Psi_l(0)f_l$.

2. À la k-ième mesure:

   - à l'obtention de la mesure $y(k)$ on calcule pour chaque $l\leq\kappa$

   $$\Psi_l(k)=\sum_{n\leq\kappa}Q_{ln}(y(k))\Psi_n(k-1), \tag{3.16}$$

   où

   $$Q_{ln}(y(k))=q_{ln}+\Delta\sum_{i=1}^r y_i(k)q_{ln}^i+\frac{\Delta^2}{2}\sum_{i=1}^r\sum j=1ry_i(k)y_j(k)q_{ln}^{ij}.$$

   - on calcule par la suite

   $$\overline{p}_k(x)=\sum_{l\leq\kappa}\Psi_l(k)e_l(x),$$

$$\overline{\Phi}_k[f] = \sum_{l \leq \kappa} \Psi_l(k) f_l,$$

et

$$\overline{f}_k = \frac{\overline{\Phi}_k[f]}{\overline{\Phi}_k[1]}.$$

Remarques 3.2.1

Premièrement : *les intégrales dans les équations 3.15 qui font intervenir la resolution de l'équation de Fokker-Planck sont calculées une seule fois et indépendamment des mesures.*

Deuxièment : *Seuls les coefficients Ψ_l doivent être calculer à chaque étape de l'algorithme.*

3.3 Méthode numérique pour approximer la solution de l'équation de Fokker-Planck.

Pour calculer les q_{ln},q^i_{ln} et q^{ij}_{ln} dans l'équation 3.15 on doit résoudre l'équation de Fokker-Planck. Dans la plupart des cas la solution de cette équation est difficile à trouver d'une manière exacte, alors on procède par des méthodes numériques pour son approximation. On propose dans cette section une méthode parmi d'autres qui nous servira par la suite pour approximer la solution de l'équation de Fokker-Planck. Étant donné X_t un processus d'état

évoluant suivant l'équation différentielle d'Itô suivante

$$dX_t = b(X(t))dt + \sigma(X_t)dW_t,$$

soit T_t l'opérateur solution de l'équation de Fokker-Planck qu'on a défini dans la section 3.1 et φ une fonction dont on veut calculer $T_\Delta\varphi(x)$ avec $\Delta > 0$. La méthode consiste à générer un nombre N de trajectoires. On choisi un entier positif m, et pour chaque $k \in \{1...N\}$ on génère X_Δ^k de la façon suivante Partant d'une valeur initiale

$$X_0 = x_\alpha,$$

pour chaque $l \in \{1...m\}$, on génére

$$X_{\frac{l\Delta}{m}} = X_{\frac{(l-1)\Delta}{m}} + b(X_{\frac{(l-1)\Delta}{m}})\frac{\Delta}{m} + \sigma(X_{\frac{(l-1)\Delta}{m}})\frac{Z_l}{\sqrt{m}}\sqrt{\Delta},$$

où les Z_l sont des vecteurs aléatoires gausiens indépendants et identiquement distribués

$$Z_l \sim N(0, I_d).$$

Pour $l = m$ on prend

$$X_\Delta^k = X_{\frac{l\Delta}{m}}.$$

Une fois les X_Δ^k générées on approxime $T_\Delta\varphi(x_\alpha)$ par la somme $\frac{1}{N}\sum_{k=1}^N \varphi(X_\Delta^k)$

$$T_\Delta\varphi(x_\alpha) \approx \frac{1}{N}\sum_{k=1}^{N} \varphi(X_\Delta^k). \tag{3.17}$$

On rappelle ici un résultat fondamental, si f et g sont deux fonctions de $L_2(\mathbf{R}^d)$ alors on a

$$\int f(x)T_\Delta^* g(x)dx = \int (T_\Delta f(x))g(x)dx. \tag{3.18}$$

Les équations 3.17 et 3.18 et la méthode numérique d'Hermite pour le calcul d'intégrale nous donnent une approximation des valeurs q_{ln}, q^i_{ln} et q^{ij}_{ln} pour tout $l, n = 1...\kappa$ et $i, j = 1...r$.

3.4 Conclusion

La méthode de filtrage qu'on a étudiée dans ce chapitre est bonne sur le plan théorique du fait que l'approximation de la densité de filtrage non normalisée converge comme le montre le théorème 1. De plus cette méthode a pour avantage les calculs qui se font avant l'obtention des mesures (off-line) ce qui rend l'implémentation de l'algorithme faisable et facile. Cependant sur le plan pratique la méthode ne donne malheureusement pas de bons résultats comme on va l'illustrer par des exemples concrets au chapitre 5. En effet le choix d'un κ petit rend la zone, où l'on veut faire nos prévisions, limité et par conséquence on risque de perdre beaucoup d'information sur la trajectoire réelle comme on peut le remarquer dans les figures 3.2 et 3.2 qui représentent l'approximation d'une densité gaussienne de moyenne 5 et d'écart type 2, par des polynômes d'Hermite. Par contre si l'on prend une valeur κ très grande la zone d'observation devient plus large, et par conséquent on augmente la chance d'avoir toute l'information sur la trajectoire, mais le problème c'est que pour des valeurs de κ très grandes, le nombre de calculs devient énorme et donc l'algorithme est impraticable. De plus, le calculs des polynômes d'Hermite devient aussi très difficile étant donné la grosseur des coefficients. Une alternative que l'on propose pour résoudre ce problème est de procéder

par un changement de la base d'Hermite chaque fois que l'on veut faire une nouvelle prévision. En effet à l'obtention d'une prévision on recentre la base d'Hermite autour de cette dernière , autrement dit on change la base $\{e_n(x)\}_{n\leq\kappa}$ par $\{e_n(x-\widehat{x})\}_{n\leq\kappa}$ où $\widehat{x}$ est la prévision réalisée. Il est bien clair qu'avec cette méthode on doit renoncer aux calculs off-line car à chaque étape de l'algorithme on change la base et donc on doit refaire tous les calculs.

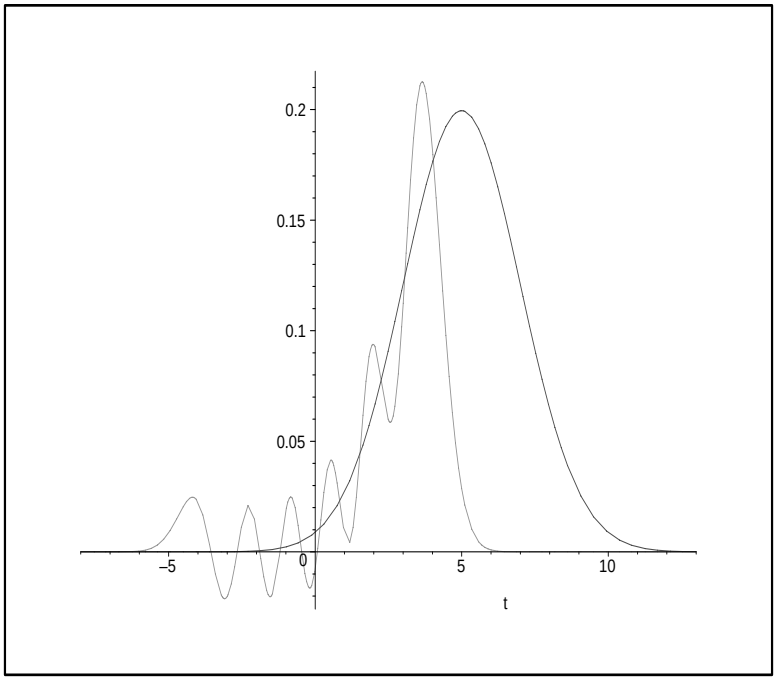

Figure 3.1: Approximation de la densité gaussienne de moyenne 5 et d'écart type 2 par la base d'Hermite pour $\kappa = 10$

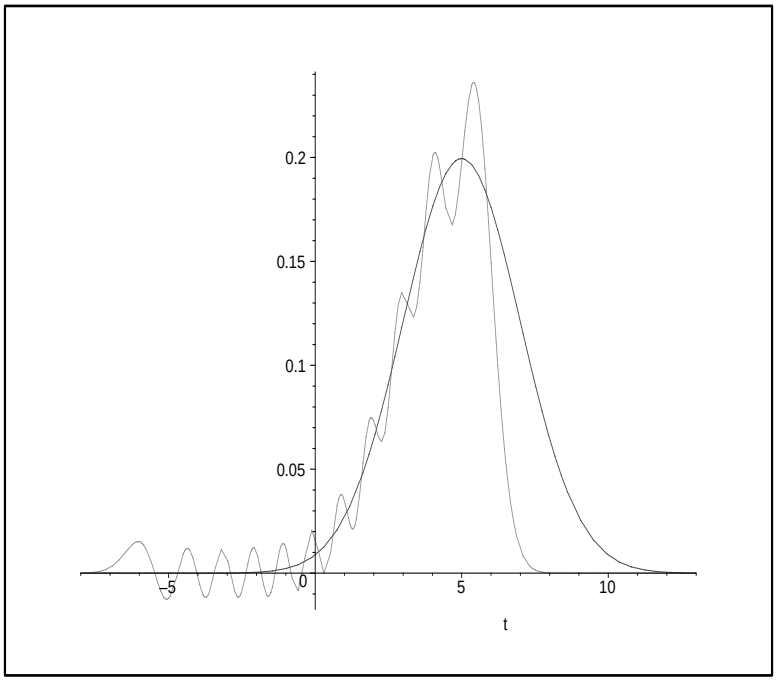

Figure 3.2: Approximation de la densité gaussienne de moyenne 5 et d'écart type 2 par la base d'Hermite pour $\kappa = 40$

Chapitre 4

Quelques exemples d'applications

L'un des domaines d'applications des méthodes de filtrage est celui de poursuite de cible (ou prévision de trajectoire). Le problème consiste à estimer les paramètres cinématiques d'un mobile (position, vitesse, accélération,...) étant donné des mesures bruitées fournies par un radar. On possède en général un modèle de dynamique décrivant l'évolution de l'état de la cible dans le temps découlant de la connaissance que l'on a du système. Les équations modélisant les vecteurs d'état et d'observation peuvent être linéaire ou non linéaire.

Dans ce chapitre nous appliquerons la méthode de filtrage décrite au chapitre 3 à trois modèles différents et nous allons analyser les résultats trouvés. Mais dans un premier temps nous allons donner une brève description du principe

de mesure d'un radar.

4.1 Principe de mesure d'un radar

Nous rappelons brièvement le principe de mesure d'un radar de poursuite. Le radar envoie un signal très bref ou impulsion dans une direction donnée. Ce signal se propage à la vitesse de la lumière, dans toutes les directions. Au bout d'un intervalle de temps δt le signal atteint la cible, ce temps est proportionnel à la distance radar-cible. Une partie du signal est réfléchie par la cible et au bout du même intervalle de temps δt, l'onde réfléchie atteint le radar. Celui-ci, après avoir traité le signal reçu fournit une estimation de la distance entre le radar et la cible et de l'angle formée par l'axe de visée du radar et un axe de référence. L'utilisation d'un tel radar de poursuite conduit aux problème de prévision de trajectoire d'une cible par mesure d'angle seulement (radar passif) ou par mesure d'angle et de distance (radar actif).

