PATRICK DALLAIRE

Apprentissage par Renforcement Bayésien de processus décisionnels de Markov partiellement observables : une approche basée sur les processus Gaussiens.

Mémoire présenté
à la Faculté des études supérieures de l'Université Laval
dans le cadre du programme de maîtrise en informatique
pour l'obtention du grade de Maître ès sciences (M.Sc.)

Faculté des sciences et de génie
UNIVERSITÉ LAVAL
QUÉBEC

2010

Résumé

L'apprentissage par renforcement est une approche d'apprentissage automatique permettant de développer des systèmes s'améliorant à partir d'interactions avec un environnement. Les processus décisionnels de Markov partiellement observables (PDMPO) font partie des modèles mathématiques fréquemment utiliser pour résoudre ce type de problème d'apprentissage. Cependant, la majorité des méthodes de résolution utilisées dans les processus décisionnels de Markov partiellement observables nécessitent la connaissance du modèle. De plus, les recherches actuelles sur le PDMPO se restreignent principalement aux espaces d'états discrets, ce qui complique son application à certains problèmes naturellement modélisés par un espace d'état continu.

Ce mémoire présente une vision des PDMPO basée sur les processus Gaussiens, une méthode d'apprentissage supervisée ayant comme propriété particulière d'être une distribution de probabilité dans l'espace des fonctions. Cette propriété est notamment très intéressante du fait qu'elle ouvre la porte à un traitement Bayésien de l'incertitude sur les fonctions inconnues d'un PDMPO continu. Les résultats obtenus avec l'approche d'apprentissage par processus Gaussien montrent qu'il est possible d'opérer dans un environnement tout en identifiant le modèle de ce celui-ci.

À partir des conclusions tirées à la suite de nos travaux sur le PDMPO, nous avons observé un certain manque pour ce qui est de l'identification du modèle sous l'incertain. Ainsi, ce mémoire expose aussi un premier pas vers une extension de l'apprentissage de PDMPO continu utilisant des séquences d'états de croyances lors de l'identification du modèle. Plus précisément, nous proposons une méthode de régression par processus Gaussiens utilisant des ensembles d'entraînement incertain pour réaliser l'inférence dans l'espace des fonctions. La méthode proposée est particulièrement intéressante, du fait qu'elle s'applique exactement comme pour le cas des processus Gaussiens classiques et qu'elle n'augmente pas la complexité de l'apprentissage.

Avant-propos

Par la présente, j'aimerais offrir mes remerciements les plus sincères à tous ceux qui m'ont appuyé depuis mon arrivée au laboratoire DAMAS.

Tout d'abord, je dois remercier la personne m'ayant offert la chance de découvrir la recherche, Prof. Brahim Chaib-draa. La confiance qu'il m'a témoignée au cours des dernières années fût des plus encourageantes et je lui en suis grandement reconnaissant. De plus, son aide et ses conseils ont toujours été fort appréciés et ont très certainement permis de faire avancer mes recherches. Par ailleurs, je remercie les membres du laboratoire DAMAS pour les nombreuses discussions enrichissantes qui ont permis de faire évoluer les idées. Un merci tout particulier à Jean-Samuel Marier et Camille Besse qui ont été des voisins de bureau incroyables et avec qui j'ai passé du temps exceptionnel.

Finalement, je remercie du fond du coeur le soutien de ma mère Louisette et ma tante Fleurette depuis le début de mes études universitaires. Elles ont été présentes au quotidien pour m'appuyer et m'ont donné l'énergie de poursuivre surtout dans les moments difficiles.

Patrick Dallaire

Table des matières

Résumé ii

Avant-propos iii

Table des matières iv

Table des figures vi

1 Introduction 1
- 1.1 Problématique et motivation . . . 5
- 1.2 Méthode et contribution . . . 6
- 1.3 Plan du mémoire . . . 6

2 Introduction aux processus Gaussiens 7
- 2.1 Historique . . . 7
- 2.2 L'apprentissage Bayésien . . . 8
  - 2.2.1 L'apprentissage hiérarchique . . . 9
  - 2.2.2 Approximation de la distribution a posteriori . . . 10
- 2.3 Les processus Gaussiens . . . 11
  - 2.3.1 Définition . . . 11
  - 2.3.2 Les fonctions de covariance . . . 12
  - 2.3.3 Les fonctions de covariance stationnaire . . . 13
  - 2.3.4 Les fonctions de covariance non-stationnaire . . . 20
- 2.4 Régression par processus Gaussiens . . . 25
  - 2.4.1 La prédiction . . . 27
- 2.5 L'apprentissage des hyperparamètres . . . 29
  - 2.5.1 Maximum de vraisemblance a posteriori . . . 31
  - 2.5.2 Identification automatique de la pertinence . . . 33
- 2.6 Experimentations . . . 34
  - 2.6.1 Exemple illustratif . . . 34
  - 2.6.2 Série temporelle de Mackey-Glass . . . 35
- 2.7 Conclusion . . . 40

3 Apprentissage de PDMPO **41**
3.1 Introduction 41
3.2 Travaux connexes 43
3.3 Processus décisionnels de Markov 44
3.4 PDMPO continu 48
3.5 Application des processus Gaussien pour les PDMPO continus 50
3.5.1 Apprentissage du modèle par processus Gaussiens 52
3.5.2 Planification en ligne 57
3.6 Experimentations 60
3.7 Conclusion 64

4 Apprentissage de processus Gaussien avec données incertaines **67**
4.1 Introduction 67
4.2 Les processus Gaussiens et l'incertain 69
4.2.1 Traitement naïf de l'incertitude 71
4.2.2 L'espérance de la fonction de covariance 72
4.2.3 L'espérance de la fonction de moyenne 76
4.2.4 L'incertitude en sortie 77
4.2.5 Apprentissage des hyperparamètres 78
4.3 Experimentations 79
4.3.1 Le problème artificiel sincsig 79
4.3.2 Le problème du pendule inversé 83
4.4 Discussion 87
4.5 Conclusion 91

5 Conclusion **93**
5.1 Contributions 94
5.2 Travaux futurs 95

Bibliographie **97**

A Dérivées partielles pour apprentissage de PDMPO **102**

B Dérivation de fonction d'espérance de covariance **105**
B.1 Fonction de covariance linéaire espérée 105
B.2 Fonction de covariance quadratique espérée 106
B.3 Fonction de covariance carré-exponentielle espérée 107

Table des figures

2.1 Fonctions échantillonnées à partir de processus Gaussien utilisant la carré-exponentielle avec différent hyperparamètres. Les fonctions en trait plein ont été générées avec $a = 2$, $\lambda = 0.5$ et les fonctions en pointillés avec $a = 0.75$, $\lambda = 2.25$. La fonction de moyennes est $\mu(x) = 0$. 14

2.2 Fonctions échantillonnées à partir de processus Gaussien utilisant la γ-exponentiel avec différent hyperparamètres. Les fonctions en pointillés ont été générées avec $\gamma = 1.8$ et les fonctions en trait plein avec $\gamma = 1$. Les autres paramètres ont été fixés à $\lambda = 1$ et $a = \sqrt{3}$. La fonction de moyennes est $\mu(x) = 0$. 16

2.3 Fonctions échantillonnées à partir de processus Gaussien utilisant la fonction Matérn avec différent hyperparamètres. Les fonctions en pointillés ont été générées avec $\nu = 3/2$ et les fonctions en trait plein avec $\nu = 1$. Les autres paramètres ont été fixés à $\lambda = 1$ et $a = \sqrt{3}$. La fonction de moyennes est $\mu(x) = 0$. 17

2.4 Fonctions échantillonnées à partir de processus Gaussien utilisant la fonction quadratique rationnelle avec différent hyperparamètres. Les fonctions en pointillés ont été générées avec $\alpha = 2$ et les fonctions en trait plein avec $\alpha = 1/10$. Les autres paramètres ont été fixés à $\lambda = 1$ et $a = \sqrt{3}$. La fonction de moyennes est $\mu(x) = 0$. 18

2.5 Fonctions échantillonnées à partir de processus Gaussien utilisant la fonction de covariance linéaire avec différents hyperparamètres. Les fonctions en pointillés ont été générées avec $\sigma_b^2 = 3$ et les fonctions en trait plein avec $\sigma_b^2 = 0$. Les autres paramètres ont été fixés tel que Σ_w est la matrice identité. La fonction de moyennes est $\mu(x) = 0$. 21

2.6 Fonctions échantillonnées à partir de processus Gaussien utilisant la fonction de covariance polynomiale. Les fonctions ont été générées avec $\sigma_b^2 = 0.5$, $p = 5$ et Σ_w diagonale de valeur 0.1. La fonction de moyennes est $\mu(x) = 0$. 23

2.7 Fonctions échantillonnées à partir de processus Gaussien utilisant la fonction de covariance réseau de neurones. Les courbes en pointillés ont été générées avec $\Sigma = \text{diag}(0.1, 10)$ et celles en trait plein avec $\Sigma = \text{diag}(100, 100)$. La fonction de moyennes est $\mu(x) = 0$. 24
2.8 Fonctions échantillonnées à partir de processus Gaussien utilisant la fonction de covariance exponentielle sur un espace périodique. Les courbes ont été générées avec $a = \sqrt{3}$ et $\lambda = \sqrt{0.2}$. La fonction de moyennes est $\mu(x) = 0$. 25
2.9 Fonction de vraisemblance des hyperparamètres d'un processus Gaussien utilisant la fonction carré-exponentielle où l'ensemble de données a été échantillonnée avec $a = 0.85$ et $\lambda = 0.65$. 32
2.10 Fonction de vraisemblance des facteurs d'échelle d'un processus Gaussien utilisant la fonction carré-exponentielle où l'ensemble de données contient une dimension non pertinente. 34
2.11 Illustration de l'apprentissage par processus Gaussien. La courbe en trait plein correspond à la fonction recherchée et les croix indiquent les exemples utilisés lors de l'entraînement. Les zones grises représentent des intervalles de confiance de 3 écarts-types. 36
2.12 Serie temporelle de Mackey-Glass avec $\beta = 0.2$, $\gamma = 0.1$ et un retard $\tau = 17$. 37
2.13 Prédictions de 200 pas de temps sur les différents intervalles de l'ensemble test. 38
2.14 Racine carrée de l'erreur quadratique moyenne normalisée en fonction de la longueur de l'intervalle de prédiction. 39

3.1 Adaptation du modèle dynamique processus Gaussien incluant actions et récompenses. 54
3.2 Récompense moyenne reçue. 63
3.3 Distance moyenne à la hauteur requise. 64
3.4 Erreur moyenne de prédiction. 65
3.5 Quantile à 2.5%, 25%, 50%, 75%, 97.5% de la distribution des trajectoires. Les moustaches contiennent donc 95% des données, les boîtes 50% et le trait central représente la médiane. 66

4.1 Distribution de probabilité $p(\mathbf{K})$ sur la matrice de covariance $\mathbf{K}$ d'un processus Gaussien à 3 variables aléatoires. Les sous-figures sont disposées de façon à représenter une matrice où les éléments correspondent aux distributions $p(k(x_i, x_j))$. Ainsi, les abscisses contiennent les covariances potentielles et les ordonnées indiquent la vraisemblance des covariances. 74

4.2 Distribution de probabilité de la covariance entre des examples à $x_i \sim \mathcal{N}(0,1)$ et $x_j \sim \mathcal{N}(1,1)$ sous une fonction de covariance carré-exponentielle dont tout les paramètres sont fixés à 1. L'espérance de la covariance donnée par k_{ECE} pour ces exemples est de 0.5. 76
4.3 La fonction sincsig avec les apprentissages typiques d'un PG-incertain et d'un PG-naïf. Les traits fins sont les fonctions de moyenne a posteriori entourées de leurs barres d'erreur donnée par la variance a posteriori. Les croix sont (uniquement) les moyennes des données d'entraînement. . . . 81
4.4 Résultats pour le problème d'apprentissage de la fonction sincsig. En trait rouge avec des carrés (PG-incertain), trait bleu avec des cercles (PG-naïf) et trait vert avec des triangles (Réseau de neurones). 82
4.5 Résultats de l'erreur quadratique moyenne. $\sigma_{x_i} \sim \mathcal{U}(0,v)$ et $\sigma_{y_i} \sim \mathcal{U}(0,v)$. En trait rouge avec des carrés (PG-incertain) et en trait bleu avec des cercles (PG-naïf). 84
4.6 Résultats des log vraisemblances. $\sigma_{x_i} \sim \mathcal{U}(0,v)$ et $\sigma_{y_i} \sim \mathcal{U}(0,v)$. En trait rouge avec des carrés (PG-incertain) et trait bleu avec des cercles (PG-naïf). 85
4.7 Illustration du problème du pendule inversé. L'objectif est de maintenir le pendule à la verticale par des mouvements latéraux. 86
4.8 Résultats de l'erreur quadratique moyenne sur le problème du pendule inversé. En trait rouge avec des carrés (PG-incertain), trait bleu avec des cercles (PG-naïf) et trait vert avec des triangles (Réseau de neurones). 88
4.9 Processus Gaussien a posteriori typique pour la fonction sinus cardinal. 90

Chapitre 1

Introduction

L'apprentissage automatique est la discipline scientifique qui cherche à développer des systèmes évoluant automatiquement avec l'expérience et à comprendre les lois fondamentales sous-tendant les processus d'apprentissage [Mitchell, 1997]. Les recherches dans ce domaine ont mené à de multiples applications importantes telles que la reconnaissance automatique du langage, la vision numérique, la biosurveillance, la classification de séquence d'ADN, le contrôle robotique, etc. Aujourd'hui, l'apprentissage automatique est une dimension importante de l'informatique, car elle met de l'avant un paradigme de programmation différent de la méthode traditionnelle qui consiste à développer des programmes manuellement. Pour certaines applications, cette méthode traditionnelle est largement impraticable, notamment lorsqu'un trop grand nombre de cas particuliers doit être implanté, ou pire, lorsque la solution est inconnue. L'approche que propose l'apprentissage automatique consiste plutôt à permettre à un programme de se développer automatiquement à partir d'exemples afin d'améliorer ses performances à réaliser une tâche grâce à l'expérience acquise.

Il existe actuellement une multitude d'algorithmes d'apprentissage qui sont généralement catégorisés par rapport au mode d'apprentissage adopté. Ainsi, face à un problème particulier, il est nécessaire de choisir la forme d'apprentissage adaptée aux types de données que l'on doit traiter. Bien entendu, un même problème peut être abordé sous plusieurs angles et, par conséquent, plus d'une méthode d'apprentissage peut s'avérer pertinente à sa résolution. Les principales formes d'apprentissage automatiques sont : l'apprentissage supervisé, l'apprentissage non-supervisé, l'apprentissage semi-supervisé et l'apprentissage par renforcement. Nous allons maintenant les passer en revue.

L'apprentissage supervisé

En contexte d'apprentissage supervisé, l'objectif est d'apprendre *une fonction* expliquant au mieux la relation existant entre des données d'entrée et de sortie. Lorsque les données de sortie appartiennent à un espace continu, il s'agit de problèmes de *régression*, et lorsqu'elles appartiennent à un espace discret, de problèmes de *classification*. Un problème de classification classique est la reconnaissance de caractères manuscrits, utilisé entre autres par certains services des postes pour identifier les codes postaux écrits à la main. Dans la catégorie des problèmes de régression, nous avons l'analyse de séries temporelles telle que la variation de la valeur des actions à la bourse. De façon générale, la fonction estimée sert à *généraliser* les connaissances acquises pour des fins de prédiction, où l'objectif est d'obtenir une estimation de la valeur en sortie pour des entrées ne figurant pas dans l'ensemble d'entraînement. Par ailleurs, les paramètres du modèle obtenus suite à l'apprentissage peuvent aussi servir à l'interprétation des données lorsqu'ils ont une signification dans le contexte du problème.

Ce qui caractérise les problèmes d'apprentissage supervisé est la forme particulière des données à traiter. Les exemples d'entraînement sont en fait des couples (x, y) provenant d'une fonction $f : X \to Y$ qu'il faut estimer. L'approche classique consiste d'abord à paramètrer la relation f entre les entrées et les sorties par un vecteur de paramètres $\mathbf{w}$ de sorte que $y = f(x|\mathbf{w})$, et à définir ensuite une fonction permettant d'évaluer la qualité d'un vecteur $\mathbf{w}$ particulier. Ainsi, l'apprentissage supervisé se traduit souvent en un problème d'*optimisation* de fonction d'erreur (ou de qualité) par rapport aux paramètres du modèle. Dès lors, les méthodes de montée et de descente de gradient sont fréquentes pour leur résolution, car très pratiques face à ce type de problème. Il est donc judicieux de choisir une fonction d'erreur qui soit dérivable afin d'obtenir un gradient, citons par exemple la fonction d'erreur quadratique en régression linéaire.

L'apprentissage non-supervisé

Cette catégorie regroupe les problèmes pour lesquels il faut identifier des structures dans les données. Par exemple, il peut s'agir de découvrir des groupes de données similaires parmi un ensemble donné, tâche que l'on nomme *partitionnement des données*. Un autre problème important consiste à déterminer la distribution des données dans l'espace auquel ces mêmes données appartiennent. Ce problème est mieux connu sous le nom d'*estimation de densité* en statistiques et probabilités. Pour sa part, la *réduction de la dimensionnalité* cherche à découvrir une projection des données vers un espace de plus faible dimension, souvent pour des fins de visualisation des données ou pour

réduire la complexité de calcul d'un problème.

Contrairement à l'apprentissage supervisé, les données traitées en apprentissage non-supervisé ne sont pas sous la forme de couple (x, y), mais simplement sous forme d'entrée x. Les données sont généralement supposées provenir d'une distribution jointe inconnue $p(x)$ et dont l'estimation peut s'avérer essentielle à la résolution du problème. Une méthode fréquemment utilisée pour modéliser cette distribution de probabilité consiste à la représenter par une somme de distributions de probabilités Gaussiennes, connues sous le nom de *mélange de Gaussiennes*. Cette méthode possède plusieurs propriétés intéressantes. Par exemple, lorsqu'il faut partitionner les données en divers groupes, il s'agit d'introduire un certain nombre de distributions Gaussiennes qui seront réparties de sorte que chacune représente un groupe particulier. On remarque ici que la quantité à introduire est potentiellement inconnue, car il faut possiblement identifier le nombre de groupes présent, ce qui correspond en fait à une *variable cachée* du problème. Ce type de variables est courant en apprentissage non-supervisé et l'estimation de celles-ci représente généralement une partie importante quant à la résolution du problème.

L'apprentissage semi-supervisé

Un problème d'apprentissage automatique se caractérise principalement par la forme des données destinées à l'entraînement du modèle et par le type d'information que l'on doit en extraire. Bien entendu, il est aussi possible que les données d'entraînement aient plusieurs formes. C'est le cas de l'apprentissage semi-supervisé, une classe de problème à cheval entre l'apprentissage supervisé et non-supervisé. Cette catégorie fait référence aux problèmes pour lesquels l'apprentissage se fait à partir de couples entrée-sortie dont toutes les entrées sont connues, mais dont les sorties ne sont connues que pour une partie des exemples. Cette forme d'apprentissage généralise donc les deux formes précédentes du fait que les cas extrêmes où toutes les données sont étiquetées, c'est-à-dire que les sorties sont connues, et où aucune n'est étiquetée reviennent respectivement aux formes supervisées et non-supervisées.

Le principal intérêt de l'apprentissage semi-supervisé est d'économiser la ressource permettant d'étiqueter les données en intégrant les données non étiquetées à l'apprentissage. Par exemple, déterminer le sujet d'un texte quelconque est une tâche qui requiert du temps de la part d'un humain. Par conséquent, fournir une grande quantité d'exemples de texte dont le sujet a été prédéterminé peut s'avérer une procédure coûteuse. D'un autre côté, les textes non étiquetés peuvent être disponibles en très grand nombre pour des fins d'apprentissage. L'apprentissage semi-supervisé intervient donc afin d'exploiter à la fois l'information fournit par les quelques exemples de textes

dont le sujet est connu, mais surtout, extraire un maximum d'information à partir des plusieurs textes sans sujet connu.

L'apprentissage par renforcement

L'apprentissage par renforcement se distingue des autres formes d'apprentissages du fait que l'apprentissage ne se fait pas directement à partir d'une base d'exemples, mais plutôt par essais et erreurs où la récompense joue un rôle d'évaluation instantané. La formulation générale des problèmes d'apprentissage par renforcement fait généralement appel à la notion d'*agent intelligent* devant exécuter séquentiellement des actions dans un environnement. L'objectif de l'agent est donc d'apprendre la *politique* d'actions à exécuter qui permet de maximiser à long terme les récompenses accumulées.

Ce type d'apprentissage est différent de l'apprentissage supervisé du fait que l'*action optimale* n'est pas fournie à l'agent par un superviseur, mais doit être apprise par expérience via des essais-erreurs. Dès lors, l'agent doit réaliser un compromis entre diversifier ses expériences dans le but de s'améliorer (exploration) et chercher à obtenir un maximum de récompenses en utilisant ses connaissances actuelles (exploitation). L'environnement de l'agent est généralement décrit par un processus décisionnel de Markov (PDM), car ce type d'environnement possède d'excellentes propriétés par rapport aux calculs de politiques et respecte les exigences de plusieurs problèmes de décisions séquentielles. Dans la mesure où les dynamiques de l'environnement sont connues, cette hypothèse permet entre autres de calculer la *politique optimale* par programmation dynamique. Cependant, les processus décisionnels de Markov ont certaines limites quant à leur expressivité, notamment pour la modélisation de l'incertitude relative à l'observabilité partielle de l'environnement.

Le processus décisionnel de Markov partiellement observable (PDMPO) est un modèle plus général qui, contrairement au processus décisionnel de Markov classique, permet de modéliser l'observation partielle de l'environnement. L'amélioration passe essentiellement par l'ajout d'un modèle d'observation partiel de l'état de l'agent, ce qui introduit naturellement de l'incertitude par rapport à l'état réel du système. Par ailleurs, l'incertitude intrinsèque aux PDMPO rend le calcul de la politique optimale beaucoup plus complexe que pour un PDM. Des adaptations de méthodes par programmation dynamique ont été proposées et offrent d'excellentes approximations, car dans les faits, le calcul de la politique optimale devient très difficile pour des problèmes de grande taille.

1.1 Problématique et motivation

Les processus décisionnels de Markov partiellement observables offrent un cadre théorique d'une grande flexibilité pour modéliser les problèmes de décisions séquentiels. Un exemple intéressant est celui du diagnostic médical où la condition de santé d'un patient (l'état) n'est pas directement observable et doit être estimée en fonction de symptômes et d'examens médicaux (les observations). L'objectif dans ce contexte est de trouver la série d'examens et de traitements (les actions) qui permettront d'améliorer la condition à long terme du patient (la récompense). Évidemment, le calcul d'une politique d'examens et traitements optimale n'est pas tractable dans le cas général en raison du très grand nombre de variables.

Les méthodes de résolution de PDMPO actuelles permettent d'obtenir une bonne estimation de la politique optimale. Cependant, la grande majorité de ces méthodes suppose que les espaces d'états, d'actions et d'observations sont discrets. Pour plusieurs problèmes, comme celui présenté précédemment, cette hypothèse n'est pas trop contraignante dans la mesure où le problème se discrétise bien. Pour d'autres, tels que les problèmes de contrôle robotique, les espaces concernés sont fondamentalement continus. Dès lors, il est préférable de préserver cette propriété du modèle et ne pas forcer une discrétisation pouvant détériorer la qualité de la solution. Quelques approches ont été proposées afin de prendre en considération la continuité, mais malheureusement elles sont souvent incomplètes (un seul espace est continu) ou elles font des hypothèses fortes sur le modèle (le modèle est linéaire).

Une seconde limitation dont souffre la majeure partie des approches actuelles est l'hypothèse que le modèle de l'environnement est entièrement connu. Ainsi, aucune adaptation n'est possible au cours de l'apprentissage afin de s'ajuster à une variation des paramètres du modèle. Un exemple simple est la détérioration à long terme d'un senseur dont le niveau de bruit peut augmenter, mais reste toujours informatif sur l'état. Il existe plusieurs approches permettant de traiter ce cas particulier, mais par contre, les approches se font plus rares lorsqu'il est question de variation des paramètres d'un modèle, et ce, d'autant plus lorsqu'il s'agit d'apprendre le modèle d'un PDMPO.

L'intérêt pour une méthode efficace pouvant à la fois traiter les PDMPO à espaces continus et identifier en même temps le modèle de l'environnement est sans équivoque. En effet, une telle méthode offrirait à un système intelligent une meilleure capacité d'adaptation dans un environnement continu. En particulier, elle pourrait minimiser l'impact d'une mauvaise implantation initiale causée par des connaissances a priori généralement imprécises du modèle. C'est dans ce contexte que se situent ces travaux

de maîtrise.

1.2 Méthode et contribution

Ce mémoire propose deux contributions principales au domaine de l'apprentissage automatique. En premier lieu, l'élaboration d'une approche permettant l'apprentissage de processus décisionnel de Markov partiellement observable continu. La "mécanique" de l'algorithme se fonde essentiellement sur des méthodes d'apprentissage Bayésienne où les *processus Gaussiens* sont utilisés afin d'apprendre le modèle. Un algorithme de planification en ligne exploite ensuite les connaissances extraites par les processus Gaussiens afin de prendre les décisions dans l'environnement.

En second lieu, nous proposons une extension à l'apprentissage par processus Gaussiens. À l'origine, cette extension vise à combler un manque au niveau de l'algorithme d'apprentissage de PDMPO continu afin de le rendre plus robuste face à l'incertitude. Nous avons donc proposé dans ce contexte une méthode permettant aux processus Gaussiens d'apprendre à partir de données incertaines autant en entrée qu'en sortie. Un avantage net à la méthode en question est qu'elle s'emploie exactement comme dans le cas classique et donc n'augmente pas la complexité des calculs. Notons que cette contribution concerne plus précisément l'apprentissage supervisé et qu'elle dépasse largement l'apprentissage de PDMPO continu. L'apprentissage à partir d'exemples d'entraînement entièrement incertains est un problème fondamental et est destiné à des usages multiples.

1.3 Plan du mémoire

Ce mémoire est organisé comme suit. Le chapitre 2 introduit l'apprentissage par processus Gaussiens dans le cadre de la régression. Nous expliquons comment les processus Gaussiens peuvent être utilisés en tant que distribution de probabilités a priori dans un cadre d'apprentissage Bayésien. Le chapitre 3 détaille le développement de la méthode d'apprentissage de PDMPO continu via les processus Gaussiens. Des résultats obtenus sur le problème de contrôle d'un dirigeable y sont présentés. Le chapitre 4 décrit l'adaptation des processus Gaussiens pour l'apprentissage à partir de données incertaines. Le sujet de la covariance incertaine y est couvert ainsi que la dérivation de fonction de covariance incertaine. Finalement, le chapitre 5 conclut et couvre certains travaux futurs prolongeant les travaux initiés dans ce même mémoire.

Chapitre 2

Introduction aux processus Gaussiens

Ce chapitre introduit les processus Gaussiens en tant que distribution de probabilité sur un espace de fonctions pour des fins d'apprentissages supervisés. L'utilisation des processus Gaussiens permet d'atteindre des performances équivalentes à celles de l'état de l'art en matière d'apprentissage de modèles non-linéaires. Le fait d'utiliser les processus Gaussiens comme méthode d'apprentissage non-paramétrique permet une grande flexibilité lors de la modélisation des données d'entraînement. En contrepartie, la complexité $\mathcal{O}(n^3)$ qu'impliquent les calculs matriciels sous-tendant une telle méthode les rend impraticable pour des problèmes de grande taille. Bien que les processus Gaussiens soient applicables autant pour la classification que pour la régression, seule la partie portant sur la régression est couverte dans ce chapitre.

2.1 Historique

Contrairement à ce que l'on pourrait imaginer, l'étude des processus Gaussiens en contexte de régression n'est pas récente. En 1880, l'astronome T.N. Thiele les utilisait pour faire l'analyse de séries temporelles [Lauritzen, 1981]. Dans les années 1940, Wiener et Kolmogorov ont indépendamment introduit des méthodes de prédiction de trajectoire utilisant des filtres de bruits Gaussiens [Kolmogoroff, 1941; Wiener, 1949]. Plus récemment, une approche issue du domaine de la géostatistique, appelée *Kriging*, a été proposée par Matheron [1963]. Cette approche est similaire à la régression par processus Gaussien traitée dans ce chapitre. Au cours des décennies suivantes, la

méthode en question a considérablement évolué au sein de la géostatistique et de la météorologie, mais en ignorant en grande partie l'interprétation probabiliste du modèle [Cressie, 1993].

C'est dans le domaine des statistiques qu'est apparue une formulation générale applicable aux problèmes de régression [O'Hagan, 1978]. En ce qui concerne le domaine de l'apprentissage automatique, l'utilisation des processus Gaussiens fût initialement décrite par Williams et Rasmussen [Williams et Rasmussen, 1996]. L'introduction s'est faite à une époque où les réseaux de neurones artificiels étaient en vogue dans le domaine. L'inspiration de Williams et Rasmussen est essentiellement venue suite au résultat qu'un processus Gaussien est équivalent à un traitement Bayésien d'une certaine classe de réseaux de neurones lorsque le réseau devient infiniment grand [Neal, 1996]. Plus récemment, des relations ont été démontrées entre les processus Gaussiens et certaines méthodes telles que les Machines à Vecteur de Support, les splines, les Machines à Vecteur d'Importance ainsi qu'avec la théorie de la régularisation.

2.2 L'apprentissage Bayésien

Comme les processus Gaussiens ont des fondements théoriques se basant sur les méthodes Bayésiennes et que, d'autre part, ils peuvent être utilisés en tant que distribution de probabilité en contexte d'apprentissage Bayésien, cette section couvre sommairement certaines notions essentielles à la compréhension du chapitre.

Les méthodes Bayésiennes sont une alternative aux méthodes traditionnelles. Dans ce cadre théorique, les probabilités servent à refléter un état de croyance par rapport à une hypothèse plutôt qu'un passage à la limite d'une fréquence. Dans un contexte d'apprentissage où l'objectif est d'estimer les paramètres $\mathbf{w}$ d'un modèle, la démarche Bayésienne requiert d'abord d'exprimer sous forme de distribution de probabilités, les connaissances a priori $p(\mathbf{w})$ concernant les valeurs envisageables des paramètres. À partir des observations obtenues, ces connaissances sont mises à jour grâce à une reformulation particulière de la probabilité jointe que l'on nomme la *règle de Bayes* :

$$p(\mathbf{w}|\mathcal{D}) = \frac{p(\mathcal{D}|\mathbf{w})\, p(\mathbf{w})}{p(\mathcal{D})} \tag{2.1}$$

Chacun des termes de cette équation possède une signification précise. D'abord, nous avons la distribution a priori $p(\mathbf{w})$, aussi appelé *prior*, qui permet d'encoder des informations préalablement acquises concernant le problème. Ensuite, nous avons la *vraisemblance* des données $p(\mathcal{D}|\mathbf{w})$ qui indique quel sont les données $\mathcal{D}$ probables lorsque

l'on fixe le modèle à un vecteur de paramètre $\mathbf{w}$ particulier. Il est aussi possible de considérer tous les modèles possibles sous la distribution a priori. On se réfère à cette quantité $p(\mathcal{D})$ en tant que *vraisemblance marginale* et sert fréquemment à valider la qualité d'un prior. Le calcul de ce terme peut parfois s'avérer complexe, car il implique une intégrale :

$$p(\mathcal{D}) = \int p(\mathcal{D}|\mathbf{w})\, p(\mathbf{w}) d\mathbf{w} \tag{2.2}$$

Heureusement, il n'est pas toujours nécessaire de réaliser ce calcul pour compléter la règle de Bayes, car la quantité en question sert uniquement de constante de normalisation pour obtenir la distribution a posteriori $p(\mathbf{w}|\mathcal{D})$, c'est-à-dire la mise à jour. Bien entendu, l'obtention de la distribution a posteriori est l'objectif central, car elle est le résultat de la combinaison des connaissances a priori et des données observées.

2.2.1 L'apprentissage hiérarchique

L'application de cette démarche visant à reviser les connaissances a priori grâce aux données acquises se nomme *inférence Bayésienne* et peut s'appliquer à plusieurs niveaux, donnant ainsi lieu à un apprentissage *hiérarchique*. Supposons que la distribution des paramètres $p_{\boldsymbol{\theta}}(\mathbf{w})$ ne soit pas connue avec exactitude et qu'elle soit contrôlable via un vecteur d'*hyperparamètres* $\boldsymbol{\theta}$ et un ensemble fini de structure de modèle $\mathcal{H}_i$. Par exemple, l'ensemble des structures pourrait en fait être un ensemble de distribution de familles différentes où les hyperparamètres sont les paramètres de chaque distribution. Le processus peut ainsi s'appliquer au niveau supérieur, qui ici correspond aux hyperparamètres.

L'inférence au niveau suivant utilise la vraisemblance marginale du niveau précédent en tant que vraisemblance pour reviser la distribution a priori sur $\boldsymbol{\theta}$. En appliquant à nouveau la règle de Bayes, on obtient :

$$p(\boldsymbol{\theta}|\mathcal{D}) = \frac{p(\mathcal{D}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\mathcal{D})} \tag{2.3}$$

où $p(\boldsymbol{\theta})$ est un *hyper-prior* sur les paramètres qui contrôle la distribution $p_{\boldsymbol{\theta}}(\mathbf{w})$. À ce niveau, la vraisemblance marginale nécessite une intégrale supplémentaire [1] tel que :

$$p(\mathcal{D}) = \int p(\mathcal{D}|\boldsymbol{\theta})p(\boldsymbol{\theta}) d\boldsymbol{\theta} \tag{2.4}$$

Elle permet de prendre en compte l'incertitude sur les hyperparamètres et correspond à la fonction de vraisemblance du dernier niveau de la hiérarchie de notre exemple.

1. La première intégrale étant sur les paramètres $\mathbf{w}$.

Finalement, la distribution a posteriori sur les structures se calcule selon :

$$p(\mathcal{H}_i \mid \mathcal{D}) = \frac{p(\mathcal{D} \mid \mathcal{H}_i)p(\mathcal{H}_i)}{p(\mathcal{D})} \tag{2.5}$$

où le nombre de modèles est fini et, par conséquent, la dernière vraisemblance marginale devient simplement une somme tel que $p(\mathbf{y}|\mathbf{X}) = \sum_i p(\mathbf{y}|\mathbf{X}, \mathcal{H}_i)p(\mathcal{H}_i)$.

2.2.2 Approximation de la distribution a posteriori

Comme le démontrent les présentes équations, l'inférence Bayésienne implique généralement plusieurs calculs complexes. Idéalement, le calcul de la distribution a posteriori devrait se faire analytiquement en choisissant adéquatement le prior et la fonction de vraisemblance, ce qui simplifie grandement la complexité. Face aux situations difficiles, il faut se rabattre sur des méthodes d'estimation de la distribution a posteriori.

L'une des méthodes les plus intuitives consiste à représenter les distributions par des *particules*. D'abord, il faut représenter la distribution a priori sous cette forme. Pour y parvenir, il suffit simplement d'échantillonner la distribution en question afin d'accumuler un ensemble de représentant de la distribution, les particules. Celles-ci sont en réalité des candidats potentiels de la variable à identifier, et par conséquent, peuvent être évaluées au moyen de la fonction de vraisemblance $p(\mathcal{D}|\mathbf{w})$. Ainsi, la distribution a posteriori $p(\mathbf{w}|\mathcal{D})$ est obtenue en assignant un poids à chaque particule qui soit proportionnel à leur aptitude à expliquer les données. Évidemment, augmenter le nombre de particules entraîne une meilleure estimation, mais s'accompagne aussi d'une augmentation du temps de calcul.

Une des méthodes utilisées au cours de ce mémoire consiste à n'utiliser qu'un seul représentant de la distribution, un cas limite d'estimation Bayésienne. Le représentant que nous cherchons à obtenir est en réalité celui ayant la plus grande probabilité sous la distribution a posteriori, donc le meilleur candidat, généralement appelé *maximum a posteriori* (MAP). Trouver le maximum a posteriori est un problème d'optimisation pour lequel il faut d'obtenir :

$$\hat{\mathbf{w}} = \arg\max_{\mathbf{w}} p(\mathcal{D} \mid \mathbf{w})\, p(\mathbf{w}) \tag{2.6}$$

Notons qu'en présence d'une distribution a priori non-informative ou en l'absence complète de distribution a priori $p(\mathbf{w})$, cette estimation correspond à l'estimateur par maximum de vraisemblance traditionnel et largement relaté dans la littérature.

2.3 Les processus Gaussiens

Un processus stochastique est un ensemble de variables aléatoires dont les interdépendances sont décrites par des distributions de probabilités. Par exemple, lorsqu'un processus stochastique est défini sur le temps, la valeur que peut prendre le processus à chaque instant est une variable aléatoire dont l'index est le temps. Les distributions de probabilités permettent de modéliser l'évolution du processus au cours du temps et, par conséquent, de déterminer la probabilité de chaque séquence d'état possible. En définissant un processus sur des espaces comme les *champs aléatoires*, les processus stochastiques peuvent servir de distribution de probabilités sur un espace de fonctions. Par contre, définir des distributions de probabilités dans de tels espaces peut s'avérer complexe selon le type de processus choisi. Parmi les processus stochastiques les plus simples à manipuler, il y a les processus Gaussiens dont la définition comprend uniquement deux fonctions. Voici donc comment définir un processus Gaussien.

2.3.1 Définition

Formellement, lorsqu'un processus stochastique $f(\mathbf{x})$ est dit Gaussien, la distribution jointe de tout sous-ensemble de ses variables aléatoires $\mathbf{f} = \{f_1, f_2, \ldots, f_N\}$ sur un ensemble fini d'index $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ prend, par hypothèse, la forme d'une *distribution normale multivariée* à N dimensions. En adoptant cette hypothèse, nous obtenons des processus Gaussiens qui sont complètement définis par leurs fonctions de moyennes μ et de covariance k. Ces fonctions permettent d'obtenir la distribution

$$p(\mathbf{f}) = \mathcal{N}(\boldsymbol{\mu}, \mathbf{K}) \tag{2.7}$$

où $\mathcal{N}$ représente la distribution normale, le vecteur de moyennes $\boldsymbol{\mu}$ est composé des éléments $\mu_i = \mu(\mathbf{x}_i)$ et la matrice de covariance $\mathbf{K}$ est construite telle que $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$. Comme les fonctions μ et k ont une influence directe sur la distribution jointe (2.7), les diverses formes que ces fonctions peuvent prendre entraînerons des processus aux comportements différents. Notons qu'il est aussi possible de définir les fonctions de moyenne et de covariance sur un espace fini, auquel cas le processus Gaussien devient simplement une distribution normale multidimensionnelle. Le processus Gaussien est donc un objet très général pouvant être employé sur des espaces variés. Parmi ceux-ci, citons les espaces de chaînes de caractères, d'arbres ou même de graphes. L'unique exigence est que les fonctions μ et k soient bien définies sur l'espace d'entrée considéré.

Comme les processus Gaussiens sont des processus stochastiques, ils peuvent servir de distribution de probabilités sur un espace de fonctions. Suivant cette idée, il est donc

possible d'échantillonner une fonction d'un processus et, inversement, une fonction se voit attribuer une mesure de vraisemblance par rapport au processus. Définir une distribution sur un tel espace équivaut à préciser les fonctions qui le constituent. Dans le cas particulier du processus Gaussien, la fonction μ est utilisée pour centrer la distribution sur la région d'intérêt de l'espace des fonctions. Pour sa part, la fonction de covariance sert à déterminer le type de fonction attendu et indique aussi le degré d'incertitude entourant la fonction $\mu(\mathbf{x})$ via la variance exprimée par $k(\mathbf{x}, \mathbf{x})$. Ainsi, la distribution de probabilités peut être vue comme un hypertube enveloppant les fonctions les plus probables où la fonction de moyennes permet de positionner le centre du tube et la fonction de covariance fixe le diamètre tout au long de celui-ci. Lors de l'apprentissage d'une fonction, les connaissances a priori sont principalement encodées par la fonction de covariance. La section 2.3.2 ci-après, explique comment le choix de la fonction k influence le comportement du processus.

2.3.2 Les fonctions de covariance

Afin de bien comprendre le processus Gaussien, il est impératif de saisir le rôle que joue la fonction de covariance dans sa définition. Une fonction de covariance permet de caractériser l'indépendance de deux variables aléatoires. Pour ce qui est du processus Gaussien, les variables aléatoires sont indexées par leur valeur d'entrée. Par conséquent, la fonction k détermine le niveau de covariance pour toutes paires de variables aléatoires $f(\mathbf{x}_i)$ et $f(\mathbf{x}_j)$, et ce, en ayant uniquement recours aux données d'entrées $\mathbf{x}_i$ et $\mathbf{x}_j$. En général, une fonction arbitraire n'est pas une fonction de covariance valide. Pour l'être, celle-ci doit, pour tout ensemble fini d'entrées possibles $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$, générer une matrice construite telle que $\mathbf{K}_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$, qui soit *définie positive*[2]. Dans la littérature, on se réfère régulièrement à la fonction k sous l'appellation *noyau* et plus spécifiquement aux *noyaux de Mercer* lorsqu'ils respectent la dernière propriété.

Le choix de la fonction de covariance d'un processus Gaussien influence son comportement en fonction des valeurs d'entrées. En se référant à l'équation (2.7), on remarque que les valeurs d'un processus peuvent être instanciées conjointement. En revanche, il est aussi possible de réaliser séquentiellement ce processus d'échantillonnage, ce qui met l'emphase sur la notion d'indépendance conditionnelle. Plus précisément, pour un vecteur $\mathbf{f}$ déjà instancié, l'ensemble des valeurs possibles qu'une variable aléatoire supplémentaire $f(\mathbf{x}_*)$ peut prendre, est influencé par celles contenues dans $\mathbf{f}$. Comme ces valeurs doivent être prises en considération, la fonction de covariance indiquera

2. Une matrice est définie positive lorsque toutes ses valeurs propres sont strictement positives, car celles-ci correspondent en fait aux variances dans les directions des vecteurs propres.

dans quelle mesure $f(\mathbf{x}_*)$ est lié aux valeurs de $\mathbf{f}$ via une règle de probabilité conditionnelle décrite à la section 2.4.1. Une hypothèse commune concernant l'effet des dépendances veut que deux variables aléatoires proches l'une de l'autre par rapport à l'espace d'entrée, aient tendance à prendre des valeurs similaires. Cette mesure de similarité est parfois définie par rapport à une distance sur l'espace d'entrée.

C'est essentiellement grâce à cette notion de similarité entre variables aléatoires, fournie par la fonction de covariance, qu'il est possible de réaliser des prédictions. Ainsi, le choix d'une fonction de covariance appropriée est primordial à l'apprentissage. Différentes fonctions permettront de découvrir ou de négliger certaines structures dans les données d'entraînements. Les sections suivantes couvrent une liste non-exhaustive de fonctions jugées utiles pour l'apprentissage avec processus Gaussiens. Chacune de ces fonctions permet de définir un type de distribution a priori particulier dans l'espace des fonctions. Ainsi, le problème de l'apprentissage par processus Gaussien revient essentiellement à déterminer la distribution a priori appropriée pour une tâche d'apprentissage supervisé.

2.3.3 Les fonctions de covariance stationnaire

Les fonctions de covariances stationnaires regroupent les fonctions invariantes aux translations dans l'espace d'entrée. Plus précisément, cela signifie que la covariance $k(\mathbf{x}, \mathbf{x}')$ entre deux variables aléatoires sera identique à la covariance $k(\mathbf{x}+\Delta\mathbf{x}, \mathbf{x}'+\Delta\mathbf{x})$ pour toutes valeurs de $\Delta\mathbf{x}$. Ainsi, la distribution de probabilités jointe des variables aléatoires du processus restera inchangée lorsque décalée sur l'espace d'entrée. Par conséquent, un processus Gaussien utilisant ce type de fonction aura un seul et même comportement sur tout son espace. Voici quelques exemples de fonction de covariance stationnaire.

2.3.3.1 La fonction de covariance carré-exponentielle

La fonction de covariance carré-exponentielle est l'une des plus utilisées, notamment en raison de son interprétation simple et son aisance à être manipulée algébriquement. Puisque cette fonction est essentiellement une Gaussienne non normalisée, on s'y réfère régulièrement en tant que fonction de covariance Gaussienne. Sous sa forme usuelle, elle s'écrit :

$$k_{CE}(\mathbf{x}, \mathbf{x}') = a^2 \exp\left(-\frac{\|\mathbf{x}-\mathbf{x}'\|^2}{\lambda^2}\right) \tag{2.8}$$

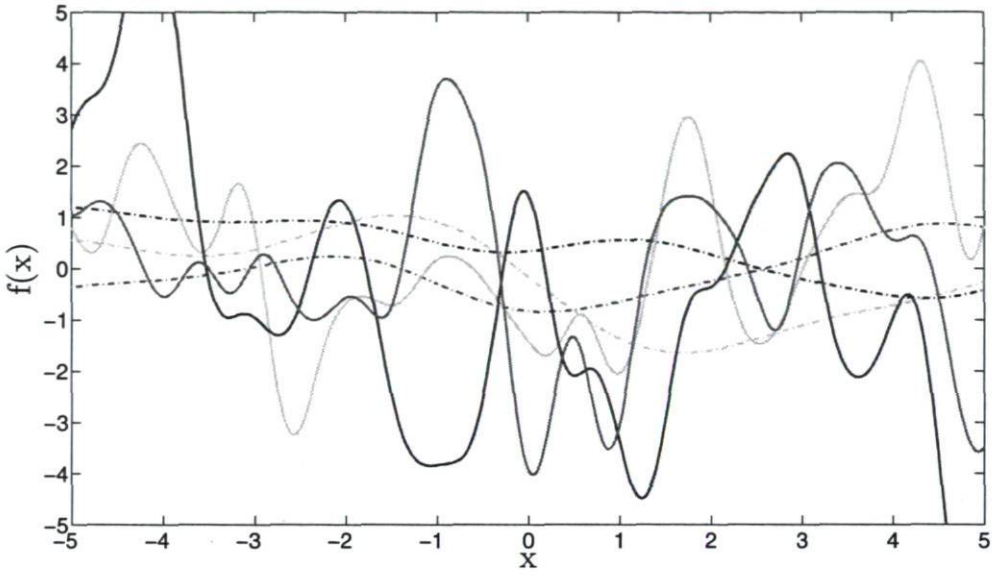


FIGURE 2.1 – Fonctions échantillonnées à partir de processus Gaussien utilisant la carré-exponentielle avec différent hyperparamètres. Les fonctions en trait plein ont été générées avec $a = 2$, $\lambda = 0.5$ et les fonctions en pointillés avec $a = 0.75$, $\lambda = 2.25$. La fonction de moyennes est $\mu(x) = 0$.

où le paramètre λ est le *facteur d'échelle de la distance* et le paramètre d'amplitude a contrôle la (co)variance maximale. Cette fonction indique que les données proches les une des autres, relativement à leurs distances sur l'espace d'entrée, auront tendance à prendre des valeurs similaires. Le paramètre λ permet d'ajuster l'impact de cette mesure de distance sur la covariance. Toute variation de ce paramètre modifie la distance nécessaire afin que deux données soient quasiment indépendantes. Du point de vue d'une loi de probabilité normale, cette variable correspond à l'écart-type et joue un rôle similaire pour la carré-exponentielle.

Dans le cas où la forme proposée en (2.8) utilise la distance Euclidienne[3], la fonction de covariance est dite *isotropique*, car le niveau de covariance est invariant aux rotations sur l'espace d'entrées. Par conséquent, cela signifie que la covariance dépend uniquement de la distance entre les variables d'entrées $\mathbf{x}$ et $\mathbf{x}'$. Dans certains cas, il est préférable d'employer la fonction carré-exponentielle sous une forme *anisotropique* (voir section 2.5.2) afin de tenir compte de l'influence sur le processus des différentes dimensions de l'espace d'entrées.

3. Il s'agit simplement de définir la norme au numérateur en conséquence de la distance désirée.

Les processus Gaussiens basés sur la fonction de covariance carré-exponentielle sont particulièrement lisses. Cette propriété intéressante provient du fait qu'une Gaussienne est infiniment dérivable. La figure 2.1 montre quelques fonctions échantillonnées à partir de deux processus Gaussien distinct utilisant la carré-exponentielle avec différent hyperparamètres[4]. Les fonctions en trait plein ont été générées avec les hyperparamètres $a = 2$ et $\lambda = 0.5$ et les fonctions en pointillés avec $a = 0.75$, $\lambda = 2.25$. On remarque qu'une augmentation du paramètre a entraîne une plus grande amplitude des fonctions, tandis qu'une diminution du paramètre λ provoque des variations rapides. Bien qu'un très petit λ permet d'obtenir des fonctions qui varient considérablement sur un court intervalle, les fonctions resteront toujours lisses si la fonction de moyennes est lisse. Cependant, cette propriété s'avère parfois déraisonnable, entre autres lors de la modélisation de systèmes physiques. Dans ce contexte, il est plus approprié de choisir d'autres fonctions de covariance. Malgré tout, la carré-exponentielle reste l'une des plus employées dans le domaine de l'apprentissage automatique.

2.3.3.2 La fonction de covariance γ-exponentielle

La fonction de covariance γ-exponentielle généralise une classe de fonctions permettant de définir des processus non lisses. Entre autres, cette fonction de covariance généralise deux cas particuliers fréquemment relatés dans la littérature, soit les fonctions de covariances carré-exponentielle (avec $\gamma = 2$) et exponentielle (avec $\gamma = 1$). La fonction de covariances γ-exponentielle est définie telle que :

$$k_{\gamma-exp}(\mathbf{x}, \mathbf{x}') = a^2 \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^\gamma}{\lambda^2}\right), \qquad 0 < \gamma \leqslant 2 \tag{2.9}$$

où λ est le *facteur d'échelle de la distance*, a contrôle la (co)variance maximale et γ est le paramètre contrôlant le niveau de rugosité du processus. Lorsque la valeur de γ est faible, le processus est dit rugueux en raison de son apparence accidentée. Ultimement, les réalisations d'un processus Gaussien utilisant un γ tendant vers 0 prendront la forme d'un bruit blanc Gaussien de moyenne non nulle. Inversement, les valeurs plus élevées diminuent l'aspect accidenté du processus pour finalement devenir lisse lorsque $\gamma = 2$, soit le cas spécial de la fonction de covariance carré-exponentielle. La figure 2.2 contient des exemples générés avec $\gamma = 1.8$ en pointillés et avec $\gamma = 1$ en trait plein. La carré-exponentielle étant un cas particulier, les exemples utilisant le paramètre $\gamma = 2$ sont présentés à la figure 2.1.

4. Rappelons qu'un processus Gaussien est une distribution sur un espace de fonction et qu'un échantillon est une fonction en soi.

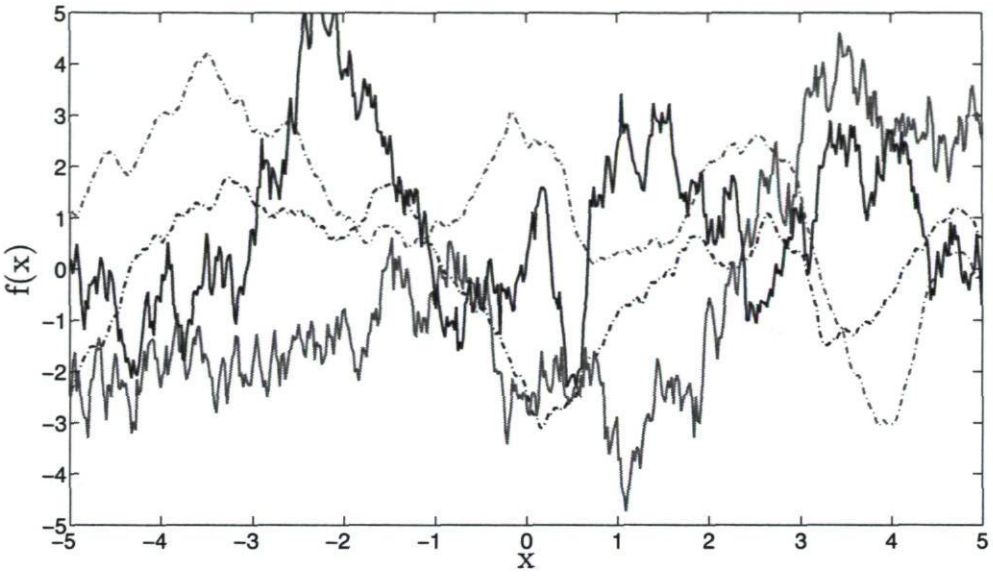


FIGURE 2.2 – Fonctions échantillonnées à partir de processus Gaussien utilisant la γ-exponentiel avec différent hyperparamètres. Les fonctions en pointillés ont été générées avec $\gamma = 1.8$ et les fonctions en trait plein avec $\gamma = 1$. Les autres paramètres ont été fixés à $\lambda = 1$ et $a = \sqrt{3}$. La fonction de moyennes est $\mu(x) = 0$.

2.3.3.3 La fonction de covariance Matérn

Le choix de la fonction de covariance influence directement le type de réalisation que peut générer un processus Gaussien. En apprentissage, choisir la douceur de la carré-exponentielle ou la rugosité de la γ-exponentielle correspond en fait à une hypothèse forte sur l'espace des fonctions. Pour éviter ce type de dichotomie, il est possible d'employer la fonction de covariance Matérn. Cette fonction permet de réaliser un compromis intéressant quant aux hypothèses de rugosité et douceur via un paramètre $\nu > 0$ contrôlant le degré de douceur du processus. Cette fonction est définie comme suit :

$$k_{Matern}(\mathbf{x}, \mathbf{x}') = a^2 \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\frac{\sqrt{2\nu}\,\|\mathbf{x} - \mathbf{x}'\|}{\lambda^2} \right)^{\nu} K_\nu \left(\frac{\sqrt{2\nu}\,\|\mathbf{x} - \mathbf{x}'\|}{\lambda^2} \right) \tag{2.10}$$

où les paramètres a et λ modélisent respectivement l'amplitude des réalisations et l'impact de la distance entre les entrées sur la corrélation. Les fonctions K_ν et $\Gamma(\nu)$ sont respectivement une fonction de Bessel modifiée du second genre d'ordre ν et la fonction gamma.

Lorsque ν est petit, le processus est rugueux, et inversement, lorsque ν est grand,

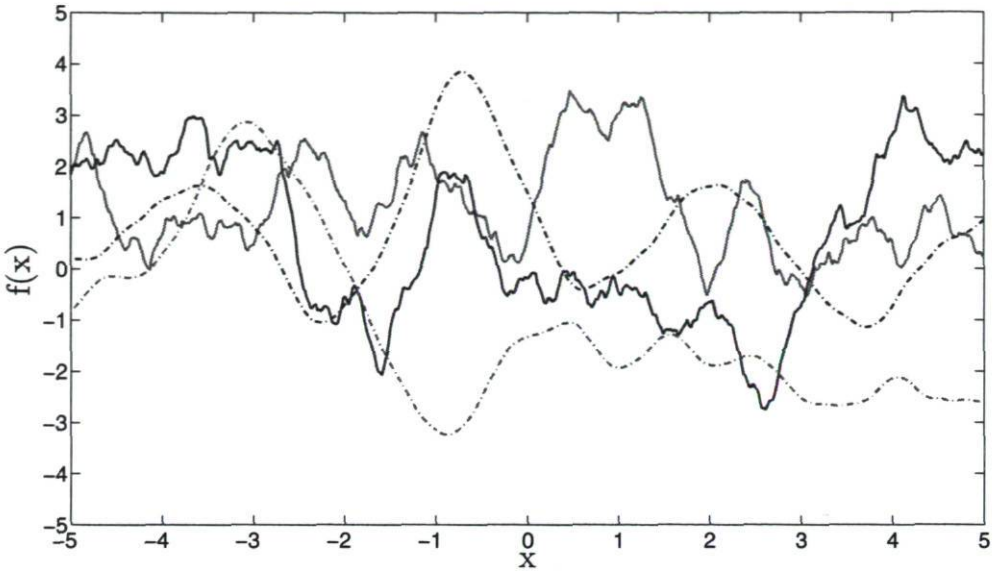


FIGURE 2.3 – Fonctions échantillonnées à partir de processus Gaussien utilisant la fonction Matérn avec différent hyperparamètres. Les fonctions en pointillés ont été générées avec $\nu = 3/2$ et les fonctions en trait plein avec $\nu = 1$. Les autres paramètres ont été fixés à $\lambda = 1$ et $a = \sqrt{3}$. La fonction de moyennes est $\mu(x) = 0$.

le processus est lisse. La principale explication est qu'un processus employant cette fonction de covariance est dit k fois dérivable en moyenne quadratique[5] pour autant que $k > \nu$. Ainsi, en contrôlant la dérivabilité d'un processus, il est possible de contrôler le niveau de douceur de ce même processus.

Bien que la fonction de covariance Matérn possède autant de paramètres que la γ-exponentielle, la fonction Matérn est considérée plus flexible en raison du paramètre de douceur. La fonction covariance carré-exponentielle est d'ailleurs un cas particulier de la fonction Matérn obtenu lorsque $\nu \to \infty$. Cette situation donne ainsi lieu à un processus infiniment dérivable en moyenne quadratique et, par conséquent, celui-ci devient très lisse. À l'opposé, le cas spécial $\nu = 1/2$ permet d'obtenir la fonction de covariance exponentielle (voir section 2.3.3.2), dont les processus basés sur celle-ci sont non dérivables en moyenne quadratique. Lorsque la dimension de la variable d'entrée est le temps, ce cas spécial est mieux connu sous le nom de processus d'Ornstein-Uhlenbeck [Uhlenbeck et Ornstein, 1930], lequel a été appliqué en finance et en physique.

5. La dérivabilité en moyenne quadratique est une extension de la dérivée usuelle aux processus stochastique.

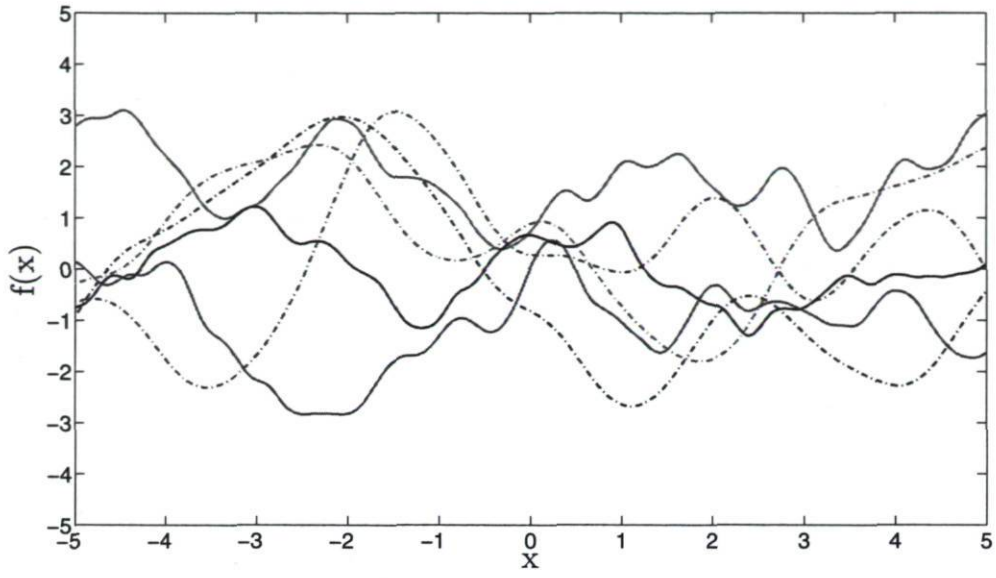


FIGURE 2.4 – Fonctions échantillonnées à partir de processus Gaussien utilisant la fonction quadratique rationnelle avec différent hyperparamètres. Les fonctions en pointillés ont été générées avec $\alpha = 2$ et les fonctions en trait plein avec $\alpha = 1/10$. Les autres paramètres ont été fixés à $\lambda = 1$ et $a = \sqrt{3}$. La fonction de moyennes est $\mu(x) = 0$.

D'autres cas intéressants sont obtenus pour les cas où $\nu = p + 1/2$, où p est un entier positif. Cette restriction aux entiers positifs fait en sorte que la fonction de covariance Matérn devient le produit d'une fonction exponentielle (voir section 2.3.3.2) et d'une fonction de covariance polynomiale d'ordre p (voir section 2.3.4.2). Il a été avancé par Rasmussen et Williams [2006] que les cas les plus intéressants pour le domaine de l'apprentissage automatique sont $\nu = 3/2$ et $\nu = 5/2$. En effet, au-delà de ces valeurs, il est difficile de distinguer l'effet de la fonction Matérn par rapport à une fonction carré-exponentielle.

La figure 2.3 montre quelques exemples obtenus avec $\nu = 3/2$ pour les courbes en pointillés et avec $\nu = 1$ pour les courbes en trait plein. La figure 2.1 permet d'observer le comportement de processus Gaussiens utilisant des valeurs $\nu \to \infty$, et finalement, les courbes en trait plein de la figure 2.2 correspondent au cas particulier $\nu = 1/2$.

2.3.3.4 La fonction de covariance quadratique rationnelle

En se basant sur le fait qu'une somme de fonctions de covariance produit une fonction de covariance valide, il est possible de construire de nouvelles fonctions de covariance. Dans le cas présent, la fonction de covariance quadratique rationnelle est obtenue par une somme infinie de fonctions carré-exponentielle pondérées par une *distribution gamma* sur le facteur d'échelle de la distance λ. Cette distribution est décrite par :

$$p(\lambda^{-2}|\alpha,\beta) \propto (\lambda^{-2})^{\alpha-1}\exp(-\lambda^{-2}/\beta) \tag{2.11}$$

où α et β sont respectivement les paramètres de forme et d'échelle de la distribution gamma. Pour obtenir la quadratique rationnelle, cette somme infinie de carré-exponentielles prend la forme d'une intégrale sur les différents facteurs d'échelles :

$$\begin{aligned} k_{RQ}(\mathbf{x},\mathbf{x}') &= \int k_{CE}(\mathbf{x},\mathbf{x}'|\lambda^{-2})p(\lambda^{-2}|\alpha,\beta)\mathrm{d}\lambda^{-2} \\ &\propto \int a^2\exp\left(-\frac{\|\mathbf{x}-\mathbf{x}'\|^2}{\lambda^2}\right)(\lambda^{-2})^{\alpha-1}\exp(-\lambda^{-2}/\beta)\mathrm{d}\lambda^{-2} \\ &\propto a^2\left(1+\frac{\beta\|\mathbf{x}-\mathbf{x}'\|^2}{\alpha}\right)^{-\alpha} \end{aligned} \tag{2.12}$$

où $\beta > 0$ correspond au facteur d'échelle de la distance de la quadratique rationnelle et $\alpha > 0$ contrôle le ratio de covariance qu'auront les données distantes par rapport à celles qui sont rapprochées[6]. Lorsque le paramètre α est faible, il tend à donner une certaine covariance aux données distantes l'une de l'autre. À l'inverse, un α plus grand aura tendance à donner une covariance pratiquement nulle à ces mêmes données distantes.

Comme la quadratique rationnelle provient d'une somme de carré-exponentielle, il n'est pas étonnant de retrouver la carré-exponentielle comme cas spécial de la quadratique rationnelle. Cette situation particulière survient lorsque le paramètre $\alpha \to \infty$. D'ailleurs, la fonction quadratique rationnelle hérite aussi de la propriété d'être infiniment dérivable en moyenne quadratique, faisant d'elle une fonction de covariance qui produit des processus lisses.

La figure 2.4 illustre le comportement de processus Gaussiens adoptant la quadratique rationnelle. Les courbes en pointillés ont été générées avec $\alpha = 2$ et celles en trait plein avec $\alpha = 1/10$. La figure 2.1 montre le cas où $\alpha \to \infty$, soit le cas de la carré-exponentielle.

6. Notons que le facteur β fait varier linéairement la mesure de distance, tandis que α contrôle l'épaisseur de la queue de la fonction de covariance.

2.3.4 Les fonctions de covariance non-stationnaire

Les fonctions de covariance non-stationnaire permettent de considérer des corrélations dépendant des valeurs d'entrée. Ainsi, la distribution de probabilités jointe (2.7) des données dans une région de l'espace est dépendante de cette même région. Par conséquent, une translation des données sur l'espace d'entrée donne lieu à une transformation de la distribution de probabilités jointe.

L'effet de cette propriété est de permettre divers comportements de fonction dépendamment des régions. Cela permet entre autres de modéliser un niveau de bruit variable au cours du temps ou même de générer des processus ayant des variations très rapides dans une certaine région et très lentes pour une autre. Par exemple, la fluctuation des prix sur un marché peut s'avérer plus volatile qu'à la normale au cours de certaines périodes. Pour modéliser un comportement variable au cours du temps, il est avantageux d'employer une fonction de covariance non-stationnaire considérant les temps en valeur absolue, ce qui dans notre exemple reviendrait à utiliser la date comme caractéristique du processus. Une fonction de covariance stationnaire quant à elle, ne considère que les valeurs relatives entre les entrées et présume ainsi qu'un seul et unique comportement influence les prix du début à la fin.

2.3.4.1 La fonction de covariance linéaire

Un exemple simple de fonction de covariance non-stationnaire est obtenu d'un traitement Bayésien de la régression linéaire. Cet exemple est intéressant, car il montre comment faire pour construire d'autres fonctions de covariance valides. Formellement, supposons un modèle $f(\mathbf{x}) = \mathbf{x}^\top \mathbf{w} + b$, où le vecteur $\mathbf{w}$ contient les pentes du modèle linéaire et le paramètre b est le biais par rapport à l'origine. La mise en place du traitement Bayésien consiste à placer des distributions de probabilités sur les paramètres inconnus. Supposons donc que les pentes suivent une distribution $\mathbf{w} \sim \mathcal{N}(0, \Sigma_w)$ de moyenne nulle et dont la matrice de covariance est Σ_w. Concernant le biais, une distribution $b \sim \mathcal{N}(0, \sigma_b^2)$ de variance σ_b^2 y est appliquée.

Cette définition constitue un modèle génératif dont nous pouvons calculer l'espérance et la covariance. D'abord, voici l'espérance que nous dénotons $\mathbb{E}$ et qui est donnée par :

$$\begin{aligned} \mathbb{E}[f(\mathbf{x})] &= \mathbb{E}[\mathbf{x}^\top \mathbf{w} + b] \\ &= \mathbf{x}^\top \mathbb{E}[\mathbf{w}] + \mathbb{E}[b] \\ \mu(\mathbf{x}) &= 0 \end{aligned} \tag{2.13}$$

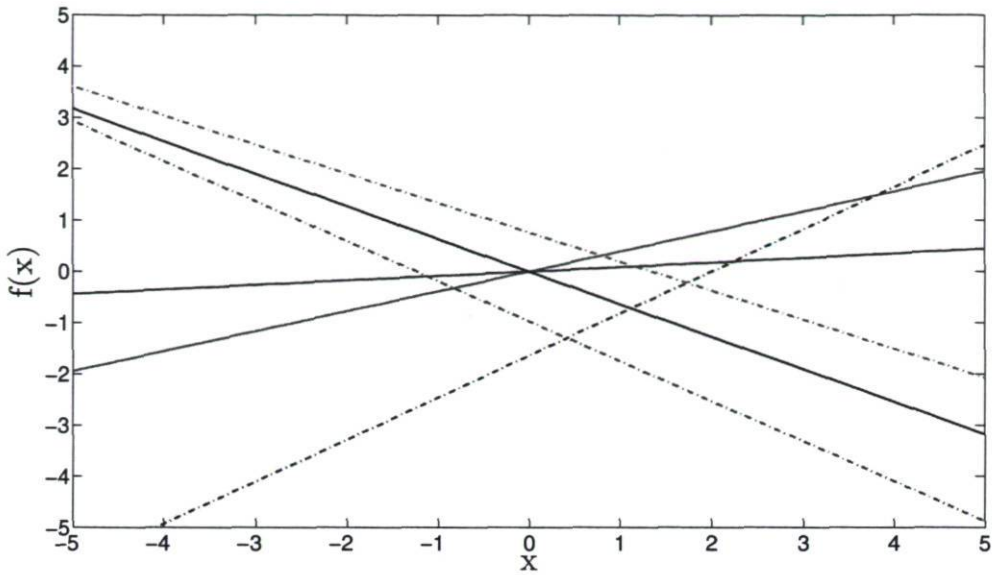


FIGURE 2.5 – Fonctions échantillonnées à partir de processus Gaussien utilisant la fonction de covariance linéaire avec différents hyperparamètres. Les fonctions en pointillés ont été générées avec $\sigma_b^2 = 3$ et les fonctions en trait plein avec $\sigma_b^2 = 0$. Les autres paramètres ont été fixés tel que Σ_w est la matrice identité. La fonction de moyennes est $\mu(x) = 0$.

où l'espérance nulle est une conséquence des priors à moyenne nulle des paramètres $\mathbf{w}$ et b. Pour sa part, la covariance que nous dénotons par le symbole Cov, est donnée par :

$$\begin{aligned} \text{Cov}[f(\mathbf{x}), f(\mathbf{x}')] &= \mathbb{E}[(f(\mathbf{x}) - 0)(f(\mathbf{x}') - 0)] \\ &= \mathbb{E}[(\mathbf{w}^\top \mathbf{x} + b)(\mathbf{w}^\top \mathbf{x}' + b)] \\ &= \mathbf{x}^\top \mathbb{E}[\mathbf{w}\mathbf{w}^\top]\mathbf{x}' + \mathbb{E}[b^2] + (\mathbf{x} + \mathbf{x}')^\top \mathbb{E}[\mathbf{w}]\mathbb{E}[b] \\ k_{Lin}(\mathbf{x}, \mathbf{x}') &= \mathbf{x}^\top \Sigma_w \mathbf{x}' + \sigma_b^2. \end{aligned} \tag{2.14}$$

où la fonction k_{Lin} est dite homogène[7] si $\sigma_b^2 = 0$ et non-homogène autrement. Ainsi, en appliquant des distributions a priori Gaussiennes aux paramètres du modèle, nous obtenons une distribution de probabilités sur l'espace des fonctions linéaires. La figure 2.5 illustre quelques fonctions échantillonnées à partir de fonctions de covariance homogènes en trait plein et non-homogènes en pointillés. On remarque que les fonctions homogènes passent par l'origine et qu'attribuer une variance σ_b^2 non nulle au biais permet de lever cette restriction. Quant à la matrice de covariance Σ_w, celle-ci permet

7. Rappelons qu'un système d'équations linéaires homogène possède une forme $Ax = 0$, et donc, admet la solution triviale.

de contrôler l'inclinaison des pentes[8] en modifiant la largeur de la Gaussienne sur les pentes. Une augmentation des variances aura l'effet d'admettre des pentes plus élevées et inversement, la diminuer restreint le processus à de faibles pentes. Par ailleurs, ce raisonnement s'applique aussi au biais, car augmenter sa variance σ_b^2 a l'effet d'admettre des fonctions s'éloignant de plus en plus de l'origine.

2.3.4.2 La fonction de covariance polynomiale

En se basant sur le fait qu'un produit de fonctions de covariance valide $k(\mathbf{x}, \mathbf{x}') = k_1(\mathbf{x}, \mathbf{x}')k_2(\mathbf{x}, \mathbf{x}')$ reste une fonction de covariance valide, il est possible de construire de nouvelles fonctions. Ainsi, en multipliant une fonction de covariance linéaire p fois par elle-même, cela permet d'obtenir la fonction de covariance polynomiale suivante :

$$k_{poly}(\mathbf{x}, \mathbf{x}') = (\mathbf{x}^\top \Sigma_w \mathbf{x}' + \sigma_b^2)^p \tag{2.15}$$

où le degré est soumis à la contrainte $p \in \mathbb{N}^*$. La figure 2.6 montre quelques polynômes de degré 5 échantillonnées d'un processus Gaussien basé sur une covariance polynomiale non-homogène.

2.3.4.3 La fonction de covariance réseau de neurones

Le réseau de neurones est un modèle mathématique fréquemment utilisé en apprentissage automatique, notamment en raison de ses excellentes capacités d'apprentissage de fonction non-linéaire. En général, un réseau de neurones contient une série de couches de neurones où chacune des couches est complètement connectée à la suivante. La dernière couche sert à construire le vecteur de sortie du réseau à partir de l'information traitée par les couches précédentes, que l'on nomme couches cachées.

Les architectures envisageables pour les réseaux de neurones sont variées. L'une des plus importantes est celle à une seule couche cachée, car il a été démontré qu'avec un nombre de neurones tendant vers l'infini, cette architecture à la propriété d'approximateur universel [Hornik, 1993]. Par ailleurs, il se trouve qu'un réseau de neurones infini converge vers un processus Gaussien utilisant la covariance du réseau comme fonction de covariance [Neal, 1996]. Une des architectures permettant ce rapprochement est

8. Si Σ_w n'est pas diagonale, ce sont les pentes de la transformation linéaire induite par Σ_w qui seront contrôlées.

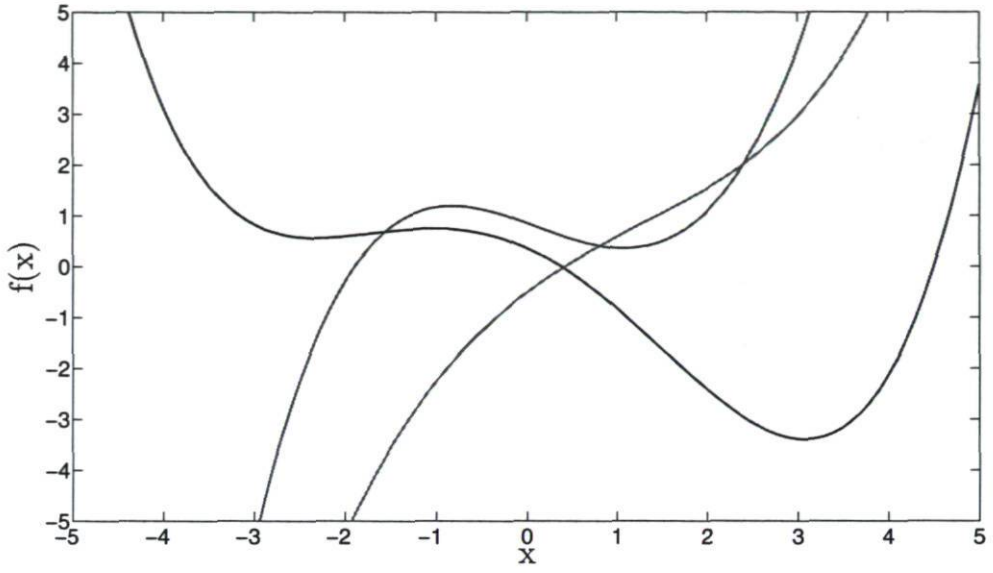


FIGURE 2.6 – Fonctions échantillonnées à partir de processus Gaussien utilisant la fonction de covariance polynomiale. Les fonctions ont été générées avec $\sigma_b^2 = 0.5$, $p = 5$ et Σ_w diagonale de valeur 0.1. La fonction de moyennes est $\mu(x) = 0$.

représentée par :

$$f(\mathbf{x}) = b + \sum_{j=1}^{H} w_j \operatorname{erf}(\mathbf{u}_j^\top \tilde{\mathbf{x}}) \tag{2.16}$$

où $\tilde{\mathbf{x}}$ est le vecteur $\mathbf{x}$ augmenté du scalaire 1 afin de représenter le biais, H correspond au nombre de neurones contenus sur la couche cachée et la fonction *erf*[9] est la fonction d'activation de la couche cachée. En supposant que b, les w_j et les u_j sont tous indépendamment distribués selon des normales à moyenne nulle, la fonction de covariance du réseau de neurones est obtenue analytiquement lorsque $H \to \infty$ et donne [Williams, 1998] :

$$k_{NN}(\mathbf{x}, \mathbf{x}') = a^2 \sin^{-1}\left(\frac{2\tilde{\mathbf{x}}^\top \Sigma \tilde{\mathbf{x}}'}{\sqrt{(1 + 2\tilde{\mathbf{x}}^\top \Sigma \tilde{\mathbf{x}})(1 + 2\tilde{\mathbf{x}}'^\top \Sigma \tilde{\mathbf{x}}')}}\right) \tag{2.17}$$

où Σ est la matrice de covariance diagonale de la distribution des $\mathbf{u}_j$. Le premier terme de la matrice Σ contrôle la variance du biais de la fonction *erf* et, par conséquent, permet de prendre en compte des fonctions pouvant varier plus loin de l'origine à mesure que ce paramètre augmente. Les autres termes de la matrice Σ constituent des facteurs d'échelle de distance et permettent de contrôler la vitesse à laquelle varient

9. La fonction d'erreur correspond à l'intégration numérique d'une distribution Gaussienne sur l'intervalle $[0, z]$ et possède une forme sigmoïde.

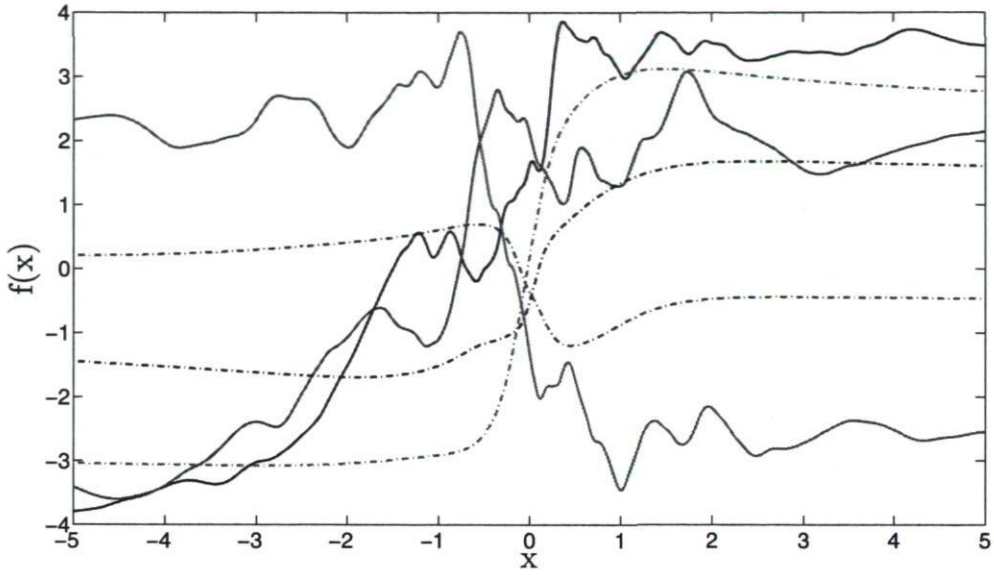


FIGURE 2.7 – Fonctions échantillonnées à partir de processus Gaussien utilisant la fonction de covariance réseau de neurones. Les courbes en pointillés ont été générées avec $\Sigma = \text{diag}(0.1, 10)$ et celles en trait plein avec $\Sigma = \text{diag}(100, 100)$. La fonction de moyennes est $\mu(x) = 0$.

les fonctions. La figure 2.7 contient quelques exemples de fonction tirés à partir de $\Sigma = \text{diag}(0.1, 10)$ en pointillés et $\Sigma = \text{diag}(100, 100)$ en trait plein.

2.3.4.4 La fonction de covariance périodique

Une méthode intéressante d'obtenir des fonctions de covariance non-stationnaire est d'utiliser une fonction de covariance stationnaire sur une transformation non-linéaire de l'espace d'entrée. En choisissant $\mathbf{u}(\mathbf{x}) = [\cos(x_1), \sin(x_1), \ldots, \cos(x_d), \sin(x_d)]$, les entrées à d dimensions sont transformées en vecteurs de $2d$ dimensions et permettent de créer des processus Gaussiens périodiques. Par exemple, en utilisant la fonction de covariance exponentielle (voir section 2.3.3.2) sur ce nouvel espace, la fonction de covariance devient :

$$k_{per}(\mathbf{x}, \mathbf{x}') = a^2 \exp\left(-\frac{\|\mathbf{u}(\mathbf{x}) - \mathbf{u}(\mathbf{x}')\|}{\lambda^2}\right) \tag{2.18}$$

où la distance entre $\mathbf{x}$ et $\mathbf{x}'$ est calculée dans le nouvel espace induit par la transformation $\mathbf{u}$. La figure 2.8 illustre le comportement périodique d'une covariance exponentielle avec $a = \sqrt{3}$ et $\lambda = \sqrt{0.2}$. L'utilisation d'autres fonctions de covariance stationnaire

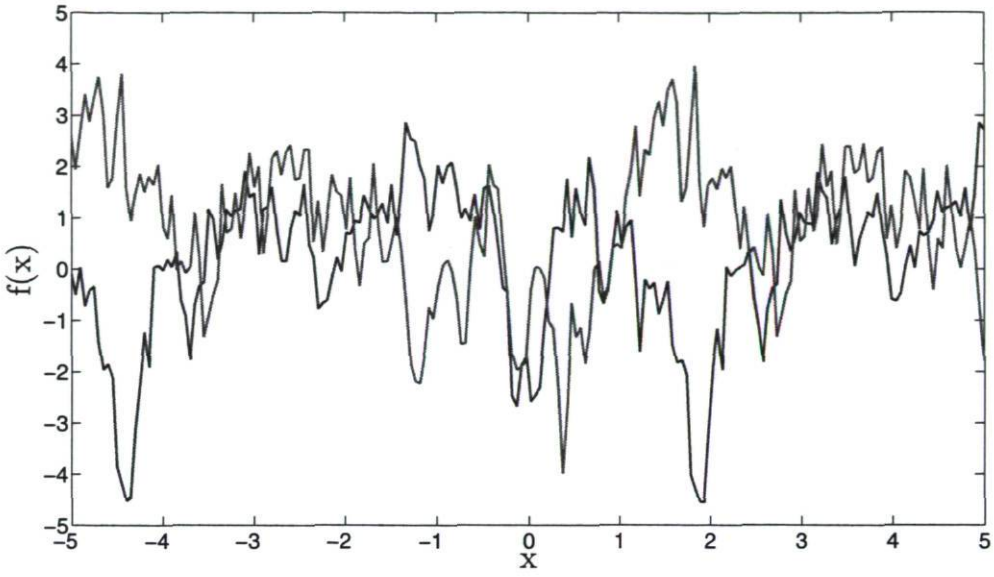


FIGURE 2.8 – Fonctions échantillonnées à partir de processus Gaussien utilisant la fonction de covariance exponentielle sur un espace périodique. Les courbes ont été générées avec $a = \sqrt{3}$ et $\lambda = \sqrt{0.2}$. La fonction de moyennes est $\mu(x) = 0$.

permet de modifier l'aspect général du processus. Par exemple, l'emploi d'une carré-exponentielle produit des fonctions lisses et sans aspérité dont l'instanciation sur une période est répétée indéfiniment.