4.2 Premier modèle (radar passif).

Ce premier modèle a été donné comme exemple d'application dans l'article de Lototsky et Rozovskii (1998) pour "montrer"la convergence de leur méthode de filtrage. On considère un processus d'état $X = (X_1(t), X_2(t))$ évoluant

suivant l'équation

$$\begin{aligned} dX_1(t) &= -189.33X_2^3dt + 9.16X_2(t)dt + 0.001dW_1(t), \\ dX_2(t) &= -\frac{1}{3}dt + 0.03dW_2(t), \end{aligned}$$

Les mesures $y = (y_k)$, sont modélisées par l'équation d'observation

$$y_k = h(X_1(t_k), X_2(t_k)) + \frac{0.2}{\sqrt{\Delta}}V_k,$$

avec

$$h(x,y) = \begin{cases} \arcsin(\frac{y}{\sqrt{x^2+y^2}}) & \text{si } x > 0 \\ \Pi - \arcsin(\frac{y}{\sqrt{x^2+y^2}}) & \text{si } x < 0 \\ 0 & \text{si } x = 0. \end{cases}$$

Le but de l'application du filtre présenté dans le chapitre 3 est d'estimer les deux coordonnées $\widehat{X_1}(t_k)$ et $\widehat{X_2}(t_k)$ étant donné les mesures $y_1, ..., y_k$. On choisira $\Delta = 0.01$ et on supposera que $(V_k)_{k\geq 1}$ sont gaussiennes de moyenne nulle et de matrice de covariance $\frac{1}{\Delta}I$, la distribution initiale est $p_0(x,y) = \frac{1}{2\pi}\exp\frac{-1}{2}(x^2+y^2)$.

Pour des valeurs de $\kappa = 10, 15, 30$ et 50 le filtre ne donne pas de bons résultats [voir les graphes B.1 à B.4]. On a remarqué que la non convergence du filtre est due à trois causes. La première cause c'est que dans ce modèle la premiere coordonnée de la trajectoire tend très vite ver l'infini [voir les graphes B.5 à B.8]. Ainsi à partir d'un certain temps, alors que le filtre continue à estimer la position de la trajectoire dans une zone limitée à cause de la valeur prise de κ qui est fixe et petite, la vraie trajectoire a deja quitté cette zone ce qui explique les mauvaises estimations réalisées par le filtre. Dans le tableau 4.1 on remarque que l'erreur entre la trajectoire réelle et

prédite diminue chaque fois que l'on augmente la valeur de κ. Cependant, même avec une valeur de $\kappa = 50$ l'erreur est encore énorme. Pour espérer la convergence du filtre on doit donc prendre des valeurs très grandes pour κ, mais ceci rendra les calculs très lourds et difficiles à réaliser. En effet, le cas $\kappa = 100$ est impossible à réaliser, même sur un ordinateur ayant $2go$ de mémoire vive.

	$\kappa = 10$	$\kappa = 15$	$\kappa = 30$	$\kappa = 50$
$N = 20$	0.7485	0.7117	0.7110	0.1971
$N = 100$	1.2887	1.9350	1.9083	1.2037
$N = 250$	8.9631	7.0542	6.8731	7.0042

Tableau 4.1: Erreurs quadratiques pour le premier modèle

La deuxième cause qui peut être aussi à l'origine des mauvaises estimations est due à la fonction d'observation prise dans ce modèle et qui mesure l'angle seulement ce qui est pour nous insuffisant pour donner la position exacte de la cible. En effet on peut bien avoir deux positions de distances différentes par rapport au radar alors qu'ils ont le même angle. Dans le deuxième modèle, que nous allons étudiés dans la prochaine section, nous allons prendre en considération ce fait.
La troisième cause du problème est due au fait que si l'on prend des valeurs de Δ très proche de 0, le niveau de bruit augmente considérablement, ce qui rend la méthode impraticable.

4.3 Deuxième modèle(radar actif)

Dans le premier modèle nous avons pris pour fonction d'observation la mesure de l'angle seulement. Dans ce deuxième modèle on prendra aussi en considération la mesure de la distance entre la cible et le radar. Soit $X(t)$ le processus modélisé par l'équation

$$\begin{aligned} dX_1(t) &= -189.33X_2^3dt + 9.16X_2(t)dt + 0.001dW_1(t), \\ dX_2(t) &= -\frac{1}{3}dt + 0.03dW_2(t), \end{aligned}$$

les observations $y = (y_k)$, sont modélisées par les équations

$$y_k = h(X_1(t_k), X_2(t_k)) + \frac{0.2}{\sqrt{\Delta}}V_k,$$

avec $h(x, y) = (h_1(x, y), h_2(x, y))$,

$$h_1(x,y) = \begin{cases} \arcsin(\frac{y}{\sqrt{x^2+y^2}}) & \text{si } x > 0 \\ \Pi - \arcsin(\frac{y}{\sqrt{x^2+y^2}}) & \text{si } x < 0 \\ 0 & \text{si } x = 0 \end{cases},$$

$$h_2(x,y) = \sqrt{x^2+y^2}.$$

On reprend les mêmes hypotheses deja faite dans le premier modèle sur Δ, $(V_k)_{k\geq 1}$ et la densité initiale.

Le tableau 4.2 donne les erreurs quadratiques entre les vraies mesures et celles prédites pour différentes valeurs de κ et pour des trajectoires qui différent par leurs nombres de points.

	$\kappa = 10$	$\kappa = 15$	$\kappa = 30$	$\kappa = 50$
$N = 20$	0.4965	0.4443	0.5124	0.6940
$N = 100$	1.1102	1.2452	0.9174	0.9410
$N = 250$	9.4739	8.4576	7.2595	6.3785

Tableau 4.2: Erreurs quadratiques pour le deuxième modèle

On peut faire la même remarque que dans le premier modèle, l'erreur entre la mesure réelle et estimée diminue chaque fois que l'on augmente la valeur de κ, et devient énorme lorsque le nombre de points de la trajectoire augmente. Même avec un $\kappa = 50$ le filtre est loin de donner des bonnes estimations comme le montrent les figures B.10, B.11 et B.12.

4.4 Troisième modèle (modèle d'Ornstein-Ulhembeck)

Le modèle que l'on considère dans cette partie diffère des deux premiers du fait que son processus d'état reste dans une zone limitée comme le montre les graphes B.13 et B.14. Soient $X(t)$ le processus d'Ornstein-Ulhembeck modélisée par l'équation

$$dX(t) = -\mathcal{A}(X(t) + \mathcal{U})dt + \sigma dW(t),$$

avec

$$\mathcal{A} = \begin{pmatrix} 1 & \frac{1}{2} \\ \frac{1}{2} & 1 \end{pmatrix},$$

$$\mathcal{U} = \begin{pmatrix} 0.3 \\ 0.3 \end{pmatrix},$$

et

$$\sigma = \begin{pmatrix} 0.01 \\ 0.03 \end{pmatrix}.$$

On veut donc estimer par l'application de la méthode de filtrage, présenté au chapitre précédent, les deux coordonnées étant donné les mesures d'observations modélisées par l'équation

$$y_k = h(X_1(t_k), X_2(t_k)) + \frac{0.2}{\sqrt{\Delta}} V_k,$$

avec $h(x, y) = (h_1(x, y), h_2(x, y))$

$$h_1(x,y) = \begin{cases} \arcsin(\frac{y}{\sqrt{x^2+y^2}}) & \text{si } x > 0 \\ \Pi - \arcsin(\frac{y}{\sqrt{x^2+y^2}}) & \text{si } x < 0 \\ 0 & \text{si } x = 0. \end{cases}$$

$$h_2(x,y) = \sqrt{x^2 + y^2}.$$

On fera les même hypothèses sur Δ, $(V_k)_{k\geq 1}$ et la densité initiale.
On remarque que les erreurs de la valeur estimée et la vraie mesure dans ce troisième exemple sont beaucoup plus mieux que dans les deux premiers modèles voir tableau4.3. Cependant le filtre ne donne toujours pas la bonne estimation de la trajectoire de la cible voir figures B.15 à B.17.

	$\kappa = 10$	$\kappa = 15$	$\kappa = 30$	$\kappa = 50$
$N = 20$	0.4719	0.4274	0.3420	0.6754
$N = 100$	0.4016	0.4731	0.3114	1.3428
$N = 250$	6.9705	2.9432	3.3537	4.2751

Tableau 4.3: Erreurs quadratiques pour le troisième modèle

4.5 Conclusion

La méthodes de filtrage non linéaire basée sur l'expansion en chaos de Wiener est théoriquement très bonne. Mais pratiquement les résultats obtenus par son application aux trois modèles présentés dans ce chapitre avec des valeurs de κ allant jusqu'à 50 montrent qu'elle ne donne malheureusement pas de bonnes estimations. Plus la valeur de κ est grande plus les calculs deviennent très lourds et difficiles à réaliser par la machine.

Chapitre 5

Conclusion

La recherche dans le domaine de filtrage non linéaire est fortement motivée par ses nombreuses applications. Cependant, il s'agit d'un problème difficile qui n'admet pas en générale de solution en dimension finie. Nous avons vu dans cette étude que la détermination du filtre optimal revient à celle de la densité de filtrage non normalisée (UFD). Cette densité que l'on peut calculer d'une manière analytique ou numérique est solution de l'équation différentielle stochastique de Fokker-Planck. L'objectif de ce mémoire était d'étudier une approche numérique récente pour l'approximation de la densité de filtrage non normalisée qui est basée sur l'expansion en chaos de wiener. L'avantage de cette méthode est due au fait que les calculs les plus difficiles et les plus longs qui font intervenir la solution de l'équation de Fokker-Planck ne dépendent pas des observations et se font de manière off-line, ce qui rend l'algorithme facile à implémenter. Théoriquement la méthode est bonne mais sur le plan pratique elle ne donne pas de bonnes estimations comme nous

l'avons montrés au chapitre 4. Nous avons remarquer que le problème est possiblement causé, par les valeurs petites de κ que l'on avait pris.
Une alternative qui nous parait être bonne est de faire un changement de base à chaque fois que l'on veut faire une nouvelle prévision, de tel sorte que la cible soit à l'intérieur de la zone dans la quelle on espère faire notre prochaine estimation de la position de cible. Pour ce fait on pourrait faire appel aux ondelettes, au lieu des polynômes d'Hermite.