2.4 Régression par processus Gaussiens

L'objectif d'une régression est de modéliser la relation existant entre des données d'entrée et des données de sortie à valeurs réelles. Cette section montre comment tirer profit des qualités des processus Gaussiens afin de résoudre ces problèmes de régression. De par leur nature aléatoire, les processus Gaussiens peuvent servir de distribution de probabilité sur un espace de fonctions (voir section 2.3.2). Par conséquent, il est concevable de les utiliser en tant que distribution de probabilité en contexte d'inférence Bayésienne.

Tel que relaté à la section 2.2, l'inférence Bayésienne se traduit par la mise à jour des probabilités a priori sur un ensemble d'hypothèses envisagées, en fonction de leur capacité respective à expliquer des observations. En ce qui concerne les problèmes de

régression, l'objectif est d'estimer une fonction f expliquant au mieux un ensemble d'observations. L'idée de la régression par processus Gaussien est de procéder à l'inférence de la fonction f en adoptant un raisonnement Bayésien directement dans l'espace des fonctions. Ainsi, en définissant une distribution de probabilité a priori dans l'espace de fonctions, l'application du processus d'inférence permettra d'augmenter la probabilité des fonctions arrivant à expliquer les observations et, par conséquent, celles qui n'ont pas cette capacité verront leur probabilité réduite.

Suivant la démarche Bayésienne, il faut d'abord et avant tout exprimer les connaissances a priori du problème sous forme de distribution de probabilités. Pour les processus Gaussiens, cette étape revient à déterminer une fonction de covariance k, qui contrôle la forme de la fonction recherchée, et une fonction de moyennes μ, qui précise le sous-espace en sortie où elle est attendue. Une fois ces fonctions définies, la formulation de la distribution a priori est donnée par :

$$p(\mathbf{f}) = \mathcal{N}(\boldsymbol{\mu}, \mathbf{K}) \tag{2.19}$$

où $\mathbf{f}$ est le vecteur de variables aléatoires représentant la fonction f, $\mathbf{K}$ est la matrice de covariance construite selon $\mathbf{K}_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$ et $\boldsymbol{\mu}$ est le vecteur de moyenne construit tel que $\mu_i = \mu(\mathbf{x}_i)$. Cette formulation découle de l'hypothèse fondamentale que nous faisons par utilisation des processus Gaussiens, soit que tout sous-ensemble fini $\mathbf{f}$ de variables aléatoires appartenant à f possède une distribution jointe Gaussienne. Ainsi, sous la distribution a priori (2.19), toute fonction envisageable se voit attribuer une certaine probabilité, avant même d'avoir pu observer quoi que ce soit qui servira lors du processus de mise à jour des probabilités.

Lorsque le vecteur $\mathbf{f}$ ne peut être observé directement et qu'uniquement une version bruitée $\mathbf{y}$ de celui-ci est disponible, il est nécessaire d'introduire un modèle de bruit. Il faut donc introduire un modèle d'observations à cet égard. Ce modèle sert de lien entre la distribution sur les fonctions définies en (2.19) et les observations bruitées. Dès lors, la définition d'un modèle d'observation conduit à une distribution de probabilités conditionnelles sur les observations $p(\mathbf{y}|\mathbf{f})$, laquelle est exploitée pour la mise à jour de la distribution sur les fonctions. Pour la suite, la notation $\mathbf{y}$ a été choisie afin de mettre l'emphase sur le fait que les observations servant à l'apprentissage ne proviennent généralement pas directement de la fonction f.

Le raisonnement à appliquer lorsque les données d'entraînement sont bruitées est identique au cas sans bruit. Lors d'une mise à jour à partir d'observations bruitées, les différentes fonctions de l'espace de fonction deviennent plus ou moins vraisemblables en proportion de leurs aptitudes à expliquer des observations qui, cette fois, sont simplement bruitées. Pour formaliser ce type d'incertitude liée aux observations, nous em-

ployons le modèle standard de régression linéaire :

$$y = f(\mathbf{x}) + \varepsilon \tag{2.20}$$

où ε est un bruit blanc Gaussien de variance σ_y^2. L'hypothèse sur le type de bruit auquel est soumis $f(\mathbf{x})$ donne lieu à une probabilité conditionnelle sous forme Gaussienne décrit par :

$$p(\mathbf{y}|\mathbf{f}) = \mathcal{N}(\mathbf{f}, \sigma_y^2\mathbf{I}) \tag{2.21}$$

Dans cette expression, le vecteur de moyennes $\mathbf{f}$ est calculé à partir de l'élément [10] de l'espace de fonctions que l'on considère et $\mathbf{I}$ est la matrice identité utilisée pour indiquer un bruit stationnaire. Ainsi, en évaluant cette expression pour une fonction arbitraire, nous obtenons la probabilité que celle-ci ait pu générer les observations, donc la *vraisemblance* de la fonction. Bien entendu, la fonction réelle n'est connue qu'avec une certaine probabilité. Afin de considérer tous les éléments de l'espace des fonctions lors de l'explication des observations, il faut prendre en compte la contribution de chacun. Il est donc essentiel de pondérer la vraisemblance de chaque fonction face aux données par rapport à leur probabilité a priori respective [11]. Ceci est réalisable grâce à l'intégrale suivante :

$$\begin{aligned} p(\mathbf{y}) &= \int p(\mathbf{y}|\mathbf{f})p(\mathbf{f})d\mathbf{f} \\ &= \mathcal{N}(\boldsymbol{\mu}, \mathbf{K} + \sigma_y^2\mathbf{I}) \end{aligned} \tag{2.22}$$

que l'on nomme *vraisemblance marginale*. Au premier abord, ce terme correspond en fait uniquement à la constante de normalisation de la distribution a posteriori. En revanche, cette vraisemblance marginale est fréquemment utilisée lors du processus de sélection du modèle dans la hiérarchie Bayésienne. La section 2.5 explique comment l'utiliser afin de déterminer les hyperparamètres de la fonction de covariance.

2.4.1 La prédiction

La prédiction est généralement l'objectif principal d'une régression, où une entrée $\mathbf{x}_*$ est passée à l'algorithme d'apprentissage pour qu'il fournisse par la suite une estimation de la valeur en sortie $\mathbf{y}_*$. Dans le cas de la régression par processus Gaussien, c'est à partir de la distribution a posteriori sur l'espace des fonctions que sont réalisées les prédictions. D'un point de vue probabiliste, le calcul de cette distribution est une opération relativement simple.

10. Un élément d'un espace de fonction est une fonction en soi, laquelle peut avoir été échantillonnée d'une distribution sur un espace de fonction.

11. Voir section 2.2 pour plus d'explication sur le calcul de distribution a posteriori.

Supposons qu'un ensemble d'entraînement $\mathcal{D} = \{\mathbf{X}, \mathbf{y}\}$ soit disponible et qu'il contienne des observations potentiellement bruitées. Selon l'hypothèse que ces observations proviennent d'un processus Gaussien, nous pouvons alors écrire directement :

$$\mathbf{y} \sim \mathcal{N}\left(\boldsymbol{\mu}, \mathbf{K} + \sigma_y^2 \mathbf{I}\right) \tag{2.23}$$

où les fonctions de moyenne $\mu(\mathbf{x})$ et covariance $k(\mathbf{x}, \mathbf{x}')$ reflètent les connaissances a priori. Comme les variables aléatoires $\mathbf{y}$ ont été instanciées aux valeurs fournies par l'ensemble d'entraînement $\mathcal{D}$, leurs probabilités peut être calculées par rapport à la distribution a priori (2.23). Cette information peut donc être utilisée pour obtenir une distribution de probabilité conditionnelle sur l'espace des fonctions qui sera utilisée pour réaliser des prédictions.

Pour ce faire, il s'agit d'introduire de nouvelles variables aléatoires $\mathbf{y}_* = \{y_{*1}, y_{*2}, \ldots\}$, dont les entrées sont $\mathbf{X}_* = \{\mathbf{x}_{*1}, \mathbf{x}_{*2}, \ldots\}$, qui seront conditionnées en fonction de $\mathcal{D}$. Par définition, ces nouvelles variables possèdent une distribution jointe Gaussienne avec l'ensemble d'entraînement tel que

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{y}_* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \boldsymbol{\mu} \\ \boldsymbol{\mu}_* \end{bmatrix}, \begin{bmatrix} \mathbf{K} + \sigma_y^2 \mathbf{I} & \mathbf{K}_*^\top \\ \mathbf{K}_* & \mathbf{K}_{**} + \sigma_y^2 \mathbf{I} \end{bmatrix}\right) \tag{2.24}$$

où $\boldsymbol{\mu}_*$ provient de la fonction de moyennes $\mu(\mathbf{x})$, $\mathbf{K}_{**} + \sigma_y^2 \mathbf{I}$ est la matrice de covariance a priori calculée à partir de $\mathbf{X}_*$ et $\mathbf{K}_*$ est la matrice de covariance croisée calculée à partir de $\mathbf{X}$ et $\mathbf{X}_*$ dont les éléments correspondent à $k(\mathbf{x}_i, \mathbf{x}_{*j})$ [12]. En appliquant la règle de conditionnement d'une Gaussienne sur la distribution jointe (2.24), nous obtenons les équations de prédiction de la régression par processus Gaussien

$$\begin{aligned} \mathbb{E}[\mathbf{y}_* | \mathcal{D}] &= \boldsymbol{\mu}_* + \mathbf{K}_*^\top [\mathbf{K} + \sigma_y^2 \mathbf{I}]^{-1} (\mathbf{y} - \boldsymbol{\mu}) \\ \mathrm{Cov}[\mathbf{y}_* | \mathcal{D}] &= \mathbf{K}_{**} - \mathbf{K}_*^\top [\mathbf{K} + \sigma_y^2 \mathbf{I}]^{-1} \mathbf{K}_* + \sigma_y^2 \mathbf{I} \end{aligned} \tag{2.25}$$

où $\mathbb{E}$ dénote l'espérance et Cov la covariance. En fait, ces dernières équations définissent exactement le processus Gaussien a posteriori recherché sur l'espace des fonctions. Par contre, la démarche menant aux équations (2.25) suppose que l'objectif est d'apprendre le processus en tant que tel. En d'autres mots, nous apprenons la fonction bruitée et non la fonction sous-jacente. Lorsque le bruit est une composante nuisible à la prédiction et que seule l'identification de la fonction non-bruitée cachée est d'intérêt, il convient d'adopter une approche légèrement différente.

Lors de la formulation de la distribution (2.24), les variables de prédictions ont été introduites en utilisant la même fonction de covariance que celle du processus (2.23),

12. La fonction de covariance permettant de générer un bruit blanc Gaussien retourne une covariance nulle et une variance correspondant au bruit.

et par conséquent, elles incluent le bruit. Comme la somme de plusieurs processus Gaussiens résulte en un processus Gaussien dont les fonctions de moyenne et de covariance ont été additionnées, nous pouvons utiliser cette propriété pour décomposer un processus et ainsi retirer le bruit. Par conséquent, si le processus à identifier est une unique composante du processus Gaussien globale, les variables de prédiction doivent suivre la même distribution que la composante en question. En formant la distribution jointe entre $\mathbf{y}$ et de nouvelles variables aléatoires $\mathbf{f}_*$ n'étant pas assujetties au bruit, contrairement à $\mathbf{y}_*$, nous pouvons alors écrire,

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{f}_* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \boldsymbol{\mu} \\ \boldsymbol{\mu}_* \end{bmatrix}, \begin{bmatrix} \mathbf{K} + \sigma_y^2 \mathbf{I} & \mathbf{K}_*^\top \\ \mathbf{K}_* & \mathbf{K}_{**} \end{bmatrix} \right) \tag{2.26}$$

expression où la matrice de covariance de $\mathbf{f}_*$ ne contient pas de variance ajoutée. Suite au conditionnement, nous obtenons des équations de prédiction légèrement modifiée

$$\begin{aligned} \mathbb{E}[\mathbf{f}_*|\mathcal{D}] &= \boldsymbol{\mu}_* + \mathbf{K}_*^\top \left[\mathbf{K} + \sigma_y^2 \mathbf{I}\right]^{-1} (\mathbf{y} - \boldsymbol{\mu}) \\ \mathrm{Cov}[\mathbf{f}_*|\mathcal{D}] &= \mathbf{K}_{**} - \mathbf{K}_*^\top \left[\mathbf{K} + \sigma_y^2 \mathbf{I}\right]^{-1} \mathbf{K}_* \end{aligned} \tag{2.27}$$

où la partie bruit de la matrice de covariance qu'il convient d'éliminer est mise en évidence. La principale différence entre les équations (2.25) et (2.27) est qu'en présence d'observations bruitées, la première sert à la prédiction de sorties bruitées et la seconde à la prédiction de sortie sans bruit. D'un autre point de vue, l'un sert à reproduire intégralement le comportement du processus et l'autre à déterminer la fonction sous-jacente.

Le calcule des équations (2.25) et (2.27) est dominé par l'inversion d'une matrice de dimension $N \times N$, où N est la taille de l'ensemble d'entraînement. La complexité de cette opération est en général $\mathcal{O}(N^3)$. Précisons qu'il existe des méthodes d'inversion sophistiquées et d'approximation d'inverse qui permettent de réduire ce temps de calcul. Il est aussi possible de précalculer certaines parties des équations afin de réduire la complexité à $\mathcal{O}(N)$ pour une prédiction de moyenne et $\mathcal{O}(N^2)$ pour une variance. Actuellement, les meilleures méthodes d'approximation consistent à développer des processus Gaussiens creux pouvant abaisser la complexité à $\mathcal{O}(M^2 N)$, où $M \times M$ est la taille de la *matrice creuse*[13] [Snelson, 2007].

2.5 L'apprentissage des hyperparamètres

Déterminer une distribution a priori sur un espace de fonctions requiert, dans le cas des processus Gaussiens, la détermination des fonctions de moyenne et de cova-

13. Une matrice contenant beaucoup de valeur nulle, mieux connue sous le nom de "sparse matrix".

riance. Pour les problèmes typiques d'apprentissage automatique, les paramètres exacts de ces fonctions ne sont pas connus a priori. Le choix de ces paramètres $\boldsymbol{\theta}$, que l'on identifie aussi comme les *hyperparamètres* d'un processus Gaussien, aura une incidence directe sur les capacités de prédiction du processus Gaussien a posteriori. À la section précédente, nous avons détaillé la procédure de mise à jour d'un processus Gaussien a priori lorsqu'un ensemble d'entraînement est disponible. À partir de ces mêmes observations, il est aussi possible de reviser les hyperparamètres préalablement choisis.

Les méthodes permettant l'apprentissage des hyperparamètres sont très variées. L'une d'entre elles est la validation croisée, couramment utilisée pour le problème de sélection de modèles. Cette méthode consiste à diviser un ensemble observations en deux ensembles, l'un destiné à l'entraînement du modèle, et l'autre à valider la qualité de celui-ci.

Avec les processus Gaussiens, cela peut se réaliser en formant d'abord la matrice de covariance des données d'entraînement pour ensuite produire une distribution a posteriori sur les données de validation. Cette distribution est Gaussienne et s'obtient grâce aux équations (2.25) ou (2.27). En fixant le vecteur de prédiction $\mathbf{y}_*$ ou $\mathbf{f}_*$ aux valeurs de sorties de l'ensemble de validation, nous obtenons la probabilité de cet ensemble, donc la vraisemblance du modèle appris. Notons que cette Gaussienne dépend des exemples d'entraînement, mais aussi des hyperparamètres $\boldsymbol{\theta}$ de la fonction de covariance. Par conséquent, il est possible d'optimiser la probabilité de l'ensemble de validation par rapport aux valeurs contenues dans $\boldsymbol{\theta}$ [14].

L'un des avantages à employer la validation croisée est la réduction du risque de *surapprentissage*. Rappelons que celui-ci consiste en une explication pratiquement parfaite des données d'entraînement, contre de piètres performances face aux données encore jamais observées, ce qui mène à un sérieux manque au niveau de la généralisation.

D'un autre côté, il est aussi possible de tirer avantage de la nature probabiliste des processus Gaussien pour remplacer la validation croisée. Dans un cadre Bayésien, l'incertitude associée aux hyperparamètres devrait être prise en compte en choisissant une distribution a priori $p(\boldsymbol{\theta})$, pour ensuite la mettre à jour. La distribution a posteriori se calcule par :

$$p(\boldsymbol{\theta}|\mathbf{y}) = \frac{p(\mathbf{y}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\mathbf{y})} \tag{2.28}$$

avec une vraisemblance marginale est donnée par

$$p(\mathbf{y}) = \iint p(\mathbf{y}|\mathbf{f})p(\mathbf{f}|\boldsymbol{\theta})p(\boldsymbol{\theta})\mathrm{d}\mathbf{f}\mathrm{d}\boldsymbol{\theta} \tag{2.29}$$

14. Voir section 2.2 pour plus de détail sur l'apprentissage Bayésien hiérarchique.

où l'intégrale sur $\mathbf{f}$ est la même que la vraisemblance marginale obtenue en (2.22). Malheureusement, l'intégral sur $\boldsymbol{\theta}$ n'a pas de forme analytique dans le cas général, ouvrant ainsi la porte aux diverses méthodes d'approximation de la distribution a posteriori décrit par l'équation (2.28).

2.5.1 Maximum de vraisemblance a posteriori

L'objectif de cette approche est de trouver les hyperparamètres qui maximisent la probabilité des exemples d'entraînement. La recherche en question est en réalité un problème d'optimisation sur l'espace des hyperparamètres dont la fonction objective est l'équation (2.28). Comme la vraisemblance marginale (2.29) n'est qu'une constante multiplicative, sa contribution à la fonction objective n'a aucun effet sur l'optimisation. Il est donc pertinent de n'utiliser que les termes au numérateur, soit le produit de la vraisemblance et la probabilité a priori, ce qui permet de réduire la complexité de la procédure. D'autre part, il est pratique et plus stable numériquement de prendre le logarithme de ces dernières quantités, ce qui donne :

$$\log p(\boldsymbol{\theta}|\mathbf{y}) \propto -\frac{1}{2}(\mathbf{y}-\boldsymbol{\mu})^\top \left[\mathbf{K}+\sigma^2\mathbf{I}\right]^{-1}(\mathbf{y}-\boldsymbol{\mu}) - \frac{1}{2}\log|\mathbf{K}+\sigma^2\mathbf{I}| - \frac{N}{2}\log 2\pi + \log p(\boldsymbol{\theta}) \tag{2.30}$$

où $\log p(\boldsymbol{\theta})$ est la log-distribution a priori des hyperparamètres, les autres termes sont la log-Gaussienne provenant de (2.23), le symbole $|\cdot|$ denote le déterminant d'une matrice et N est le nombre d'observations. Dans le cas où $p(\boldsymbol{\theta})$ est une distribution uniforme sur l'espace des hyperparamètres, le dernier terme disparaît, ce qui donne lieu à une optimisation de la vraisemblance marginale classique. Soit pour des raisons évidentes, les distributions de la famille exponentielle sont particulièrement intéressantes pour définir $p(\boldsymbol{\theta})$, car elles permettent de simplifier le calcul de la fonction objective (2.30).

La figure 2.9 illustre un exemple de distribution a posteriori (2.28) à optimiser par rapport à des hyperparamètres de distribution uniforme. Pour l'obtenir, nous avons défini un processus Gaussien à moyenne nulle et de fonction de covariance carré-exponentielle dont les hyperparamètres ont été fixés à $a = 0.85$ et $\lambda = 0.65$. Ce processus Gaussien est ensuite échantillonné pour obtenir un ensemble d'entraînement $(\mathbf{X}, \mathbf{y})$ qui servira à retrouver le processus Gaussien initial. Finalement, nous avons évaluer (2.28) en quadrillant l'espace des hyperparamètres afin d'illustrer la forme de la fonction.

Tel qu'illustré sur le graphique 2.9, le couple d'hyperparamètres $a = 0.85$ et $\lambda = 0.65$ est très vraisemblable par rapport aux données d'entraînement. Évidemment, face à un problème d'apprentissage supervisé, la fonction de covariance ainsi que ses paramètres sont généralement inconnus. Pour notre exemple, l'optimisation de la fonction

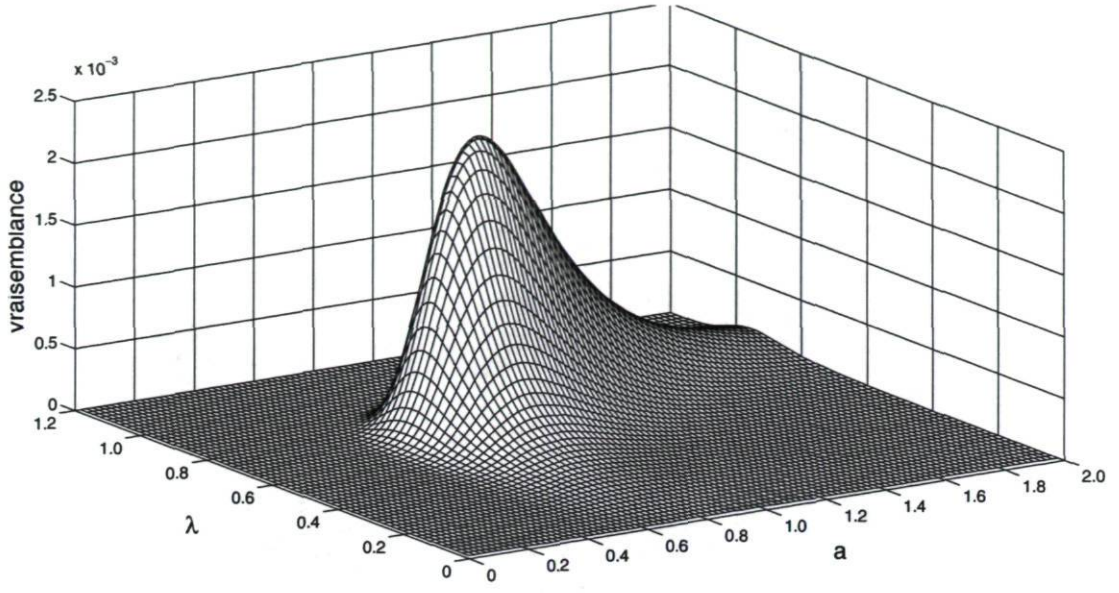


FIGURE 2.9 – Fonction de vraisemblance des hyperparamètres d'un processus Gaussien utilisant la fonction carré-exponentielle où l'ensemble de données a été échantillonnée avec $a = 0.85$ et $\lambda = 0.65$.

de vraisemblance illustrée à la figure 2.9 permettrait de trouver d'excellents hyperparamètres, car on remarque que la vraisemblance a un maximum local très près des hyperparamètres utilisés. L'apprentissage des hyperparamètres est donc un moyen pour déterminer une fonction de covariance qui arrive à expliquer un ensemble d'observations avec une certaine probabilité et ainsi d'obtenir un meilleur processus Gaussien a posteriori. C'est d'ailleurs le choix de la fonction de covariance qui déterminera comment l'information de l'ensemble d'entraînement sera exploitée lors de la prédiction.

L'une des méthodes les plus répandues pour résoudre les problèmes d'optimisation est celle du gradient conjugué. Cette approche nécessite par conséquent la connaissance des dérivées partielles de l'équation (2.30) par rapport aux hyperparamètres, soit :

$$\frac{\partial}{\partial \theta_j} \log p(\boldsymbol{\theta}|\mathbf{y}) = \frac{1}{2}(\mathbf{y} - \boldsymbol{\mu})^\top \mathbf{K}^{-1} \frac{\partial \mathbf{K}}{\partial \theta_j} \mathbf{K}^{-1}(\mathbf{y} - \boldsymbol{\mu}) - \frac{1}{2}\mathrm{Tr}\left(\mathbf{K}^{-1}\frac{\partial \mathbf{K}}{\partial \theta_j}\right) \tag{2.31}$$

où Tr denote la trace et $\frac{\partial \mathbf{K}}{\partial \theta_j}$ est construit avec la fonction de covariance dérivée par rapport aux θ_j. Dans son ensemble, l'optimisation est une procédure assez coûteuse. Le temps de calcul des dérivés partielles est principalement dominé par l'inversion de la matrice de covariance, qui requière $\mathcal{O}(N^3)$ opérations. Dans ces conditions, la complexité globale pour M itérations d'optimisation est de $\mathcal{O}(N^3 M)$.

2.5.2 Identification automatique de la pertinence

L'un des aspects intéressants de l'apprentissage des hyperparamètres est la possibilité d'identifier les dimensions pertinentes de la variable d'entrée. La pertinence qu'il convient de mesurer ici est l'information qu'une dimension d'entrée apporte quant à la bonne prédiction de la variable de sortie. D'autre part, cette procédure permet d'ajuster les paramètres de la fonction de covariance afin de prendre en compte le fait que les différentes dimensions d'entrée peuvent varier plus ou moins rapidement. L'idée est, en quelque sorte, de normaliser les dimensions par une transformation linéaire sur l'espace d'entrée pour que chaque dimension varie similairement.

Les fonctions de covariances vues à la section 2.3.3 ont été décrites sous une forme isotropique, car la norme utilisée était supposée Euclidienne. Lorsqu'une fonction de covariance est basée sur un autre type de distance, il se peut que la covariance qu'elle retourne ne soit plus invariante aux rotations sur l'espace d'entrée. Dès lors, cette fonction de covariances est dite anisotopiques. Dans le cas de la distance de Mahalonobis

$$d(\mathbf{x}, \mathbf{x}') = \sqrt{(\mathbf{x} - \mathbf{x}')C^{-1}(\mathbf{x}, \mathbf{x}')}, \tag{2.32}$$

la matrice semi-définie positive C peut être paramétrée et apprise lors de la phase d'optimisation des hyperparamètres. Dans le cas général, le résultat est une transformation linéaire de l'espace d'entrée qui adapte la covariance entre les variables de sortie. Lorsqu'on restreint la matrice C à être diagonale, chaque dimension est associée à un facteur d'échelle spécifique. Apprendre ces facteurs d'échelle revient à pondérer la pertinence de chaque dimension par l'inverse du facteur d'échelle. Par conséquent, une dimension ayant un facteur d'échelle très élevé suite à l'optimisation est en quelque sorte inutile, et peut ainsi être éliminée [Neal, 1996].

La figure 2.10 montre à nouveau la distribution a posteriori (2.28), mais cette fois en fonction de différent facteur d'échelle. Le problème est le même qu'à la figure 2.9 à l'exception que l'entrée possède maintenant une dimension supplémentaire non pertinente générée à partir d'une distribution uniforme. Cette dimension non pertinente devrait pouvoir être identifiable à partir de la distribution (2.28). Conséquemment, on remarque sur le graphique que la fonction favorise des facteurs d'échelle λ_2 toujours plus grand, ce qui nous indique que cette dimension est non pertinente. Pour ce qui est la dimension pertinente, le facteur d'échelle $\lambda_1 = 0.65$ original se voit attribuer une grande probabilité. Ainsi, une optimisation de la distribution a posteriori permettrait de trouver les hyperparamètres $\lambda_2 \to \infty$ et $\lambda_1 \approx 0.65$.

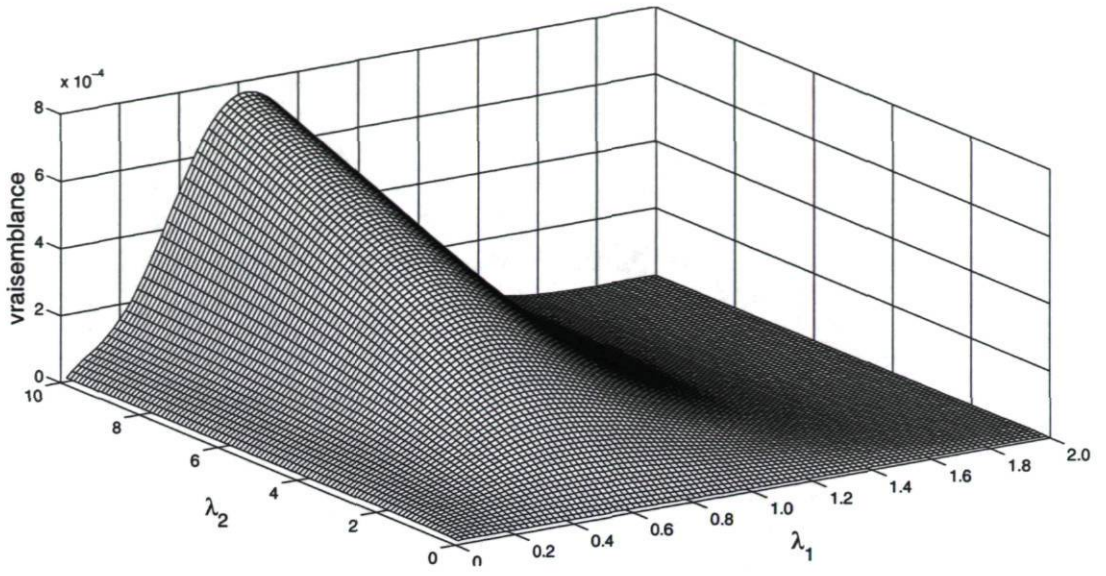


FIGURE 2.10 – Fonction de vraisemblance des facteurs d'échelle d'un processus Gaussien utilisant la fonction carré-exponentielle où l'ensemble de données contient une dimension non pertinente.

2.6 Experimentations

2.6.1 Exemple illustratif

Cette section offre un exemple d'apprentissage par processus Gaussien afin de consolider les concepts relatés au cours de ce chapitre. Il s'agira donc de placer une distribution a priori sur l'espace des fonctions en définissant un processus Gaussien. Par la suite, le calcul de la distribution a posteriori sur les fonctions permettra d'obtenir une estimation de la fonction inconnue.

La fonction à apprendre a été générée aléatoirement en échantillonnant un processus Gaussien basé sur une fonction de covariance réseau de neurones (voir section 2.3.4.3). De cette façon, nous obtenons une fonction exhibant un comportement complexe. Par contre, nous n'utiliserons pas la fonction de covariance réseau de neurones pour l'apprentissage, car en pratique cette information est généralement inconnue. La distribution a priori est plutôt déterminée par un processus Gaussien à moyenne nulle utilisant une fonction de covariance carré-exponentielle (voir section 2.3.3.1). Cette distribution est illustrée à la figure 2.11(a) par la zone grise autour de la fonction nulle et la courbe représente la fonction à identifier. Le calcul de la distribution a posteriori

a été fait à partir d'un ensemble d'entraînement contenant 20 exemples aléatoires et en déterminant les hyperparamètres par maximum de vraisemblance. La figure 2.11(b) montre la distribution a posteriori obtenue suite à l'apprentissage à partir des données d'entraînement indiquées par des croix.

Dans cet exemple d'apprentissage, on remarque qu'aux endroits éloignés des données d'entraînement, l'incertitude représentée par les zones grises est plus large. Il faut bien comprendre que cette mesure d'éloignement dépend des hyperparamètres, plus précisément du facteur d'échelle de la distance. L'optimisation des hyperparamètres permet d'apprendre à quoi correspond cette distance pour ce problème, car elle n'est pas connue a priori. Bien sûr, l'ajout de nouveaux exemples permettrait d'affiner à la fois l'estimation de la fonction et de rendre celle-ci plus certaine, ce qui se traduirait par un resserrement de la zone grise autour de la courbe.

2.6.2 Série temporelle de Mackey-Glass

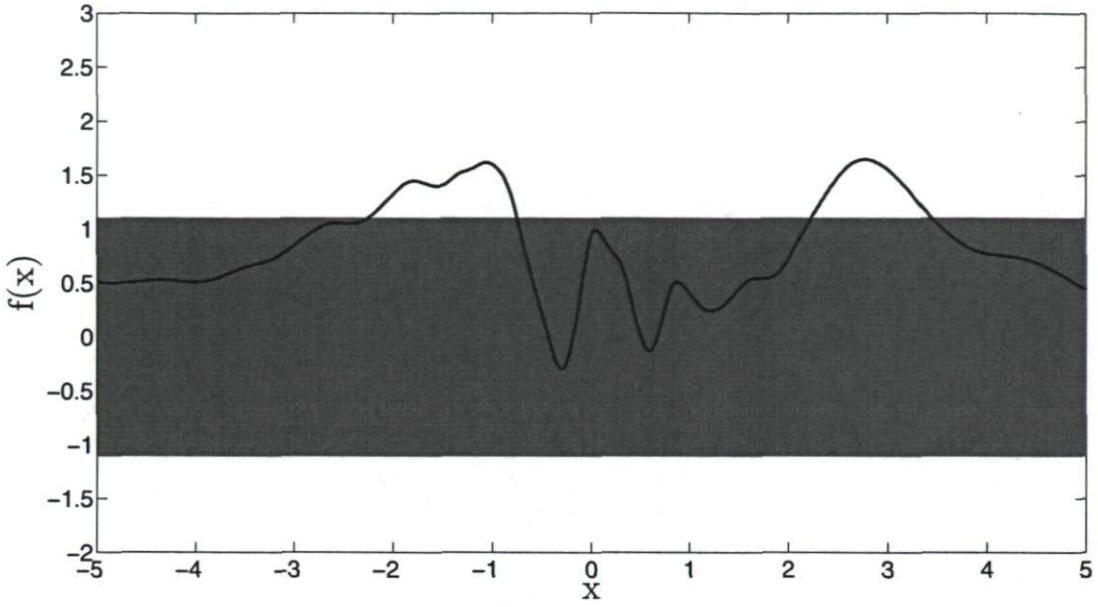
Afin de démontrer les performances de la régression par processus Gaussien, cette section propose d'apprendre à prédire une série temporelle basée sur les équations différentielles de Mackey-Glass. Les séries temporelles de Mackey-Glass sont une référence reconnue en ce qui a trait au problème de prédire le futur d'une série. Cette série, décrite par l'équation différentielle avec retard [Mackey et Glass, 1977]

$$\frac{dy(t)}{dt} = \beta \frac{y(t-\tau)}{1+y(t-\tau)^{10}} - \gamma y(t), \tag{2.33}$$

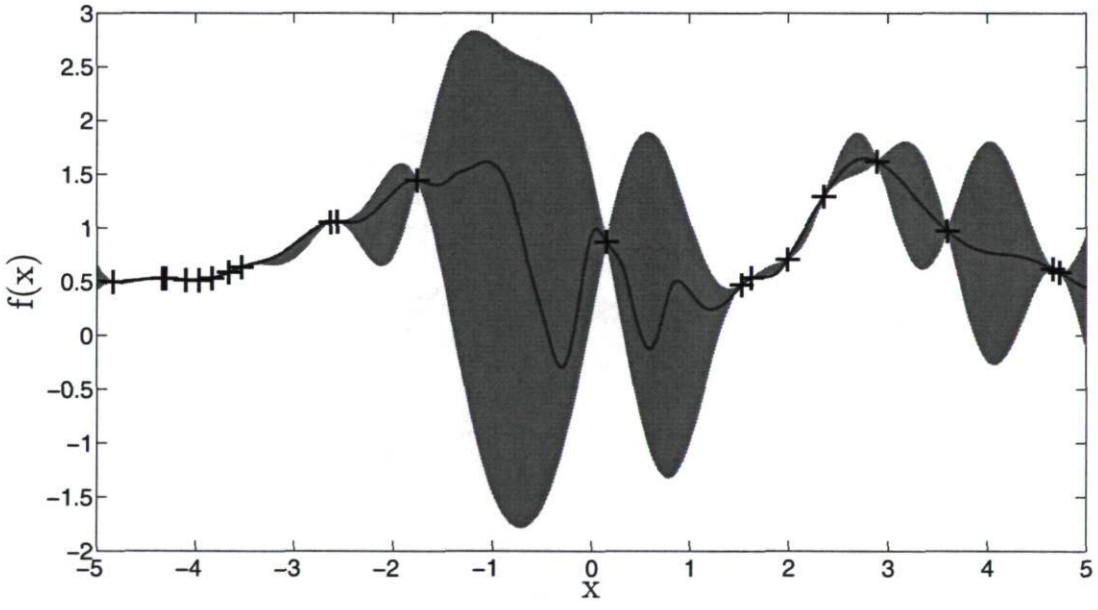
exhibe un comportement chaotique pour le cas que nous considérons, soit $\beta = 0.2$, $\gamma = 0.1$, un retard de $\tau = 17$ et un état initial $y(0) = 1.2$. Une série de 10000 valeurs discrétisées sur 1 pas de temps a été générée en utilisant la méthode de Runge-Kutta d'ordre 4. La série obtenue est par la suite transformée sous la forme d'un ensemble de couples entrée-sortie afin de pouvoir appliquer une méthode d'apprentissage supervisé. Formellement, en dénotant la série par $y(t)$, il s'agit d'abord d'identifier les changements $\Delta y(t)$ survenus à chaque pas de temps afin d'apprendre une forme différentielle. Par la suite, l'ensemble de données est construit tel que :

$$\Omega = \{(\mathbf{e}_i, s_i) \mid \mathbf{e}_i = [y(i), y(i-1), \ldots, y(i-\varphi)], s_i = \Delta y(i)\}_{i=\varphi}^{9999} \tag{2.34}$$

où l'entrée $\mathbf{e}_i$ est une fenêtre de temps de taille φ qui est associée à une sortie s_i correspondant au changement de l'état courant $y(i)$. Ainsi, cela revient à faire l'hypothèse qu'une séquence de φ états consécutifs est informative sur la variation de l'état qui doit survenir, et donc, informative sur l'état suivant. Le résultat est une formulation



(a) Distribution a priori sur l'espace des fonctions.



(b) Distribution a posteriori sur l'espace des fonctions.

FIGURE 2.11 – Illustration de l'apprentissage par processus Gaussien. La courbe en trait plein correspond à la fonction recherchée et les croix indiquent les exemples utilisés lors de l'entraînement. Les zones grises représentent des intervalles de confiance de 3 écarts-types.

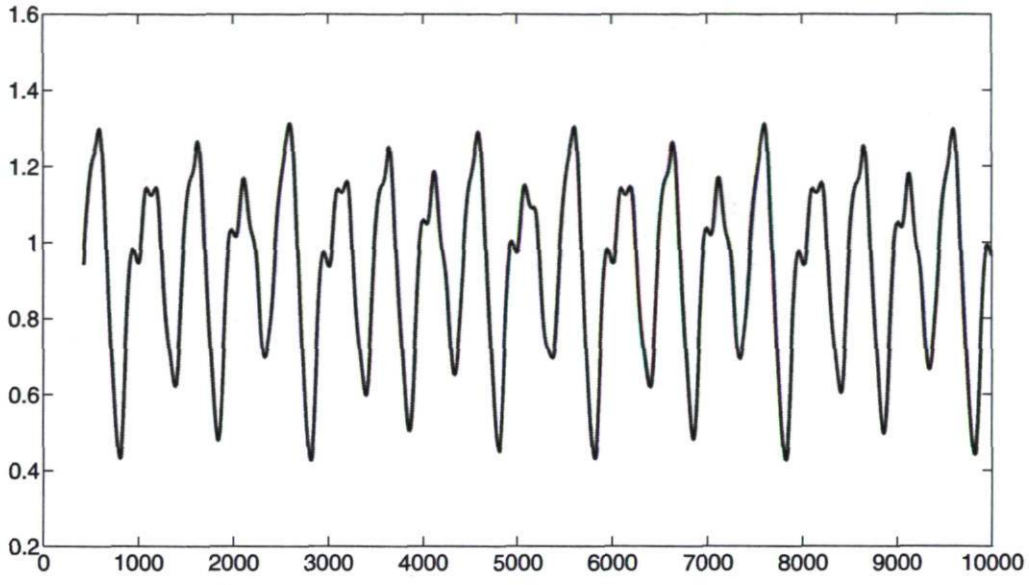


FIGURE 2.12 – Serie temporelle de Mackey-Glass avec $\beta = 0.2$, $\gamma = 0.1$ et un retard $\tau = 17$.

permettant d'apprendre le comportement de l'équation (2.33) sans connaître son retard τ.

Pour réaliser un apprentissage par processus Gaussien et mesurer ses performances, un ensemble de données Ω avec fenêtre $\varphi = 25$ a été divisé en 3 sous-ensembles. Le premier ensemble contient 1000 éléments équidistants de l'intervalle $[1000, 6000]$ et vise à entraîner le processus Gaussien. Le second ensemble sert à la validation et contient les éléments de l'intervalle $[6000, 8000]$. Finalement, les éléments de l'intervalle $[8000, 10000]$ forme l'ensemble de test et sont utilisés pour mesurer la performance de prédiction.

Lors de l'apprentissage du processus Gaussien, l'ensemble d'entraînement et une fonction de covariance carré-exponentielle ont été utilisés pour calculer les équations de prédictions (2.27). Les hyperparamètres ont été appris par optimisation stochastique grâce à l'ensemble de validation. L'évaluation des hyperparamètres est mesurée en fonction de la performance qui en résulte lorsqu'il faut prédire l'*évolution* de certains intervalles de l'ensemble de validation. L'idée est de fournir une première fenêtre $\mathbf{e}_i$ à l'algorithme qui par la suite doit retourner l'espérance et la variance de la sortie s_i. Comme s_i est en fait un changement $\Delta y(i)$ au niveau de l'état, il suffit de l'appliquer à la composante $y(i)$ appartenant à la fenêtre d'état $\mathbf{e}_i$ afin obtenir une estimation naïve [15]

15. Le terme naïf désigne le fait que nous n'utilisons pas l'incertitude lors de la prédiction, mais

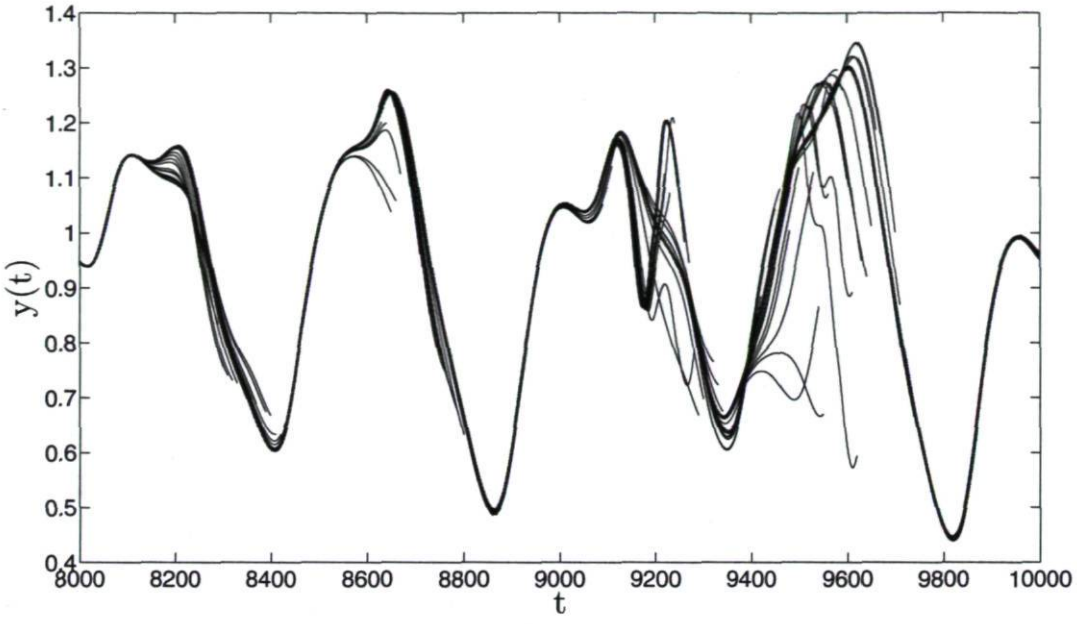


FIGURE 2.13 – Prédictions de 200 pas de temps sur les différents intervalles de l'ensemble test.

$\widehat{y}(i+1)$ de l'état suivant.

En appliquant cette mécanique répétitivement, cela revient à faire glisser la fenêtre temporelle de sorte à éliminer l'élément le plus ancien et y introduire l'estimation du nouvel état. Au fur et à mesure que la série temporelle est prédite, l'erreur de prédiction s'accumule, car éventuellement la fenêtre contient uniquement des estimations naïves d'états. Ultimement, nous désirons mesurer l'erreur de prédiction à long terme que fait le processus Gaussien lorsqu'on le combine à cette méthode de prédiction naïve. La mesure choisie est la racine carré de l'erreur quadratique moyenne normalisée (REQMN) :

$$\xi(n) = \sqrt{\sum_{i}^{i+n} \frac{(y(i) - \widehat{y}(i))^2}{n\sigma^2}} \tag{2.35}$$

où σ^2 est la variance de l'entière série temporelle et n est la longueur de l'intervalle à prédire.

Comme la série de Mackey-Glass adopte des comportements différents en fonction du temps, toute mesure d'erreur doit être moyennée sur plusieurs intervalles distincts. En minimisant la REQMN sur l'ensemble de validation par rapport aux hyperparamètres,

uniquement l'espérance.

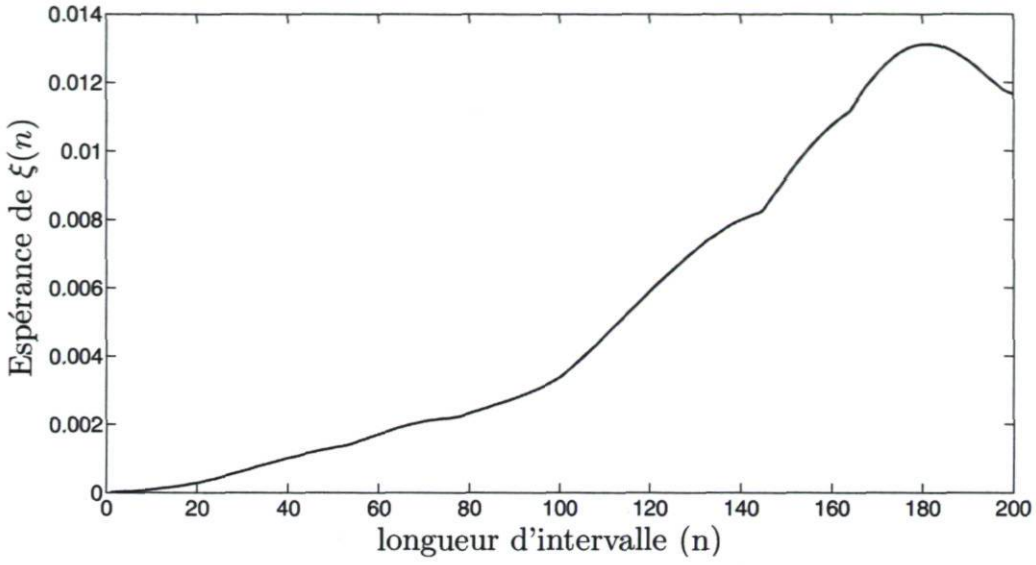


FIGURE 2.14 – Racine carrée de l'erreur quadratique moyenne normalisée en fonction de la longueur de l'intervalle de prédiction.

nous avons obtenu $a = 0.0035$ et $\lambda = 0.1672$. Les intervalles de prédictions sont au maximum $n = 200$ et une translation de 50 pas de temps est utilisée entre chaque intervalle. Les performances obtenues sur l'ensemble de validation sont excellentes, mais doivent être vérifiées sur l'ensemble de test afin de vérifier les capacités de généralisation de cet apprentissage.

La figure 2.13 montre les séquences de 200 prédictions (en traits fins) réalisées sur les différents intervalles. La figure 2.14 rapporte la moyenne des REQMN obtenus pour différentes longueurs d'intervalle de prédictions. Les résultats obtenus montrent que le processus Gaussien est parvenu à identifier l'équation efficacement. Tel qu'attendu, l'erreur s'accroît à mesure que les prédictions demandées sont de plus en plus loin dans le futur, ce qui est entièrement normal lorsqu'on utilise recursivement les prédictions antérieures. En particulier, la REQMN pour une prédiction à un seul pas de temps dans le futur est de 1.25^{-5}, ce qui est un excellent résultat s'interprétant comme un apprentissage précis de l'équation (2.33). Bien que l'approche naïve visant à prédire l'évolution de la série n'utilise que l'espérance, elle permet toutefois d'obtenir de bons résultats. Évidemment, l'utilisation de la variance lors des prédictions aurait ouvert la porte à de meilleures performances. Par contre, cela nécessite de réaliser des prédictions à partir de données incertaines, ce dont il sera question au chapitre 4.

2.7 Conclusion

Dans ce chapitre, nous avons couvert les bases de l'apprentissage par processus Gaussien. Cet outil puissant d'apprentissage supervisé est particulièrement intéressant de par sa nature probabiliste. En fait, les processus Gaussiens sont des distributions de probabilités dans l'espace des fonctions et permettent d'appliquer l'inférence Bayésienne, vue à la section 2.2, de façon plus directe. Dans ce contexte, le problème d'apprentissage revient ainsi à définir une bonne distribution a priori dans l'espace des fonctions, c'est-à-dire de déterminer la bonne fonction de covariance à utiliser, pour ensuite obtenir une distribution a posteriori appropriée.

Les performances qu'offre l'apprentissage par processus Gaussiens sont comparables aux méthodes de l'état de l'art en matière d'apprentissage supervisé, ce qui leur a permis de se forger une place respectable dans le domaine de l'apprentissage automatique. Dès lors, ils sont aujourd'hui couramment utilisés dans des contextes d'apprentissage plus large afin de résoudre des problèmes plus complexes. Le chapitre 3 couvre l'un de ces problèmes et explique comment les processus Gaussiens peuvent être exploités pour l'apprentissage de modèles cachés. Par la suite, une extension des processus Gaussiens aux données d'entrée incertaines est développée au chapitre 4.

Chapitre 3

Apprentissage de PDMPO

Les processus décisionnels de Markov partiellement observables (PDMPO) offrent une riche modélisation mathématique pour la prise de décisions séquentielles dans le monde réel. Cependant, la majorité des méthodes de résolution utilisées dans les processus décisionnels de Markov partiellement observables nécessitent la connaissance du modèle. De plus, les recherches actuelles sur le PDMPO se restreignent principalement aux espaces d'états discrets, ce qui complique son application à certains problèmes naturellement modélisés par un espace d'états continus. Dans ce chapitre, nous considérons le problème de contrôle optimal dans un environnement continu partiellement observable dont les paramètres du modèle sont inconnus. Face à ce problème, nous proposons une application des processus Gaussiens en vue d'apprendre le modèle en ligne, ainsi qu'un algorithme de planification en ligne permettant d'agir dans l'environnement.

3.1 Introduction

L'apprentissage par renforcement est une méthode générale permettant à un *agent* d'apprendre le comportement idéal à adopter à partir de ses interactions avec l'environnement. Ces dernières années, cette méthode est graduellement apparue comme une élégante technique pour résoudre les problèmes de prise de décisions séquentielles lorsque le modèle du monde est inconnu ou mal défini. Face à ce type de problèmes, il est essentiel de maintenir un équilibre entre l'exploration et l'exploitation du monde, un problème fondamental toujours ouvert en apprentissage par renforcement [Szepesvári, 2009]. Cet équilibre revient pour un agent à définir un compromis lui permettant de choisir ses actions pendant la phase d'apprentissage, de sorte qu'il améliore à la fois

ses connaissances du modèle pour mieux atteindre ses objectifs dans le futur (*l'exploration*), tout en satisfaisant ses objectifs immédiats sur la base des connaissances actuelles de l'environnement (*l'exploitation*). Il a été démontré que sous certaines conditions sur le comportement exploratoire, l'apprentissage par renforcement permet d'apprendre le choix d'action optimal [Sutton et Barto, 1998]. Ces conditions ne spécifient toutefois pas comment réaliser le compromis exploration-exploitation en vue d'atteindre l'optimalité.

Plus récemment, le paradigme standard d'inférence Bayésienne (voir section 2.2) a été appliqué à l'apprentissage par renforcement, donnant ainsi naissance à l'apprentissage par renforcement Bayésien basé sur le modèle. L'apprentissage par renforcement Bayésien permet d'optimiser naturellement les performances d'un agent lors de son apprentissage, ainsi que ses performances futures. Cette méthode consiste à représenter l'information a priori du problème, incluant l'incertitude, sous une forme paramétrique qui est ensuite intégrée dans un processus d'inférence Bayésienne afin d'incorporer les nouvelles informations acquises concernant le modèle. Ainsi, le dilemme exploration-exploitation est traité comme un problème explicite de prise de décision séquentielle où l'agent cherche à remplir ses objectifs en respectant son incertitude sur le modèle. Une limitation importante de cette approche réside dans le fait que la prise de décision devient significativement plus complexe lorsqu'il faut raisonner sur tous les modèles possibles et, par conséquent, toutes les séquences d'actions possibles. De plus, la majorité des travaux à ce jour sur l'apprentissage par renforcement Bayésien se sont concentrés sur le cas où l'agent a accès à une observabilité parfaite de l'état à chaque instant [Duff, 2002; Wang *et al.*, 2005b; Castro et Precup, 2007; Poupart *et al.*, 2006].

Actuellement, les cadres théoriques d'apprentissage par renforcement Bayésien avec modèle, appliqués à l'apprentissage de PDMPO, sont uniquement applicables aux domaines disposants d'un nombre fini d'états, d'actions et d'observations [Doshi *et al.*, 2008; Ross *et al.*, 2008a]. En se restreignant à ce type d'espace, cela permet d'utiliser les distributions de Dirichlets comme distribution a posteriori sur les distributions discrètes. En pratique, c'est une limitation importante due au fait que plusieurs problèmes du monde réel sont naturellement décrits par des ensembles continus d'états, d'actions et d'observations. Par exemple, pour les tâches de navigation robotique, l'état est généralement défini par un ensemble de variables à valeurs continues telles que la position, la vitesse, l'orientation, etc. De plus, les actions contrôlant l'accélération et l'orientation sont fréquemment continues, de même que peuvent l'être les observations bruitées de l'état fourni par les capteurs.

Une approche d'apprentissage par renforcement Bayésien de PDMPO continu a récemment été proposée par Ross *et al.* [2008b]. Cette approche fait toutefois l'hypothèse qu'une forme paramétrique des fonctions de transition et d'observation est disponible

et que seul le vecteur de moyenne et la matrice de covariance des variables aléatoires de bruit sont inconnus. Ainsi, ce modèle est difficilement applicable dans un contexte où les fonctions de transition et observation sont entièrement inconnues.

Pour palier à cette restriction, ce chapitre présente une approche d'apprentissage par renforcement Bayésien basée sur le modèle pouvant traiter des environnements à la fois partiellement observables et continus, et ce, sans aucune connaissance a priori sur les fonctions de transition, d'observation et de récompense [Dallaire *et al.*, 2009b]. Dans ce qui suit, nous exposons tout d'abord comment utiliser les processus Gaussiens pour apprendre le modèle du PDMPO. Nous faisons ensuite état d'un algorithme de planification en ligne permettant la sélection des actions en fonction de la distribution a posteriori sur le modèle.

3.2 Travaux connexes

La littérature sur les PDMPO continus est relativement rare. En général, l'approche la plus commune consiste à discrétiser l'espace d'états, ce qui par conséquent mène à un modèle incomplet au niveau du système sous-jacent.

Une première approche consacrée aux PDMPO à espace d'états continus est celle de Thrun [2000] où l'état de croyance est vu comme un ensemble pondéré d'échantillons (filtre à particule), ce qui peut être considéré comme un cas particulier de représentation par mixture de Gaussiennes. L'auteur présente un algorithme Monte Carlo pour apprendre à agir dans les PDMPO lorsque les états et les actions sont à valeurs réelles, reconnaissant ainsi le fait que plusieurs problèmes communs sont naturellement continus. Une *itération de la valeur* est ensuite utilisée afin d'apprendre la fonction de valeur sur les états de croyance. Une version par échantillonnage de la méthode des plus proches voisins est finalement appliquée pour généraliser sur l'espace d'états.

Une seconde approche a également été proposée par Porta *et al.* [2005]. Dans leur article, les auteurs montrent qu'avec un horizon fini, la fonction de valeur optimale sur l'espace infini des états de croyances du PDMPO est linéaire par morceaux et convexe. Ils montrent également cette même propriété pour une classe assez générale de PDMPO dont toutes les fonctions du modèle sont représentées par des mixtures de Gaussiennes. Ainsi, toutes les mises à jour de l'état de croyance et celles de l'itération de la valeur peuvent s'effectuer exactement et analytiquement. Les premiers résultats expérimentaux de ces auteurs montrent l'applicabilité de cette approche pour un problème simple de planification robotique en environnement continu. Pour l'implémentation, ils ont utilisé

un algorithme d'itération de la valeur à base de points aléatoires nommés PERSEUS [Spaan et Vlassis, 2005]. Ces mêmes auteurs ont par la suite prolongé leurs travaux au cas des actions et des observations continues en utilisant un échantillonnage efficace [Porta *et al.*, 2006]. Pour cela, ils ont réduit la forme générale du PDMPO continu à la classe des PDMPO à états continus seulement (en laissant donc les actions et les observations discrètes) en utilisant des stratégies d'échantillonnage. Contrairement à la méthode présentée dans ce chapitre, leur méthode fait l'hypothèse que le modèle du PDMPO est connu, ce qui constitue une limitation importante pour le type d'application réel que nous considérons.

En résumé, l'approche commune pour résoudre un PDMPO continu consiste à discrétiser l'espace d'états, ce qui peut entraîner une piètre modélisation du système en question. La reformulation de la forme générale du PDMPO continu vers le paradigme PDMPO à états continus ne change rien au niveau de la qualité de la modélisation. L'approche proposée dans ce chapitre permet d'éviter ce problème en proposant un modèle complètement continu du PDMPO où l'espace d'états, l'espace d'actions et l'espace d'observations sont tous continus et potentiellement multidimensionnels.

Une approche reliée aux travaux présentés dans ce chapitre fait référence à la modélisation de séries temporelles en utilisant les systèmes dynamiques [Wang *et al.*, 2008]. Dans le cas de l'analyse de séries temporelles non linéaires, Wang et ses collègues ont introduit les *modèles dynamiques basés sur les processus Gaussiens* (MDPG) afin d'apprendre des modèles de mouvements et de poses humains à partir de capteurs de mouvements fournissant des données de grande dimension. Leur MDPG comprend un espace d'état latent de faible dimension régi par des dynamiques Markoviennes, ainsi qu'une fonction de cet espace latent vers un espace d'observation. Les auteurs ont intégré les paramètres du modèle sous forme analytique en utilisant des distributions Gaussiennes a priori sur les paramètres des fonctions d'observations et de dynamiques. Il en résulte une modélisation non paramétrique de systèmes dynamiques qui peut être vue comme un modèle de Markov caché continu n'incluant ni action ni récompense.

3.3 Processus décisionnels de Markov

Avant de poursuivre sur le processus décisionnel de Markov partiellement observable, cette section revoit le cas particulier du processus décisionnel de Markov (PDM) entièrement observable qui est à base de la majorité des approches modernes d'apprentissage par renforcement [Szepesvári, 2009]. Ce modèle est essentiellement un produit de la théorie des probabilités, de la théorie de la décision et de la théorie de l'uti-

lité. Le processus décisionnel de Markov est généralement employé pour modéliser des problèmes de prises de décisions séquentielles où les dynamiques de l'environnement sont incertaines. En pratique, les PDM se sont avérés très efficaces pour résoudre de multiples problèmes jugés complexes, allant du contrôle en temps réel [Abbeel *et al.*, 2007] jusqu'au développement d'agent pour des jeux entre adversaires [Tesauro, 1995].

La modélisation d'un problème par un processus décisionnel de Markov entraîne certaines hypothèses sur l'environnement à traiter. Formellement, un PDM est composé de quatre éléments, soit l'espace d'états $\mathcal{S}$, l'espace d'actions $\mathcal{A}$, la fonction de transition $\mathcal{T}$ et la fonction de récompense $\mathcal{R}$:

- **L'espace d'états** : $\mathcal{S}$ représente l'ensemble de toutes les configurations possibles de l'environnement. Cet ensemble peut être fini, infini dénombrable ou continu. Par exemple, si un système est décrit par un certain nombre de variables (position, vitesse, orientation, etc.), l'espace d'états est l'ensemble des combinaisons de valeurs admissibles.
- **L'espace d'actions** : $\mathcal{A}$ représente l'ensemble des actions exécutables dans l'environnement. À chaque instant requérant qu'une décision soit prise, l'agent choisi parmi les actions disponibles, celle influençant au mieux l'environnement, de sorte qu'il atteigne ses objectifs. Encore une fois, cet ensemble peut être fini, infini dénombrable ou continu.
- **Les probabilités de transitions** : $\mathcal{T}$ est une fonction qui permet d'exprimer les dynamiques de l'environnement sous forme de distributions de probabilités. Plus précisément, lorsqu'une action $\mathbf{a} \in \mathcal{A}$ est exécutée à partir d'un état $\mathbf{s} \in \mathcal{S}$, cette fonction de transition $\mathcal{T}(\mathbf{s}, \mathbf{a}, \mathbf{s}') = p(\mathbf{s}'|\mathbf{s}, \mathbf{a})$ indique la probabilité conditionnelle de l'état suivant $\mathbf{s}' \in \mathcal{S}$. Remarquons que d'après cette définition, l'état suivant ne dépend que de l'état courant et aucunement des états et actions antérieurs. Cette caractéristique importante correspond à la *propriété de Markov* et stipule que l'état courant est une statistique suffisante pour prédire le futur de l'environnement.
- **Les probabilités de récompenses** : $\mathcal{R}$ permet d'encoder les objectifs à remplir dans l'environnement sous forme de récompenses immédiates. Cette fonction de récompenses indique la probabilité conditionnelle $\mathcal{R}(\mathbf{s}, \mathbf{a}, \mathbf{s}', r) = p(r|\mathbf{s}, \mathbf{a}, \mathbf{s}')$ d'obtenir une récompense $r \in \mathbb{R}$ lorsqu'une action $\mathbf{a} \in \mathcal{A}$ est exécutée dans un état $\mathbf{s} \in \mathcal{S}$ et que l'état résultant est $\mathbf{s}' \in \mathcal{S}$. Ces signaux de récompenses servent en fait à renforcer les comportements désirables et à réduire ceux qui ne le sont pas via des pénalités [1]. Dès lors, l'agent doit développer une stratégie lui permettant de combiner rationnellement la réalisation des différents objectifs afin d'obtenir

1. Par opposition à la récompense qui est généralement un nombre positif, la pénalité et le coût sont des nombres négatifs visant à limiter ou éliminer certains comportements.

le maximum de récompenses.

À partir de cette définition du processus décisionnel de Markov, un agent est en mesure d'agir dans un environnement dont les dynamiques sont régies par $\mathcal{T}$ et de recevoir une série de récompenses correspondant à la qualité de ses décisions. Comme le PDM est un processus stochastique à temps discret, les instants de prise de décision sont prédéfinis et généralement indexés par une variable de temps, ici t. À chaque étape de prise de décision t, l'agent reçoit d'abord une *perception* exacte de l'état $\mathbf{s}_t$ dans lequel il se trouve. Une fois l'état de l'environnement connu, l'agent choisit ensuite l'action $\mathbf{a}_t$ à exécuter afin d'influencer la probabilité de l'état suivant $\mathbf{s}_{t+1}$ et la probabilité de récompense immédiate r_t. Évidemment, il ne faut pas oublier qu'il existe une dépendance temporelle entre états consécutifs lorsque cette procédure est répétée séquentiellement.

Les décisions que pose l'agent dans l'environnement caractérisent son comportement général. Pour les PDM, ce comportement est formalisé par une fonction π, que l'on nomme *politique*[2] et qui dicte l'action qu'il convient d'exécuter en fonction de l'état courant. Cette fonction peut être aussi définie sous forme de probabilité conditionnelle $\pi(\mathbf{s}, \mathbf{a}) = p(\mathbf{a}|\mathbf{s})$, auquel cas la politique est dite stochastique.

Ce que l'on vient de décrire constitue donc un modèle entièrement probabiliste offrant une représentation de l'environnement et du comportement de l'agent. Ainsi, lorsqu'un agent applique une politique fixe π, il va générer des séquences d'états, actions et récompenses aléatoirement. Rappelons que le succès d'une séquence correspond à la somme des récompenses qu'elle contient. Bien entendu, cette somme de récompenses est une variable aléatoire dépendant à la fois de la politique π et de la fonction de transition $\mathcal{T}$, mais aussi de l'état initial $\mathbf{s}_0$ débutant la séquence. C'est à partir de la distribution de probabilité de cette somme de récompense qu'il est possible de mesurer l'utilité des états d'un PDM. En particulier, l'espérance de la somme des récompenses escomptées sert à définir la *fonction de valeur*

$$V^{\pi}(\mathbf{s}) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_t \mid \pi, \mathbf{s}_0 = \mathbf{s}\right] \tag{3.1}$$

qui retourne la *valeur* de l'état $\mathbf{s}$. Une telle valeur de l'état $\mathbf{s}$ indique la désirabilité de cet état sous la politique π. Notons aussi l'introduction d'un *facteur d'escompte* $\gamma \in [0, 1]$ permettant de favoriser les récompenses actuelles plutôt que les récompenses futures. Lorsque $\gamma = 0$, l'agent est entièrement vorace et néglige les récompenses à venir. Le

2. Nous traiterons uniquement le cas des politiques indépendantes du temps, et par conséquent, le terme politique sous-entend une politique stationnaire.

cas $\gamma = 1$ est un cas particulier où la valeur est exactement la somme de toutes les récompenses. Autrement, le facteur γ^t est le poids attribué à une récompense reçue à t étapes de temps dans le futur.

À partir de l'équation (3.1), il est possible de mettre en évidence une relation directe entre la valeur d'un état et la valeur des états adjacents. Pour l'obtenir, il suffit de sortir la récompense immédiate et d'observer que le reste de la somme correspond en fait à la valeur de l'état suivant $V^\pi(\mathbf{s}')$, ce qui s'écrit :

$$V^\pi(\mathbf{s}) = \sum_{\mathbf{a}\in\mathcal{A}} \pi(\mathbf{s},\mathbf{a}) \left[\mathbb{E}[\mathcal{R}(\mathbf{s},\mathbf{a},\mathbf{s}',r)] + \gamma \sum_{\mathbf{s}'\in\mathcal{S}} \mathcal{T}(\mathbf{s},\mathbf{a},\mathbf{s}') V^\pi(\mathbf{s}') \right] \tag{3.2}$$

Le calcul de la fonction de valeur V^π d'un processus décisionnel de Markov joue un rôle fondamental lors de l'*évaluation* de politiques. En calculant la valeur $V^\pi(\mathbf{s}_0)$ de l'état de départ de l'agent, cela revient à évaluer la capacité d'une politique π à récolter des récompenses à long terme. Cette mesure de performance, qui est en fait l'espérance de la somme des récompenses escomptées, permet de comparer les politiques entres elles, et donc, de rechercher la *politique optimale.*

Par définition, la politique optimale π^* est celle qui permet d'obtenir le plus grand nombre de récompenses escomptées espérées :

$$\pi^* = \arg\max_\pi V^\pi(\mathbf{s}_0) \tag{3.3}$$

Fait intéressant, Bellman [1957] a montré que pour tout processus décisionnel de Markov, il existe toujours une politique optimale déterministe π^* permettant d'obtenir au moins autant de récompenses que toute autre politique. Cela veut donc dire que pour tout état $\mathbf{s}$, il existe une action optimale qui maximise l'espérance des récompenses. Évidemment, la politique optimale doit respecter cette contrainte d'optimalité sur les actions dans tous les états du PDM, sinon elle ne serait pas optimale. Ainsi, la fonction de valeur de la politique optimale est définie telle que :

$$V^*(\mathbf{s}) = \max_{\mathbf{a}\in\mathcal{A}} \left[\mathbb{E}[\mathcal{R}(\mathbf{s},\mathbf{a},\mathbf{s}',r)] + \gamma \sum_{\mathbf{s}'\in\mathcal{S}} \mathcal{T}(\mathbf{s},\mathbf{a},\mathbf{s}') V^*(\mathbf{s}') \right] \tag{3.4}$$

où la politique optimale π^* correspond à une solution de l'opérateur *max* pour tout les états. On se réfère fréquemment à l'équation (3.4) sous le nom d'*équation de Bellman.* Il faut bien comprendre qu'il n'y a pas qu'une seule équation de Bellman, mais bien une équation par état. Dès lors, résoudre le système d'équations non linéaires relatif à ces états est équivalent à calculer la politique optimale du PDM. C'est d'ailleurs en se basant sur ces équations que l'algorithme d'*itération de valeurs* a été développé.

3.4 PDMPO continu

À la section 3.3, nous avons introduit le processus décisionnel de Markov entièrement observable. Cependant, l'hypothèse que l'état du système est accessible avec exactitude est parfois trop forte pour certains problèmes. Le processus décisionnel de Markov partiellement observable (PDMPO) est une extension du précédent modèle permettant de considérer des observations indirectes, partielles ou bruitées de l'état. Cette section présente le processus décisionnel de Markov partiellement observable sous sa forme continue. Formellement, nous pouvons définir le PDMPO continu par un tuple (S, A, Z, T, O, R) où :

- **L'espace d'états** : $\mathcal{S} \subset \mathbb{R}^m$ est continu et potentiellement multidimensionnel.
- **L'espace d'actions** : $\mathcal{A} \subset \mathbb{R}^n$ est continu et potentiellement multidimensionnel. Nous supposons que A est un sous-ensemble fermé de $\mathbb{R}^n$, afin que le contrôle optimal, défini plus bas, existe.
- **L'espace d'observations** : $\mathcal{Z} \subset \mathbb{R}^p$ est continu et potentiellement multidimensionnel. Cet espace contient les perceptions que peut recevoir l'agent suite à l'exécution d'une action. Une observation $\mathbf{z} \in \mathcal{Z}$ contient une certaine quantité d'information ou caractéristique de l'état, contrairement à l'état lui-même qui contient toute l'information.
- **Les probabilités de transitions** : La fonction de transition $\mathcal{T}$ détermine maintenant des densités de probabilités en raison de la continuité de $\mathcal{S}$.
- **Les probabilités de récompenses** : La fonction de récompense $\mathcal{R}$ est identique à celle du PDM.
- **Les probabilités d'observations** : La fonction d'observation $\mathcal{O}$ permet de déterminer l'information que l'agent reçoit concernant l'état courant de l'environnement. Plus précisément, elle détermine la densité de probabilité conditionnelle $O(\mathbf{s}, \mathbf{a}, \mathbf{s}', \mathbf{z}') = p(\mathbf{z}'|\mathbf{s}, \mathbf{a}, \mathbf{s}')$ d'observer $\mathbf{z}'$ lorsque l'agent atteint l'état $\mathbf{s}'$ suite à l'exécution de l'action $\mathbf{a}$ à partir de l'état $\mathbf{s}$.

Le PDMPO est particulièrement utile pour faire face aux problèmes de décision séquentielle en contexte d'observations partielles. La principale difficulté qu'occasionne l'observabilité partielle est que l'agent ne connaît l'état dans lequel il se trouve qu'avec une certaine probabilité. Cette distribution de probabilité représentant l'incertitude de l'agent concernant l'état courant se nomme l'*état de croyance*. Évidemment, l'état de croyance est appelé à changer au cours du temps en fonction des actions qu'exécute l'agent et des observations qu'il reçoit de l'environnement. Suite à une action $\mathbf{a}$ et une observations $\mathbf{z}'$, la distribution sur l'état courant $\mathbf{s}$, noté $b(\mathbf{s})$, est mise à jour via la règle

de Bayes :

$$b^{\mathbf{az}'}(\mathbf{s}') \propto \mathcal{O}(\mathbf{s}, \mathbf{a}, \mathbf{s}', \mathbf{z}') \int_S \mathcal{T}(\mathbf{s}, \mathbf{a}, \mathbf{s}') b(\mathbf{s}) d\mathbf{s} \tag{3.5}$$

C'est en appliquant successivement cette dernière équation à mesure que les nouvelles observations sont disponibles que l'agent peut maintenir à jour son état de croyance.

Malgré l'incertitude sur l'état de croyance, l'agent doit prendre les décisions lui permettant d'atteindre son but qui consiste à recevoir le maximum de récompense. Comme pour le PDM, ces décisions peuvent prendre la forme d'un plan ou d'une politique qui, cette fois, est fonction d'un état de croyance plutôt que d'un état. Ainsi, dans le cas du PDMPO, une politique se note $\pi(b, \mathbf{a})$ et dicte le comportement qu'un agent adopte pour tout état de croyance possible. Lorsqu'une politique π est fixée, il est possible d'évaluer l'utilité des états de croyances en fonction de cette politique. L'utilité d'un état de croyance b, notée $V^{\pi}(b)$, est similaire à l'utilité de l'état pour un PDM. La seule différence est que maintenant il faut considérer l'incertitude sur l'état pour calculer la récompense escomptée espérée. Conséquemment, le calcul de la fonction de valeur permet de mesurer la qualité d'une politique pour un processus décisionnel de Markov partiellement observable.

Ce que l'agent recherche en réalité est la politique optimale $\pi^*(b, \mathbf{a})$, celle qui permet de maximiser la somme des récompenses escomptées espérées sur un *horizon* infini où l'escompte vise à favoriser les récompenses immédiates plutôt que celle nécessitant plusieurs actions. Dans le cas du PDMPO, la politique optimale peut être obtenue par la résolution des équations de Bellman définies selon :

$$V^*(b) = \max_{\mathbf{a} \in A} \left[g(b, \mathbf{a}) + \gamma \int_Z p(\mathbf{z}'|b, \mathbf{a}) V^*(b^{\mathbf{az}'}) d\mathbf{z}' \right] \tag{3.6}$$

où V^* est la fonction de valeur de la politique optimale et est aussi le point fixe de l'équation de Bellman. Afin de simplifier l'équation (3.6), nous avons introduit la fonction de densité

$$p(\mathbf{z}'|b, \mathbf{a}) = \int_S b(s) \int_S \mathcal{T}(\mathbf{s}, \mathbf{a}, \mathbf{s}') \, \mathcal{O}(\mathbf{s}, \mathbf{a}, \mathbf{s}', \mathbf{z}') d\mathbf{s} d\mathbf{s}' \tag{3.7}$$

correspondant à la probabilité conditionnelle d'observer $\mathbf{z}$ suite à l'exécution de l'action $\mathbf{a}$ dans l'état de croyance b et la quantité

$$g(b, \mathbf{a}) = \int_S b(\mathbf{s}) \int_S \mathcal{T}(\mathbf{s}, \mathbf{a}, \mathbf{s}') \int_{\mathbb{R}} r \mathcal{R}(\mathbf{s}, \mathbf{a}, \mathbf{s}', r) dr d\mathbf{s}' d\mathbf{s} \tag{3.8}$$

représentant la récompense espérée lorsque l'action $\mathbf{a}$ est exécuté dans l'état de croyance b. Le calcul des équations (3.7) et (3.8) n'est généralement pas possible analytiquement. Il est malgré tout possible de simplifier ces calculs grâce à des approximations ou en

choisissant judicieusement les distributions de probabilité pour obtenir une forme analytique. Malheureusement, comme l'opérateur *max* de l'équation (3.6) est un problème d'optimisation en soi pour chacun des états de croyances, le calcul de la fonction de valeur optimale d'un PDMPO continu est difficilement calculable en général.

Bien entendu, les précédentes équations supposent que le modèle du PDMPO est parfaitement connu, ce qui peut s'avérer une hypothèse forte pour certains problèmes. Une approche permettant de lever cette restriction consiste à incorporer l'apprentissage du modèle du PDMPO à l'apprentissage par renforcement par des méthodes Bayésiennes. La section suivante présente la solution proposée.

3.5 Application des processus Gaussien pour les PDMPO continus

Dans cette section, nous considérons le problème d'apprendre la politique optimale d'un PDMPO continu, tel que décrit à la section 3.4, lorsque les fonctions de transitions, observations et récompenses ne sont pas connues a priori. Afin d'envisager la prise de décisions optimales, le modèle est estimé via une modélisation par processus Gaussiens (voir section 2), et ce, uniquement à partir de séquences d'actions-observations. Les processus Gaussiens sont bien adaptés à l'identification de modèle en raison de leur interprétation probabiliste. L'un des avantages est qu'ils permettent non seulement d'exprimer les connaissances a priori d'un modèle en passant par la fonction de moyenne, mais aussi l'incertitude entourant ce modèle. Ainsi, au fur et à mesure que les données sur l'environnement sont acquises, telles des séquences d'actions-observations, le processus Gaussien s'affine aux moyen de ces données qui sont intégrées *en ligne*, et ce, pour obtenir une estimation a posteriori du modèle.

Afin d'apprendre le modèle d'un PDMPO continu grâce aux processus Gaussiens, il faut d'abord faire une hypothèse sur les fonctions de transitions $\mathcal{T}$, d'observations $\mathcal{O}$ et de récompenses $\mathcal{R}$. Nous supposerons que le modèle du PDMPO continu s'exprime sous la forme :

$$\begin{aligned} \mathbf{s}_t &= T'(\mathbf{s}_{t-1}, \mathbf{a}_{t-1}) + \varepsilon_T \\ \mathbf{z}_t &= O'(\mathbf{s}_t, \mathbf{a}_{t-1}) + \varepsilon_O \\ r_t &= R'(\mathbf{s}_t, \mathbf{a}_{t-1}) + \varepsilon_R \end{aligned} \tag{3.9}$$

où ε_T, ε_O et ε_R sont des variables de bruit blanc Gaussiens à moyenne nulle dont les variances sont inconnues. Les fonctions T', O' et R' sont des fonctions *déterministes*

qui retournent respectivement l'état suivant $\mathbf{s}_t$, l'observation $\mathbf{z}_t$ associé à cet état et la récompense r_t pour avoir atteint ce même état. Cette formulation du PDMPO revient à dire que les fonctions sont représentables par des distributions de probabilités Gaussiennes dont les moyennes sont données par ces fonctions déterministes et dont les variances sont fixes (bruit stationnaire). Cette hypothèse diminue légèrement la complexité du PDMPO que nous considérons, mais reste néanmoins plus flexible que le PDMPO déterministe [Besse et Chaib-draa, 2009]. D'autre part, le bruit blanc Gaussien est fréquemment employé face aux problèmes du monde réel présentant de l'incertitude.

Le modèle de PDMPO proposée en 3.9 apporte certains avantages par rapport aux travaux relatés à la section 3.2. D'abord, cette formulation considère que toutes les fonctions composant le PDMPO sont continues. Cet aspect est fondamental afin d'éviter une discrétisation du PDMPO. D'autre part, elle correspond essentiellement au modèle proposé par Ross *et al.* [2008b] où les fonctions déterministes du modèle sont aussi affectées par des bruits Gaussiens. Toutefois, nous n'ajoutons aucune hypothèse concernant l'inversibilité des parties déterministes des fonctions (tel un modèle linéaire par exemple), ce qui rend notre modèle plus général. Enfin, cette formulation où les fonctions sont déterministes et corrompues par un bruit Gaussien additif s'aligne exactement sur les hypothèses fondamentales de la régression par processus Gaussiens, permettant ainsi une utilisation aisée de ceux-ci.

Maintenant que le modèle est adapté à l'apprentissage avec processus Gaussiens, il faut définir une méthode permettant à la fois d'identifier le modèle (*exploration*) mais aussi de maximiser les récompenses à long terme (*exploitation*). Rappelons que l'apprentissage du modèle visé doit se faire en ligne à partir de séquences d'actions-observations. Par contre, il manque un élément important aux informations contenues dans les séquences, soit l'estimation des états parcourus. La méthode pour les obtenir consiste normalement à appliquer séquentiellement la règle de mise à jour (3.5). Cependant, l'application de cette dernière règle produit une séquence de distributions de probabilités sur les états visités, ce qui est problématique concernant l'apprentissage avec processus Gaussien. En effet, l'apprentissage conventionnel par processus Gaussien ne permet pas des données d'entrées sous forme de distributions de probabilités, mais uniquement des vecteurs contenant des scalaires[3].

Face à cet obstacle, une solution possible consiste à représenter l'estimation de l'état par un point afin de pouvoir appliquer l'apprentissage par processus Gaussien. Idéalement, ces points devraient être choisis astucieusement de sorte qu'ils représentent au mieux la séquence d'états réellement parcourues. Il se trouve qu'estimer cette séquence

3. Cet aspect sera corrigé via une extension des processus Gaussiens à l'incertain proposée au chapitre 4

par maximum de vraisemblance a posteriori (voir section 2.2.2) est un choix judicieux, car cette méthode d'estimation équivaut essentiellement à rechercher la séquence d'état la plus probable.

La section 3.5.1 qui suit explique en détail la routine servant à identifier le modèle du PDMPO conjointement à cette estimation de la séquence d'états la plus probable. La méthode en question est une adaptation des travaux sur l'apprentissage de modèle dynamique par processus Gaussien de Wang *et al.* [2008]. La solution que ces auteurs proposent permet l'apprentissage de certains *modèles de Markov cachés* continus à partir de séquences d'observations, un modèle simplifié omettant récompenses et actions, mais néanmoins très proche du PDMPO. Les informations apprisent sont par la suite fournies à un algorithme de planification en ligne, décrit à la section 3.5.2, qui a pour tâche de sélectionner l'action à exécuter dans l'environnement.