Chapitre 6

Codes des programmes sur matlab

6.1 Codes des programmes pour le premier modèle.

```
%%%%%%%  Simulation du processus %%%%%%%%%%%%%%%%%%%%%%

function data = paths(nu,delta) data = zeros(nu,7);

sigma = 0.2/sqrt(delta);

nn = 1+m*(nu-1);

x1= zeros(nn,1);

y1 = zeros(nn,1);
x= zeros(nu,1);
```

```
y= zeros(nu,1);
r1= zeros(nu,1);
theta1= zeros(nu,1);
theta2= zeros(nu,1);
z1= zeros(nu,1); z2 =
zeros(nu,1);
x1(1) = 0;
y1(1) = 0;
deltanew = delta/m;

for i=2:nn,
  x0 = x1(i-1);  y0 = y1(i-1);
  x1(i)= x0 +deltanew*(-189.33*(y0)^3 + 9.16*y0) +0.001*sqrt(deltanew)*randn(1,1);
  y1(i)= y0 -(1/3)*deltanew + 0.03*sqrt(deltanew)*randn(1,1);
end

for k=1:nu,
   v = m*(k-1)+1;
   x(k) = x1(v);
   y(k) = y1(v);
   r1(k) = sqrt(x1(k)^2+ y1(k)^2) + sigma*randn(1,1);
   theta1(k) = asin( y1(k)/(0.00001+sqrt(x1(k)^2+ y1(k)^2))) + sigma*randn(1,1);
   theta2(k) = asin( (y1(k)-10)/(0.00001+ sqrt(x1(k)^2+ (y1(k)-10)^2))) + sigma*randn(1,1);
   z1(k) = x(k) + sigma*randn(1,1);
   z2(k) = y(k) + sigma*randn(1,1);

end

data(:,1) = x; data(:,2) = y; data(:,3) = r1; data(:,4) = theta1;
data(:,5) = theta2; data(:,6) = z1; data(:,7) = z2;


plot(x,y)

 %%%%%%%% Fin de la simulation %%%%%%%%%%%%%%%%%%

%%%%% Programme pour calculer h1(x)%%%%%%%%%%%%
```

```
function h1 = obserh1(x,y)

       r= sqrt(x^2 + y^2);

      if(x>0)
         ff = asin(y/r);
      else
       if(x<0)
           ff = pi-asin(y/r);
       else
           ff=0.0;


       end
   end

     h1 = ff;
  %%% Fin du programme %%%%%


 %%%%%%% Programme pour calculer les e_n %%%%%%%%

function f = ehermite(n,t)
      ent = floor(n/2);
      val = 0;
      mun = ones(1,(1+ent));
      mun(1) = -1;
        for i=2:(1+ent)
            mun(i) = -mun(i-1);
        end

      for j=0:ent
       val = val + mun(j+1)*(2*t).^(n-2*j)/( factorial(j)*factorial(n-2*j) );
      end

        f = sqrt(factorial(n))*exp(-t.^2/2).*val/sqrt( 2^n *sqrt(pi) );
```

```
  %%%%%% Fin du programme ehermite %%%%%%%%



%%%%%%%%%%%%Processus de Lotovski et Rosoksky  %%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear all tic
%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Parameters %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
kappa = 10; nkappa = (kappa+1)*(kappa+2)/2;
nkappa2 = nkappa^2;

nn = 100; ;
m= 10;
nz = 30;
dim = nz^2;
delta = 0.01;

deltanew = delta/m;
PI = 3.141592654;
nz = 30;
dim = nz^2; A =
dlmread('C:\GNAT\\progb2\hermite30.txt',' ');

poids = A(:,2); p  = A(:,1);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
ew = zeros(nkappa,dim);

f = zeros(dim,1); f1 = zeros(dim,1); f2 = zeros(dim,1);

fpsi = zeros(dim,1);


ind = zeros(nkappa,2);
l = 1;
 for i = 0:kappa
```

```
    for j = 0: (kappa-i)

        ind(l,1) = i;
        ind(l,2) = j;
        l = l+1;
    end
end


for i=1:nz
     x1 = p(i);
     %d1 = exp(-x1^2/4);
   for j = 1:nz
       k = nz*(i-1)+j;
       x2 = p(j);
       %d2 = d1*exp(-x2^2/4)/(4*PI);
        d2 = den(x1,x2)
       f(k) = 1;
       f1(k) = x1;
       f2(k) = x2;
       fpsi(k) = d2;

   end
 end



for n = 1: nkappa;
  for i=1:nz
      part = ehermite(ind(n,1),p(i))*poids(i);
    for j = 1:nz
        k = nz*(i-1)+j;
        ew(n,k) = part*ehermite(ind(n,2),p(j))*poids(j);
    end
  end
end

ff =  ew*f; ffx = ew*f1; ffy = ew*f2; psi0 = ew*fpsi;
```

```
toc fprintf('1st step')




%%%%%%%  Simulation du processus  %%%%%%%%%%%%%%%%%%%%%
tic
xsim = zeros(nn,dim); ysim = zeros(nn,dim);

for i = 1:nz
      for j=1:nz
          y0 = p(j)*ones(nn,1);
          x0 = p(i)*ones(nn,1);
          n = nz*(i-1)+j;
       for k=1:m,
        xp = x0 + deltanew*(-189.33*y0.^3 + 9.16*y0) +0.001*sqrt(deltanew)*randn(nn,1);
        yp = y0 -1.0/3.0*deltanew + 0.03*sqrt(deltanew)*randn(nn,1);
        y0=yp; x0=xp;

    end
    xsim(:,n) = x0;
    ysim(:,n) = y0;
end end

toc
 fprintf('Simulation done')

%%%%%%%% Fin de la simulation %%%%%%%%%%%%%%%%%%
tic
eew = zeros(nkappa,dim);
eew1 = zeros(nkappa,dim);

eew2 = zeros(nkappa,dim);
eew11 = zeros(nkappa,dim);

eew12 = zeros(nkappa,dim);
 eew22 = zeros(nkappa,dim);
```

```
un = ones(1,nn)/m;
h = zeros(nn,dim);
for i=1:nn
    for j=1:dim
    h(i,j) = obserh1(xsim(i,j),ysim(i,j));
end end


part = exp(-0.5*delta*h.^2);

for n = 1:nkappa
   toto = part.*ehermite(ind(n,1),xsim).*ehermite(ind(n,2),ysim);
   eew(n,:)= un*toto;
   eew1(n,:)= un*(toto.*h);
   eew11(n,:)= un*(toto.*h.*h);
end

toc fprintf('mat1 termine')

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
tic

M = ew';
q = eew*M;
q1 = eew1*M;
q11 = eew11*M;

toc

fprintf('Matrices are ready!')

save datad1pk10n100 ff ffx ffy psi0 q q1 q11


%%%% filtre  hermitien de l'article Lot-Roz

clear all tic
```

```
nu = 250;   % taille de la trajectoire \‘{a} simuler
delta =0.01;

load datad1k10n100


%%%%%%%  Simulation %%%%%%%%%%%%%%%%%%%%%

C = paths(nu,delta);

x= C(:,1);
y = C(:,2);
r = C(:,3);
theta1 = C(:,4);

theta2 = C(:,5);
 x1 = C(:,6);
 y1 = C(:,7);
 xp4= zeros(nu,1);
 yp4= zeros(nu,1);
 denom4=zeros(nu,1);
 Inter = psi0 /(psi0'*ff);

   for k=1:nu,

      z1 = theta2(k);
      %z1 = x1(k);
      %z2 = y1(k);
      Qm= q+ delta*( z1*q1) + (delta^2/2)*q11*z1^2 ;
      psi=Qm*Inter;
      denom4(k) = psi'*ff;
      Inter = psi/denom4(k);
      xp4(k) = Inter'*ffx;
      yp4(k) = Inter'*ffy;
    end

   z1 = zeros(nu,1); z2 = zeros(nu,1);
 for k=1:nu,
```

```
   z1(k) = mean(xp4(1:k));
   z2(k) = mean(yp4(1:k));
 end


erq2 = sqrt((yp4-y).^2 +(xp4 - x).^2);
ereurq = (1/nu)*sum(erq2)
 k =1;
 plot(x(k:nu),y(k:nu))
 title('trajectoire r\'{e}elle')

 figure
 plot(x1(k:nu),y1(k:nu))
 title('trajectoire observ\'{e}e')

 figure
  plot(x1(k:nu),y1(k:nu),'b-')
 hold on
 plot(x(k:nu),y(k:nu),'r-')
 title('trajectoires observ\'{e}e et r\'{e}elle')
  h = legend('Obs', 'R\'{e}elle',0);

  figure
  plot(xp4(k:nu),yp4(k:nu),'b+')
 hold on
 plot(x(k:nu),y(k:nu),'r-')

  h = legend('Pred', 'R\'{e}elle',0);

  figure
  plot(xp4(k:nu),yp4(k:nu),'b-')
 hold on
 plot(x1(k:nu),y1(k:nu),'r-')
 title('trajectoires pr\'{e}dite et observ\'{e}e')
  h = legend('Pred', 'Obs',0);

 figure
  plot(xp4(k:nu),yp4(k:nu),'b-')
```

```
 title('trajectoire pr\'{e}dite')

figure;
plot([denom4])
title('constantes normalis\'{e}es')




figure;
plot([xp4-x])
title('Erreurs relatives aux pr\'{e}dictions de x')
%h = legend('4',0);
figure
plot([yp4-y])
title('Erreurs relatives aux pr\'{e}dictions de y')
%h = legend('4',0);
figure
plot(xp4,'r+')
hold on
plot(x,'b-')
h=legend('Pr\'{e}dite', 'r\'{e}elle',0);
figure
plot(yp4,'r+')
hold on
plot(y,'b-')
h=legend('Pr\'{e}dite', 'r\'{e}elle',0);
toc
```

6.2 Codes des programmes pour le deuxième modèle.

```
%%%%%%  Simulation processus dans le deuxieme modele %%%%%%%%%%%%%%%%%%%%%


function data = paths(nu,delta)

%nu = 1000;
delta = 0.01;
m= 100;
data = zeros(nu,7);
sigma = 0.2/sqrt(delta);
nn = 1+m*(nu-1);
x1= zeros(nn,1);
y1 = zeros(nn,1);

x= zeros(nu,1);
y= zeros(nu,1);
r1= zeros(nu,1);
theta1=zeros(nu,1);
theta2= zeros(nu,1);
z1= zeros(nu,1);
z2 = zeros(nu,1);
x1(1) = 0;
y1(1) = 0;
deltanew = delta/m;

for i=2:nn,
  x0 = x1(i-1);  y0 = y1(i-1);
  x1(i)= x0 +deltanew*(-189.33*(y0)^3 + 9.16*y0) +0.001*sqrt(deltanew)*randn(1,1);
  y1(i)= y0 -(1/3)*deltanew + 0.03*sqrt(deltanew)*randn(1,1);
end

for k=1:nu,
```

```
   v = m*(k-1)+1;
   x(k) = x1(v);
   y(k) = y1(v);
   r1(k) = sqrt(x1(k)^2+ y1(k)^2) + sigma*randn(1,1);
   theta1(k) = asin( y1(k)/(0.00001+sqrt(x1(k)^2+ y1(k)^2))) + sigma*randn(1,1);
   theta2(k) = asin( (y1(k)-10)/(0.00001+ sqrt(x1(k)^2+ (y1(k)-10)^2))) + sigma*randn(1,1);
   z1(k) = x(k) + sigma*randn(1,1);
   z2(k) = y(k) + sigma*randn(1,1);

end

data(:,1) = x; data(:,2) = y; data(:,3) = r1; data(:,4) = theta1;
data(:,5) = theta2; data(:,6) = z1; data(:,7) = z2; plot(x,y)

%%%%%%%% Fin de la simulation %%%%%%%%%%%%%%%%%

%%% Programme pour calculer la fonction observation %%%%%

function f = obs1(x,y)
  r = sqrt(x^2+y^2);
  f = zeros(1,2);
  if(x>0)
      ff = asin(y/r);
  else
       if(x<0)
           ff = pi-asin(y/r);
       else
           ff=0.0;



       end
   end

f(1) = ff;
f(2) = r;

%%%%%% Fin du programme %%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%%%%%%%%%%%%%%%%%%%%% Calcules off-line pour le deuxieme modele %%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear all
tic
%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Parameters %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
kappa = 50;
nkappa = (kappa+1)*(kappa+2)/2;
nkappa2 = nkappa^2;

nn = 100;
m= 10;
nz = 30;
dim = nz^2;

delta = 0.01;
deltanew = delta/m;
PI = 3.141592654;
nz = 30;

dim = nz^2;

A = dlmread('C:\GNAT\\progb2\hermite30.txt',' ');

poids = A(:,2);
p  = A(:,1);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
ew = zeros(nkappa,dim);

f = zeros(dim,1);
f1 = zeros(dim,1);
f2 = zeros(dim,1);

fpsi = zeros(dim,1);
ind = zeros(nkappa,2);

l = 1;
```

```
 for i = 0:kappa
     for j = 0: (kappa-i)

         ind(l,1) = i;
         ind(l,2) = j;
         l = l+1;
     end
 end

 for i=1:nz
      x1 = p(i);
      d1 = exp(-x1^2/4);
    for j = 1:nz
        k = nz*(i-1)+j;
        x2 = p(j);
        d2 = d1*exp(-x2^2/4)/(4*PI);
        f(k) = 1;
        f1(k) = x1;
        f2(k) = x2;
        fpsi(k) = d2;

    end
  end

for n = 1: nkappa;
  for i=1:nz
      part = ehermite(ind(n,1),p(i))*poids(i);
    for j = 1:nz
        k = nz*(i-1)+j;
        ew(n,k) = part*ehermite(ind(n,2),p(j))*poids(j);
    end
  end
end

ff = ew*f;
ffx = ew*f1;
ffy = ew*f2;
psi0 = ew*fpsi;
```

```
toc
fprintf('1st step')

%%%%%%%  Simulation processus  %%%%%%%%%%%%%%%%%%%%%

tic
xsim = zeros(nn,dim);
ysim = zeros(nn,dim);
 for i = 1:nz
      for j=1:nz
          y0 = p(j)*ones(nn,1);
          x0 = p(i)*ones(nn,1);
          n = nz*(i-1)+j;
       for k=1:m,
        xp = x0 + deltanew*(-189.33*y0.^3 + 9.16*y0) +0.001*sqrt(deltanew)*randn(nn,1);
        yp = y0 - 1.0/3.0*deltanew + 0.03*sqrt(deltanew)*randn(nn,1);
        y0=yp; x0=xp;

    end
    xsim(:,n) = x0;
    ysim(:,n) = y0;
  end
end

toc
fprintf('Simulation done')

 %%%%%%%% Fin de la simulation %%%%%%%%%%%%%%%%%%

tic
eew = zeros(nkappa,dim);
eew1 = zeros(nkappa,dim);
eew2 = zeros(nkappa,dim);
eew11 = zeros(nkappa,dim);
eew12 = zeros(nkappa,dim);
eew22 = zeros(nkappa,dim);
```

```
un = ones(1,nn)/m;

h = zeros(nn,dim,2);

for i=1:nn
    for j=1:dim
      h(i,j,:) = obsrad(xsim(i,j),ysim(i,j));
    end
end


part = exp(-0.5*delta* (h(:,:,1).*h(:,:,1) + h(:,:,2).*h(:,:,2)));

for n = 1:nkappa
   toto = part.*ehermite(ind(n,1),xsim).*ehermite(ind(n,2),ysim);
   eew(n,:)= un*toto;
   eew1(n,:)= un*(toto.*h(:,:,1));
   eew2(n,:)= un*(toto.*h(:,:,2));
   eew11(n,:)= un*(toto.*h(:,:,1).*h(:,:,1));
   eew12(n,:)= un*(toto.*h(:,:,1).*h(:,:,2));
   eew22(n,:)= un*(toto.*h(:,:,2).*h(:,:,2));
end

toc
fprintf('mat1 termine')

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% mat2 ********************************

tic

M = ew';
q = eew*M;
q1 = eew1*M;
q2 = eew2*M;
q11 = eew11*M;
q12 = eew12*M;
q22 = eew22*M;
toc
```

```
fprintf('Matrices are ready!')
save datad2k50n100 ff ffx ffy psi0 q q1 q2 q11 q12 q22

%%%% filtre  hermitien pour le deuxieme modele %%%%%%%%
tic
nu = 250;   % taille de la trajectoire \`{a} simuler
delta =0.01;
%C = pathslin(nu,delta);



load datad2k50n100


%%%%%%%  Simulation %%%%%%%%%%%%%%%%%%%%%%

C = paths(nu,delta);

x= C(:,1);
y = C(:,2);
r = C(:,3);
theta1 = C(:,4);
theta2 = C(:,5);
x1 = C(:,6);
y1 = C(:,7);
xp4= zeros(nu,1);
yp4= zeros(nu,1);
denom4=zeros(nu,1);
Inter = psi0 /(psi0'*ff);
   for k=1:nu,
      Qm= q+ delta*( theta1(k)*q1) + (delta^2/2)*(q11*theta1(k)^2 +theta2(k)*q2);
      Qm = Qm+ (delta^2/2)*(2*q12*theta1(k)*theta2(k)+q11*theta1(k)^2+ q22*theta2(k)^2);
      psi=Qm*Inter;
      denom4(k) = psi'*ff;
      Inter = psi/denom4(k);
      xp4(k) = Inter'*ffx;
      yp4(k) = Inter'*ffy;
```

```
    end

   z1 = zeros(nu,1); z2 = zeros(nu,1);
for k=1:nu,
    z1(k) = mean(xp4(1:k));
    z2(k) = mean(yp4(1:k));
end
 erq2 = sqrt((yp4-y).^2 +(xp4 - x).^2);
 ereurq = (1/nu)*sum(erq2)

 k =100;
 plot(x(k:nu),y(k:nu))
 title('trajectoire r\'{e}elle')

 figure
 plot(x1(k:nu),y1(k:nu))
 title('trajectoire observ\'{e}e')

 figure
  plot(x1(k:nu),y1(k:nu),'b-')
 hold on
 plot(x(k:nu),y(k:nu),'r-')
 title('trajectoires observ\'{e}e et r\'{e}elle')
  h = legend('Obs', 'R\'{e}elle',0);

  figure
  plot(xp4(k:nu),yp4(k:nu),'b+')
 hold on
 plot(x(k:nu),y(k:nu),'r-')
 %title('trajectoires pr\'{e}dite et r\'{e}elle')
  h = legend('Pred', 'R\'{e}elle',0);

  figure
  plot(xp4(k:nu),yp4(k:nu),'b+')
 hold on
 plot(x1(k:nu),y1(k:nu),'r-')
 title('trajectoires pr\'{e}dite et observ\'{e}e')
  h = legend('Pred', 'Obs',0);
```

```
figure
plot(xp4(k:nu),yp4(k:nu),'b-')
title('trajectoire pr\'{e}dite')
figure;
plot([denom4])
title('constantes normalis\'{e}es')
figure;
plot([xp4-x])
title('Erreurs relatives aux pr\'{e}dictions de x')
%h = legend('4',0);
figure
plot([yp4-y])
title('Erreurs relatives aux pr\'{e}dictions de y')
%h = legend('4',0);
figure
plot(xp4,'r-')
hold on
plot(x,'b-')
h=legend('Pr\'{e}dite', 'r\'{e}elle',0);
figure
plot(yp4,'r-')
hold on
plot(y,'b-')
h=legend('Pr\'{e}dite', 'r\'{e}elle',0);
toc

fprintf('Fin du programme')
```