Afin de simplifier la lecture, nous utiliserons pour le reste du chapitre une notation matricielle où $\mathbf{S} = [\mathbf{s}_0, \mathbf{s}_1, \ldots, \mathbf{s}_N]^\top$ représente la matrice de séquence d'état, $\mathbf{A} = [\mathbf{a}_0, \mathbf{a}_1, \ldots, \mathbf{a}_{N-1}]^\top$ la matrice de séquence d'action, $\mathbf{Z} = [\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_N]^\top$ pour les observations et $\mathbf{r} = [r_1, r_2, \ldots, r_N]^\top$ le vecteur contenant les récompenses obtenues. Pour cette notation, chaque matrice est construite par rapport à la variable N, laquelle représente le nombre d'exemples d'entraînement qui sera disponible à l'apprentissage par processus Gaussien.

3.5.1 Apprentissage du modèle par processus Gaussiens

Le modèle dynamique utilisant les processus Gaussiens (MDPG) consiste d'abord en une fonction dont l'ensemble de départ est un espace latent et dont l'espace d'arrivée est celui des observations. Ce modèle d'observations, lorsqu'adapté aux PDMPO, est défini tel que $S \times A \to Z \times R$, et correspond à une jonction des fonctions d'observation et de récompense du PDMPO où l'action est complètement observable et l'état est une variable latente. Une seconde fonction permet de régir les dynamiques Markovienne du premier ordre[4] de l'espace latent. Ce modèle des dynamiques, qui originalement permet seulement de générer de simples chaînes de Markov, est maintenant défini tel que $S \times A \to S$, et correspond à la fonction de transition du PDMPO grâce à l'ajout d'une variable d'action. Formellement, nous pouvons définir les fonctions du MDPG

4. Le modèle s'étend facilement aux ordres plus élevés.

comme des combinaisons linéaires de fonctions de base non-linéaires de sorte que :

$$\mathbf{s}_t = \sum_i \mathbf{b}_i \phi_i(\mathbf{s}_{t-1}, \mathbf{a}_{t-1}) + \varepsilon_T \tag{3.10}$$

$$\mathbf{y}_t = \sum_j \mathbf{c}_j \psi_j(\mathbf{s}_t, \mathbf{a}_{t-1}) + \varepsilon_Y \tag{3.11}$$

où $\mathbf{B} = [\mathbf{b}_1, \mathbf{b}_2, \dots]$ et $\mathbf{C} = [\mathbf{c}_1, \mathbf{c}_2, \dots]$ sont les poids accordés à chacunes des fonctions de base ϕ_i et ψ_j, ε_T et ε_Y sont des bruits blancs Gaussiens. L'espace d'observation-récompense conjoint est défini tel que $Y = Z \times R$, et par conséquent, $\mathbf{y}_t$ est un vecteur d'observation $\mathbf{z}_t$ augmenté d'une récompense r_t. Cette adaptation du MDPG permet de prendre en considération les trois fonctions du modèle PDMPO sous la forme de deux fonctions. D'autre part, nous avons ajouté des variables d'action aux fonctions pour considérer le contrôle par un agent. Ainsi, cette modélisation permet d'identifier le modèle d'un PDMPO par l'application de méthodes destinées à l'apprentissage de MDPG.

Pour suivre la méthodologie Bayésienne, les paramètres inconnus d'un modèle doivent idéalement être marginalisés par l'intégration de leur distribution respective. D'abord, pour la fonction (3.11), ceci peut se faire sous forme analytique [Mackay, 2003; Neal, 1996] en appliquant une distribution Gaussienne isotropique a priori sur les colonnes de $\mathbf{C}$ et ainsi mener à la fonction de vraisemblance Gaussienne multivariée :

$$p(\mathbf{Y} \mid \mathbf{S}, \mathbf{A}, \bar{\alpha}, \mathbf{W}) = \frac{|\mathbf{W}|^N}{\sqrt{(2\pi)^{N(p+1)} |\mathbf{K}_Y|^{(p+1)}}} \exp\left(-\frac{1}{2}\mathbf{W}\mathbf{Y}^\top \mathbf{K}_Y^{-1} \mathbf{Y}\mathbf{W}\right) \tag{3.12}$$

où $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_N]^\top$ est la matrice contenant la séquence des observations et récompenses, $\mathbf{K}_Y$ est la matrice de covariance calculée à partir des hyperparamètres $\bar{\alpha} = \{\alpha_1, \alpha_2, \dots, \mathbf{W}\}$. La matrice $\mathbf{W}$ est diagonale et contient $p+1$ facteurs de mise à l'échelle permettant de tenir compte des différentes variances parmi les dimensions d'observation [5]. En utilisant une seule fonction de covariance pour les variables d'états et d'actions, les éléments de la matrice de covariance sont calculés tel que $(\mathbf{K}_Y)_{i,j} = k_Y([\mathbf{s}_i, \mathbf{a}_{i-1}], [\mathbf{s}_j, \mathbf{a}_{j-1}])$. Notons que dans le cas du modèle d'observation, l'action à considérer est celle ayant été exécutée dans l'état précédent. La fonction de covariance choisie pour cette fonction est la combinaison d'une carré-exponentielle et d'un bruit stationnaire :

$$k_Y(\mathbf{x}, \mathbf{x}') = \alpha_1 \exp\left(-\frac{\alpha_2}{2} \|\mathbf{x} - \mathbf{x}'\|^2\right) + \alpha_3^{-1} \delta_{\mathbf{x}\mathbf{x}'} \tag{3.13}$$

où nous posons $\mathbf{x}_i = [\mathbf{s}_i, \mathbf{a}_{i-1}]$ en guise de vecteur d'entrée, l'hyperparamètre α_1 représente la variance en sortie du modèle d'observations, α_2 est le facteur d'échelle de la distance (pour plus de détails, voir section 2.3.3.1) et α_3^{-1} est la variance du bruit blanc Gaussien ε_Y perturbant les observations et les récompenses.

5. Rappelons que dans la définition du PDMPO continu, p représente la dimension de l'espace d'observations.

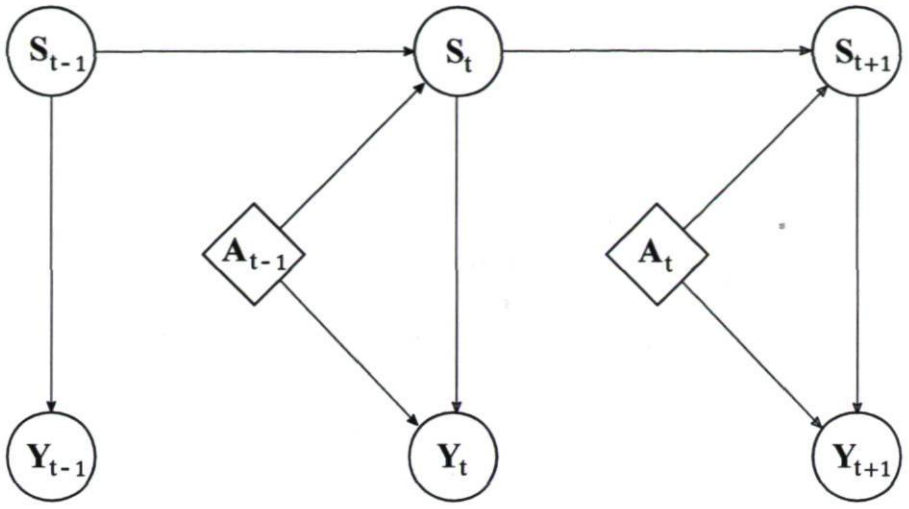


FIGURE 3.1 – Adaptation du modèle dynamique processus Gaussien incluant actions et récompenses.

En ce qui concerne la fonction (3.10) représentant les dynamiques de l'espace latent, la méthode de modélisation est similaire, mais nécessite cette fois l'application de la propriété de Markov. En définissant une distribution Gaussienne isotropique a priori sur les colonnes de $\mathbf{B}$, l'intégration peut à nouveau se faire analytiquement. Il en résulte la densité de probabilité suivante sur les séquences d'états latents :

$$p(\mathbf{S} \mid \mathbf{A}, \bar{\beta}) = \frac{p(\mathbf{s}_0)}{\sqrt{(2\pi)^{N(m+n)} \left|\mathbf{K}_X\right|^{m+n}}} \exp\left(-\frac{1}{2}\mathbf{S}_{sor}^\top \mathbf{K}_X^{-1} \mathbf{S}_{sor}\right) \tag{3.14}$$

où $p(\mathbf{s}_0) = b_0$ est la distribution sur l'état initial que l'on définit par une Gaussienne isotropique et $\mathbf{S}_{sor} = [\mathbf{s}_1, \ldots, \mathbf{s}_N]^\top$ est la matrice de variables latentes correspondant aux états non-observés. Notons que la matrice $\mathbf{S}_{sor}$, que nous désignons comme matrice des états en sortie, contient uniquement les états constituant le résultat d'une transition, d'où l'élimination de l'état initial. La matrice de covariance $\mathbf{K}_X$, dont la taille est $N \times N$, est construite à partir de $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]^\top$, où nous posons cette fois $\mathbf{x}_i = [\mathbf{s}_{i-1}, \mathbf{a}_{i-1}]$. La fonction de covariance choisie pour la modélisation des dynamiques est la somme d'une carré-exponentielle, d'une linéaire homogène et d'un bruit stationnaire :

$$k_X(\mathbf{x}, \mathbf{x}') = \beta_1 \exp\left(-\frac{\beta_2}{2} \left\|\mathbf{x} - \mathbf{x}'\right\|^2\right) + \beta_3 \mathbf{x}^\top \mathbf{x}' + \beta_4^{-1} \delta_{\mathbf{x}\mathbf{x}'} \tag{3.15}$$

où l'hyperparamètre β_3 représente la contribution de la fonction de covariance linéaire.

Ce que nous venons d'obtenir par la dérivation des équations (3.12) et (3.14) est en réalité une interprétation des fonctions composant le PDMPO comme des processus

Gaussiens. Nous avons donc fait l'hypothèse que la fonction de transition est un processus Gaussien à moyenne nulle dont la fonction de covariance est donnée par (3.15). D'autre part, la fonction jointe des observations et des récompenses est aussi un processus Gaussien dont la covariance est donnée par l'équation (3.13). Ces dernières fonctions de covariances referment en fait des produits scalaires de fonctions de bases sur ϕ et sur ψ, c'est-à-dire que le choix des fonctions de bases en (3.10) et (3.11) est implicite aux fonctions de covariances utilisées pour l'apprentissage. La subtilité qui nous permet de s'attaquer à ce problème d'apprentissage complexe est que nous restreignons considérablement les séquences d'états possibles, car autrement, il pourrait exister une infinité de modèles de transitions permettant d'expliquer une séquence d'observations.

La restriction de l'espace des modèles de PDMPO envisageable se fonde essentiellement sur les fonctions de covariance caractérisées par leur ensemble d'hyperparamètres. Comme nous considérons le cas où aucune connaissance a priori n'est disponible, tous les hyperparamètres sont par conséquent inconnus. Afin de les identifier, nous suivons la démarche de Wang *et al.* [2005a] et appliquons des distributions $p(\bar{\alpha}) \propto \prod_i \alpha_i^{-1}$ et $p(\bar{\beta}) \propto \prod_i \beta_i^{-1}$ a priori sur ces hyperparamètres, indiquant ainsi une préférence pour des valeurs moins élevées. En ce qui concerne les facteurs de mise à l'échelle $\mathbf{W}$, ils sont supposés uniformes a priori. Ainsi, nous obtenons une interprétation entièrement probabiliste des séquences d'état $\mathbf{S}$, d'observations $\mathbf{Z}$ et de récompenses $\mathbf{r}$ que peut générer un PDMPO à partir d'une séquence d'actions $\mathbf{A}$:

$$p(\mathbf{Z}, \mathbf{r}, \mathbf{S}, \bar{\alpha}, \bar{\beta}, \mathbf{W}|\mathbf{A}) = p(\mathbf{Y}|\mathbf{S}, \mathbf{A}, \bar{\alpha}, \mathbf{W}) p(\mathbf{S}|\mathbf{A}, \bar{\beta}) p(\bar{\alpha}) p(\bar{\beta}) p(\mathbf{W}) \tag{3.16}$$

Cette dernière équation forme la distribution jointe de toutes les variables appartenant à notre PDMPO. Parmi les variables observables, nous retrouvons les observations reçues $\mathbf{Z}$, les récompenses méritées $\mathbf{r}$ et les actions $\mathbf{A}$ posées par l'agent. Quant aux variables cachées, nous avons les états visités dans l'environnement $\mathbf{S}$ et les hyperparamètres $\bar{\alpha}, \bar{\beta}$ et $\mathbf{W}$.

Maintenant que nous avons un modèle probabiliste pour interpréter les données, il faut l'utiliser pour compléter l'information manquant aux calculs des distributions a posteriori sur les fonctions du PDMPO. Le principal problème vient du fait que les séquences d'états sont inconnues et qu'elles sont nécessaires pour former les ensembles d'entraînement des fonctions (3.10) et (3.11). Par conséquent, il faut conditionner les variables cachées $\mathbf{S}$, $\bar{\alpha}$, $\bar{\beta}$ et $\mathbf{W}$ en fonction de l'information observable $\mathbf{Z}$, $\mathbf{r}$, $\mathbf{A}$ pour obtenir une estimation des variables cachées. La distribution de probabilité conditionnelle recherchée correspond en fait à la distribution a posteriori :

$$p(\mathbf{S}, \bar{\alpha}, \bar{\beta}, \mathbf{W}|\mathbf{Z}, \mathbf{r}, \mathbf{A}) \propto p(\mathbf{Y}|\mathbf{S}, \mathbf{A}, \bar{\alpha}, \mathbf{W}) p(\mathbf{S}|\mathbf{A}, \bar{\beta}) p(\bar{\alpha}) p(\bar{\beta}) p(\mathbf{W}) \tag{3.17}$$

où la constante de normalisation (vraisemblance marginale) a été omise en introduisant une proportionnalité entre les dernières quantités.

Il va sans dire que le calcul de la distribution a posteriori n'est pas tractable dans le cas général. Il est malgré tout possible d'estimer cette distribution, entre autres, par des méthodes de *Monte Carlo*, mais celles-ci sont généralement assez coûteuse. Pour procéder à l'apprentissage, Wang *et al.* [2008] proposent de minimiser le logarithme de la distribution a posteriori négative $-\ln p(\mathbf{S}, \bar{\alpha}, \bar{\beta}, \mathbf{W}|\mathbf{Z}, \mathbf{r}, \mathbf{A})$ par rapport aux paramètres inconnus. Dans nôtre cas, cela revient à prendre le logarithme de la distribution de probabilité obtenue en (3.17), dont le résultat est :

$$\begin{aligned} -\ln p(\mathbf{S}, \bar{\alpha}, \bar{\beta}, \mathbf{W}|\mathbf{Z}, \mathbf{r}, \mathbf{A}) \propto & \frac{m+n}{2} \ln |\mathbf{K}_X| + \frac{1}{2} \mathrm{Tr}(\mathbf{S}_{sor}^\top \mathbf{K}_X^{-1} \mathbf{S}_{sor}) \\ & + \frac{p+1}{2} \ln |\mathbf{K}_Y| + \frac{1}{2} \mathrm{Tr}(\mathbf{W}\mathbf{Y}^\top \mathbf{K}_Y^{-1} \mathbf{Y}\mathbf{W}) - N \ln |\mathbf{W}| \\ & + \frac{1}{2} \mathbf{s}_0^\top \mathbf{s}_0 + \sum_i \ln \alpha_i + \sum_i \ln \beta_i \end{aligned} \tag{3.18}$$

où m et n sont respectivement les dimensions de l'espace d'états et l'espace d'actions et la fonction Tr dénote la trace sommant les termes sur la diagonale. Ainsi, l'estimation des fonctions de transitions, observations et récompenses d'un PDMPO à partir de séquences d'actions, d'observations et récompense revient essentiellement à minimiser la quantité définie en (3.18).

Pour ce problème d'optimisation, nous avons utilisé une méthode par *descente de gradient conjugué mis à l'échelle* (scaled conjugate gradient) Møller [1993] afin d'accélérer l'apprentissage du modèle. Le paramètre $\mathbf{W}$ est quant à lui mis à jour analytiquement. Les équations nécessaires à l'apprentissage sont fournies à l'annexe A.

Lorsque cette procédure d'optimisation est complétée, nous obtenons une estimation par maximum de vraisemblance a posteriori (MAP) des variables cachées du modèle. La particularité importante de cette approche est qu'une estimation de la séquence d'état la plus probable est maintenant disponible, et dans la circonstance, une estimation de l'état courant. Ces nouvelles informations viennent donc compléter les données initialement manquantes à l'apprentissage du PDMPO, menant ainsi aux calculs des distributions a posteriori sur les fonctions de transitions et d'observations-récompenses. Ces distributions a posteriori sont des processus Gaussiens dont les moyennes sont données par :

$$\mathbb{E}[\mathbf{s}_t|\mathcal{D}] = \mathbf{k}_X([\mathbf{s}_{t-1}, \mathbf{a}_{t-1}])^\top \mathbf{K}_X^{-1} \mathbf{S}_{sor} \tag{3.19}$$

$$\mathbb{E}[\mathbf{y}_t|\mathcal{D}] = \mathbf{k}_Y([\mathbf{s}_t, \mathbf{a}_{t-1}])^\top \mathbf{K}_Y^{-1} \mathbf{Y} \tag{3.20}$$

et dont les fonctions de covariance sont telles que :

$$\begin{aligned}&\mathrm{Cov}[\mathbf{s}_t, \mathbf{s}'_t|\mathcal{D}] \\ &\quad = k_X([\mathbf{s}_{t-1}, \mathbf{a}_{t-1}], [\mathbf{s}_{t-1}, \mathbf{a}_{t-1}]) - \mathbf{k}_X([\mathbf{s}_{t-1}, \mathbf{a}_{t-1}])^\top \mathbf{K}_X^{-1} \mathbf{k}_X([\mathbf{s}_{t-1}, \mathbf{a}_{t-1}])\end{aligned} \tag{3.21}$$

$$\begin{aligned}&\mathrm{Cov}[\mathbf{y}_t, \mathbf{y}'_t|\mathcal{D}] \\ &\quad = k_Y([\mathbf{s}_t, \mathbf{a}_{t-1}], [\mathbf{s}_t, \mathbf{a}_{t-1}]) - \mathbf{k}_Y([\mathbf{s}_t, \mathbf{a}_{t-1}])^\top \mathbf{K}_Y^{-1} \mathbf{k}_Y([\mathbf{s}_t, \mathbf{a}_{t-1}])\end{aligned} \tag{3.22}$$

où $\mathbf{k}_X$ et $\mathbf{k}_Y$ sont les vecteurs de covariance croisée, calculé à partir des équations (3.15) et (3.13), entre la donnée à prédire et les données contenues dans les ensembles d'entraînement respectifs. Pour la fonction de transition, l'ensemble d'entraînement est composé à partir de $\mathbf{S}$ et $\mathbf{A}$. Pour la fonction d'observation-récompense, l'ensemble d'entraînement est composé à partir de $\mathbf{S}$, $\mathbf{A}$ et $\mathbf{Y}$. Les covariances sont calculées en utilisant les hyperparamètres $\bar{\alpha}$ et $\mathbf{W}$ pour le modèle d'observations et $\bar{\beta}$ pour le modèle de transition. Finalement, $\mathcal{D}$ est introduit pour représenter l'ensemble de l'information rendu disponible par observation directe ainsi que par l'apprentissage.

L'approche exposée au cours de cette section montre comment apprendre le modèle d'un PDMPO continu à partir de l'information disponible en ligne. L'apprentissage du modèle en question prend essentiellement la forme d'une procédure d'optimisation où il faut minimiser la quantité définie en (3.18). Pour sa part, le calcul des distributions a posteriori sur les fonctions du modèle correspond uniquement à des calculs usuels de régression par processus Gaussiens vue au chapitre 2. Ainsi, l'agent dispose maintenant d'une estimation de chaque fonction du PDMPO qu'il peut utiliser pour prendre ces décisions dans l'environnement.

3.5.2 Planification en ligne

Tel que mentionné à la section précédente, l'agent est dorénavant en mesure, à partir de l'estimation par maximum a posteriori, de mettre à jour son état de croyance en conditionnant sur les observations reçues de l'environnement. Par cette approche, l'état de croyance de l'agent prend la forme d'un vecteur correspondant à l'état le plus probable. Il faut maintenant déterminer comment l'agent doit agir dans l'environnement à partir de cet état de croyance, et dans le contexte actuel, en se basant sur l'estimation courante du modèle.

Rappelons que pour un PDMPO dont le modèle est connu, l'*action optimale* est celle qui permet de maximiser l'espérance des récompenses escomptées telle qu'indiquée par l'équation (3.6). Comme l'espace d'actions est continu, trouver l'action optimale par rapport à un modèle est un problème d'optimisation difficile. Pour simplifier ce

problème, nous utilisons un algorithme de planification en ligne qui a pour objectif d'estimer l'action recherchée en simulant des séquences d'états, d'observations et de récompenses. En d'autres mots, nous utilisons une approche par échantillonnage de chaîne de Markov pour nous attaquer à ce problème d'optimisation.

L'approche par échantillonnage est en réalité une approche par simulation de l'environnement. Bien entendu, comme ce dernier est inconnu, il faut utiliser l'estimation du modèle pour réaliser ces simulations. Pour prédire l'évolution de l'environnement en fonction des actions possibles, l'algorithme emploie les équations (3.19) et (3.20), c'est-à-dire les fonctions de moyenne de chaque processus Gaussien respectif. Dans le même ordre d'idées que l'estimation par maximum a posteriori, les prédictions n'utilisent que l'espérance des processus Gaussiens, car l'espérance d'une distribution Gaussienne est aussi sa valeur la plus probable, et ignore l'incertitude reliée à ces estimations (équations (3.21) et (3.22)), car prendre en compte l'incertitude nécessiterait davantage d'échantillons, ce qui augmente considérablement le temps de calcul[6]. Suite aux multiples simulations, l'action immédiate générant la plus grande espérance de récompenses escomptées est choisie et exécuter, ce qui correspond en fait à l'opérateur *max* de l'équation (3.6).

Globalement, cette procédure est une estimation de la fonction de valeur sur un horizon fini. Bien entendu, des approches de planification en ligne plus performantes amélioreraient le comportement final de l'agent. Cependant, l'algorithme 1 est largement suffisant pour évaluer la qualité de la méthode d'apprentissage de PDMPO continu présentée à la section 3.5.1. Voici la procédure proposée :

Algorithm 1 PLANIFICATION(b, M, E, d)

if $d = 0$ **then Return** $0, null$
$r^* \leftarrow -\infty$
for $i = 1$ to M **do**
  Échantillonner $\mathbf{a} \in A$ uniformément
  Prédire $\mathbf{z}|b, \mathbf{a}$ et $r|b, \mathbf{a}$
  $b' \leftarrow$ OPTIMISATION$(b, \mathbf{a}, \mathbf{z}, r)$
  $r', \mathbf{a}' \leftarrow$ PLANIFICATION$(b', E, E, d-1)$
  $r \leftarrow r + \gamma r'$
  if $r > r^*$ **then** $(r^*, \mathbf{a}^*) \leftarrow (r, \mathbf{a})$
end for
Return $r^*, \mathbf{a}^*$

6. Une version employant l'incertitude a été implantée et fût écarté justement en raison du trop grand temps de calcul requis par celle-ci et qu'elle n'améliorait pas significativement les performances par rapport à la version présentée.

Lors de l'application en ligne de cet algorithme, à chaque étape de temps t, l'agent fait un appel à la fonction récursive PLANIFICATION(b_t, M, E, D), où b_t représente l'état de croyance courant, M est le nombre d'actions à échantillonner au premier niveau (sélection) de l'arbre de recherche, E est le nombre d'actions à échantillonner pour les niveaux subséquent (évaluation) et d est la profondeur de l'arbre, c'est-à-dire l'horizon de planification. Lors de l'initialisation de l'algorithme, la récompense espérée escomptée optimale r^* est initialisé à $-\infty$ de sorte que la première estimation vient remplacer cette valeur. À chaque fois qu'une nouvelle séquence est générée, sa récompense espérée escomptée est comparée avec la meilleure connue (ligne 9) et l'action optimale qui lui est rattachée est conservée si l'estimation anticipe une plus grande récompense.

La génération de séquences d'états, d'observations et de récompenses est l'essentiel de l'algorithme 1. Pour ce faire, l'agent doit d'abord fournir son état de croyance b_t, lequel est une estimation de l'état courant $\mathbf{s}_t$. Tel qu'illustré par le schéma 3.1, il faut maintenant produire une estimation de l'état de suivant $\mathbf{s}_{t+1}$, ce qui nécessite l'échantillonnage d'une action. Naturellement, il faut utiliser l'équation (3.19), qui est en fait la fonction de transition la plus probable, avec l'action obtenue et l'estimation de l'état courant pour alors produire l'estimation voulu. Une fois l'état suivant estimé, nous pouvons produire une estimation de l'observation-récompense $\mathbf{y}_{t+1}$ anticipé en utilisant l'équation (3.20), duquel nous pouvons extraire l'observation $\mathbf{z}_{t+1}$ et la récompense r_{t+1}. Jusqu'à maintenant, nous n'avons expliqué que la *prédiction* à un pas de temps dans le future. Il est bien connue qu'en environnement incertain, l'étape de *lissage* d'une séquence d'états de croyance permet de raffiner les estimations. Cette étape revient, dans le contexte actuel, à procéder à l'optimisation de l'équation (3.18) en ajoutant l'information obtenue de $\mathbf{y}_{t+1}$ et l'action simulée [7]. Le résultat est un état de croyance estimé b_{t+1} consistant avec le modèle et la séquence d'états de croyance. Finalement, il s'agit simplement de repasser l'état de croyance récursivement à l'algorithme de planification afin de construire récursivement l'arbre de recherche.

Une fois l'algorithme terminé, l'agent exécute l'action optimale estimé $\mathbf{a}_t = \mathbf{a}^*$ dans l'environnement et calcul, avec la fonction OPTIMISATION($b_t, \mathbf{a}_t, \mathbf{z}_{t+1}, r_{t+1}$), son nouvel état de croyance b_{t+1} où $\mathbf{z}_{t+1}$ et r_{t+1} sont l'observation et à la récompense réellement obtenues suite à l'exécution de l'action choisie. Pour notre part, nous avons choisi une méthode d'optimisation par gradients conjugués. Bien entendu, il existe une grande variété de méthode d'optimisation permettant de minimiser la fonction (3.18), lesquels pourraient améliorer les performances générales de l'algorithme. Toutefois, nous nous en sommes tenus aux méthodes conventionnelles par gradient conjugué.

7. Cette optimisation est en quelque sorte locale à l'algorithme, car elle n'influence pas la séquence d'états de croyance réelle de l'agent.

Tel que mentionné précédemment, l'apprentissage du PDMPO par la procédure d'optimisation retourne une estimation MAP de la séquence d'états, mais aussi les hyperparamètres les plus probables pour le modèle. Ces informations sont requises pour mettre à jour les ensembles d'entraînement destinés aux équations de prédictions (3.19) et (3.20) en vue de la prochaine étape de temps. D'autre part, une variation des hyperparamètres change la façon dont les données sont interprétées et ont donc une grande influence sur l'estimation du modèle. Lorsque l'agent doit à nouveau prendre une décision dans l'environnement, l'algorithme de planification est appelé avec le nouvel état de croyance b_{t+1} afin de sélectionner une action à exécuter et se basera sur les nouveaux ensembles d'entraînement et les nouveaux hyperparamètres.

3.6 Experimentations

Pour valider notre approche, plus particulièrement la partie concernant l'apprentissage du modèle d'un PDMPO, nous avons convenu de contrôler un dirigeable en ligne sans connaître le modèle physique de celui-ci. Nous avons choisi le dirigeable, car, comparativement aux autres appareils, il présente l'avantage d'opérer à une vitesse relativement lente et peut conserver son altitude sans nécessairement avoir à se déplacer. De plus, il est moins sujet aux erreurs de contrôle, si nous le comparons à un hélicoptère par exemple [Rottmann *et al.*, 2007]. En fait, le problème de contrôler un dirigeable a été étudié intensivement par le passé, et plus particulièrement par la communauté du contrôle. Zhang et ses collègues ont développé un contrôleur Proportionnel, Intégral et Dérivé (PID) combiné à un système de vision permettant de guider un dirigeable [Zhang et Ostrowski, 1999]. Pour leur part, Wyeth et Barron [1997] ont utilisé un contrôleur réactif, tandis que Rao *et al.* [2006] ont opté pour la logique floue. D'autres chercheurs ont par ailleurs utilisé des modèles à dynamique non linéaires afin de contrôler différentes phases du vol [Gomes et Ramos, 1998; Hygounenc *et al.*, 2004].

Toutes ces approches ont par contre fait des hypothèses quant à la connaissance a priori des dynamiques de l'environnement ou d'un modèle prédéfini. Pour notre part, nous proposons une approche ne faisant pas de telles hypothèses [8] et qui vise à apprendre directement la politique sans passer par des connaissances a priori sur la charge utile, la température, la pression atmosphérique, etc. Par rapport à ce contexte, l'approche se rapprochant le plus de celle présentée dans ce chapitre est celle proposée par Rottmann *et al.* [2007]. Les auteurs ont appliqué l'apprentissage par renforcement de type Monte Carlo sans modèle et ont utilisé les processus Gaussiens comme méthode de généralisation des connaissances sur un espace d'état-action continu. En réalité, ils ont

8. En fait, la connaissance a priori du modèle se limite à un processus Gaussien à moyenne nulle.

fait l'hypothèse que les couples état-action étaient complètement observables. Par la suite, ils ont utilisé l'estimation des états obtenus par filtrage pour calculer la *fonction de Q-valeur* via l'apprentissage par processus Gaussien. De notre côté, notre approche basée sur le modèle vise à apprendre les fonctions de transition, d'observation et de récompense par l'utilisation des processus Gaussiens et à estimer par la suite la fonction de valeur grâce à la distribution a posteriori sur le modèle. En fait, le problème du dirigeable est seulement un exemple illustratif pour soutenir l'application de l'apprentissage par renforcement Bayésien dans les PDMPO continus avec les processus Gaussiens, une idée pouvant certainement être étendue aux tâches complexes du monde réel.

Le but des expérimentations est de valider l'application de l'apprentissage avec processus Gaussien exposé à la section 3.5.1 pour des fins d'identification en ligne du modèle de PDMPO et d'estimation de l'état. L'ajout d'un algorithme de planification en ligne permet d'évaluer la qualité du modèle dans un contexte de prise de décisions par un agent. Par rapport au contexte établi, l'agent à pour objectif de maintenir au mieux l'altitude d'un dirigeable à une *hauteur nulle*, et ce, en utilisant le *minimum d'énergie* possible. Lors de chaque simulation, l'agent débute à une hauteur de 0 ainsi qu'à une vitesse nulle et dure 100 étapes de temps. Pour ce qui est de l'environnement de l'agent, les dynamiques auxquelles est soumis le dirigeable ont été simulée à partir des équations suivantes :

$$\frac{dh(t)}{dt} = \frac{M_d}{F_d}(v(t) - \frac{f_a(t)}{F_d})(1 - \exp(-\frac{M_d}{F_d}t)) + \frac{f_a(t)}{F_d}t \tag{3.23}$$

où $M_d = 0.36$ est la masse du dirigeable, $F_d = 0.06$ représente le coefficient de friction du dirigeable et $h(t)$ correspond à la hauteur au temps t. Pour sa part, $f_a(t)$ est la force générée par l'action $a(t)$ et est expliqué plus bas. Pour mettre à jour la hauteur du dirigeable, l'équation (3.23) prend en compte sa vitesse $v(t)$ est la vitesse au temps t. :

$$\frac{dv(t)}{dt} = (v(t) - \frac{f_a(t)}{F_d})\exp(-\frac{F_d}{M_d}t) + \frac{f_a(t)}{F_d} - v(t) \tag{3.24}$$

D'autre part, le contrôle de l'hélice du dirigeable permet d'appliquer une force donnée par :

$$f_a(t) = 0.098a(t)^3 + 0.017a(t)^2 + 0.063a(t) \tag{3.25}$$

où $f_a(t)$ est la force générée suite à l'exécution de l'action $a(t)$. L'ensemble d'actions disponible à l'agent est continu et défini tel que $a(t) \in A = [-1, 1]$ où les bornes représentent respectivement les poussées maximales vers le bas et vers le haut. La simulation de ces dynamiques a été réalisée avec une discrétisation du temps de 1 seconde. Les observations disponibles à l'agent sont constituées de la hauteur (m) et de la vitesse (m/s), tous deux perturbées par un bruit blanc Gaussien de moyenne nulle et d'écart-type de $1cm$. À chaque pas de temps, l'agent reçoit une récompense corrompue par le même bruit blanc Gaussien. Bien que les dynamiques du dirigeable correspondent à une fonction de transition déterministe, les variables d'état en sortie, soit la hauteur

$h(t)$ et la vitesse $v(t)$, sont perturbées par un bruit blanc Gaussien de moyenne nulle et d'écart-type de $0.5cm$. Ainsi, nous obtenons un PDMPO dont le modèle respecte les hypothèses faites à l'équation (3.9), soit que les fonctions sont déterministes et soumises à un bruit blanc Gaussien.

En vue d'apprendre le modèle et de planifier dans l'environnement, nous avons appliqué l'algorithme 1 à chaque pas de temps. Pour s'assurer d'un bon compromis entre exploration et exploitation de l'environnement, l'agent exécute une action choisie aléatoirement, plutôt que l'estimation de l'action optimale retournée par l'algorithme de planification. La probabilité de choisir une action aléatoire est décroissante au cours du temps, favorisant ainsi l'exploration en début de simulation. Formellement, une action aléatoire est sélectionnée avec une probabilité 0.9^t. L'introduction de cette exploration forcée est principalement due au fait que nous n'utilisons pas l'incertitude sur le modèle dans l'algorithme 1. Par ailleurs, les paramètres de planification ont été fixés à $M = 25$ actions échantillonnées pour le niveau de sélection, $E = 10$ échantillons pour le niveau d'évaluation de l'état, $d = 3$ pour la profondeur de l'arbre et $\gamma = 0.9$ comme facteur d'escompte. Il est préférable d'avoir un nombre plus élevé d'actions échantillonnées au premier niveau de l'arbre, afin d'offrir un plus vaste choix d'actions à l'agent. Le facteur de branchement subséquent sert uniquement à l'estimation de la valeur d'un état. En ce qui a trait à la profondeur de l'arbre, l'augmenter améliore évidemment l'estimation de l'état, mais à un coût exponentiel. Le choix d'une profondeur $d = 3$ est donc entièrement dû à ce facteur de complexité. Cependant, pour le problème de contrôle du dirigeable, cette valeur, combinée avec un facteur d'escompte $\gamma = 0.9$ ont permis d'obtenir de bons résultats.

Lors de nos expérimentations, la fonction de récompense fut cruciale en raison du cours laps de temps que l'agent disposait pour apprendre. Par conséquent, nous avons défini cette fonction par [Dallaire *et al.*, 2009b] :

$$r(t) = \frac{1}{2}\left[\exp\left(-\frac{h(t)^2}{2}\right) + \exp\left(-\frac{h(t)^2}{2(0.05)^2}\right)\right] - \frac{a(t-1)^2}{5} \tag{3.26}$$

ce qui correspond essentiellement à une récompense positive définie par deux Gaussiennes et une pénalité quadratique sur les actions appliquées. La première Gaussienne d'écart-type de $5cm$ permet de symboliser le but qui est de maintenir une altitude nulle tout en offrant au maximum un demi-point. La seconde Gaussienne, dont l'écart-type est de 1 mètre, sert à indiquer qu'il est malgré tout préférable de se trouver à quelques mètres du but et permet d'atteindre la récompense positive totale de 1 point. De plus, la pénalité quadratique est bornée à -0.2. Ce coût sur les actions est appliqué afin de forcer l'agent à utiliser une quantité d'énergie minimale. Finalement, toutes les récompenses observées sont corrompues par un bruit blanc Gaussien d'écart-type équivalent à $1cm$.

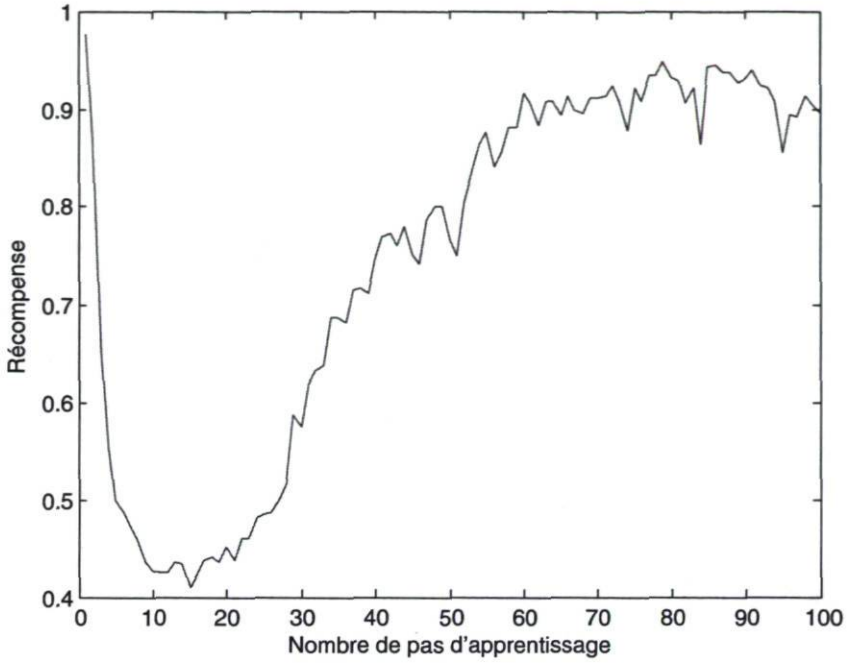


FIGURE 3.2 – Récompense moyenne reçue.

La figure 3.2 montre la récompense moyenne reçue par l'agent-dirigeable. Notons que la courbe se stabilise autour d'une récompense de 0.9. Cette valeur correspond à la récompense octroyée lorsque le dirigeable est à une distance de $5cm$ de l'altitude cible et qu'aucune action significative n'est exécutée. Sur la figure 3.3, nous observons que la distance moyenne du dirigeable par rapport à l'altitude cible se stabilise autour de $10cm$. La figure 3.4 montre l'erreur de prédiction moyenne de la séquence d'observation-récompense lorsque l'agent utilise l'équation (3.20) et l'estimation du dernier état. Nous avons défini l'erreur comme étant la somme des erreurs absolues sur la prédiction de l'observation et la récompense bruitée. La figure 3.5 montre que la majorité des trajectoires du dirigeable ont une grande variance au début de l'épisode et que celle-ci diminue au fur et à mesure que l'agent reçoit des observations. Évidemment, cette variance est en partie due à l'exploration forcée qui passe sous la barre des 5% à la 28ième étape de temps. D'autre part, chaque boîte représente les 25ième et 75ième percentiles, la marque centrale est la médiane et les moustaches englobent les données non aberrantes. Par ailleurs, une évaluation de la politique aléatoire a montré que cette stratégie diverge rapidement, et ce, jusqu'à 3 mètres de l'altitude cible.

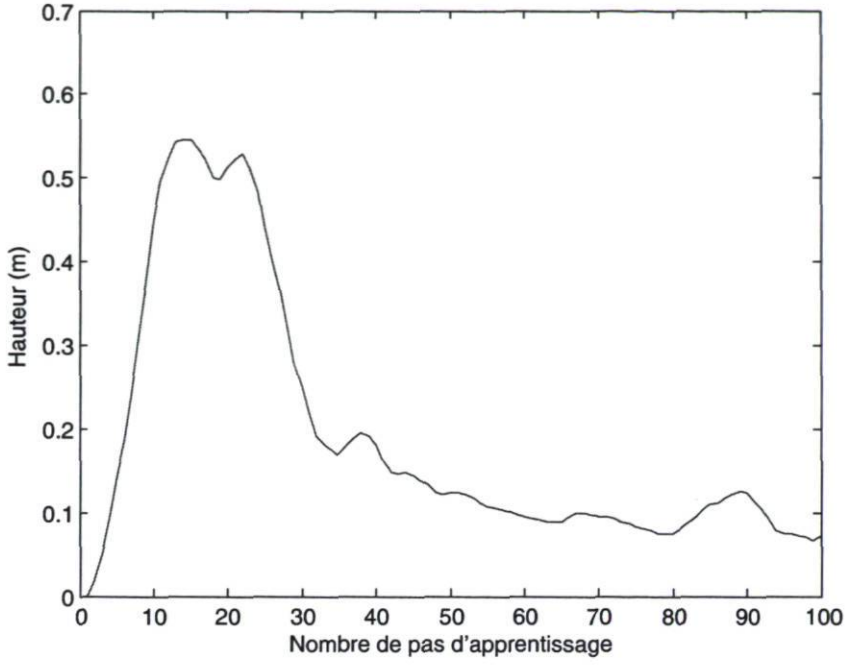


FIGURE 3.3 – Distance moyenne à la hauteur requise.