6.3 Codes des programmes pour le troisième modèle.

```
%%%%%%  Simulation processus O-U %%%%%%%%%%%%%%%%%%%%%

function data = pathslin(nu, delta, sigma)


m= 100;
data = zeros(nu,7);
nn = 1+m*(nu-1);
x1= zeros(nn,1);
y1 = zeros(nn,1);
x= zeros(nu,1);
y= zeros(nu,1);
r1= zeros(nu,1);
theta1= zeros(nu,1);
theta2= zeros(nu,1);
z1= zeros(nu,1);
z2 = zeros(nu,1);
x1(1) = -0.33;
y1(1) = -0.04;

deltanew = delta/m;
for i=2:nn,
  x0 = x1(i-1);  y0 = y1(i-1);
  x1(i)= x0 -deltanew*((x0+0.3) +0.5*(y0+0.03))+0.01*sqrt(deltanew)*randn(1,1);
  y1(i)= y0 -deltanew*((y0+0.03)+0.5*(x0+0.3)) + 0.03*sqrt(deltanew)*randn(1,1);
end

for k=1:nu,
   v = m*(k-1)+1;
   xx = x1(v);
   yy = y1(v);
   r = sqrt(xx^2+ yy^2);
```

```
   r1(k) = r + sigma*randn(1,1);
   theta1(k) = asin( yy/r ) +  (sigma/sqrt(delta))*randn(1,1);
   theta2(k) = asin( (y(k))/sqrt((xx-0.1)^2+ yy^2)) + (sigma/sqrt(delta))*randn(1,1);
   z1(k) = xx +0.01*sqrt(deltanew)*randn(1,1)+sigma*randn(1,1);
   z2(k) = yy + 0.03*sqrt(deltanew)*randn(1,1)+ sigma*randn(1,1);
   x(k) = xx;
   y(k) = yy;
end

data(:,1) = x;
data(:,2) = y;
data(:,3) = r1;
data(:,4) = theta1;
data(:,5) = theta2;
data(:,6) = z1;
data(:,7) = z2;

plot(x,y)

%%%%%%%% Fin de la simulation %%%%%%%%%%%%%%%%%%

%%%%%% Calcules off-line pour le modele O-U %%%%%
clear all
tic
%%%%%%%%%%%%%%%%%%%%%%%%%%%% Parameters %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
kappa = 10;
nkappa = (kappa+1)*(kappa+2)/2;
nkappa2 = nkappa^2;
nn = 100;
m= 10;
nz = 30;
dim = nz^2;
delta = 0.01;
deltanew = delta/m;
PI = 3.141592654;
nz = 30;
dim = nz^2;
A = dlmread('C:\GNAT\\progb2\hermite30.txt',' '); poids = A(:,2);
```

```
p  = A(:,1);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
ew = zeros(nkappa,dim);
f = zeros(dim,1);
f1 = zeros(dim,1);
f2 = zeros(dim,1);
fpsi = zeros(dim,1);
ind = zeros(nkappa,2);

n = 1;
 for i = 0:kappa
     for j = 0: (kappa-i)

          ind(n,1) = i;
          ind(n,2) = j;
          n = n+1;
     end
 end


 for i=1:nz
      x1 = p(i);
      d1 = exp(-x1^2/4);
    for j = 1:nz
        k = nz*(i-1)+j;
        x2 = p(j);
        d2 = d1*exp(-x2^2/4)/(4*PI);
        f(k) = 1;
        f1(k) = x1;
        f2(k) = x2;
        fpsi(k) = d2;

    end
  end


for n = 1: nkappa;
```

```
  for i=1:nz
      part = ehermite(ind(n,1),p(i))*poids(i);
    for j = 1:nz
        k = nz*(i-1)+j;
        ew(n,k) = part*ehermite(ind(n,2),p(j))*poids(j);
    end
  end
end

ff =  ew*f;
ffx = ew*f1;
ffy = ew*f2;
psi0 = ew*fpsi;

toc
fprintf('1st step')




%%%%%%%  Simulation processus O-U %%%%%%%%%%%%%%%%%%%%%
tic
xsim = zeros(nn,dim);
ysim = zeros(nn,dim);

for i = 1:nz
      for j=1:nz
          y0 = p(j)*ones(nn,1);
          x0 = p(i)*ones(nn,1);
          n = nz*(i-1)+j;
       for k=1:m,
        xp = x0 -deltanew*((x0+0.3) +0.5*(y0+0.03)) +0.01*sqrt(deltanew)*randn(nn,1);
        yp = y0 -deltanew*((y0+0.03)+0.5*(x0+0.3)) + 0.03*sqrt(deltanew)*randn(nn,1);
        y0=yp; x0=xp;

       end
        xsim(:,n) = x0;
        ysim(:,n) = y0;
```

```
      end
end
toc
fprintf('Simulation done')

%%%%%%%% Fin de la simulation %%%%%%%%%%%%%%%%%%
tic
eew = zeros(nkappa,dim);
eew1 = zeros(nkappa,dim);
eew2 = zeros(nkappa,dim);
eew11 = zeros(nkappa,dim);
eew12 = zeros(nkappa,dim);
eew22 = zeros(nkappa,dim);
un = ones(1,nn)/m;

part = exp(-0.5*delta*(xsim.*xsim+ysim.*ysim));

for n = 1:nkappa
   toto = part.*ehermite(ind(n,1),xsim).*ehermite(ind(n,2),ysim);
   eew(n,:)= un*toto;
   eew1(n,:)= un*(toto.*xsim);
   eew2(n,:)= un*(toto.*ysim);
   eew11(n,:)= un*(toto.*xsim.*xsim);
   eew12(n,:)= un*(toto.*xsim.*ysim);
   eew22(n,:)= un*(toto.*ysim.*ysim);
end
toc
fprintf('mat1 termine')

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% mat2 **********
tic
M = ew';
q = eew*M;
q1 = eew1*M;
q2 = eew2*M;
q11 = eew11*M;
q12 = eew12*M;
q22 = eew22*M;
```

```
toc
fprintf('Matrices are ready!')
save dataouk10n100 ff ffx ffy psi0 q q1 q2 q11 q12 q22



%% filtre hermetienne pour le modele O-U dim = 2 %%%%%%
clear all
tic
load dataouk10n100
nu = 250;   % taille de la trajectoire \'{a} simuler
delta =0.01;
C = pathslin(nu,delta, 0.1);

xp=zeros(1,nu);
yp=zeros(1,nu);
z = zeros(1,nu);
z0 = zeros(1,nu);

%%%%%%%  Simulation %%%%%%%%%%%%%%%%%%%%%%
x= C(:,1);
y = C(:,2);
r = C(:,3);
theta1 = C(:,4);
theta2 = C(:,5);
x1 = C(:,6);
y1 = C(:,7);
xp4= zeros(nu,1);
yp4= zeros(nu,1);
denom4=zeros(nu,1);
Inter = psi0/(psi0'*ff);


 for k=1:nu,
     z1 = r(k);
     z2 = theta1(k);
     %z1 = x1(k);
     %z2 = y1(k);
```

```
    Qm = q+ delta*( z1*q1 +z2*q2 ) + (delta^2/2)*(2*q12*z1*z2+ q11*z1^2 + q22*z2^2) ;
    psi=Qm*Inter;  %Q ou Q' ?????????
    denom4(k) = psi'*ff;
    Inter = psi/denom4(k);
    xp4(k) = Inter'*ffx;
    yp4(k) = Inter'*ffy;
 end

  z1 = zeros(nu,1);
  z2 = zeros(nu,1);

  for k=1:nu,
       z1(k) = mean(xp4(1:k));
       z2(k) = mean(yp4(1:k));
  end


erq2 = sqrt((yp4-y).^2 +(xp4 - x).^2);
ereurq = (1/nu)*sum(erq2)
nu1 = 100;
k =1;
plot(x(k:nu1),y(k:nu1))
title('trajectoire r\'{e}elle')

figure
plot(x1(k:nu1),y1(k:nu1))
title('trajectoire observ\'{e}e')

figure
 plot(x1(k:nu1),y1(k:nu1),'b+')
hold on
plot(x(k:nu1),y(k:nu1),'r-')
title('trajectoires observ\'{e}e et r\'{e}elle')
 h = legend('Obs', 'R\'{e}elle',0);

 figure
 plot(xp4(k:nu1),yp4(k:nu1),'b+')
hold on
```

```
plot(x(k:nu1),y(k:nu1),'r-')
%title('trajectoires pr\'{e}dite et r\'{e}elle')
 h = legend('Pred', 'R\'{e}elle',0);

 figure
 plot(xp4(k:nu1),yp4(k:nu1),'b+')
hold on
plot(x1(k:nu1),y1(k:nu1),'r-')
title('trajectoires pr\'{e}dite et observ\'{e}e')
 h = legend('Pred', 'Obs',0);

figure
 plot(xp4(k:nu1),yp4(k:nu1),'b-')
 title('trajectoire pr\'{e}dite')

figure;
plot([denom4])
title('constantes normalis\'{e}es')
figure;
plot([xp4-x])
title('Erreurs relatives aux pr\'{e}dictions de x')
%h = legend('4',0);
figure
plot([yp4-y])
title('Erreurs relatives aux pr\'{e}dictions de y')
%h = legend('4',0);
figure
plot(xp4,'r-')
hold on
plot(x,'b+')
h=legend('Pr\'{e}dite', 'r\'{e}elle',0);
figure
plot(yp4,'r-')
hold on
plot(y,'b+')
h=legend('Pr\'{e}dite', 'r\'{e}elle',0);
toc
```

```
fprintf('Programme termine')
```

Appendice A

Processus de Markov et mouvement brownien

A.1 Processus de Markov

Étant donné un espace de probabilité $(\Omega, \mathcal{A}, \mathbf{P})$, on appellera processus stochastique (ou fonction aléatoire) une collection $\{X(t); t \geq 0\}$ de vecteurs aléatoire de dimension d. Pour chaque $t \geq 0$, on a donc une application :

$$\begin{aligned} \Omega &\rightarrow \mathbf{R}^d, \\ \omega &\mapsto X(t, \omega). \end{aligned}$$

X est donc aussi une application de $\mathbf{R}_+ \times \Omega$ dans $\mathbf{R}^d$, qui à chaque couple (t, ω) fait correspondre $X(t, \omega)$. Le processus vérifie la propriété de mesurabilité par rapport au couple (t, ω), pour la tribu produit $\mathcal{B}_+ \otimes \mathcal{A}$, où $\mathcal{B}_+$

désigne la tribu borélienne de $\mathbf{R}_+$.
On appelle trajectoire du processus X l'application :

$$t \mapsto X(t, \omega)$$

de $\mathbf{R}_+$ dans $\mathbf{R}^d$, ω étant fixé. L'ensemble des trajectoires est donc une collection indexée par ω d'application de $\mathbf{R}_+$ dans $\mathbf{R}^d$.
On considère un ensemble mesurable G muni d'une tribu $\mathcal{G}$. Nous donnons maintenant la définition d'un processus de Markov à valeur dans G.

Définition A.1.1 *Un processus stochastique $\{Z(t); t \geq 0\}$ à valeur dans un espace mesurable $(G, \mathcal{G})$ est dite markovien (ou processus de markov) si pour tous $0 < s < t$, et pour :*

$$0 = t_0 < t_1 < ... < t_n < s$$

et

$$z_0, z_1, ..., z_n, z \in G,$$

on a :

$$\mathbf{P}(Z(t) \in B | Z(0) = z_0, Z(t_1) = z_1, ..., Z_{t_n} = z_n, Z(s) = z) = \mathbf{P}(Z(t) \in B | Z(s) = z)$$

$$\forall B \in \mathcal{G}.$$

A.2 Mouvement brownien

Définition A.2.1 *Soit (Ω, A, P) un espace probabilisé, on appelle mouvement brownien un processus stochastique $(X_t)_{t \geq 0}$ à valeur réelles, qui est un*

processus à accroissement indépendants et stationnaires dont les trajectoires sont continues. Ce la signifie que:

- *Continuité: $\mathbf{P}$ p.s la fonction $s \to X_s(\omega)$ est une fonction continue.*
- *indépendance des accroissements: si $s \leq t$, $X_t - X_s$ est indépendant de la tribu $\mathcal{F}_s = \sigma(X_u, u \leq s)$.*
- *stationnarité des accroissements: si $s \leq t$, la loi de $X_t - X_s$ est identique à celle de $X_{t-s} - X_0$.*

Cette définition permet de caractériser la loi de la variable aléatoire X_t

Théorème A.2.1 *Si $(X_t)_{t\geq 0}$ est un mouvement brownien, alors $X_t - X_0$ est une variable aléatoire gaussienne de moyenne rt et de variance $\sigma^2 t$, r et σ étant des constantes réelles.*

Preuve: la preuve de ce théorème découle facilement de la définition ci-dessus.

Définition A.2.2 *Un mouvement brownien X_t est dit standard si :*

$$X_0 = 0 \ \mathbf{P}.p.s. \ \mathbf{E}(X_t) = 0, \ \mathbf{E}(X_t^2) = t.$$

Appendice B

Résultats graphiques.

B.1 Résultats graphiques pour le premier modèle:

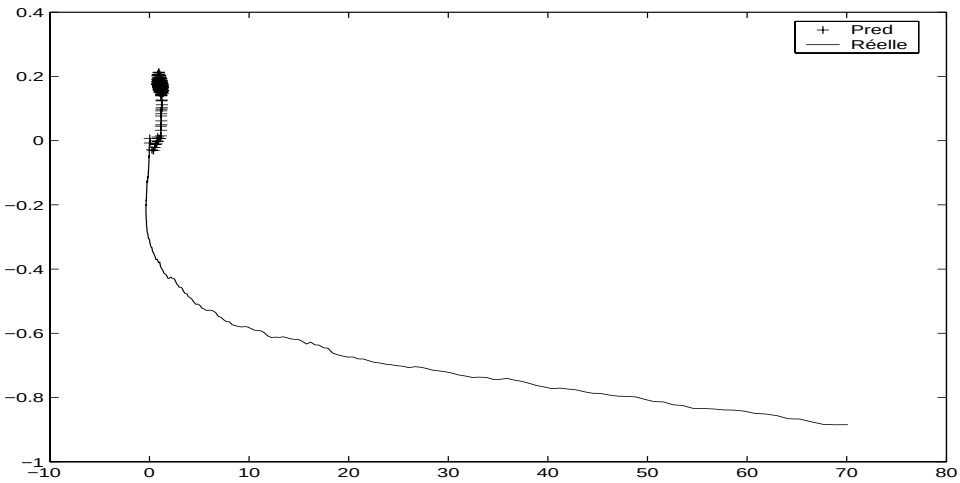

Figure B.1: Trajectoires réelle et prédite pour $\kappa = 10$.

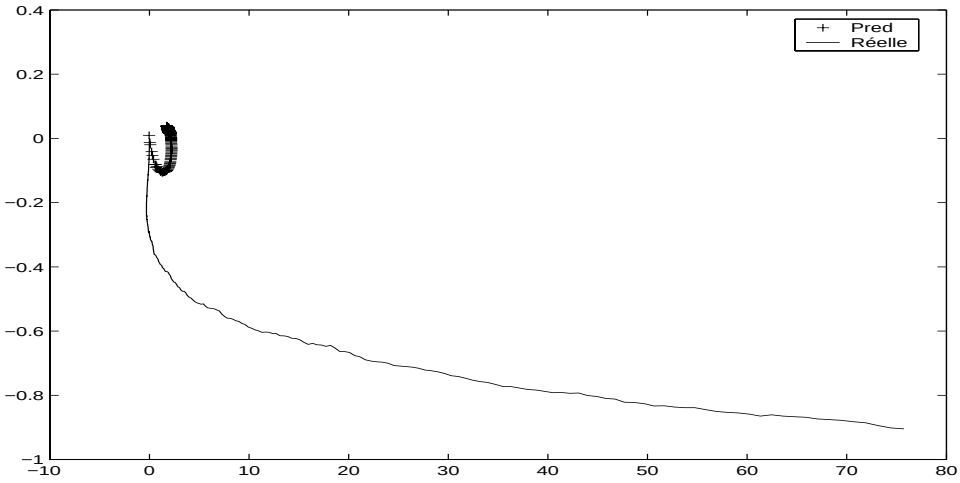

Figure B.2: Trajectoires réelle et prédite pour $\kappa = 15$.

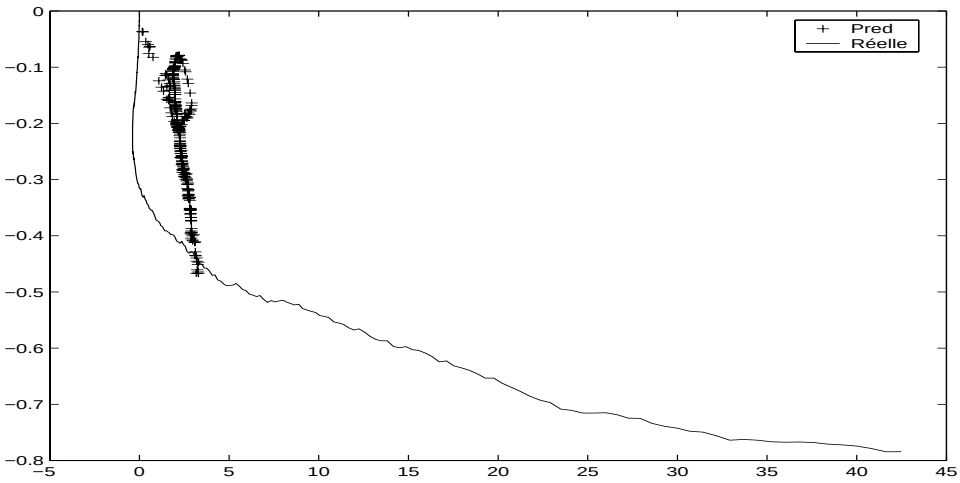

Figure B.3: Trajectoires réelle et prédite pour $\kappa = 30$.

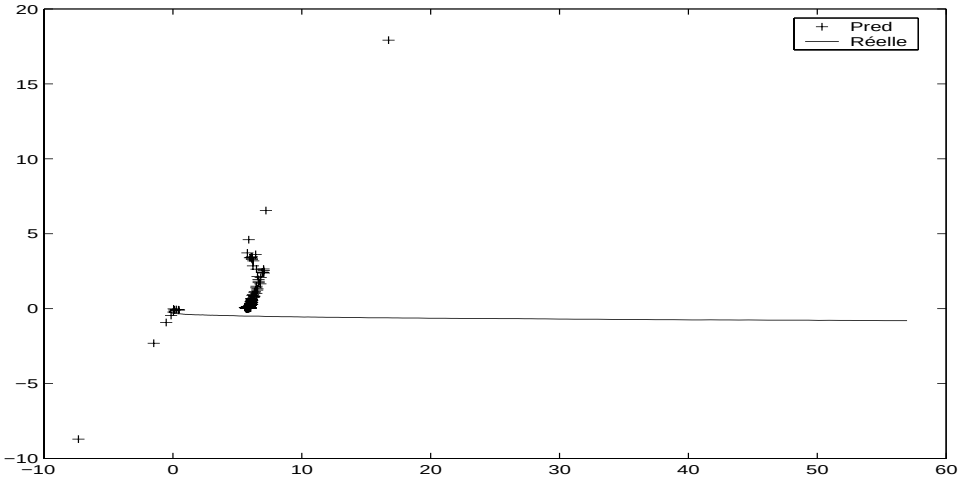

Figure B.4: Trajectoires réelle et prédite pour $\kappa = 50$.

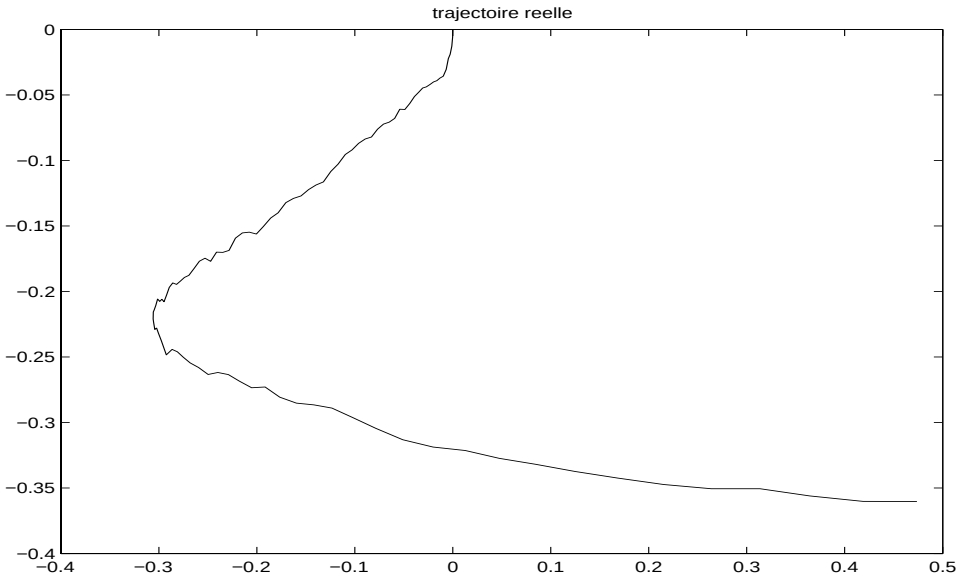

Figure B.5: Trajectoire réelle avec un nombre de points égal à 100.

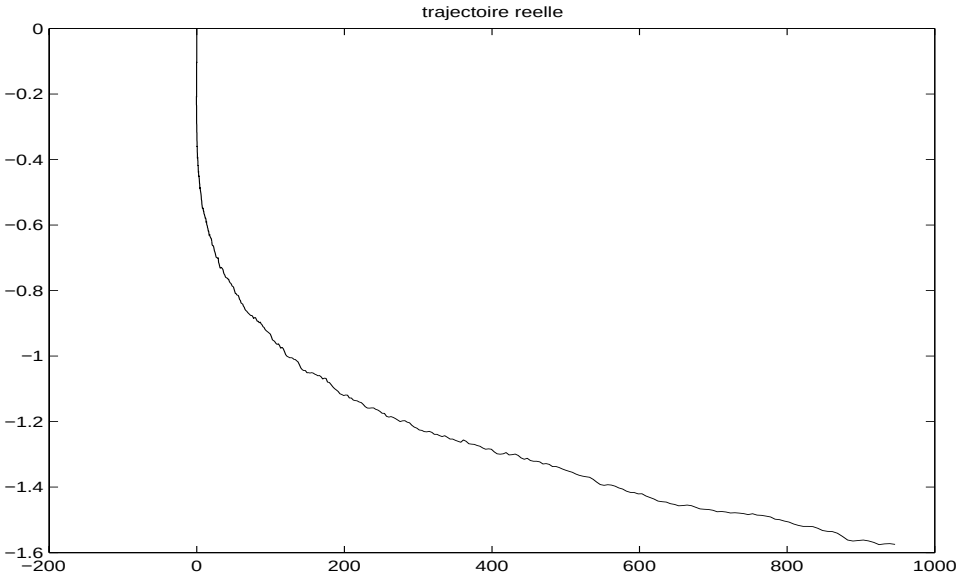

Figure B.6: Trajectoire réelle avec un nombre de points égal à 500.

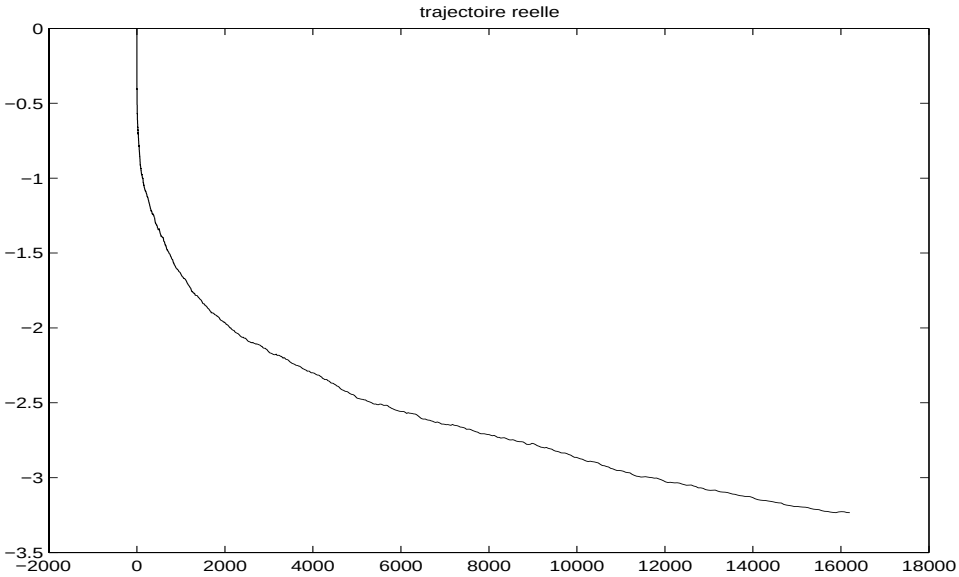

Figure B.7: Trajectoire réelle avec un nombre de points égal à 1000.

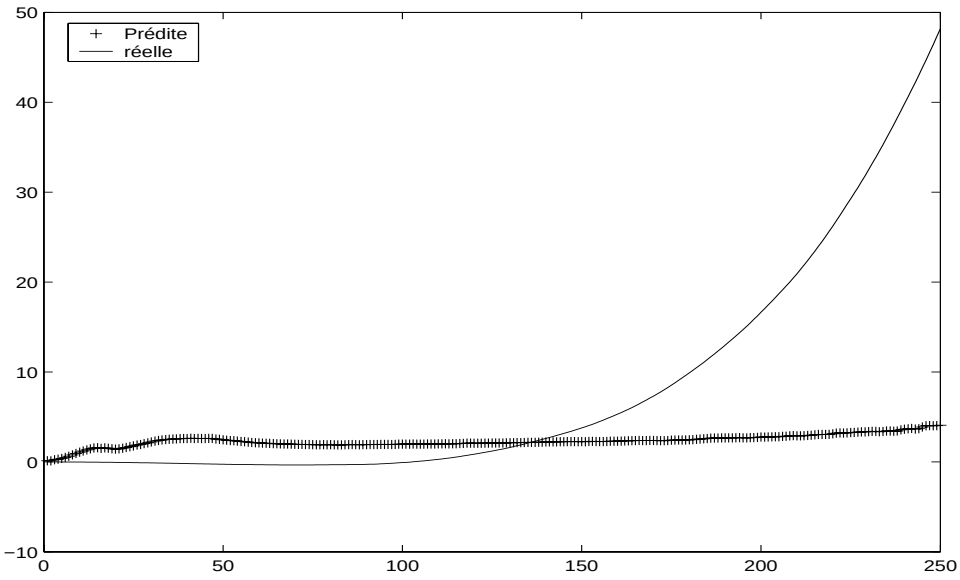

Figure B.8: Trajectoire réelle et prédite de l'abscisse x.