3.7 Conclusion

Prendre des décisions séquentielles dans un environnement stochastique et partiellement observable lorsque le modèle est inconnu est un problème important au niveau des applications de la vie réelle. De plus, choisir le bon ensemble de paramètres pour caractériser le modèle de cet environnement est aussi difficile. Pour résoudre ces deux problèmes, nous avons proposé une formulation continue du processus décisionnel de Markov partiellement observable où les fonctions du modèle sont supposées être des processus Gaussiens. Cette hypothèse nous permet d'utiliser les travaux sur les *modèles dynamiques utilisant les processus Gaussiens* de Wang *et al.* [2008] où des variables de contrôle et de récompense sont ajoutées pour répondre aux exigences du PDMPO. Bien qu'aucune information a priori sur l'environnement ne soit disponible, les résultats expérimentaux sur le problème du dirigeable montrent que le modèle appris fournit des prédictions utiles à la planification en ligne. Les résultats obtenus par la planification sont encourageants au regard du bruit introduit sur les transitions et les observations.

La comparaison de l'approche proposée dans ce chapitre par rapport aux approches existantes est difficile puisque notre modèle est déterminé via l'apprentissage, et ce, à partir de très peu de données. Pour comparer efficacement, plusieurs points doivent être considérés. Premièrement, il faut abaisser la complexité d'apprentissage afin de

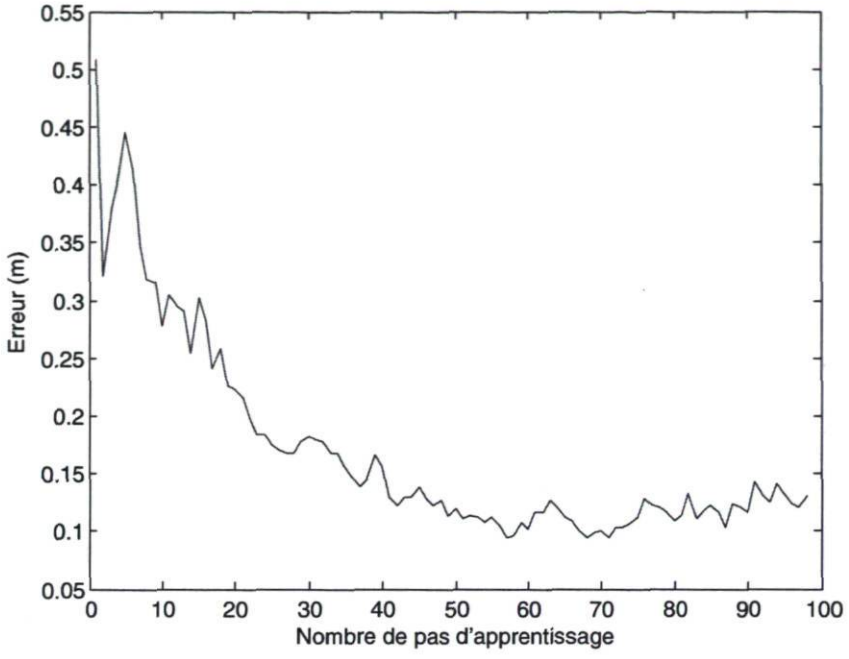


FIGURE 3.4 – Erreur moyenne de prédiction.

composer avec la grande quantité de données disponibles. Une technique d'approximation du processus Gaussien a posteriori par *matrice creuse* comme le propose Snelson [2007] permettraient d'inclure plus de données et ainsi d'améliorer les performances de prédiction. Deuxièmement, une fois le modèle de l'environnement appris, il serait préférable d'appliquer un *filtrage* de l'état plutôt qu'une approche par maximum a posteriori pour l'identification de l'état sous-jacent. Cela permettra sans aucun doute une meilleure mise à jour des paramètres de la fonction de transition, car le filtrage produit une séquence de distribution plutôt qu'une séquence de points, ce qui réduit la perte d'information. Finalement, définir des processus Gaussiens a priori plus informatifs via une fonction de moyenne non-nulle permettrait la comparaison avec des techniques de contrôle existantes qui suppose généralement une connaissance partielle du modèle. Cet ajout améliorerait très certainement les performances globales de notre approche.

Évidemment, l'approche proposée dans ce chapitre est sujette à l'amélioration. D'abord, l'algorithme de planification en ligne emploie uniquement les moyennes fournies par les processus Gaussiens. Il serait donc intéressant de considérer l'incertitude de la distribution a posteriori au cours de l'apprentissage. Entre autres, l'incertitude sur le modèle est utile pour déterminer à quel moment il est préférable d'explorer plutôt qu'exploiter les connaissances acquises. Par ailleurs, l'apprentissage par renforcement se réalise fréquemment sur un grand nombre d'expériences dans l'environnement. Dans notre cas, il faudrait permettre l'apprentissage à partir de plusieurs trajectoires, ce

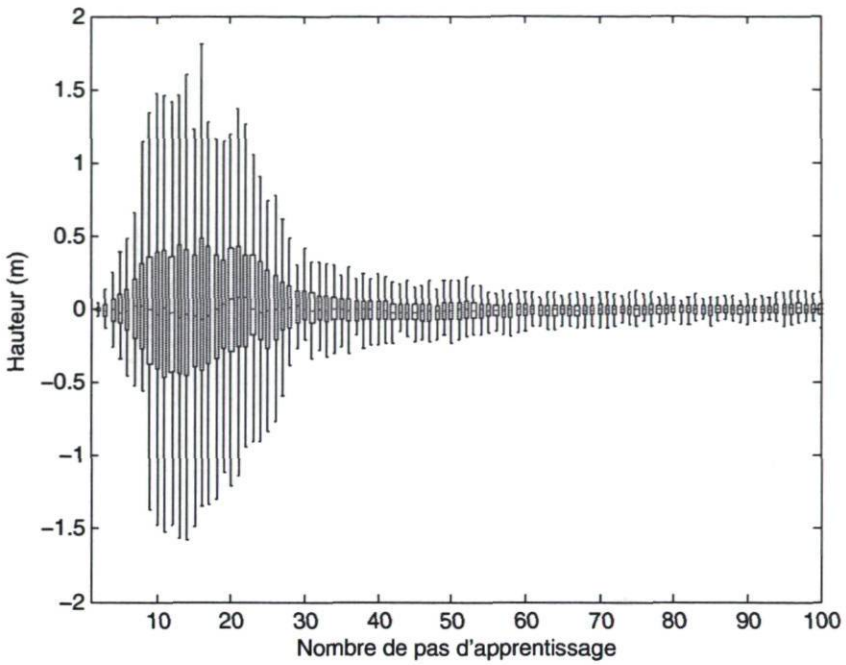


FIGURE 3.5 – Quantile à 2.5%, 25%, 50%, 75%, 97.5% de la distribution des trajectoires. Les moustaches contiennent donc 95% des données, les boîtes 50% et le trait central représente la médiane.

qui offrirait à l'agent l'avantage d'explorer davantage son environnement. L'utilisation d'un produit de fonctions de covariance, l'une sur l'*état* et l'autre sur l'*action*, serait aussi un bon point à investiguer, car ces éléments peuvent généralement être supposé indépendants par rapport à l'espace d'entrée.

Parmi les avenues de recherches proposées ci-haut, nous avons entamé celle qui semblait la plus prometteuse, soit remplacer l'estimation par maximum de vraisemblance de l'état par un filtrage. Cependant, nous faisons face à un problème lorsqu'il est question d'apprendre un modèle avec des données d'entraînement incertaines. L'incertitude dont il est question ici est exactement la distribution obtenue lors du filtrage. Les processus Gaussiens tels que présentés au chapitre 2 font l'hypothèse que les données fournies sont des points et non des distributions. Par conséquent, remplacer la méthode d'estimation de l'état nécessite au préalable une méthode d'apprentissage supervisé capable d'apprendre à partir de données incertaines. Ce problème d'apprentissage est fondamental en soi, et dépasse largement l'application pour laquelle elle est requise. Le chapitre 4 propose une solution à ce problème via une extension des processus Gaussiens.

Chapitre 4

Apprentissage de processus Gaussien avec données incertaines

La majorité des méthodes d'apprentissage supervisé sont basées sur l'hypothèse qu'uniquement les données en sortie sont incertaines. Toutefois, cette hypothèse peut s'avérer trop forte pour certains problèmes. Pour y remédier, ce chapitre expose une approche utilisant les processus Gaussiens pour l'inférence de fonction à partir d'ensemble d'exemples contenant des entrées-sorties incertaines. En supposant que les couples entrées-sorties suivent des distributions Gaussiennes de moyenne et covariance connues et en employant certaines fonctions de covariances particulières, il est possible de calculer analytiquement la matrice de covariance espérée des exemples pour obtenir une distribution a posteriori sur les fonctions. En d'autres mots, cette approche consiste essentiellement à calculer le processus Gaussien a posteriori espéré en guise d'estimation de la distribution a posteriori complète dans l'espace des fonctions.

4.1 Introduction

L'apprentissage supervisé fait référence à la catégorie de problèmes où l'objectif est d'inférer une fonction $f : X \to Y$, à partir d'exemple d'entrées-sorties. Une hypothèse courante concernant les exemples destinés à l'entraînement est que les données d'entrées sont connues exactement et que seules les variables de sortie sont sujettes aux perturbations. Cette hypothèse n'est toutefois pas réaliste pour plusieurs applications. Par exemple, certains effets incontrôlables peuvent affecter les entrées lors d'un processus d'acquisition de données. Si ce fait est ignoré dans la formulation du problème, cela

peut potentiellement résulter en une diminution de la qualité du modèle estimé. Il est donc préférable de tenir compte de l'incertitude des données en entrée lorsque cela est possible.

Le problème d'apprendre en contexte d'incertitudes autant sur les entrées que sur les sorties a été approché de plusieurs façons. En informatique et en ingénierie, on utilise généralement une méthode connue sous le nom de *moindres carrés totaux* (total least squares) qui consiste à trouver la correction minimale à appliquer sur les exemples de façon à ce que les données modifiées satisfassent une relation linéaire [Golub et Van Loan, 1980]. Pour leur part, les statisticiens ont exploré le modèle d'*erreurs dans les variables* (error-in-variables) pour faire face aux erreurs sur les variables dépendantes aussi bien que sur les variables indépendantes [Caroll *et al.*, 1995]. En général, l'approche par erreur dans les variables vise à créer un ensemble de variables cachées correspondant aux vraies observations et qui suivent la relation recherchée. Il existe un lien étroit entre les modèles d'erreurs dans les variables et les méthodes de moindres carrés totaux. Le modèle statistique qui correspond à l'approche par moindres carrés totaux de base est le modèle par erreur dans les variables restreint à la condition que les perturbations soient de moyenne nulle, indépendante et identiquement distribuée [Markovsky et Van Huffel, 2007].

Les chercheurs en *apprentissage automatique* ont également abordé le problème de l'apprentissage à partir de données incertaines. Tresp *et al.* [1994] ont par exemple proposé d'inclure ces données imparfaites à l'entraînement d'un réseau de neurones en intégrant l'incertitude sur les entrées par une estimation de la distribution de probabilité de ces entrées. En outre, ils ont montré que l'espérance de la fonction apprise sera biaisée si les entrées sont altérées par un bruit Gaussien et que les intervalles de confiance pour les prédictions seront élargis. Wright *et al.* [2000] ont présenté un cadre formel pour les réseaux de neurones Bayésiens pour lesquels ils ont inféré une régression avec des données d'entrée sans bruit via un échantillonnage *Monte Carlo par chaîne de Markov* (Markov chain Monte Carlo) des variables cachées. Plus récemment, Ting *et al.* [2006] ont proposé un modèle de régression linéaire Bayésien comportant un paramètre de précision forçant une interdépendance entre les modèles de bruits en entrée et en sortie. Leur algorithme tente par la suite de retirer le bruit des données incertaines en utilisant le principe d'*espérance-maximisation.*

Les méthodes à noyaux ont aussi été employées pour apprendre à partir d'entrées incertaines. Par exemple, pour les problèmes de classifications, Bi et Zhang [2004] ont proposé un modèle statistique qui étend les méthodes de *classification par vecteurs de support* afin de traiter l'incertitude sur les entrées. À cette fin, ils ont considéré que chaque entrée non observée est associée à une composante unique d'un *mélange de*

Gaussiennes. Par la suite, en estimant les paramètres de leur modèle de bruit Gaussien, ils furent en mesure de prendre en considération l'incertitude des entrées.

Les vertus des processus Gaussiens pour apprendre à partir d'entrées incertaines ont été exploitées par quelques chercheurs. D'abord, Girard et Murray-smith [2003] ont montré que la covariance entre les données peut être estimée en utilisant un nouveau processus Gaussien corrélé avec l'original. Ce processus utilise un développement de Taylor afin d'obtenir une fonction de covariance corrigée prenant en compte le bruit en entrée. D'autre part, Candela [2004] ont indiqué que le calcul de la vraisemblance marginale en présence de bruit sur les entrées est généralement impossible à réaliser analytiquement. Cependant, optimiser une borne inférieure de cette vraisemblance peut mener à une version corrigée des entrées bruitées tout en apprenant le modèle. Dans un autre ordre d'idées, plutôt que d'estimer la matrice de covariance, les mêmes auteurs ont tous deux avancés [Girard, 2004; Quiñonero-Candela et Roweis, 2003] qu'utiliser l'espérance de cette matrice pourrait mener à une meilleure estimation de la vraisemblance marginale.

Suite à ces différents constats, nous avons développé une méthode permettant non seulement de faire des prédictions pour des données d'entrées incertaines via les processus Gaussiens, mais aussi d'apprendre à partir d'ensemble d'entraînement incertain [Dallaire *et al.*, 2009a]. Pour y parvenir, nous avons convenu de marginaliser l'incertitude sur les entrées afin d'obtenir la matrice de covariance espérée et ainsi conserver une distribution a posteriori en forme analytique sur l'espace des fonctions. Le principal avantage de cette approche est qu'elle permet d'apprendre à partir d'entrées et sorties incertaines en employant la régression par processus Gaussien (voir chapitre 2) exactement comme dans le cas où les données sont non bruitées. Nos résultats ont montré que prendre en compte l'incertitude par cette méthode diminue l'erreur quadratique moyenne et augmente la vraisemblance de la fonction recherchée. Remarquons qu'à mesure que l'incertitude décroît, l'approche proposée tend vers la méthode sans bruit traditionnelle, ce qui nous amène à la considérer comme une généralisation de la régression par processus Gaussien classique.

4.2 Les processus Gaussiens et l'incertain

Comme nous l'avons vu au chapitre 2, les processus Gaussiens peuvent être utilisés pour décrire des distributions de probabilités directement dans l'espace des fonctions. Pour décrire une distribution, il suffit de choisir une fonction de moyenne μ et une fonction de covariance k, lesquelles définissent entièrement le processus Gaussien, et par conséquent, la distribution a priori sur les fonctions. Nous avons aussi vu qu'il est

possible d'utiliser ce type de distribution dans un processus d'inférence Bayésienne, lorsqu'un ensemble d'exemples est disponible, afin d'obtenir une estimation a posteriori de la fonction recherchée. Cependant, la méthode conventionnelle d'apprentissage par processus Gaussien fait l'hypothèse que seules les sorties sont potentiellement bruitées, et que les entrées servant d'index sont sans bruit.

Lorsque l'ensemble d'entraînement d'un processus Gaussien contient des données avec de l'incertitude autant en entrée qu'en sortie, il est nécessaire d'adapter l'apprentissage en conséquence. Supposons donc un ensemble d'entraînement $\mathcal{D} = \{(p(\mathbf{x}_i), p(y_i))\}_{i=1}^{N}$ contenant des couples de distributions de probabilités. Ces couples reflètent en fait les connaissances que nous avons sur les couples de variables cachées $(\mathbf{x}_i, y_i)$. Par conséquent, l'ensemble $\mathcal{D}$ peut être vue comme une distribution de probabilité sur *l'espace des ensembles d'entraînement*, c'est-à-dire une distribution sur les exemples réels n'ayant pas été directement observés lors de l'acquisition des données. Ce concept fait de $\mathcal{D}$ un *ensemble d'entraînement a priori* pouvant être mis à jour par inférence Bayésienne [1].

Le principal problème auquel nous faisons face en cas d'exemples incertains est qu'il est plus difficile d'appliquer la définition des processus Gaussiens. Rappelons qu'en utilisant les processus Gaussiens, nous supposons que toute paire de variables aléatoires $f(\mathbf{x})$ et $f(\mathbf{x}')$ indexés sur l'espace d'entrée par $\mathbf{x}$ et $\mathbf{x}$, a une certaine covariance donnée par la fonction $k(\mathbf{x}, \mathbf{x}')$ choisie a priori. Cependant, le calcul de cette covariance n'est plus si simple lorsque nous avons uniquement accès aux distributions $p(\mathbf{x})$ et $p(\mathbf{x}')$. Pire encore, la covariance de deux variables aléatoires fait que la covariance elle-même devient incertaine. Autrement dit, la covariance devient une variable aléatoire. Dès lors, sachant maintenant que les covariances possèdent leur propre distribution $p(k(\mathbf{x}, \mathbf{x}'))$, il faut comprendre que ces valeurs sont destinées à former la matrice de covariance d'un processus Gaussien, laquelle doit éventuellement être inversée pour produire des prédictions. Or, l'inversion d'une matrice de covariance incertaine [2] est un problème très difficile.

Afin de surmonter les obstacles qu'occasionnent les ensembles d'entraînement incertain, nous proposons d'abord de faire l'hypothèse que l'ensemble $\mathcal{D}$ contient uniquement des distributions Gaussiennes dont les paramètres sont évidemment connues. En d'autres mots, nous supposons $\forall i$ que $p(\mathbf{x}_i) = \mathcal{N}(\boldsymbol{\mu}_{x_i}, \boldsymbol{\Sigma}_{x_i})$ et $p(y_i) = \mathcal{N}(\mu_{y_i}, \sigma_{y_i})$ où les paramètres désignent respectivement la moyenne et la (co)variance des distributions Gaussiennes $\mathcal{N}$. Grâce à cette hypothèse, il nous est possible de produire une estimation

1. Comme $\mathcal{D}$ est en fait une distribution, il est possible de calculer sa distribution a posteriori suite à l'inférence du modèle des données.

2. Pour une matrice de covariance $\mathbf{K}$ suivant une distribution $p(\mathbf{K})$, l'inversion de la matrice $\mathbf{K}$ consiste à calculer la distribution $p(\mathbf{K}^{-1})$ de la matrice inverse.

du processus Gaussien a posteriori sur l'espace des fonctions. Cette estimation peut se concrétiser en calculant uniquement l'*espérance* de la distribution $p(k(\mathbf{x}, \mathbf{x}'))$ sur la covariance afin de former la matrice de covariance du processus Gaussien. L'avantage de conserver uniquement l'espérance de la covariance est qu'en faisant cela nous obtenons une matrice de covariance contenant des *scalaires*, ce qui est grandement plus simple à inverser qu'une matrice dont les éléments sont des distributions.

4.2.1 Traitement naïf de l'incertitude

Avant de poursuivre sur la méthode proposée dans ce chapitre, nous faisons d'abord état d'une approche naïve permettant l'application des processus Gaussiens usuels en contexte d'incertitude sur les entrées. Comme nous le savons, les processus Gaussiens peuvent facilement gérer l'incertitude sur les variables de sorties par l'ajout d'une fonction de covariance de bruit. Le principal problème de l'incertitude est lorsque celle-ci touche les variables d'entrées. L'application naïve des processus Gaussiens sur un ensemble d'exemples incertains consiste à réaliser l'apprentissage en n'utilisant qu'un seul représentant par donnée d'entrée, comme la moyenne ou un échantillon par exemple, et en ignorant toute autre information concernant sa distribution. Cette approximation se base sur le fait qu'un bruit Gaussien en entrée peut se transposer en sortie via la transformation de l'entrée Gaussienne par la fonction à estimer. Évidemment, la distribution du bruit transformée n'est généralement plus Gaussienne en sortie, sauf pour le cas particulier du modèle linéaire.

Cette approche naïve peut malgré tout être efficace pour certains problèmes. En effet, à mesure que l'incertitude sur une entrée diminue, certaines fonctions auront tendance à se comporter de plus en plus linéairement sur l'intervalle couvert par la Gaussienne. Ainsi, le bruit Gaussien en entrée restera pratiquement Gaussien suite à la transformation et pourra se maîtriser facilement par une simple augmentation du bruit en sortie. Malheureusement, lorsque l'incertitude est plus forte ou que les entrées ont différents degrés d'incertitudes, l'apprentissage de la fonction sera hautement dégradé. Il est donc souhaitable qu'un algorithme d'apprentissage puisse choisir automatiquement de faire confiance à un exemple sur la base de son incertitude et de n'extraire que l'information utile de cet exemple. L'une des méthodes permettant l'automatisation de cette tâche consiste à développer des fonctions de covariances particulières prenant en compte l'incertitude sur les exemples via un calcul d'espérance de la covariance.

4.2.2 L'espérance de la fonction de covariance

Le calcul de l'espérance de la covariance possède une interprétation purement Bayésienne. En effet, face à des variables incertaines ne nécessitant aucune estimation a posteriori, la méthode Bayésienne veut que celles-ci soient marginalisées afin de focaliser sur les variables intéressantes. Rappelons que la marginalisation se réalise en sommant sur tout les cas possibles de la variable à éliminer. En ce qui concerne la covariance $k(\mathbf{x}, \mathbf{x}')$, l'intégration des variables d'entrées par rapport à leur distribution $p(\mathbf{x})$ et $p(\mathbf{x}')$ permet d'obtenir la *covariance espérée* $k_{\mathbb{E}}$ donnée par :

$$k_{\mathbb{E}}(p(\mathbf{x}), p(\mathbf{x}')) = \mathbb{E}[k(\mathbf{x}, \mathbf{x}')] = \iint k(\mathbf{x}, \mathbf{x}')p(\mathbf{x})p(\mathbf{x}')d\mathbf{x}d\mathbf{x}' \tag{4.1}$$

Notons que cette dernière équation mesure la covariance espérée entre deux variables aléatoires $f(\mathbf{x})$ et $f(\mathbf{x}')$ distinctes. Lorsqu'il est question d'une seule et même variable $f(\mathbf{x})$, c'est-à-dire que nous recherchons la *variance* espérée plutôt qu'une covariance espérée, le calcul est différent, car l'espérance de la variance se mesure par rapport à une unique distribution $p(\mathbf{x})$ et est donnée par :

$$k_{\mathbb{E}}(p(\mathbf{x}), p(\mathbf{x})) = \mathbb{E}[k(\mathbf{x}, \mathbf{x})] = \int k(\mathbf{x}, \mathbf{x})p(\mathbf{x})d\mathbf{x} \tag{4.2}$$

Le précédent calcul n'est pertinent que lorsque la fonction de covariance k est non-stationnaire, car dans le cas stationnaire, $k(\mathbf{x}, \mathbf{x})$ est une constante par définition. Ainsi, pour une covariance stationnaire, la variance espérée vaut simplement $k(\mathbf{x}, \mathbf{x})$.

Grâce aux équations (4.1) et (4.2), nous avons maintenant une formulation permettant de calculer la matrice de covariance espérée d'un processus Gaussien. Par contre, les intégrales de ces équations sont intraitables dans le cas général. Toutefois, pour certaines fonctions de covariance particulière, ces équations peuvent être calculées analytiquement lorsque les entrées sont Gaussiennement distribuée. C'est notamment le cas pour la *fonction de covariance linéaire* (voir section 2.3.4.1), dont l'espérance est donnée par (voir annexe B.1) :

$$k_{\mathbb{E}lin}((\boldsymbol{\mu}_{x_i}, \boldsymbol{\Sigma}_{x_i}), (\boldsymbol{\mu}_{x_j}, \boldsymbol{\Sigma}_{x_j})) = \boldsymbol{\mu}_{x_i}^{\top}\Sigma_w\boldsymbol{\mu}_{x_j} + \sigma_b^2 + \delta_{ij}\mathrm{Tr}(\boldsymbol{\Sigma}_{x_i}) \tag{4.3}$$

où Σ_w et σ_b^2 sont les paramètres de la fonction de covariance et contrôle la pente et le biais des fonctions aléatoires. Le terme δ_{ij} désigne le delta de Kronecker[3] et sert au calcul de la variance. La fonction Tr dénote la trace et indique la sommation des termes sur la diagonale de la matrice de covariance de l'entrée.

La fonction de covariance polynomiale (voir section 2.3.4.2) est une généralisation de la fonction de covariance linéaire. Dans ces conditions, déterminer une forme analytique

3. Le delta de Kronecker vaut 0 si $i \neq j$, et 1 lorsque $i = j$.

pour l'espérance d'une telle covariance devient plus complexe dès que nous atteignons le deuxième degré. Pour le cas particulier de la *fonction de covariance quadratique*, l'espérance est donnée par (voir annexe B.2) :

$$\begin{aligned} k_{\mathbb{E}qua}((\boldsymbol{\mu}_{x_i}, \boldsymbol{\Sigma}_{x_i}), (\boldsymbol{\mu}_{x_j}, \boldsymbol{\Sigma}_{x_j})) = \operatorname{Tr}\left([\boldsymbol{\Sigma}_{x_i} + \boldsymbol{\mu}_{x_i}\boldsymbol{\mu}_{x_i}^\top][\boldsymbol{\Sigma}_{x_j} + \boldsymbol{\mu}_{x_j}\boldsymbol{\mu}_{x_j}^\top]\right)(1+\delta_{ij}) \\ + 2\sigma_b^2 \boldsymbol{\mu}_{x_i}^\top \boldsymbol{\mu}_{x_j} + \sigma_b^4 \\ + \delta_{ij}\left[\operatorname{Tr}\left(\boldsymbol{\Sigma}_{x_i} + \mathbf{u}_{x_i}\mathbf{u}_{x_i}^\top\right)^2 + 2\sigma_b^2 \operatorname{Tr}\left(\boldsymbol{\Sigma}_{x_i}\right) - 2(\mathbf{u}_{x_i}^\top \mathbf{u}_{x_i})^2\right] \end{aligned} \tag{4.4}$$

où nous avons fait l'hypothèse que $\Sigma_w = \mathbf{I}$, car en fait, cette matrice sert uniquement à transformer linéairement l'espace du produit scalaire [4]. On remarque immédiatement que cette nouvelle fonction fait une plus importante utilisation des matrices de covariance $\boldsymbol{\Sigma}_{x_i}$ et $\boldsymbol{\Sigma}_{x_j}$ par rapport à la forme linéaire obtenue à l'équation (4.3). Ainsi, déjà avec une fonction de covariance polynomiale du second degré, l'incertitude sur les entrées est sérieusement prise en considération, et ce, pareillement pour les fonctions de degrés subséquents. Cependant, à mesure que le degré de la fonction de covariance polynomiale augmente, la complexité de calcul de son espérance augmente aussi. De plus, l'emploi de la fonction de covariance polynomiale est une hypothèse relativement forte sur les fonctions admissibles. Il peut donc s'avérer fort utile de se tourner vers des fonctions de covariance plus flexibles, comme la carré-exponentielle, pour éviter ce type d'hypothèses et rendre l'apprentissage plus flexible.

Pour la *fonction de covariance carré-exponentielle* (voir section 2.3.3.1), une forme analytique pour l'espérance de la covariance est obtenue assez aisément. Le calcul de cette espérance implique des produits et intégrations de Gaussiennes dont le résultat est une forme Gaussienne particulière (voir annexe B.3) :

$$k_{\mathbb{E}CE}((\boldsymbol{\mu}_{x_i}, \boldsymbol{\Sigma}_{x_i}), (\boldsymbol{\mu}_{x_j}, \boldsymbol{\Sigma}_{x_j})) = \frac{\sigma_f^2 \exp(-(\mathbf{u}_{x_i} - \mathbf{u}_{x_j})^\top (W + \boldsymbol{\Sigma}_{x_i} + \boldsymbol{\Sigma}_{x_j})^{-1}(\mathbf{u}_{x_i} - \mathbf{u}_{x_j}))}{|I + W^{-1}(\boldsymbol{\Sigma}_{x_i} + \boldsymbol{\Sigma}_{x_j})(1-\delta_{ij})|^{\frac{1}{2}}} \tag{4.5}$$

ce qui est en réalité la fonction carré-exponentielle originale où les facteurs de mise à l'échelle sont augmentées proportionnellement à l'incertitude sur l'entrée, et le facteur d'amplitude σ_f^2 est maintenant réduit à mesure que l'incertitude augmente. Ainsi, les données distantes et incertaines tendent maintenant vers une légère covariance ; tandis que les données rapprochées et incertaines se voient réduire leur covariance. Notons que dans le cas d'une fonction de covariance stationnaire, la variance est fixe et est obtenue de la fonction de covariance originale par $k(\mathbf{0}, \mathbf{0})$. Afin d'intégrer le cas particulier de la variance à l'équation (4.5), l'inverse du delta de Kronecker [5] $(1 - \delta_{ij})$ est ajouté au

4. Réintroduire Σ_w revient à calculer $k_{\mathbb{E}quad}((\boldsymbol{\mu}_{z_i}, \boldsymbol{\Sigma}_{z_i}), (\boldsymbol{\mu}_{z_j}, \boldsymbol{\Sigma}_{z_j}))$ sur la projection U des variables originales, soit $z \sim \mathcal{N}(U\boldsymbol{\mu}_x, U\boldsymbol{\Sigma}_x U^\top)$, où $\Sigma_w = U^\top U$ est une décomposition de Cholesky.

5. L'inverse du delta de Kronecker vaut 1 si $i \neq j$, et 0 lorsque $i = j$.

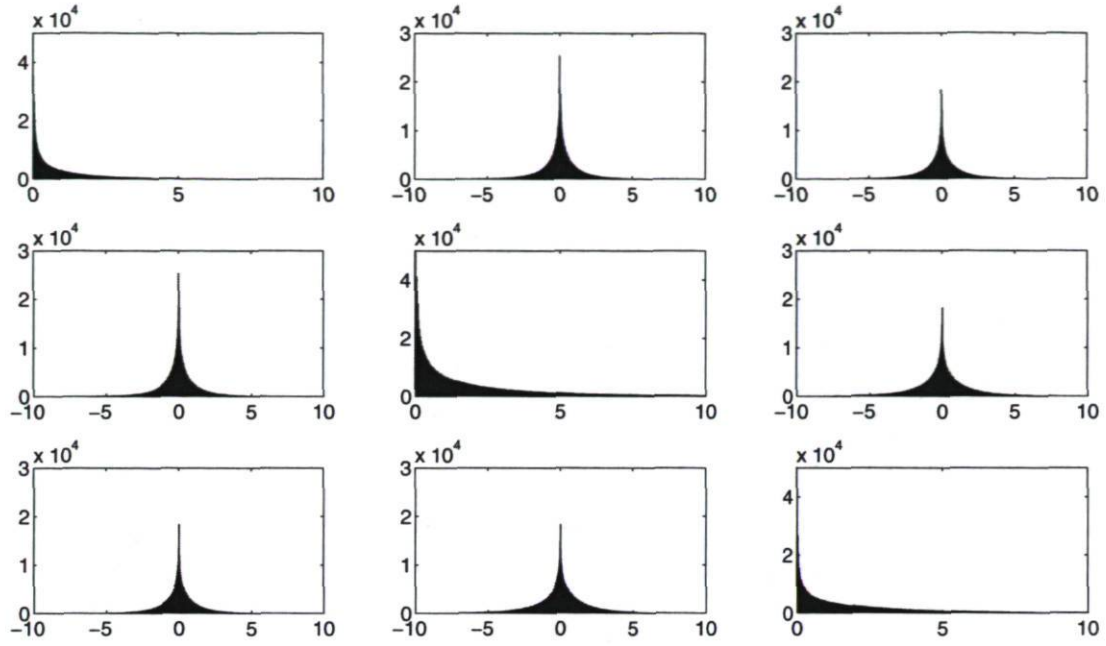


FIGURE 4.1 – Distribution de probabilité $p(\mathbf{K})$ sur la matrice de covariance $\mathbf{K}$ d'un processus Gaussien à 3 variables aléatoires. Les sous-figures sont disposées de façon à représenter une matrice où les éléments correspondent aux distributions $p(k(x_i, x_j))$. Ainsi, les abscisses contiennent les covariances potentielles et les ordonnées indiquent la vraisemblance des covariances.

dénominateur, pour contre-carré la pondération en fonction de l'incertitude et retrouver uniquement des variances égalant σ_f^2.

Comme nous l'avons mentionné au préalable, calculer l'espérance de la matrice de covariance est seulement une estimation de la distribution sur les matrices de covariance possible. Afin de bien préciser ce que signifie cette estimation et dans quel contexte elle prend place, voici un exemple simple. Supposons 3 variables aléatoires $f(x_1)$,$f(x_2)$ et $f(x_3)$ d'un processus Gaussien dont la moyenne est 0 et dont la covariance linéaire est défini par $k(x, x') = xx'$. Comme les variables d'entrées ne sont pas observées directement, nous supposerons avoir accès à des distributions Gaussiennes, soit $x_1 \sim \mathcal{N}(0, 1)$, $x_2 \sim \mathcal{N}(0, 2)$ et $x_3 \sim \mathcal{N}(1, 1)$. D'après ces distributions, la matrice de covariance $\mathbf{K}$ suit une certaine distribution $p(\mathbf{K})$, laquelle a été estimée par échantillonnage Monte Carlo pour être visualisée et illustrée à la figure 4.1. On remarque que les distributions sont unimodales et que l'espérance constitue une bonne estimation de ces distributions, ce qui est aussi le cas pour la grande majorité des covariances.

Malheureusement, il existe certains cas où estimer une distribution par l'espérance

de sa covariance n'est pas la bonne solution. Par exemple, si nous reprenons les distributions sur les entrées x_1 et x_3 ainsi qu'une fonction de covariance carré-exponentielle dont les paramètres sont tous à 1, nous obtenons la distribution affichée à la figure 4.2. Dans ce cas précis, l'espérance de la covariance vaut 0.5 et correspond effectivement à la valeur la moins vraisemblable de la distribution. Une meilleure estimation de celle-ci serait d'indiquer que les données ont une grande covariance, soit la pointe de vraisemblance de droite, ou une covariance nulle, celle de gauche. Cette situation montre qu'en théorie, l'utilisation de la matrice de covariance espérée peut potentiellement mener à une mauvaise estimation de la distribution a posteriori sur les fonctions.

Une solution à ce problème consiste à reviser les distributions des entrées contenues dans l'ensemble d'entraînement. Idéalement, cette mise à jour de la distribution a priori définie par $\mathcal{D}$ devrait être réalisée par un processus d'inférence Bayésienne afin de produire une distribution a posteriori sur l'ensemble d'entraînement. Malheureusement, cette procédure n'est pas tractable et doit donc être approximée.

Une méthode d'approximation intéressante fait usage de l'inégalité de Jensen pour obtenir une borne inférieure sur la vraisemblance marginale des données Ghahramani et Roweis [1999]. Cette borne permet d'obtenir une distribution a posteriori sur les données d'entraînement en utilisant une procédure d'optimisation. La quantité à maximiser s'obtient alors comme suit :

$$\begin{aligned} \log p(\mathbf{y}|\boldsymbol{\theta}) &= \log \int q(X) \frac{p(\mathbf{y}|\boldsymbol{\theta})p(X)}{q(X)} dX && (4.6) \\ &\geqslant \int q(X) \log \frac{p(\mathbf{y}|\boldsymbol{\theta})p(X)}{q(X)} dX && (4.7) \\ &= \int q(X) \log p(\mathbf{y}|\boldsymbol{\theta}) dX - D_{KL}(q(X)||p(X)) && (4.8) \end{aligned}$$

où la distribution $p(X)$ est la distribution a priori définie par $\mathcal{D}$, la distribution $q(X)$ représente la distribution a posteriori Gaussienne à estimer et $\boldsymbol{\theta}$ contient les hyperpamètres. L'égalité (4.6) vise simplement à introduire les termes servant aux calcules, où $p(X)$ est introduit et marginalisé de l'équation. Ensuite, nous pouvons appliquer l'inégalité de Jensen servant à obtenir la borne inférieure, ce qui donne l'inégalité (4.7). Une réécriture de cette dernière équation permet d'obtenir une formulation s'interprétant en terme connu. Le premier terme de cette équation peut être *estimé* via la log-vraisemblance marginale d'un processus Gaussien utilisant la matrice de covariance espérée par rapport à la distribution $q(X)$, c'est-à-dire l'approche proposée dans ce chapitre. Le second terme correspond à la divergence de Kullback-Leibler de la distribution a posteriori $q(X)$ [6] par rapport au prior $p(X)$. L'apprentissage par optimisation

6. Cette distribution a posteriori est en fait une distribution a posteriori "optimale", car elle est déterminée par optimisation dans un espace de distribution.

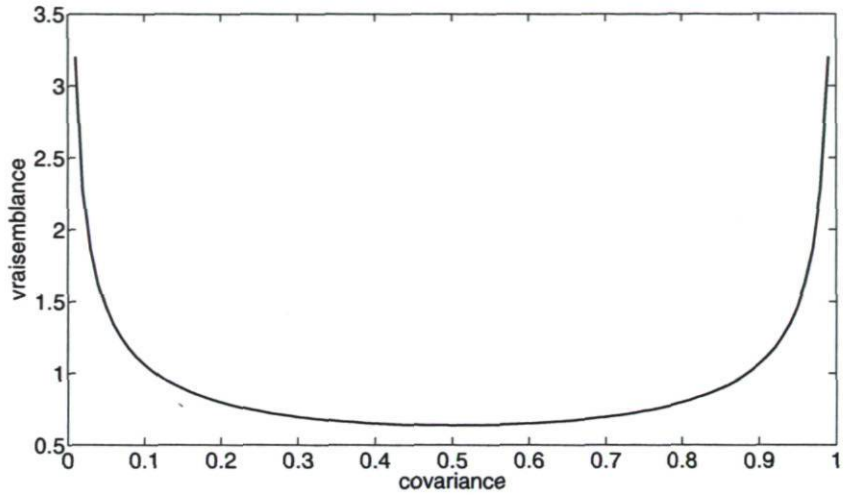


FIGURE 4.2 – Distribution de probabilité de la covariance entre des examples à $x_i \sim \mathcal{N}(0,1)$ et $x_j \sim \mathcal{N}(1,1)$ sous une fonction de covariance carré-exponentielle dont tout les paramètres sont fixés à 1. L'espérance de la covariance donnée par $k_{\mathbb{E}CE}$ pour ces exemples est de 0.5.

de l'équation (4.8) revient en fait à déterminer un déplacement des Gaussiennes en entrées permettant d'améliorer la vraisemblance des données de sorties. L'avantage de cette approche est d'une part une revision des données d'entraînement, mais aussi une amélioration des estimations des covariances permettant de corriger les mauvaises estimations par espérance. Par ailleurs, il se trouve que sans optimisation, cette méthode est exactement l'approche par espérance de covariance. Ajouter la partie optimisation offre donc la possibilité de modifier l'ensemble d'entraînement, mais à un certain coût déterminer par la divergence de Kullback-Leibler, ce qui constitue une approche d'apprentissage raisonnable.

4.2.3 L'espérance de la fonction de moyenne

Évidemment, l'incertitude sur l'entrée ne doit pas uniquement être prise en compte par la fonction de covariance. La fonction de moyenne doit aussi refléter ce fait pour former une version adaptée de la distribution jointe (2.7). Il faut donc calculer l'espérance de chaque élément du vecteur de moyenne $\boldsymbol{\mu}$ du processus Gaussien. En général, cette valeur est obtenue par intégration de l'incertitude :

$$\mathbb{E}[\mu(\mathbf{x})] = \int \mu(\mathbf{x}) p(\mathbf{x}) d\mathbf{x} \tag{4.9}$$

Pour le cas particulier où les entrées sont Gaussiennement distribuée, ce calcul est simplifié, mais reste malgré tout un problème en soi. Par ailleurs, la fonction de moyenne des processus Gaussiens est fréquemment supposée égal à zéro, c'est-à-dire $\mu(\mathbf{x}) = 0$, ce qui constitue une hypothèse faible[7] permettant d'éliminer le problème directement à la source.

4.2.4 L'incertitude en sortie

Jusqu'à maintenant, nous n'avons traité que du problème de l'incertitude sur les variables d'entrées. Les processus Gaussiens, de par leur définition purement probabiliste, permettent de gérer facilement l'incertitude Gaussienne sur les variables de sorties. Comme nous l'avons vue au chapitre 2, l'apprentissage par processus Gaussien passe essentiellement par la définition d'une fonction de covariance appropriée. Tenir compte de l'incertitude en sortie les exemples d'entraînement ne fait pas défaut à l'approche usuelle.

Afin de simplifier l'explication de l'approche permettant l'utilisation de l'incertitude sur les sorties, nous considérerons ici que les entrées des exemples sont exactes. L'objectif est donc d'identifier une f à partir d'un ensemble d'entraînement contenant des entrées sans bruit et des sorties incertaines. Comme l'incertitude sur les variables de sortie est supposée Gaussienne, l'information sur les valeurs réelles $f(x_i)$ de la fonction à identifier, c'est-à-dire les variables de sortie cachées, s'exprime par :

$$f(x_i) \sim \mathcal{N}(\mu_{y_i}, \sigma^2_{y_i}) \tag{4.10}$$

où μ_{y_i} est l'espérance et $\sigma^2_{y_i}$ la variance de la sortie incertaine que l'on identifie par y_i. Cette distribution, plus particulièrement la variance, reflète en fait le niveau d'incertitude reliée à l'exemple.

L'une des particularités intéressantes de la distribution Gaussienne est que la variable cachée et la moyenne sont interchangeables. Dès lors, appliquer cette propriété à l'équation (4.10) permet d'obtenir

$$\mu_{y_i} \sim \mathcal{N}(f(x_i), \sigma^2_{y_i}) \tag{4.11}$$

ce qui revient à dire que μ_{y_i} est une donnée bruitée issue de la fonction recherchée perturbée par un bruit Gaussien, ce qui est exactement la formulation traitée au chapitre

7. En soustrayant le biais des données d'entraînement, nous pouvons appliquer l'apprentissage sur ces données et ajouter le biais estimé lors des prédictions.

2. En fait, l'ensemble des exemples décrit par les couples (x_i, μ_{y_i}) est un processus Gaussien (fini) de distribution jointe

$$\begin{bmatrix} \mu_{y_1} \\ \mu_{y_2} \\ \vdots \\ \mu_{y_N} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} f(x_1) \\ f(x_2) \\ \vdots \\ f(x_N) \end{bmatrix}, \begin{bmatrix} \sigma_{y_1}^2 & 0 & \dots & 0 \\ 0 & \sigma_{y_2}^2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \sigma_{y_N}^2 \end{bmatrix} \right) \tag{4.12}$$

où la matrice de covariance est diagonale et N est le nombre d'exemples contenues dans l'ensemble d'entraînement. Il faut donc voir que le processus Gaussien défini en (4.12) est simplement un processus de bruit tel que vu au chapitre 2. L'unique différence est qu'ici, nous avons affaire à un processus Gaussien de bruit non-stationnaire défini seulement pour les exemples d'entraînement, alors que nous n'avons couvert que les processus de bruit à variance fixe, donc stationnaire[8].

Pour compléter l'explication de l'apprentissage avec données de sorties incertaines, supposons qu'un processus Gaussien a priori de moyenne nulle et de covariance $k_f(x, x')$ est choisi pour l'identification de la fonction $f(x)$. Comme les sorties observées μ_{y_i} ont été perturbées par le processus (4.12), la distribution a priori des variables aléatoires μ_{y_i} doit aussi comprendre l'incertitude fournit par l'ensemble d'entraînement. Ainsi, la distribution jointe des données d'entraînement est donnée par une distribution de probabilité Gaussienne de moyenne nulle et dont la fonction de covariance est définie telle que $k_y(x_i, x_j) = k_f(x_i, x_j) + \delta_{ij}\sigma_{y_i}^2$. Par conséquent, tenir compte de l'incertitude en sortie revient simplement à additionner les variances supplémentaires fournies par l'ensemble d'entraînement à la matrice de covariance du processus à identifier. Ce résultat découle du fait qu'une somme de processus Gaussiens reste un processus Gaussien dont les fonctions moyenne et de covariance ont été additionnées, soit le principe de décomposition de processus utilisé pour la prédiction (voir section 2.4.1).

4.2.5 Apprentissage des hyperparamètres

Théoriquement, en utilisant l'espérance de la fonction de covariance, apprendre avec des données incertaines est aussi simple que dans le cas sans bruit. L'apprentissage des hyperparamètres de la fonction de covariance ne fait pas défaut à cette propriété. Pour ce faire, il suffit d'appliquer la méthode d'apprentissage usuelle (voir section 2.5) en formant la matrice de covariance espérée. Par contre, la log-vraisemblance du processus

8. Rappelons qu'un processus de bruit stationnaire perturbe les sorties avec la même variable peut importe la valeur x, alors qu'un processus de bruit non-stationnaire affecte les sorties de façon arbitraire en fonction de x.

Gaussien est maintenant assujetti à plusieurs maxima locaux, lesquels consistent en des explications différentes des données. Dès lors qu'il faut traiter des données incertaines, il est souhaitable d'appliquer des distributions a priori sur les hyperparamètres afin de prévenir les résultats aberrants[9]. D'autre part, certaines distributions auront tendance à rendre la fonction de vraisemblance plus lisse, ce qui facilite l'optimisation.

4.3 Experimentations

Les expérimentations comparent les performances d'un processus Gaussien prenant en compte l'incertitude via le calcul de la matrice de covariance espérée (PG-incertain) et un processus Gaussien utilisant uniquement la moyenne des entrées (PG-naïf). L'objectif de ces expérimentations vise à déterminer si l'utilisation de la matrice de covariance espérée mène à une amélioration des performances par rapport à une utilisation naïve de la méthode traditionnelle.

Évidemment, la régression par processus Gaussiens classique ne permet généralement pas de traiter les entrées incertaines. Par conséquent, le comportement attendu de cette méthode serait qu'elle explique l'incertitude sur les variables d'entrées par un bruit Gaussien supplémentaire sur les variables de sorties. Généralement, ce bruit supplémentaire en sortie n'est pas Gaussien, mais il peut malgré tout être estimé par une telle forme de bruit.

Dans ce qui suit, nous allons d'abord évaluer le comportement de chacune des méthodes sur un problème artificiel à une seule dimension et par la suite, sur un problème plus difficile consistant à apprendre les dynamiques du système non-linéaire qu'est le pendule inversé.

4.3.1 Le problème artificiel sincsig

Afin de bien visualiser les différents comportements de chacune des méthodes d'apprentissage, nous avons choisi un problème à une seule dimension consistant à apprendre une fonction composée d'une part de la fonction *sinus cardinale* et d'autre part, d'une

9. Face à l'incertitude de l'ensemble d'entraînement, une explication simple consiste à expliquer les données, non pas par une fonction, mais uniquement par du bruit.

sigmoïde de sorte que

$$f(x) = \begin{cases} \text{sinc}(x) & \text{si } x \geqslant 0 \\ 0.5\,[1 + \exp(-10x - 5)]^{-1} + 0.5 & \text{sinon} \end{cases} \tag{4.13}$$

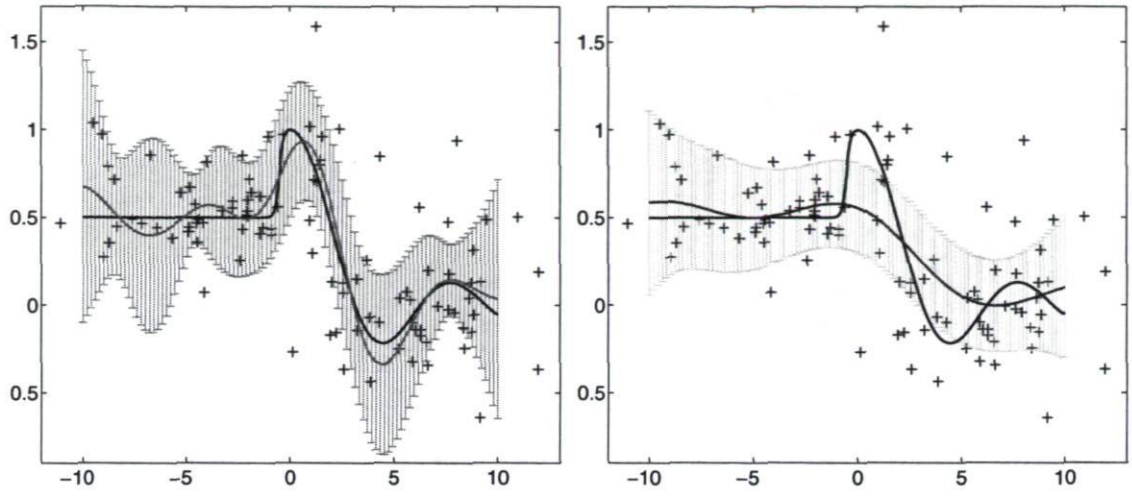
auquel nous nous référerons par l'appellation *sincsig* par la suite.

Les évaluations ont été réalisées à partir d'ensemble d'entraînement tiré au hasard. Chaque ensemble de données est construit en échantillonnant d'abord N points dans l'intervalle $[-10, 10]$. Ces points sont les entrées sans bruit dénoté par $\{\mathbf{x}_i\}_{i=1}^N$. Pour obtenir les sorties bruitées qui leur sont associées, chaque sortie est échantillonnée selon la distribution $y_i \sim \mathcal{N}(\text{sincsig}(x_i), \sigma_y^2)$. L'ensemble des données d'entrées incertaines est construit en échantillonnant d'abord la variance $\sigma_{x_i}^2$ qui sera appliquée à une entrée particulière. Ensuite, cette variance échantillonnée est utilisée pour corrompre l'entrée x_i tel que $u_i \sim \mathcal{N}(x_i, \sigma_{x_i}^2)$. Il est facile de voir que $x_i|u_i, \sigma_{x_i}^2 \sim \mathcal{N}(u_i, \sigma_{x_i}^2)$, ce qui consiste à intervertir la moyenne et la variable d'une Gaussienne. Finalement, l'ensemble d'entraînement complet est défini tel que $\mathcal{D} = \{(u_i, \sigma_{x_i}^2), y_i\}_{i=1}^N$.

La figure 4.3 montre un exemple typique d'ensemble d'entraînement aléatoire (les croix), ainsi que la fonction à inférer (en trait gras) et le résultat des régressions (en trait mince) pour PG-incertain (haut) et PG-naïf (bas). Les barres d'erreur autour des lignes minces indiquent la confiance de deux écarts-types que le processus Gaussien a envers ses prédictions. Notons que lorsque la fonction en trait plein s'écarte de la région couverte par les barres d'erreurs, tel qu'observé à la figure du bas, la fonction sous-jacente recherchée devient invraisemblable par rapport aux prédictions faites, ce qui peut être considéré comme un apprentissage inconsistant.

La première expérimentation a été conçue afin d'évaluer les performances d'apprentissage face à des entrées incertaines uniquement. Par conséquent, nous l'avons menée avec un bruit en sortie d'écart-type $\sigma_y = 0.01$ et pour différentes tailles d'ensemble d'entraînement. Les variances sur le bruit d'entrée σ_{x_i} ont été tirées aléatoirement d'une loi uniforme sur l'intervalle $[0, 2.5]$. Les écarts types ont été choisis assez hauts de sorte que le bruit sur les entrées puisse s'expliquer en ajoutant un bruit indépendant sur la sortie au cours de la procédure d'optimisation et pour mesurer l'impact des exemples hautement incertains sur le processus Gaussien a posteriori.

Toutes les comparaisons de PG-naïf et de PG-incertain ont été réalisées en les entraînant avec les mêmes ensembles de données. Pour des fins de comparaison, nous avons aussi entraîné un perceptron multicouche composé de 2 couches cachées dotées de 5 neurones chacune. Les poids du réseau de neurones ont été déterminés avec l'algorithme de Levenberg-Marquardt [Gill *et al.*, 1981]. La figure 4.4(a) montre, pour différentes tailles,

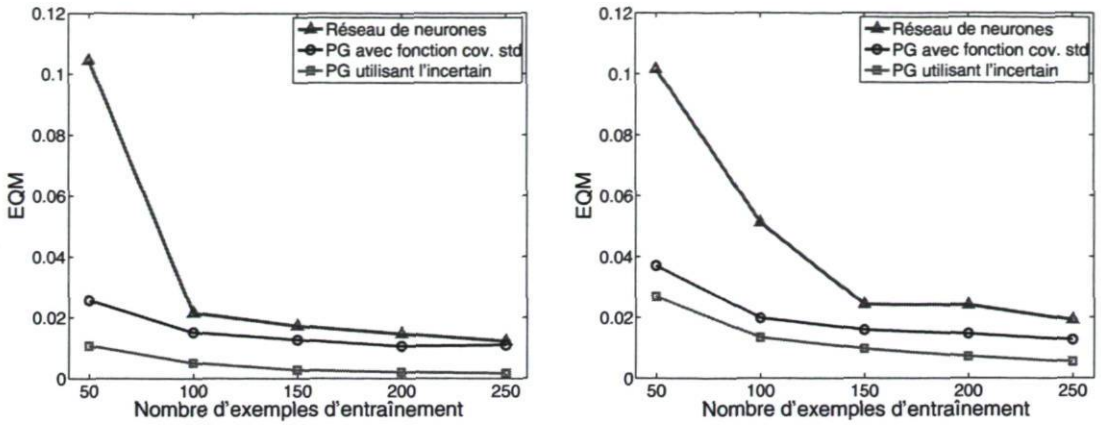


(a) PG-incertain (b) PG-naïf

FIGURE 4.3 – La fonction sincsig avec les apprentissages typiques d'un PG-incertain et d'un PG-naïf. Les traits fins sont les fonctions de moyenne a posteriori entourées de leurs barres d'erreur donnée par la variance a posteriori. Les croix sont (uniquement) les moyennes des données d'entraînement.

la moyenne de l'erreur quadratique moyenne calculée sur 25 ensembles d'entraînement. Les résultats montrent que lorsque très peu de données sont disponibles, les deux processus ont tendance à expliquer les données comme un bruit blanc de grande variance. Il convient également de remarquer que lorsque la taille des ensembles d'entraînement augmente, le PG-naïf donne une plus grande importance au fait que les données observées sont principalement du bruit. De l'autre côté, le PG-incertain discrimine les données les plus incertaines et privilégie celles étant plus certaines, ce qui revient à inférer la fonction à partir de données plus fiables.

Pour la seconde expérimentation, nous faisons l'hypothèse que les processus Gaussiens connaissent la variance du bruit sur les observations afin d'évaluer les performances sur des exemples entièrement incertains. Par conséquent, l'hyperparamètre de bruit σ_ε^2 est fixé à zéro, car le processus sait exactement quelle matrice de bruit ajouter lors du calcul de la matrice de covariance. Pour chaque sortie, l'écart-type σ_{y_i} est tiré aléatoirement par une loi uniforme sur l'intervalle $[0, 0.5]$, ce qui donne l'ensemble d'entraînement complet défini par $\mathcal{D} = \{(u_i, \sigma_{x_i}^2), (z_i, \sigma_{y_i}^2)\}_{i=1}^N$. La figure 4.4(b) montre les performances du réseau de neurones, du PG-naïf et du PG-incertain pour ce type d'ensemble d'entraînement. Retirer le processus de bruit indépendant des fonctions de covariance des processus Gaussiens a deux effets : d'abord, cela empêche le PG-naïf d'expliquer l'incertitude en entrée par une augmentation du bruit sur les sorties, et



(a) Erreur quadratique moyenne avec $\sigma_y = 0.1$

(b) Erreur quadratique moyenne avec incertitude variable $\sigma_{y_i} \sim \mathcal{U}(0, 0.5)$

FIGURE 4.4 – Résultats pour le problème d'apprentissage de la fonction sincsig. En trait rouge avec des carrés (PG-incertain), trait bleu avec des cercles (PG-naïf) et trait vert avec des triangles (Réseau de neurones).

ensuite, cela force le PG-incertain à expliquer les données avec l'information qui lui est disponible concernant les variances en entrée.

La troisième expérimentation est mise en place pour mesurer l'amélioration qu'amène la méthode par matrice de covariance espérée. Nous avons donc laissé la procédure d'optimisation des processus Gaussiens choisir les bonnes fonctions de covariance qu'il convient d'utiliser afin de traiter l'incertitude des entrées. Conséquemment, nous avons construit deux combinaisons de fonctions de covariance. La première est basée sur la fonction de covariance carré-exponentielle additionnée d'un processus de bruit non-stationnaire connu correspondant aux variances des sorties et d'un processus de bruit indépendant :

$$k(x_i, x_j) = k_{CE}(x_i, x_j) + \delta_{ij}\sigma_{y_i}^2 + \delta_{ij}\sigma_\varepsilon^2 \tag{4.14}$$

où les deux premiers termes sont supposés expliquer les données et, le dernier, de compenser pour l'erreur due au bruit d'entrée.

La seconde combinaison est basée sur la fonction de covariance carré-exponentielle espérée additionnée du processus de bruit non-stationnaire correspondant aux variances

des sorties et d'un processus de bruit indépendant :

$$k((u_i, \sigma_{x_i}^2), (u_j, \sigma_{x_j}^2)) = k_{ECE}((u_i, \sigma_{x_i}^2), (u_j, \sigma_{x_j}^2)) + k_{CE}(u_i, u_j) + \delta_{ij}\sigma_{y_i}^2 + \delta_{ij}\sigma_\varepsilon^2 \quad (4.15)$$

où il est prévu que la procédure d'optimisation équilibre l'effet de chaque fonction de covariance

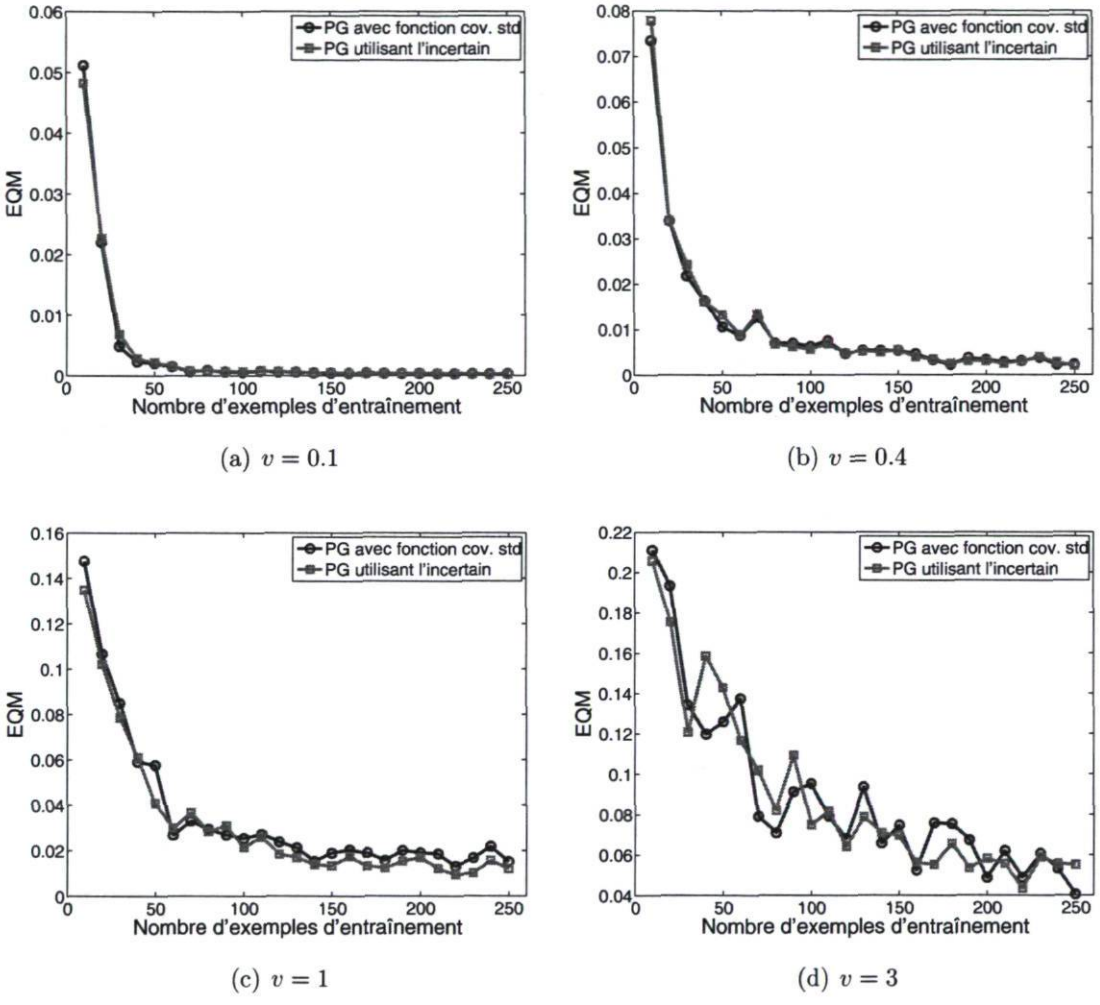
Les évaluations ont été menées avec différents niveaux de bruit et plusieurs tailles d'ensembles d'entraînement. Pour ces expérimentations, l'incertitude sur les données est bornée par v. Similairement à l'expérimentation précédente, les variances de bruit ont été tirées d'une loi uniforme et ensuite appliquées aux données. Dans le cas présent, nous avons que $\sigma_{x_i} \sim \mathcal{U}(0, v)$ et $\sigma_{y_i} \sim \mathcal{U}(0, v)$. Les performances de PG-naïf et de PG-incertain ont été mesurées avec l'erreur quadratique moyenne ainsi que la log-vraisemblance. Les résultats sont des moyennes sur 10 essais pour une borne v couplée à une taille d'ensemble. La vraisemblance est estimée à partir de 100 points équidistants tirés de la vraie fonction. Les résultats pour la moyenne de l'erreur quadratique moyenne sont illustrés à la figure 4.5 et la moyenne des log-vraisemblances à la figure 4.6 [10].

Tel que mentionné à la section 4.2.5, il est possible d'apprendre les hyperparamètres du processus Gaussien à partir d'un ensemble d'entraînement incertain. Comme les méthodes basées sur le gradient conjugué n'offrent pas de bonnes performances sur l'optimisation de la log-vraisemblance du PG-incertain, nous avons opté pour une optimisation stochastique. Pour ces expérimentations, nous n'avons pas utilisé de distribution a priori sur les hyperparamètres afin d'éviter d'aider la procédure d'apprentissage en introduisant de l'information supplémentaire concernant la fonction.

4.3.2 Le problème du pendule inversé

Nous considérons maintenant la problématique plus complexe qu'est l'apprentissage des dynamiques d'un système à pendule inversé ; système que nous avons expérimenté pour valider notre approche. La figure 4.7 illustre rudimentairement le système dont nous voulons apprendre les dynamiques. L'état de ce système est décrit par la position (φ) du chariot, la vitesse de celui-ci ($\dot{\varphi}$), l'angle du pendule (α) ainsi que la vitesse angulaire ($\dot{\alpha}$) du pendule. Un contrôle est aussi utilisé afin d'appliquer des forces latérales sur le chariot. En suivant les équations de Florian [2007] qui régissent les dynamiques

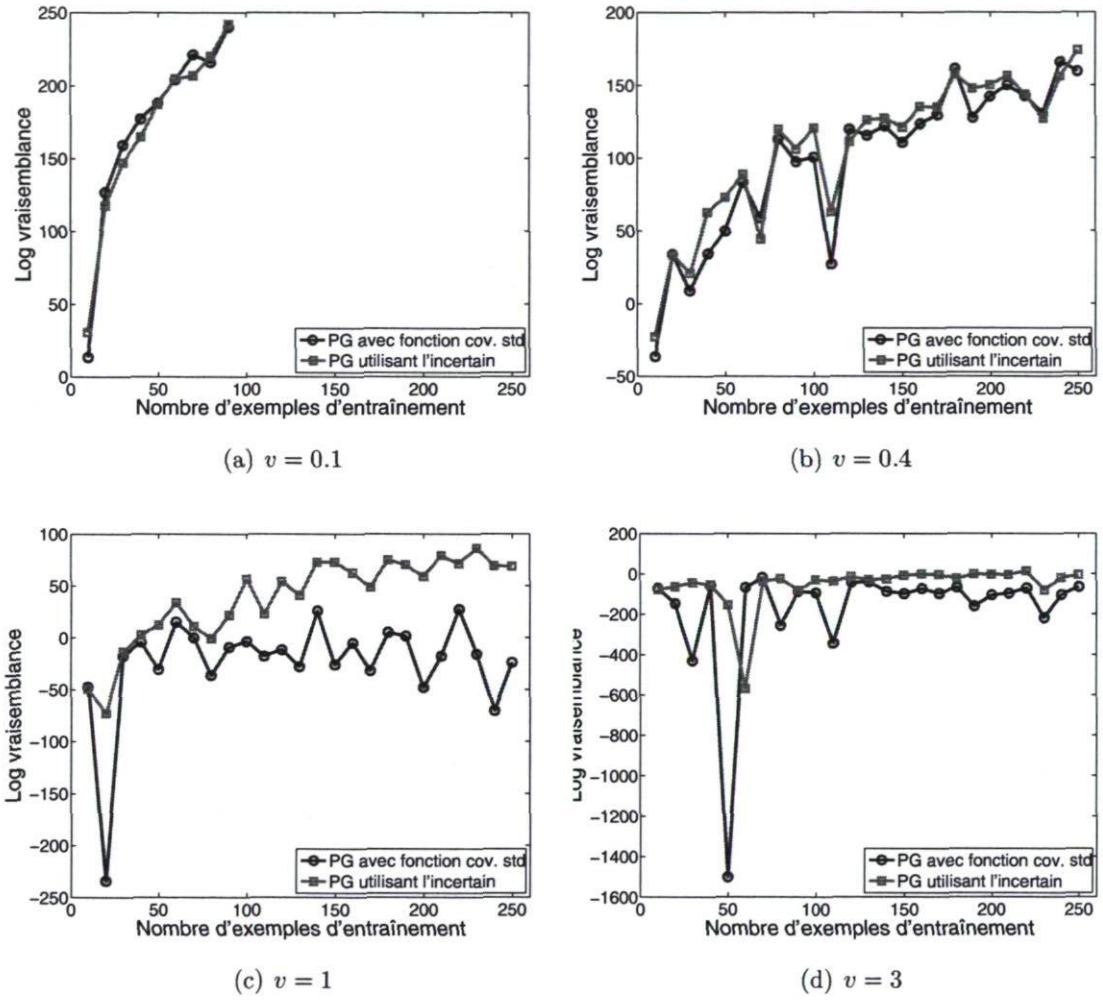
10. Pour des raisons de précision numérique, la vraisemblance de la figure 4.6(a) atteint l'infini à partir de 100, ce qui rend le graphique incomplet.



(a) $v = 0.1$ (b) $v = 0.4$

(c) $v = 1$ (d) $v = 3$

FIGURE 4.5 – Résultats de l'erreur quadratique moyenne. $\sigma_{x_i} \sim \mathcal{U}(0, v)$ et $\sigma_{y_i} \sim \mathcal{U}(0, v)$. En trait rouge avec des carrés (PG-incertain) et en trait bleu avec des cercles (PG-naïf).



(a) $v = 0.1$

(b) $v = 0.4$

(c) $v = 1$

(d) $v = 3$

FIGURE 4.6 – Résultats des log vraisemblances. $\sigma_{x_i} \sim \mathcal{U}(0, v)$ et $\sigma_{y_i} \sim \mathcal{U}(0, v)$. En trait rouge avec des carrés (PG-incertain) et trait bleu avec des cercles (PG-naïf).

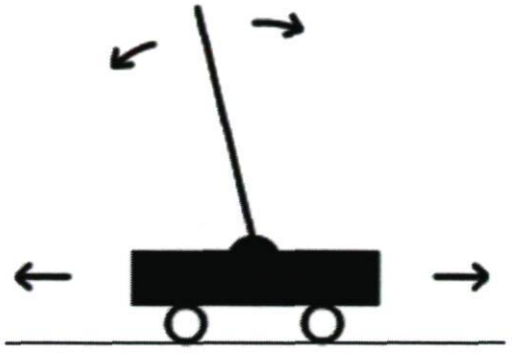


FIGURE 4.7 – Illustration du problème du pendule inversé. L'objectif est de maintenir le pendule à la verticale par des mouvements latéraux.

du système, nous avons utilisé la méthode d'Euler pour mettre à jour l'état du système :

$$\begin{aligned} \ddot{\alpha} &= \frac{g \sin\alpha + \cos\alpha \left(\frac{-F - m_p l \dot{\alpha}^2 \sin\alpha}{m_c + m_p} \right)}{l \left(\frac{4}{3} - \frac{m_p \cos^2\alpha}{m_c + m_p} \right)} \\ \ddot{\varphi} &= \frac{F + m_p l (\dot{\alpha}^2 \sin\alpha - \ddot{\alpha} \sin\alpha)}{m_c + m_p} \end{aligned}$$

où g est la force de gravité, F est la force reliée à l'action, l est la moitier de la longueur du chariot, m_p est la masse du pendule et m_c la masse du chariot.

Pour ce problème, l'ensemble d'entraînement a été tiré aléatoirement de la même façon que pour *sincsig*. Les couples état-action ont été uniformément échantillonnés sur leur domaine respectif. Les sorties, c'est-à-dire les états suivants, ont d'abord été obtenues via les vraies dynamiques du système, pour ensuite être perturbées par un bruit de variance aléatoire. Comme les variances de ces variables en sorties sont connues, l'ensemble d'entraînement peut être vu comme des entrées Gaussiennes renvoyant vers des distributions Gaussiennes en sorties. Par conséquent, certain pourrait utiliser une séquence d'*états de croyance* Gaussiens en tant qu'ensemble d'entraînement pour apprendre les dynamiques d'un système dynamique partiellement observable. Suivant cette idée, il n'y a aucune raison que les distributions sur les sorties soient de variances significativement différentes par rapport aux variances des distributions en entrée.

Lors de cette expérimentation, les variances du bruit sur les entrées et les sorties sont tirées aléatoirement d'une loi uniforme sur l'intervalle $[0, 2.5]$ pour chacune des dimensions. Chaque dimension de sortie est traitée indépendamment des autres en utilisant un processus Gaussien a priori différent pour chacune des dimensions. La figure 4.8 montre la moyenne de l'erreur quadratique moyenne sur 25 ensembles d'entraînement

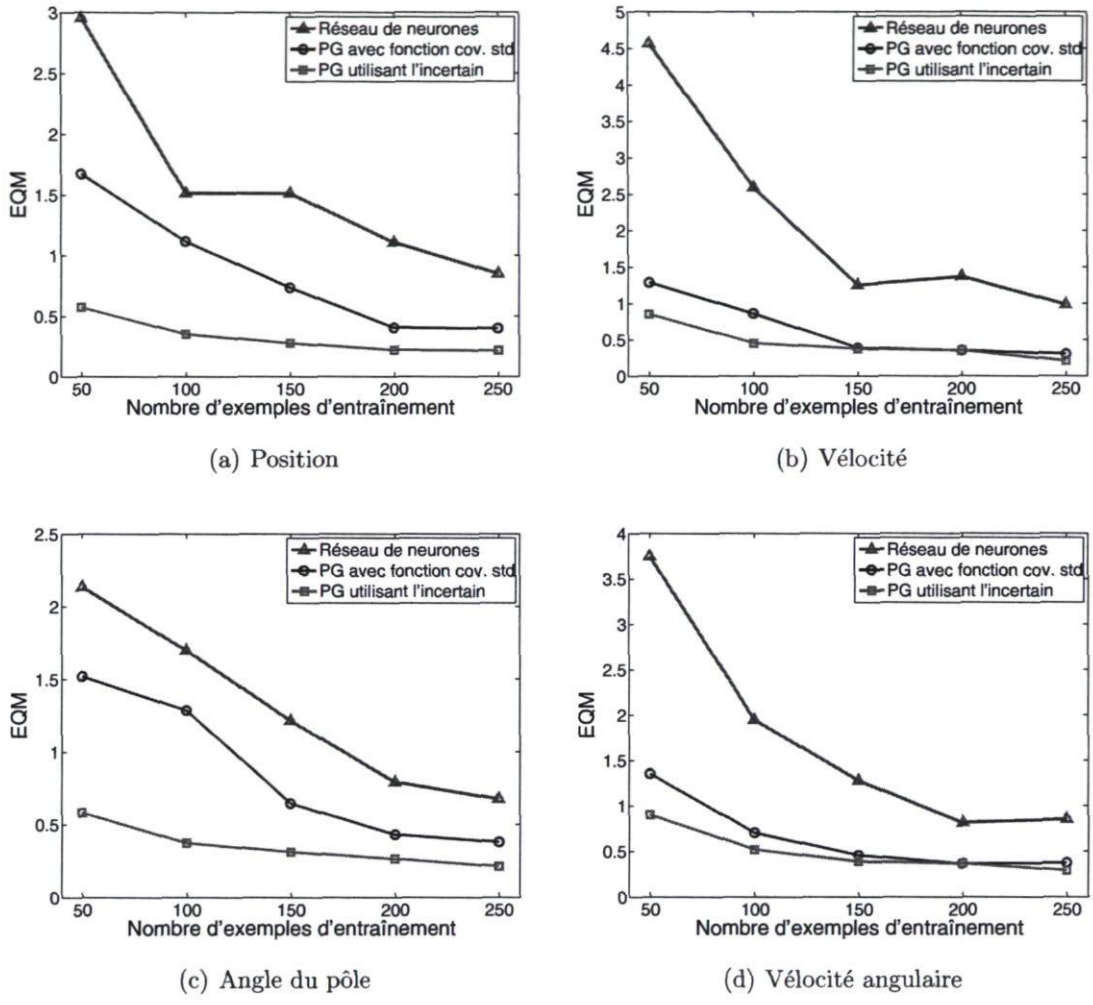
aléatoires pour différentes tailles et pour chaque dimension.

4.4 Discussion

Apprendre à partir de données incertaines est reconnu comme étant difficile. Sous l'hypothèse du processus Gaussien, un ensemble de données ayant une distribution a priori possèdent une covariance incertaine et, conséquemment, une distribution sur les matrices de covariances. Considérer l'entière distribution sur l'espace des matrices de covariances afin de calculer le processus Gaussien a posteriori est une opération intractable en général, et ce, même pour le cas de la fonction de covariance carré-exponentielle couplée avec des entrées incertaines Gaussiennes. Cependant, ce dernier cas particulier nous permet de calculer facilement l'espérance de cette distribution sur les matrices de covariance, ce qui rend la procédure d'inférence avec entrées incertaines aussi simple que dans le cas sans bruit. Les expérimentations ont été menées en vue de déterminer si l'usage de ces matrices de covariance espérées améliore la qualité de l'apprentissage lorsqu'il est question de données incertaines.

Les premières expérimentations ont été réalisées sur l'apprentissage de la fonction artificielle *sincsig*. Cette fonction est conçue de sorte à contenir certaines caractéristiques difficiles à apprendre en contexte d'incertitude. En particulier, la croissance exponentielle du centre peut s'avérer difficile à percevoir à partir d'ensemble d'entraînement contenant des entrées bruitées. La première expérimentation visait donc à isoler les performances à utiliser la matrice de covariance espérée par rapport à des méthodes classiques. À cette fin, nous avons généré des ensembles d'entraînement comportant des entrées à incertitude aléatoire et des sorties faiblement bruitées. Ensuite, nous avons mesuré les performances via l'erreur quadratique moyenne, lesquels sont affichée à la figure 4.4(a). On remarque que le PG-incertain atteint une erreur très faible, ce qui est certainement dû à l'information apportée par le calcul de la matrice de covariance espérée. En effet, celle-ci permet, contrairement au PG-naïf, de départager les données d'entraînement pertinentes de celles qui sont largement incertaines.

Suite aux résultats positifs concernant la méthode proposée, nous sommes passés aux ensembles d'entraînement entièrement incertain en ajoutant maintenant des sorties incertaines. Notons que pour des raisons de mise à l'échelle, le niveau de bruit en sortie doit être plus faible que celui en entrée pour le problème sincsig. Ainsi, nous observons à la 4.4(b) que les performances sont à nouveau positives, mais que cette fois nous pouvons clairement distinguer les avantages de chaque méthode. D'abord, le réseau de neurones n'utilise que les données bruitées (sans les variances) et nous n'apprenons pas



(a) Position

(b) Vélocité

(c) Angle du pôle

(d) Vélocité angulaire

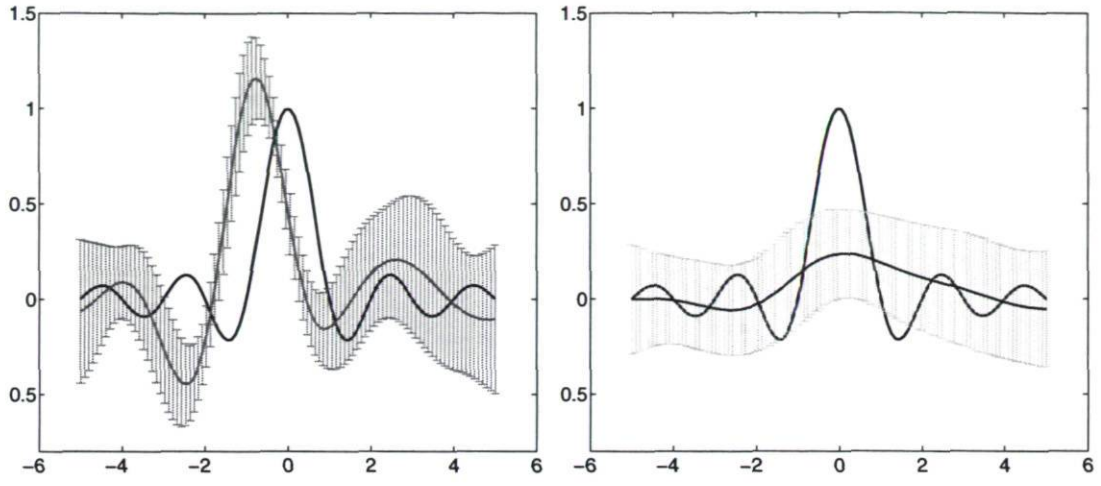
FIGURE 4.8 – Résultats de l'erreur quadratique moyenne sur le problème du pendule inversé. En trait rouge avec des carrés (PG-incertain), trait bleu avec des cercles (PG-naïf) et trait vert avec des triangles (Réseau de neurones).

la structure du réseau, laquelle est sensible lors des apprentissages avec peu de données. Pour ce qui est de la méthode PG-naïf, celle-ci à d'une part accès aux variances en sortie, et d'autre part, l'optimisation des hyperparamètres peut être vue comme un apprentissage de la structure du réseau de neurones infini sous-entendu par un processus Gaussien. Évidemment, la méthode PG-incertain hérite de ces derniers avantages, mais surtout, elle permet d'atteindre de meilleures performances par la matrice de covariance espérée.

Ces expérimentations ont été réalisées avec des ensembles de données contenant des entrées largement incertaines, ou plus précisément, l'écart-type du bruit affectant les entrées est borné supérieurement par 2.5. Comme il n'y a que peu d'exemples connus avec quasi certitude pour guider l'inférence, cette contrainte rend la tâche d'apprentissage très difficile. Face à des ensembles d'entraînement aussi flous, l'utilisation du processus Gaussien espéré offre un avantage significatif par rapport à un usage naïf de la méthode usuelle.

Afin de confirmer la pertinence de la matrice de covariance espérée, une seconde série d'expérimentations a été réalisée où la méthode PG-incertain est maintenant basée sur une composition de la fonction de covariance carré-exponentielle espérée et de la carré-exponentielle simple. Lors de la phase d'optimisation de la log-vraisemblance marginale, chaque fonction de covariance se voit attribuer un poids, lequel est un indicateur de sa pertinence par rapport aux données. Cette fois, les performances sont mesurées face à des données d'entraînement dont le niveau d'incertitude est variable. Les résultats pour l'erreur quadratique moyenne sont affichés à la figure 4.5 et ceux pour la log-vraisemblance sont affichés à la figure 4.6. Malgré qu'elle ait maintenant une fonction de covariance plus expressive, la méthode PG-incertain démontre une faible amélioration des performances en erreur quadratique. Par contre, face aux ensembles d'entraînement incertain, la log-vraisemblance est supérieure. Une explication à ces résultats est qu'en général, les poids des fonctions de covariance étaient similaires, ce qui pourrait avoir détérioré les performances. Nous nous attendions en fait à un poids plus élevé pour la fonction de covariance espérée contre un poids faible pour la version standard.

Afin de nous rapprocher du problème d'apprentissage de processus décisionnel de Markov partiellement observable (voir chapitre 3), nous avons élaboré une expérimentation sur l'apprentissage de système dynamique. Cet apprentissage se fait à partir d'exemples de transition incertaine, c'est-à-dire que les données d'entraînement sont des couples d'états de croyance, provenant d'un simulateur de pendule inversé, un problème bien connu dans la communauté de l'apprentissage par renforcement. Les résultats sont affichés à la figure 4.8. Les performances des méthodes PG-incertain et PG-naïf sont comparées sur la base de l'erreur quadratique moyenne. À nouveau, l'utilisation de



(a) PG-incertain (b) PG-naïf

FIGURE 4.9 – Processus Gaussien a posteriori typique pour la fonction sinus cardinal.

l'espérance de la covariance produit une estimation plus fidèle de la vraie fonction. Par contre, beaucoup plus de données aurait été nécessaires afin d'identifier correctement les dynamiques, mais comme la complexité de calculer le processus Gaussien a posteriori est $\mathcal{O}(N^3)$, et que l'ajout de l'apprentissage des hyperparamètres requiert $\mathcal{O}(N^3M)$, nous avons dû limiter la quantité de données à un maximum de 250. Par ailleurs, le calcul de la matrice de covariance avec la fonction de covariance carré-exponentielle espérée (4.5) nécessite plus de temps, car celle-ci implique des calculs de déterminants potentiellement coûteux.

Lors des précédentes expérimentations, nous avons observé qu'une augmentation de l'erreur quadratique moyenne est fréquemment causée par un apprentissage d'une version biaisée de la fonction recherchée, et par conséquent, la log-vraisemblance décroît lorsque le biais augmente. Voici un petit exemple pour la fonction *sinus cardinale* permettant de comprendre le type d'apprentissage sous-jacent aux résultats exposés. La figure 4.9 montre un exemple de distribution a posteriori sur l'espace des fonctions obtenu de PG-incertain. On remarque que la fonction apprise est biaisée, mais que celle-ci exhibe un comportement similaire à la vraie fonction. Pour sa part, la méthode PG-naïf apprend une version convoluée de la fonction. Évidemment, certains pourraient sans doute préférer une fonction biaisée ayant la forme générale de la vraie fonction plutôt qu'une estimation capturant moins de structure dans les données, notamment en ce qui a trait à l'identification de système.