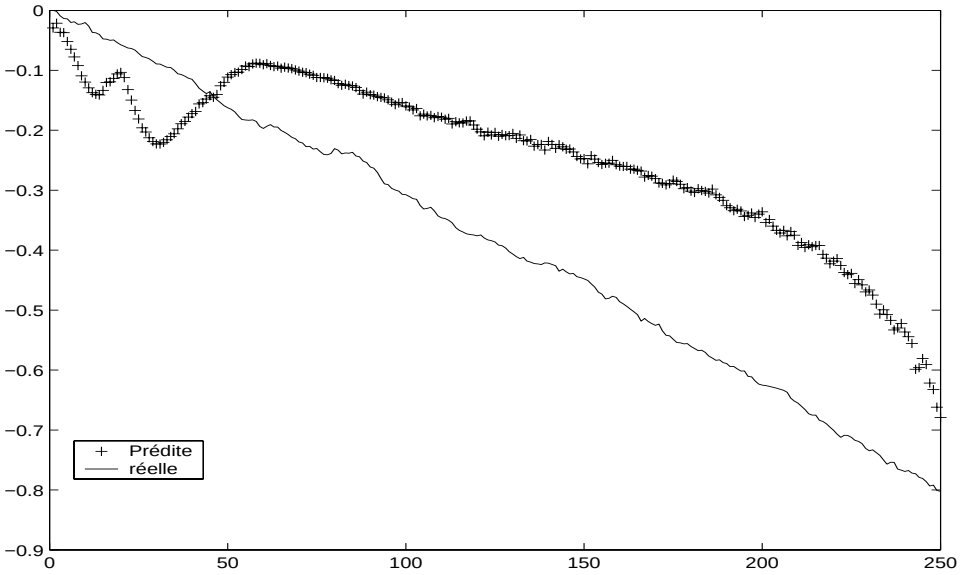

Figure B.9: Trajectoire réelle et prédite de l'ordonné y.

B.2 Résultats graphiques pour le deuxième modèle:

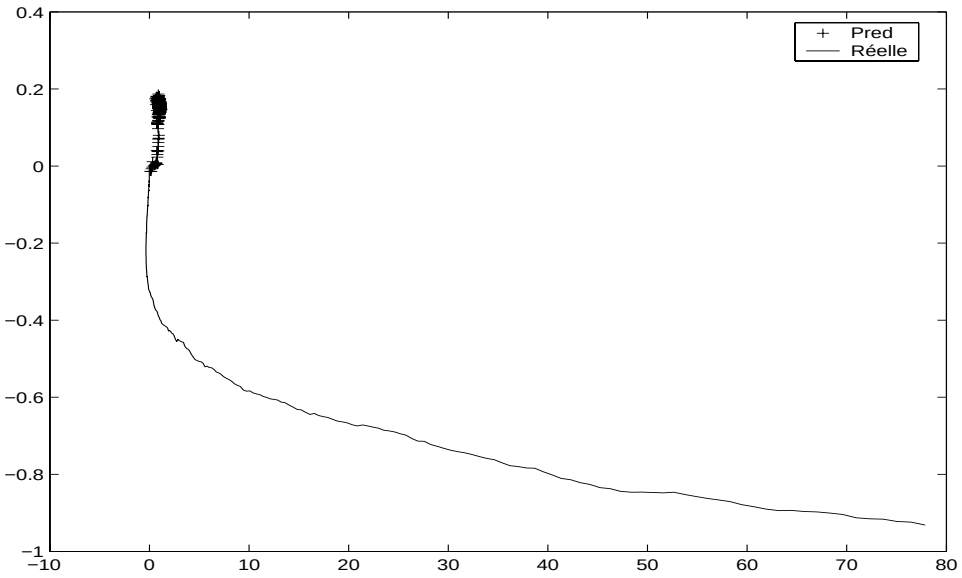

Figure B.10: Trajectoire réelle et prédite pour $\kappa = 10$.

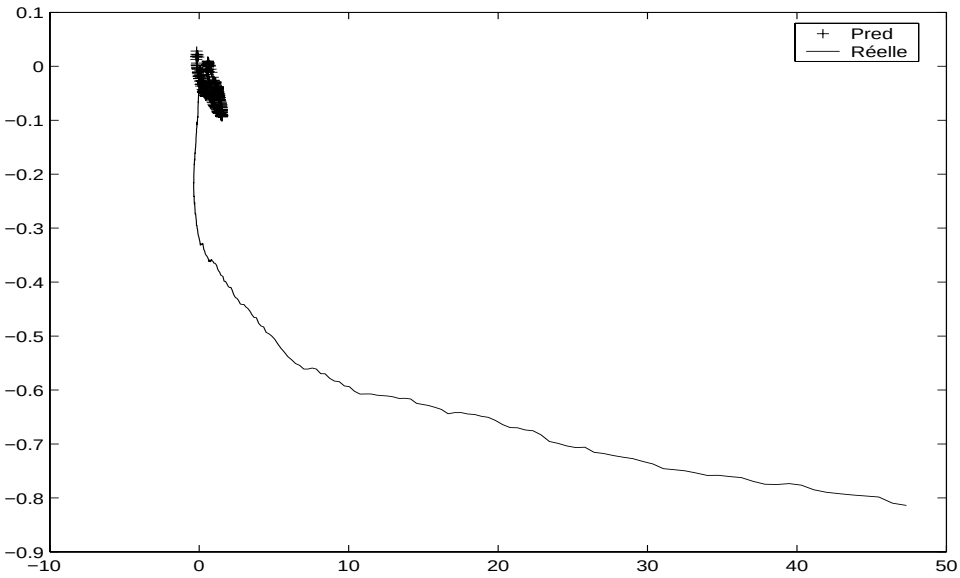

Figure B.11: Trajectoire réelle et prédite pour $\kappa = 15$.

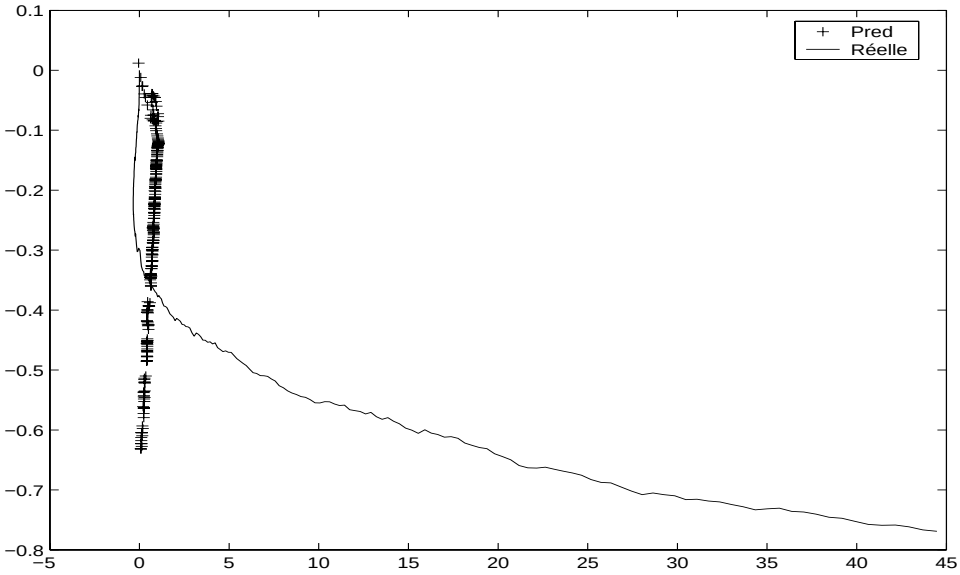

Figure B.12: Trajectoire réelle et prédite pour $\kappa = 30$.

B.3 Résultats graphiques pour le troisième modèle:

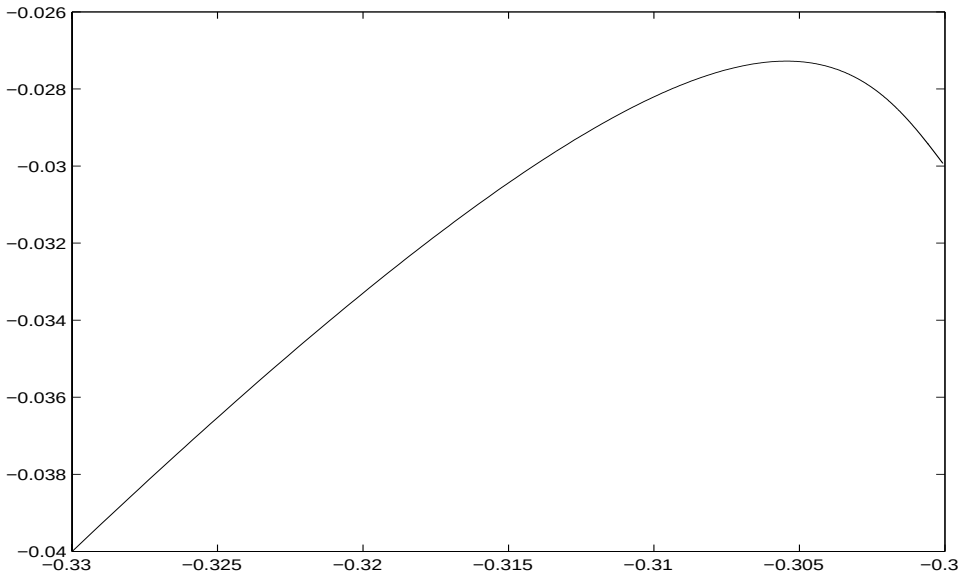

Figure B.13: Trajectoire réelle pour un nombre de points égales à 1000.

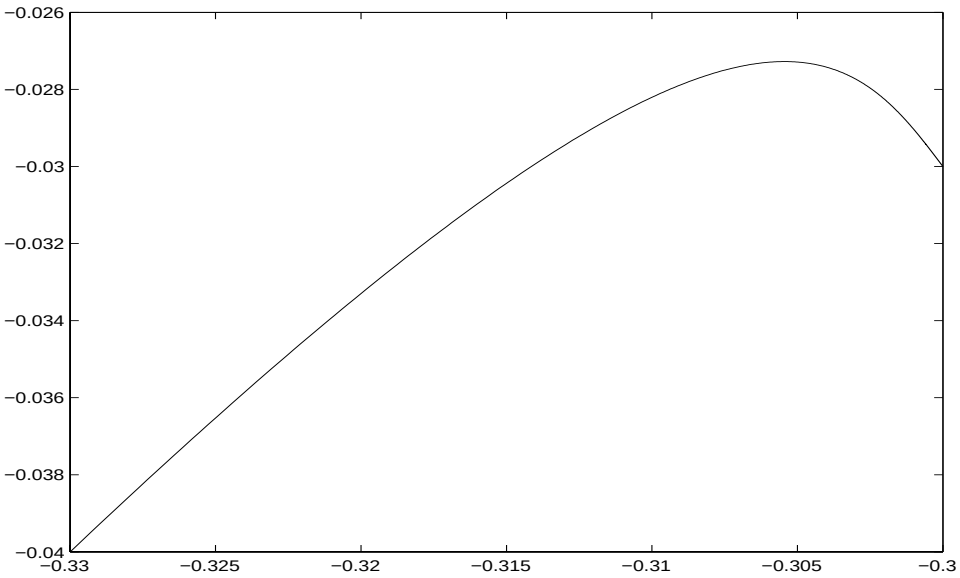

Figure B.14: Trajectoire réelle pour un nombre de points égales à 10000.

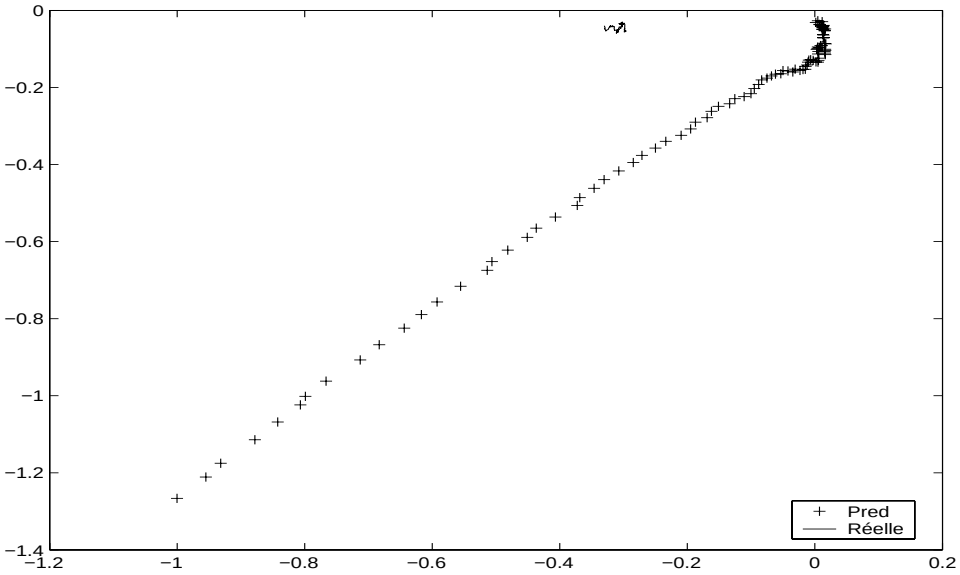

Figure B.15: Trajectoire réelle et prédite pour $\kappa = 10$.

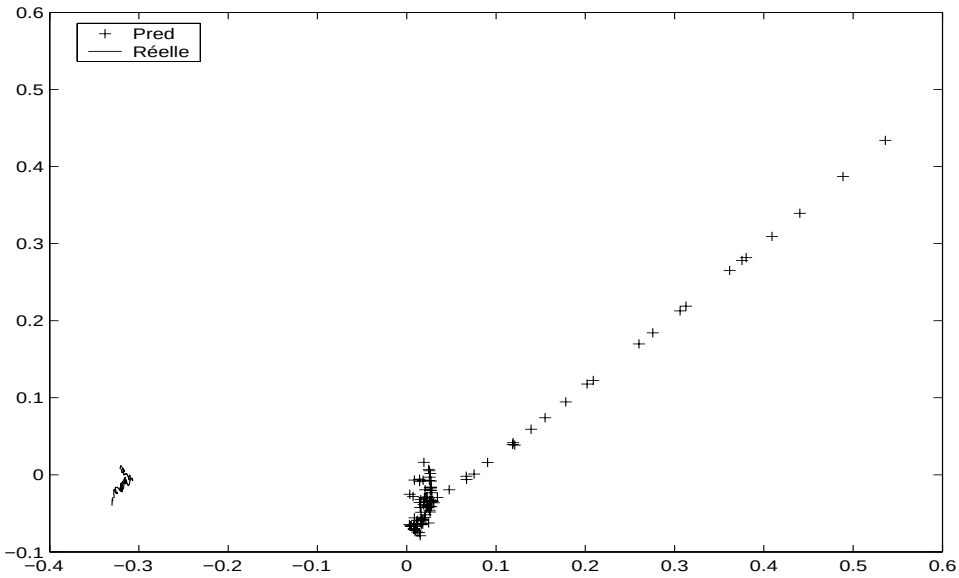

Figure B.16: Trajectoire réelle et prédite pour $\kappa = 15$.

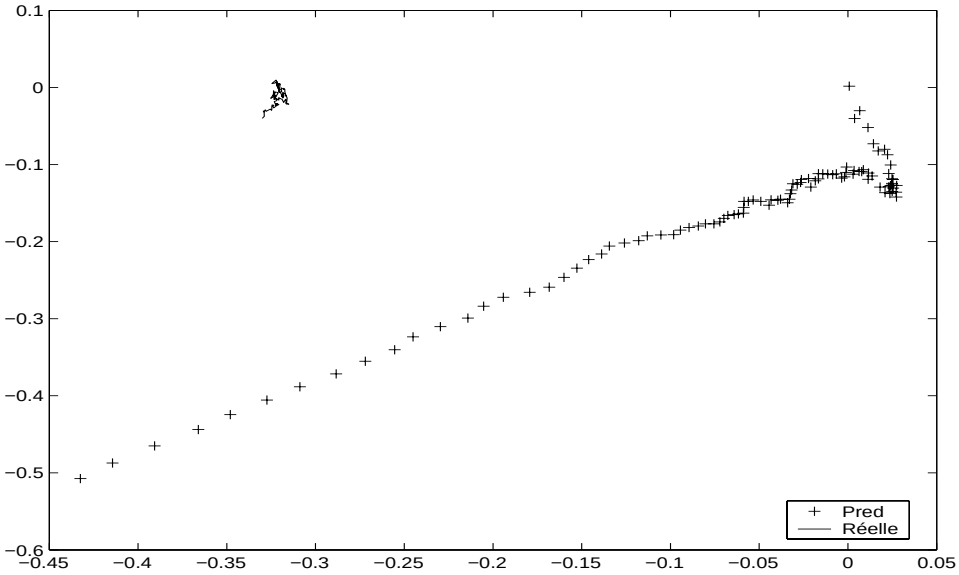

Figure B.17: Trajectoire réelle et prédite pour $\kappa = 30$.

Références

[1] Y. Bar-Shalom, K. Chang and H. Blom. Tracking a maneuvering target using input estimation versus the Interacting Multiple Model Algorithm. IEEE Transactions on Aeorospace and Electronic Systems, AES-25(2): 296-300, 1982.

[2] Y. Bar-Shalom. Multitarget-Multisensor Tracking : Application and Advances. Volume II. Artech House Boston. London : 1-28, 1992.

[3] J.F. Bennaton. Discrete Time Galerkin Approximation to the Nonlinear Filtering Solution. Journal of Mathematical Analysis and Applications, 110(2): 364-383, 1985.

[4] H. Bernier, F. Campillo , F. Cérou, F. Le Gland and Rivo Ratozafy. Parallélisme de données et Filtrage nom linéaire, Analyse de Performance. INRIA, Rapport technique N0167, 1994.

[5] J.C. Bertein and R. Ceschi. Processus stochastiques et filtrage de Kalman. Editions HERMES, Paris, 1998.

[6] Henk A. P. Blom and Y. Bar-Shalom. The interackting multiple model algorithm for systems whith Markovian switching coefficient. IEEE Transactions on Automatic Control, AC-33,8, 780-783, 1988

[7] Derbez, E. Jouan, A. and Remillard, B. (2000) A comparison of fixed gain IMM against two other filters, Fusion 2000, Paris, 7 pages.

[8] R.J. Elliott. Stochastic Calculs and Applications. Springer-Verlag New York, 1982.

[9] P. Florchinger and F. LeGland. Time discretisation of the Zakai Equation for Diffusion Processes Observed in correlated Noise. Stochastics and Stochastics Reports, 35, 233-256, 1991.

[10] C. P. Fung and S. Lototsky. Nonlinear Filtering: Separation of Parameters and Observations Using Galerkin Approximation and Wiener Chaos Decomposition, IMA Preprint Series 1458, February 1997.

[11] Andrew C. Harvey. Forcasting, Structural time series models and the Kalamn filter, Cambridge University Press, New York, 1989.

[12] Yozhong Hu, Gopinath Kaliampur and Jie Xiong. An Approximation for Zakai Equation. Applied Mathematics Optimization 45:23-44(2002).

[13] K. Itô. Approximation of the Zakai Equation for Nonlinear Filtering. SIAM Journal on Control and Optimisation, 34(2): 620-634, 1996.

[14] A.H. Jazwinski. Stochastic Process and Filtering Theory, Academic Press, New York, 1970.

[15] G. Kallianpur. Stochastic Filtering Theory. Springer-Verlag New York 1980.

[16] Michael A. Kouritzin. On Exact Filters dor Continuous Signals with Discrete Observation. IEEE Transactions On Automatic Control, Vol. 43, NO 5 May 1998.

[17] M. Kouritzin, B. Rémillard and C. Chan. Parameter Estimation, Volatility Estimation of a Mean Reverting Volatility Pricing Model. Proceedings of 4th Annual Conference on Information Fusion, Vol. I, WeB127-WeB130, 2001.

[18] M.A. Kouritzin, Exact infinite dimensional filters and explicit solutions, in Stochastic Models Eds. Luis G. Gorostiza and B. Gail Ivanoff (2000) 265–282.

[19] H.J. Kushner. Probability methods for approximations in stochastic control and for elliptic equations. New York, Academic Press , 1977.

[20] V. Lacoste. Wiener Chaos : A New Approch to Option Hedging. Mathematical Finance. Vol. 6, N2 (April 1996), 197-213.

[21] D. Lamberton and B. Lapeyre. Introduction au Calcul Stochastique appliqué à la finance, Ellipses, 1997.

[22] R.S. Lipster and A.N. Shiryayev. Statistics of Random Processes, volume I,II. Springer-Verlag, New York 1978.

[23] S. Lototsky, R. Mikulevicius and B.L. Rozovskyi. Nonlinear Filtering Revisited : A Speactral Approch. SIAM Journal on Control and Optimisation, 35(2), 435–461, 1997.

[24] S. Lototsky and B.L. Rozovskii. Recursive Multiple Wiener Integral Expansion for Nonlinear Filtering of Diffusion Processes, in stochastic processes and functional analysis. Lecture Notes in Pur and Applied Mathematics, vol. 186, J. Goldstein, N. Gretsky, and J. Uhl, Eds. New York: Marcel Dekker, 199–208, 1997.

[25] S. Lototsky and B.L. Rozovskii. Recursive Nonlinear Filter for a Continuous - Discrete Time Model: Separation of Parameters and Observations, IEEE Transactions on Automatic Control, Vol. 43, No. 8, 1154-1158, August 1998.

[26] R. Mikulevicius and B.L. Rozovskii. Separation of Observations and Parameters in Nonlinear Filtering. IEEE Control Systems Society, 1565–1559, 1993.

[27] D. Morrell. Extended Kalman Filter Lecture Notes. EEE 581-Spring 1997.

[28] Michael K. Pitt and Neil Shephard. Filtering via simulation: Auxiliary particle filter. Journal of the American Statistical Association, 94:590-599, 1999.

[29] E. Pardoux, Filtrage Non Linéaire et Équations aux Dérivées Partielles Stochastiques Associées, Ecole d'été de Probabilités de Saint-Flour, Spriger- Verlag, New York, Berlin 1989.

[30] E. Pardoux, et D. Tlay, Discretisation and simulation of Stochastic Differential Equations, Acta Applicandae Mathematicae, 3(1985), p. 23-47

[31] B. Rémillard. Méthodes Numériques en Ingénierie Financières. HEC, Note de Cours, 2002.