4.5 Conclusion

Lorsqu'il est question d'apprentissage supervisé, la plupart des méthodes font l'hypothèse que l'ensemble d'entraînement inclut des données d'entrées exactes. Évidemment, cette hypothèse peut s'avérer déraisonnable face à certains problèmes. Au cours de ce chapitre, nous avons proposé une approche permettant l'apprentissage lorsque les exemples d'entraînement sont incertains autant en entrée qu'en sortie.

En approchant le problème sous un angle Bayésien, nous en sommes arrivés à une formulation des processus Gaussiens basée sur l'espérance de la covariance permettant d'incorporer l'incertitude des exemples. Pour y arriver, nous avons d'abord fait l'hypothèse que les données d'entraînement sont Gaussiennement distribuée pour ensuite intégrer l'incertitude à la fonction de covariance, ce qui résulte en une fonction de covariance espérée. Cette méthode est particulièrement intéressante de par le fait qu'elle n'implique qu'un choix différent de fonction de covariance pour former le processus Gaussien a priori sur des données incertaines. Les résultats obtenus sur un problème artificiel et sur l'identification d'un système dynamique ont permis de montrer que l'approche offre de meilleures performances que certaines méthodes classiques.

Évidemment, l'approche peut encore être développée. Il est clair que l'apprentissage à partir de données incertaines est difficile et que pour inférer efficacement la fonction recherchée, un plus grand nombre d'exemples pourrait être nécessaire. Cependant, le temps de calcul s'en verra augmentée. Il serait donc pertinent de voir comment la méthode d'approximation de processus Gaussien proposée par Snelson [2007] peut être adapter à la méthode exposée au cours de ce chapitre. Par ailleurs, une avenue de recherche intéressante concerne les distributions de probabilité Wishart sur les matrices définies positives. Cette distribution pourrait permettre des matrices de covariance comportant plus d'information concernant les ensembles d'entraînement incertains, ce qui améliorerait bien entendu les capacités de prédiction d'un processus Gaussien.

Une autre avenue intéressante consisterait à expérimenter les performances qu'offre le calcul d'un ensemble d'entraînement a posteriori exposé au cours de ce chapitre. Nous avançons qu'utiliser les processus Gaussiens espérés avec une divergence de Kullback-Leibler en guise de pénalité pour modifier l'ensemble d'entraînement permettrait d'obtenir de meilleurs résultats. Par contre, il est impératif pour cette méthode d'utiliser des méthodes d'optimisation plus sophistiquées. Nous suggérons d'investiguer les approches par gradient naturel pour ce problème d'optimisation, lesquels sont bien adapté pour cette tâche [Le Roux *et al.*, 2008]. D'ailleurs, cette méthode d'optimisation pourrait être appliquée à l'apprentissage des hyperparamètres, car, tel que rapporté, la fonction de

log-vraisemblance contient généralement un nombre potentiellement élevé de maxima locaux pour lesquels les méthodes de gradient classique ne conviennent pas.

Finalement, comme notre principal intérêt face au développement du processus Gaussien en contexte d'incertitude est une application à l'apprentissage du processus décisionnel de Markov partiellement observable (PDMPO), il serait intéressant d'ajouter les nouveaux aspects développés à la méthode proposée au chapitre 3. Nous croyons qu'une inclusion directe à la méthode pourrait être faite en ajoutant la pénalité de la divergence de Kullback-Leibler à la phase d'optimisation permettant l'apprentissage du PDMPO.

Chapitre 5

Conclusion

L'apprentissage automatique est un domaine de recherche en pleine effervescence qui s'est forgé une place de choix en informatique de par le paradigme de programmation qu'il propose. Contrairement à l'approche traditionnelle consistant à développer un algorithme permettant à un système de réaliser une tâche, l'apprentissage automatique propose des systèmes qui ont les capacités d'évoluer en accroissant leurs performances via des algorithmes d'apprentissage.

L'apprentissage par renforcement est l'une des méthodes proposées pour le développement automatique de systèmes. Cette approche consiste essentiellement à récompenser un système en fonction des objectifs accomplis et de le pénaliser lorsque celui-ci s'en éloigne. Sous-tendant l'apprentissage par renforcement, nous retrouvons fréquemment les modèles Markoviens tels que le processus décisionnel de Markov ainsi que sa version plus générale, le processus décisionnel de Markov partiellement observable (PDMPO). À ce jour plusieurs algorithmes d'apprentissage ont été proposés pour résoudre les problèmes basés sur ces modèles, lesquels se fondent quelquefois sur certaines hypothèses fortes concernant le problème à étudier.

L'objet premier de cette recherche touche plus particulièrement les méthodes d'apprentissage pour les processus décisionnels de Markov partiellement observables. Le premier consta fût que la majorité des approches actuelles font l'hypothèse que le problème est modélisable par des espaces discrets, ce qui rend leur application à des problèmes naturellement continus, tels le contrôle robotique et la navigation. Le second constat est que le modèle du monde est généralement supposé entièrement connu. Par conséquent, lorsque le modèle est inconnu et qu'il doit être identifié au cours de la prise de décisions, ces méthodes peuvent s'avérer déficientes. Dans ce contexte, nous avons exploré certaines avenues permettant de contourner ces limitations et donc d'étendre l'application

de ce type d'apprentissage à de nouveaux problèmes.

Pour pallier aux nouvelles lacunes des algorithmes d'apprentissage actuels au niveau d'un PDMPO, il faut sans aucun doute apprendre son modèle. Lorsque celui-ci est décrit sur des espaces d'états, d'observations, d'actions et de récompenses continus, nous avons essentiellement affaire à des problèmes de régression, une forme particulière d'apprentissage supervisé. Dès lors, nous avons choisi d'utiliser les processus Gaussiens pour s'attaquer à ces problèmes de régression, une méthode ayant fait ses preuves au cours des dernières années et qui offrent des performances équivalentes aux autres méthodes de l'état de l'art.

5.1 Contributions

Cette section résume les contributions exposées dans ce mémoire relatives au domaine de l'apprentissage automatique, plus particulièrement pour l'apprentissage de PDMPO.

- Une formulation des processus décisionnels de Markov partiellement observables sous forme de processus Gaussien. Cette formulation permet de contrôler l'espace des fonctions admissibles et attribue de nouvelles propriétés intéressantes aux PDMPO. Nous avançons que l'une des plus importantes avancées concerne la réduction des fonctions de valeurs possibles. Citons aussi un filtrage de l'état de croyance simplifié.

- Une formulation permettant d'exprimer des connaissances a priori concernant le modèle d'un PDMPO. Grâce à cette formulation, nous pouvons définir un modèle incertain de PDMPO, c'est-à-dire un modèle probable comportant des intervalles de confiance. Cela permet notamment de construire un modèle partiellement connu ayant des zones de grandes incertitudes et d'autres, où les dynamiques sont identifiées.

- Un algorithme d'apprentissage permettant d'identifier le modèle d'un PDMPO continu en ligne. Par rapport au point précédent, cela revient à améliorer les connaissances du modèle par un calcul de distribution a posteriori sur l'espace des modèles admissibles.

- Un algorithme permettant de combiner la prise de décision et l'apprentissage dans les PDMPO basés sur les processus Gaussiens.

- Une approche efficace permettant l'apprentissage de processus Gaussiens à partir de données d'entrée incertaines ou bruitées. L'avantage de cette méthode est qu'elle n'augmente pas la complexité de calcul de la régression par processus Gaussiens usuelle.

- Une méthode permettant l'apprentissage de processus Gaussiens à partir de données de sortie incertaines.

- Dérivation de fonctions de covariance espérées par rapport à des données d'entrée Gaussiennes. Ces fonctions de covariance sont applicables à n'importe quelle méthode d'apprentissage utilisant des méthodes à noyaux sur des données incertaines.

5.2 Travaux futurs

Les travaux exposés dans ce mémoire ont permis de mettre en lumière une approche permettant l'apprentissage des processus décisionnels de Markov partiellement observables continus. Bien que les résultats présentés au chapitre 3 ont été concluants quant à la validité de l'approche, celle-ci peut évidemment être améliorée. Dans un premier temps, il est impératif d'utiliser une méthode d'approximation du processus Gaussien pour une application à plus grande échelle où la quantité de données serait plus large. Au meilleur de nos connaissances, la méthode d'approximation la plus adaptée au problème traité dans ce mémoire est la méthode développée par Snelson [2007]. D'autre part, nous avons aussi soulevé que l'apprentissage par optimisation de la vraisemblance où l'état de croyance n'est représenté que par un point, entraîne une perte d'information. La solution que nous avons proposée consiste à utiliser des distributions de probabilités Gaussiennes pour représenter les états de croyance de l'agent. Cependant, cette nouvelle forme d'état de croyance n'est pas adaptée à l'apprentissage par processus Gaussiens qui requiert des entrées certaines. Nous avons donc adapté la régression par processus Gaussien afin de pouvoir identifier un système dynamique à partir de séquences d'états de croyance. Cependant, plusieurs points restent en suspens avant d'aller de l'avant avec un nouvel algorithme d'apprentissage de PDMPO plus efficace.

Pour parvenir à un algorithme efficace, nous proposons d'abord de combiner la méthode d'approximation de Snelson [2007] avec l'approche de l'incertitude par cova-

riance espérée couverte au chapitre 4. Cette méthode de régression devra par la suite être intégrée à l'apprentissage de PDMPO. Nous suggérons de porter une attention particulière aux méthodes de projection de l'espace d'entrée afin de réduire le temps de calcul et la détection automatique de la dimensionalité de l'espace d'état.

Une fois l'algorithme d'identification du modèle du PDMPO développé, il faut pouvoir suivre l'évolution incertaine du système en mettant à jour l'état de croyance. Selon l'objectif fixé, il serait nécessaire de développer une méthode d'inférence des séquences d'états utilisant les propriétés des processus Gaussiens. Pour réaliser cette partie, nous suggérons d'adapter le lissage de Rauch-Tung-Striebel non-parfumé de Sarkka [2008] aux processus Gaussiens.

Finalement, nous avons avancé plus haut que le modèle PDMPO sous l'hypothèse de processus Gaussiens possède de nouvelles propriétés fort intéressantes. L'une des plus importantes est probablement la simplification de la fonction de valeur. Cette avenue purement théorique permettrait d'accélérer le calcul de la politique optimale d'un PDMPO continu, mais aussi de dégager certaines bornes sur la fonction estimée.

Bibliographie

ABBEEL, P., COATES, A., QUIGLEY, M. et NG, A. Y. (2007). An application of reinforcement learning to aerobatic helicopter flight. *In Advances in Neural Information Processing Systems 19 (NIPS'06)*, pages 1–8.

BELLMAN, R. (1957). *Dynamic Programming.* Princeton University Press.

BESSE, C. et CHAIB-DRAA, B. (2009). Quasi-deterministic partially observable markov decision processes. *In Proceedings of 16th International Conference On Neural Information Processing (ICONIP'09)*, pages 237–246.

BI, J. et ZHANG, T. (2004). Support vector classification with input data uncertainty. *In Advances in Neural Information Processing Systems 17 (NIPS'04)*, pages 161–168.

CANDELA, J. Q. (2004). *Learning with Uncertainty - Gaussian Processes and Relevance Vector Machines.* Thèse de doctorat, Technical University of Denmark.

CAROLL, R., RUPPERT, D. et STEFANSKI, L. (1995). *Measurement Error in Nonlinear Models.* Chapman and Hall.

CASTRO, P. S. et PRECUP, D. (2007). Using linear programming for bayesian exploration in markov decision processes. *In Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI'07)*, pages 2437–2442.

CRESSIE, N. A. C. (1993). *Statistics for Spatial Data.* Wiley Series in Probability and Statistics. Wiley-Interscience.

DALLAIRE, P., BESSE, C. et CHAIB-DRAA, B. (2009a). Learning gaussian process models from uncertain data. *In Proceedings of 16th International Conference On Neural Information Processing (ICONIP'09)*, pages 433–440.

DALLAIRE, P., BESSE, C., ROSS, S. et CHAIB-DRAA, B. (2009b). Bayesian reinforcement learning in continuous pomdps with gaussian processes. *In Proceedings of the 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'09)*, pages 2604–2609.

DOSHI, F., PINEAU, J. et ROY, N. (2008). Reinforcement learning with limited reinforcement : using Bayes risk for active learning in POMDPs. *In Proceedings of the Twenty-Fifth International Conference (ICML'08)*, pages 256–263.

DUFF, M. (2002). *Optimal Learning : Computational procedures for Bayes-adaptive Markov decision processes.* Thèse de doctorat, University of Massachusetts Amherst, Amherst, MA.

FLORIAN, R. (2007). Correct Equations for the Dynamics of the Cart-pole System. Rapport technique, Center for Cognitive and Neural Studies.

GHAHRAMANI, Z. et ROWEIS, S. T. (1999). Learning nonlinear dynamical systems using an em algorithm. *In Advances in Neural Information Processing Systems 11 (NIPS'98)*, pages 431–437.

GILL, P. E., MURRAY, W. et WRIGHT, M. H. (1981). *Practical Optimization.* London Academic Press.

GIRARD, A. (2004). *Approximate Methods for Propagation of Uncertainty with Gaussian Process Models.* Thèse de doctorat, University of Glasgow.

GIRARD, A. et MURRAY-SMITH, R. (2003). Learning a gaussian process model with uncertain inputs. Rapport technique TR-2003-144, University of Glasgow.

GOLUB, H. G. et VAN LOAN, F. C. (1980). An analysis of the total least squares problem. *SIAM Journal on Numerical Analysis*, (17):883–893.

GOMES, S. B. V. et RAMOS, J. J. G. (1998). Airship dynamic modeling for autonomous operation. *In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'98)*, pages 3462–3467.

HORNIK, K. (1993). Some new results on neural network approximation. *Neural Networks*, 6(8):1069 – 1072.

HYGOUNENC, E., JUNG, I., SOUERES, P. et LACROIX, S. (2004). The Autonomous Blimp Project of LAAS-CNRS : Achievements in Flight Control and Terrain Mapping. *International Journal of Robotics Research*, 23(4):473–511.

KOLMOGOROFF, A. (1941). Interpolation und extrapolation von stationaren zufalligen folgen. *Izvestiya Rossiiskoi Akademii Nauk. Seriya Matematicheskaya*, 5(1):3–14.

LAURITZEN, S. L. (1981). Time series analysis in 1880 : A discussion of contributions made by T.N. Thiele. *International Statistical Review / Revue Internationale de Statistique*, 49(3):319–331.

LE ROUX, N., MANZAGOL, P.-A. et BENGIO, Y. (2008). Topmoumoute Online Natural Gradient Algorithm. *In Advances in Neural Information Processing Systems 20 (NIPS'07)*, pages 849–856.

MACKAY, D. J. C. (2003). *Information Theory, Inference, and Learning Algorithms.* Cambridge University Press.

MACKEY, M. C. et GLASS, L. (1977). Oscillation and chaos in physiological control systems. *Science*, 197:287–289.

MARKOVSKY, I. et VAN HUFFEL, S. (2007). Overview of total least-squares methods. *Signal processing*, 87(10):2283–2302.

MATHERON, G. (1963). Principles of geostatistics. *Economic Geology*, 58(8):1246–1266.

MITCHELL, T. M. (1997). *Machine Learning*. McGraw-Hill.

MØLLER, M. (1993). A scaled conjugate gradient algorithm for fast supervised learning. *Neural networks*, 6(4):525–533.

NEAL, R. M. (1996). *Bayesian Learning for Neural Networks (Lecture Notes in Statistics)*. Springer Verlag.

O'HAGAN, A. (1978). On curve fitting and optimal design for regression. *Journal of the Royal Statistical Society, B*, 40:1–42.

PETERSEN, K. et PEDERSEN, M. (2006). The matrix cookbook. *Technical University of Denmark*.

PORTA, J. M., SPAAN, M. T. J. et VLASSIS, N. A. (2005). Robot planning in partially observable continuous domains. *In Proceedings of the International Conference on Robotics : Science and Systems (RSS'05)*, pages 217–224.

PORTA, J. M., VLASSIS, N. A., SPAAN, M. T. J. et POUPART, P. (2006). Point-based value iteration for continuous pomdps. *Journal of Machine Learning Research*, 7:2329–2367.

POUPART, P., VLASSIS, N., HOEY, J. et REGAN, K. (2006). An analytic solution to discrete Bayesian reinforcement learning. *In Proceedings of the 23rd international conference on Machine learning*, pages 697–704. ACM New York, NY, USA.

QUIÑONERO-CANDELA, J. et ROWEIS, S. T. (2003). Data imputation and robust training with gaussian processes.

RAO, J., GONG, Z., LUO, J. et XIE, S. (2006). A flight control and navigation system of a small size unmanned airship. *In Proceedings of the 2005 IEEE International Conference on Mechatronics and Automation (ICMA'05)*, volume 3, pages 1491–1496.

RASMUSSEN, C. E. et WILLIAMS, C. K. (2006). *Gaussian Processes for Machine Learning*. Adaptive Computation and Machine Learning. MIT Press.

ROSS, S., CHAIB-DRAA, B. et PINEAU, J. (2008a). Bayes-adaptive POMDPs. *In Advances in Neural Information Processing Systems 20 (NIPS'07)*, pages 1225–1232.

ROSS, S., CHAIB-DRAA, B. et PINEAU, J. (2008b). Bayesian reinforcement learning in continuous POMDPs with application to robot navigation. *In Proceedings of the 2008 IEEE International Conference on Robotics and Automation (ICRA'08)*, pages 2845–2851.

ROTTMANN, A., PLAGEMANN, C., HILGERS, P. et BURGARD, W. (2007). Autonomous blimp control using model-free reinforcement learning in a continuous state and action space. *In Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'07)*, pages 1895–1900.

SARKKA, S. (2008). Unscented Rauch-Tung-Striebel Smoother. *IEEE Transactions on Automatic Control*, 53(3):845–849.

SNELSON, E. L. (2007). *Flexible and efficient Gaussian process models for machine learning.* Thèse de doctorat, University of London.

SPAAN, M. et VLASSIS, N. (2005). PERSEUS : Randomized point-based value iteration for POMDPs. *Journal of Artificial Intelligence Research*, 24:195–220.

SUTTON, R. et BARTO, A. (1998). *Reinforcement Learning : An introduction.* The MIT Press.

SZEPESVÁRI, C. (2009). Reinforcement learning algorithms for MDPs. Rapport technique TR-09-13, University of Alberta.

TESAURO, G. (1995). Temporal difference learning and TD-gammon. *Communication of the ACM*, 38(3):58–68.

THRUN, S. (2000). Monte Carlo POMDPs. *In Advances in Neural Information Processing Systems 12 (NIPS'99)*, pages 1064–1070.

TING, J.-A., D'SOUZA, A. et SCHAAL, S. (2006). Bayesian regression with input noise for high dimensional data. *In Proceedings of the 23rd international conference on Machine learning (ICML'06)*, pages 937–944.

TRESP, V., AHMAD, S. et NEUNEIER, R. (1994). Training neural networks with deficient data. *In Advances in Neural Information Processing Systems 6 (NIPS'93)*, pages 128–135.

UHLENBECK, G. E. et ORNSTEIN, L. S. (1930). On the theory of the brownian motion. *Physical Review*, 36(5):823–841.

WANG, J., FLEET, D. et HERTZMANN, A. (2008). Gaussian Process Dynamical Models for Human Motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 283–298.

WANG, J. M., FLEET, D. J. et HERTZMANN, A. (2005a). Gaussian process dynamical models. *In Advances in Neural Information Processing Systems 18 (NIPS'05)*, pages 1441–1448.

WANG, T., LIZOTTE, D., BOWLING, M. et SCHUURMANS, D. (2005b). Bayesian sparse sampling for on-line reward optimization. *In Proceedings of the Twenty-Second International Conference on Machine Learning (ICML'05).*

WIENER, N. (1949). *Extrapolation, Interpolation, and Smoothing of Stationary Time Series.* The MIT Press.

WILLIAMS, C. K. I. (1998). Computation with infinite neural networks. *Neural Computation*, 10(5):1203–1216.

WILLIAMS, C. K. I. et RASMUSSEN, C. E. (1996). Gaussian processes for regression. *In Advances in Neural Information Processing Systems 8 (NIPS'95)*, pages 514–520.

WRIGHT, W. A., RAMAGE, G., CORNFORD, D. et NABNEY, I. T. (2000). Neural network modelling with input uncertainty : Theory and application. *VLSI Signal Processing*, 26(1-2):169–188.

WYETH, G. et BARRON, I. (1997). An Autonomous Blimp. *In Proceedings of the IEEE International Conference on Field and Service Robotics (FSR'97)*, pages 464–470.

ZHANG, H. et OSTROWSKI, J. (1999). Visual servoing with dynamics : control of an unmanned blimp. *In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'99)*, pages 618–623.

Annexe A

Dérivées partielles pour apprentissage de PDMPO

Afin d'optimiser l'équation (3.18), nous avons adopté une approche par descente de gradient. Nous présentons donc les dérivées partielles nécessaires aux calculs de gradients. D'abord, rappelons la fonction à optimiser :

$$\begin{aligned} f(\mathbf{S}, \bar{\alpha}, \bar{\beta}, \mathbf{W}) = & \frac{m+n}{2} \ln |\mathbf{K}_X| + \frac{1}{2} \mathrm{tr}(\mathbf{S}_{sor}^\top \mathbf{K}_X^{-1} \mathbf{S}_{sor}) \\ & + \frac{p+1}{2} \ln |\mathbf{K}_Y| + \frac{1}{2} \mathrm{tr}(\mathbf{W}\mathbf{Y}^\top \mathbf{K}_Y^{-1} \mathbf{Y}\mathbf{W}) - N \ln |\mathbf{W}| \\ & + \frac{1}{2} \mathbf{s}_0^\top \mathbf{s}_0 + \sum_i \ln \alpha_i + \sum_i \ln \beta_i \end{aligned}$$

où les dérivées requises sont par rapport à $\mathbf{S}$, $\bar{\alpha}$, $\bar{\beta}$ et $\mathbf{W}$. Comme certaines dérivées ne peuvent être faites directement, il faut appliquer la règle de dérivation en chaîne. Voici les dérivées par rapport aux fonctions composants f :

$$\frac{\partial f}{\partial \mathbf{K}_X} = \frac{m+n}{2} \mathbf{K}_X^{-1} - \frac{1}{2} \mathbf{K}_X^{-1} \mathbf{S}_{sor} \mathbf{S}_{sor}^\top \mathbf{K}_X^{-1} \tag{A.1}$$

$$\frac{\partial f}{\partial \mathbf{K}_Y} = \frac{p+1}{2} \mathbf{K}_Y^{-1} - \frac{1}{2} \mathbf{K}_Y^{-1} \mathbf{W}\mathbf{Y}\mathbf{Y}^\top \mathbf{W}^\top \mathbf{K}_Y^{-1} \tag{A.2}$$

$$\frac{\partial f}{\partial \mathbf{S}_{sor}} = \mathbf{K}_X^{-1} \mathbf{S}_{sor} \tag{A.3}$$

La dérivée par rapport à $\mathbf{S}$ est obtenue avec :

$$\frac{\partial f}{\partial \mathbf{S}} = \frac{\partial f}{\partial \mathbf{K}_X} \circ \frac{\partial \mathbf{K}_X}{\partial \mathbf{S}} + \frac{\partial f}{\partial \mathbf{K}_Y} \circ \frac{\partial \mathbf{K}_Y}{\partial \mathbf{S}} + \frac{\partial f}{\partial \mathbf{S}_{sor}} \circ \frac{\partial \mathbf{S}_{sor}}{\partial \mathbf{S}} + \mathbf{S}_{0,:} \tag{A.4}$$

où la matrice $\mathbf{S}_{0,:}$ contient la dérivée sur sa première ligne ainsi que des zéros pour le reste. D'autre part, $\frac{\partial \mathbf{K}_X}{\partial \mathbf{S}}$, $\frac{\partial \mathbf{K}_Y}{\partial \mathbf{S}}$ et $\frac{\partial \mathbf{S}_{sqr}}{\partial \mathbf{S}}$ nécessite les dérivées de la matrice de covariance. Pour les former, il faut prendre les dérivées partielle de chaque élément de la matrice par rapport à $\mathbf{S}$:

$$\frac{\partial k_X(\mathbf{x},\mathbf{x}')}{\partial \mathbf{s}} = -\beta_1\beta_2(\mathbf{s}-\mathbf{s}')\exp\left(-\frac{\beta_2}{2}\|\mathbf{x}-\mathbf{x}'\|^2\right)+\beta_3\mathbf{s}' \tag{A.5}$$

$$\frac{\partial k_Y(\mathbf{x},\mathbf{x}')}{\partial \mathbf{s}} = -\alpha_1\alpha_2(\mathbf{s}-\mathbf{s}')\exp\left(-\frac{\alpha_2}{2}\|\mathbf{x}-\mathbf{x}'\|^2\right)+\alpha_3^{-1}\delta_{\mathbf{x}\mathbf{x}'} \tag{A.6}$$

où nous rappelons que $\mathbf{s}$ est la partie état du vecteur $\mathbf{x} = [\mathbf{s}, \mathbf{a}]$.

La dérivée par rapport aux hyperpamètres $\bar{\alpha}$ est obtenue avec :

$$\frac{\partial f}{\partial \bar{\alpha}} = \frac{\partial f}{\partial \mathbf{K}_X} \circ \frac{\partial \mathbf{K}_X}{\partial \bar{\alpha}} + \frac{\partial f}{\partial \mathbf{K}_Y} \circ \frac{\partial \mathbf{K}_Y}{\partial \bar{\alpha}} + \sum_i \frac{1}{\alpha_i} \tag{A.7}$$

où $\frac{\partial \mathbf{K}_X}{\partial \bar{\alpha}}$ nécessite à nouveau des dérivées pour la matrice de covariance :

$$\frac{\partial k_Y(\mathbf{x},\mathbf{x}')}{\partial \alpha_1} = \exp\left(-\frac{\alpha_2}{2}\|\mathbf{x}-\mathbf{x}'\|^2\right) \tag{A.8}$$

$$\frac{\partial k_Y(\mathbf{x},\mathbf{x}')}{\partial \alpha_2} = -\alpha_1\|\mathbf{x}-\mathbf{x}'\|^2\exp\left(-\frac{\alpha_2}{2}\|\mathbf{x}-\mathbf{x}'\|^2\right) \tag{A.9}$$

$$\frac{\partial k_Y(\mathbf{x},\mathbf{x}')}{\partial \alpha_3} = -\alpha_3^{-2}\delta_{\mathbf{x}\mathbf{x}'} \tag{A.10}$$

(A.11)

La dérivée par rapport aux hyperpamètres $\bar{\beta}$ est obtenue avec :

$$\frac{\partial f}{\partial \bar{\beta}} = \frac{\partial f}{\partial \mathbf{K}_X} \circ \frac{\partial \mathbf{K}_X}{\partial \bar{\beta}} + \frac{\partial f}{\partial \mathbf{K}_Y} \circ \frac{\partial \mathbf{K}_Y}{\partial \bar{\beta}} + \sum_i \frac{1}{\beta_i} \tag{A.12}$$

où $\frac{\partial \mathbf{K}_X}{\partial \beta}$ requière les dérivées :

$$\frac{\partial k_X(\mathbf{x},\mathbf{x}')}{\partial \beta_1} = \exp\left(-\frac{\beta_2}{2}\|\mathbf{x}-\mathbf{x}'\|^2\right) \tag{A.13}$$

$$\frac{\partial k_X(\mathbf{x},\mathbf{x}')}{\partial \beta_2} = -\frac{\beta_1}{2}\|\mathbf{x}-\mathbf{x}'\|^2\exp\left(-\frac{\beta_2}{2}\|\mathbf{x}-\mathbf{x}'\|^2\right) \tag{A.14}$$

$$\frac{\partial k_X(\mathbf{x},\mathbf{x}')}{\partial \beta_3} = \mathbf{x}^\top\mathbf{x}' \tag{A.15}$$

$$\frac{\partial k_X(\mathbf{x},\mathbf{x}')}{\partial \beta_4} = -\alpha_4^{-2}\delta_{\mathbf{x}\mathbf{x}'} \tag{A.16}$$

(A.17)

La dérivée par rapport aux hyperpamètres $\mathbf{W}$ est tel que :

$$\frac{\partial f}{\partial \mathbf{W}} = \mathbf{W}\mathbf{Y}^{\top}\mathbf{K}_{Y}^{-1}\mathbf{Y} - N\mathbf{W}^{-1} \tag{A.18}$$

laquelle peut peut être mise à zéro afin d'obtenir le maximum en forme analytique :

$$\mathbf{W} = N\sqrt{(\mathbf{W}\mathbf{Y}^{\top}\mathbf{K}_{Y}^{-1}\mathbf{Y})^{-1}} \tag{A.19}$$

Les calculs de dérivation réalisés dans cette section sont essentiellement basés sur l'ouvrage de [Petersen et Pedersen, 2006].

Annexe B

Dérivation de fonction d'espérance de covariance

B.1 Fonction de covariance linéaire espérée

L'espérance de la fonction de covariance linéaire par rapport à des entrées incertaines, dont les distributions sont $p(\mathbf{x}_i) = \mathcal{N}_{\mathbf{x}_i}(\mathbf{u}_i, \mathbf{\Sigma}_i)$ et $p(\mathbf{x}_j) = \mathcal{N}_{\mathbf{x}_j}(\mathbf{u}_j, \mathbf{\Sigma}_j)$, est donnée par :

$$\begin{aligned} K_{ij} &= \iint k_{Lin}(\mathbf{x}_i, \mathbf{x}_j)\mathcal{N}_{\mathbf{x}_i}(\mathbf{u}_i, \mathbf{\Sigma}_i)\mathcal{N}_{\mathbf{x}_j}(\mathbf{u}_j, \mathbf{\Sigma}_j)d\mathbf{x}_i d\mathbf{x}_j \\ &= \mathbb{E}_i\left[\mathbb{E}_j\left[\mathbf{x}_i^\top \mathbf{x}_j + \sigma_b^2\right]\right] \\ &= \mathbb{E}_i[\mathbf{x}_i^\top]\ \mathbb{E}_j[\mathbf{x}_j] + \sigma_b^2 \\ &= \mathbf{u}_i^\top \mathbf{u}_j + \sigma_b^2 \end{aligned}$$

Le calcul de la variance espéré se fait tel que :

$$\begin{aligned} K_{ii} &= \int k_{Lin}(\mathbf{x}_i, \mathbf{x}_i)\mathcal{N}_{\mathbf{x}_i}(\mathbf{u}_i, \mathbf{\Sigma}_i)d\mathbf{x}_i \\ &= \mathbb{E}_i\left[\mathbf{x}_i^\top \mathbf{x}_i + \sigma_b^2\right] \\ &= \mathbb{E}_i\left[\mathbf{x}_i^\top \mathbf{x}_i\right] + \sigma_b^2 \\ &= \mathrm{Tr}(\mathbf{\Sigma}_i) + \mathbf{u}_i^\top \mathbf{u}_j + \sigma_b^2 \end{aligned}$$

D'après l'observation que l'unique différence entre la variance et la covariance est

la présence d'une trace pour le cas de la variance, nous pouvons définir la fonction d'espérance de covariance linéaire par :

$$k_{\mathbb{E}lin}((\mathbf{u}_i, \boldsymbol{\Sigma}_i), (\mathbf{u}_j, \boldsymbol{\Sigma}_j)) = \mathbf{u}_i^\top \mathbf{u}_j + \sigma_b^2 + \delta_{ij}\mathrm{Tr}(\boldsymbol{\Sigma}_i)$$

en introduisant un delta de Kronecker s'activant lorsqu'il est question d'obtenir une variance.

B.2 Fonction de covariance quadratique espérée

L'espérance de la fonction de covariance quadratique par rapport à des entrées incertaines, dont les distributions sont $p(\mathbf{x}_i) = \mathcal{N}_{\mathbf{x}_i}(\mathbf{u}_i, \boldsymbol{\Sigma}_i)$ et $p(\mathbf{x}_j) = \mathcal{N}_{\mathbf{x}_j}(\mathbf{u}_j, \boldsymbol{\Sigma}_j)$, est donnée par :

$$\begin{aligned}
K_{ij} &= \iint k_{qua}(\mathbf{x}_i, \mathbf{x}_j)\mathcal{N}_{\mathbf{x}_i}(\mathbf{u}_i, \boldsymbol{\Sigma}_i)\mathcal{N}_{\mathbf{x}_j}(\mathbf{u}_j, \boldsymbol{\Sigma}_j)d\mathbf{x}_i d\mathbf{x}_j \\
&= \mathbb{E}_i\left[\mathbb{E}_j\left[(\mathbf{x}_i^\top \mathbf{x}_j + \sigma_b^2)^2\right]\right] \\
&= \mathbb{E}_i\left[\mathbb{E}_j\left[\mathbf{x}_i^\top \mathbf{x}_j \mathbf{x}_i^\top \mathbf{x}_j + 2\sigma_b^2 \mathbf{x}_i^\top \mathbf{x}_j + \sigma_b^4\right]\right] \\
&= \mathbb{E}_i\left[\mathbb{E}_j\left[\mathbf{x}_i^\top \mathbf{x}_j \mathbf{x}_i^\top \mathbf{x}_j\right]\right] + 2\sigma_b^2\, \mathbb{E}_i\left[\mathbb{E}_j\left[\mathbf{x}_i^\top \mathbf{x}_j\right]\right] + \sigma_b^4 \\
&= \mathbb{E}_i\left[\mathbf{x}_i^\top\, \mathbb{E}_j\left[\mathbf{x}_j \mathbf{x}_j^\top\right] \mathbf{x}_i\right] + 2\sigma_b^2 \mathbf{u}_i^\top \mathbf{u}_j + \sigma_b^4 \\
&= \mathbb{E}_i\left[\mathbf{x}_i^\top \left(\boldsymbol{\Sigma}_j + \mathbf{u}_j \mathbf{u}_j^\top\right) \mathbf{x}_i\right] + 2\sigma_b^2 \mathbf{u}_i^\top \mathbf{u}_j + \sigma_b^4 \\
&= \mathrm{Tr}\left(\left[\boldsymbol{\Sigma}_j + \mathbf{u}_j \mathbf{u}_j^\top\right] \boldsymbol{\Sigma}_i\right) + \mathbf{u}_i^\top \left[\boldsymbol{\Sigma}_j + \mathbf{u}_j \mathbf{u}_j^\top\right] \mathbf{u}_i + 2\sigma_b^2 \mathbf{u}_i^\top \mathbf{u}_j + \sigma_b^4
\end{aligned}$$

où nous utilisons certaines propriétés de la trace afin d'obtenir une forme plus intuitive, c-à-d. une forme symétrique, pour les 2 premiers termes :

$$\begin{aligned}
&\mathrm{Tr}\left(\left[\boldsymbol{\Sigma}_j + \mathbf{u}_j \mathbf{u}_j^\top\right] \boldsymbol{\Sigma}_i\right) + \mathbf{u}_i^\top \left[\boldsymbol{\Sigma}_j + \mathbf{u}_j \mathbf{u}_j^\top\right] \mathbf{u}_i \\
=&\mathrm{Tr}\left(\left[\boldsymbol{\Sigma}_j + \mathbf{u}_j \mathbf{u}_j^\top\right] \boldsymbol{\Sigma}_i\right) + \mathrm{Tr}\left(\mathbf{u}_i^\top \left[\boldsymbol{\Sigma}_j + \mathbf{u}_j \mathbf{u}_j^\top\right] \mathbf{u}_i\right) \\
=&\mathrm{Tr}\left(\left[\boldsymbol{\Sigma}_j + \mathbf{u}_j \mathbf{u}_j^\top\right] \boldsymbol{\Sigma}_i\right) + \mathrm{Tr}\left(\left[\boldsymbol{\Sigma}_j + \mathbf{u}_j \mathbf{u}_j^\top\right] \mathbf{u}_i \mathbf{u}_i^\top\right) \\
=&\mathrm{Tr}\left(\left[\boldsymbol{\Sigma}_j + \mathbf{u}_j \mathbf{u}_j^\top\right] \boldsymbol{\Sigma}_i + \left[\boldsymbol{\Sigma}_j + \mathbf{u}_j \mathbf{u}_j^\top\right] \mathbf{u}_i \mathbf{u}_i^\top\right) \\
=&\mathrm{Tr}\left(\left[\boldsymbol{\Sigma}_j + \mathbf{u}_j \mathbf{u}_j^\top\right] \left[\boldsymbol{\Sigma}_i + \mathbf{u}_i \mathbf{u}_i^\top\right]\right)
\end{aligned}$$

En substituant dans la fonction d'espérance de covariance quadratique, nous obtenons une forme quadratique de cette fonction :

$$K_{ij} = \mathrm{Tr}\left(\left[\boldsymbol{\Sigma}_j + \mathbf{u}_j \mathbf{u}_j^\top\right] \left[\boldsymbol{\Sigma}_i + \mathbf{u}_i \mathbf{u}_i^\top\right]\right) + 2\sigma_b^2 \mathbf{u}_i^\top \mathbf{u}_j + \sigma_b^4$$

Le calcul de la variance espérée se fait tel que :

$$\begin{aligned}
K_{ii} &= \int k_{qua}(\mathbf{x}_i, \mathbf{x}_i)\mathcal{N}_{\mathbf{x}_i}(\mathbf{u}_i, \mathbf{\Sigma}_i)d\mathbf{x}_i \\
&= \mathbb{E}_i\left[\mathbf{x}_i^\top \mathbf{x}_i \mathbf{x}_i^\top \mathbf{x}_i + 2\sigma_b^2 \mathbf{x}_i^\top \mathbf{x}_i + \sigma_b^4\right] \\
&= \mathbb{E}_i\left[\mathbf{x}_i^\top \mathbf{x}_i \mathbf{x}_i^\top \mathbf{x}_i\right] + 2\sigma_b^2\ \mathbb{E}_i\left[\mathbf{x}_i^\top \mathbf{x}_i\right] + \sigma_b^4 \\
&= \mathbb{E}_i\left[\mathbf{x}_i^\top \mathbf{x}_i \mathbf{x}_i^\top \mathbf{x}_i\right] + 2\sigma_b^2\ \mathrm{Tr}\left(\mathbf{\Sigma}_i + \mathbf{u}_i\mathbf{u}_i^\top\right) + \sigma_b^4 \\
&= 2\mathrm{Tr}\left(\mathbf{\Sigma}_i^2\right) + 4\mathbf{u}_i^\top \mathbf{\Sigma}_i \mathbf{u}_i + \mathrm{Tr}\left([\mathbf{\Sigma}_i + \mathbf{u}_i\mathbf{u}_i^\top]\right)^2 \\
&\quad + 2\sigma_b^2\ \mathrm{Tr}\left(\mathbf{\Sigma}_i + \mathbf{u}_i\mathbf{u}_i^\top\right) + \sigma_b^4
\end{aligned}$$

En observant les différences entre les équations de variance et de covariance, nous pouvons définir la fonction d'espérance de covariance quadratique par :

$$\begin{aligned}
&k_{\mathbb{E}qua}((\mathbf{u}_i, \mathbf{\Sigma}_i), (\mathbf{u}_j, \mathbf{\Sigma}_j)) \\
&\qquad = \mathrm{Tr}\left([\mathbf{\Sigma}_i + \mathbf{u}_i\mathbf{u}_i^\top][\mathbf{\Sigma}_j + \mathbf{u}_j\mathbf{u}_j^\top]\right)(1 + \delta_{ij}) \\
&\qquad + 2\sigma_b^2 \mathbf{u}_i^\top \mathbf{u}_j + \sigma_b^4 \\
&\qquad + \delta_{ij}\left[\mathrm{Tr}\left(\mathbf{\Sigma}_i + \mathbf{u}_i\mathbf{u}_i^\top\right)^2 + 2\sigma_b^2 \mathrm{Tr}\left(\mathbf{\Sigma}_i\right) - 2(\mathbf{u}_i^\top \mathbf{u}_i)^2\right]
\end{aligned}$$

en introduisant un delta de Kronecker d'activant lorsqu'il faut calculer une variance.

B.3 Fonction de covariance carré-exponentielle espérée

L'espérance de la fonction de covariance carré-exponentielle par rapport à des entrées incertaines, dont les distributions sont $p(\mathbf{x}_i) = \mathcal{N}_{\mathbf{x}_i}(\mathbf{u}_i, \mathbf{\Sigma}_i)$ et $p(\mathbf{x}_j) = \mathcal{N}_{\mathbf{x}_j}(\mathbf{u}_j, \mathbf{\Sigma}_j)$, est donnée par :

$$K_{ij} = \iint k_{CE}(\mathbf{x}_i, \mathbf{x}_j)\mathcal{N}_{\mathbf{x}_i}(\mathbf{u}_i, \mathbf{\Sigma}_i)\mathcal{N}_{\mathbf{x}_j}(\mathbf{u}_j, \mathbf{\Sigma}_j)d\mathbf{x}_i d\mathbf{x}_j$$

En utilisant l'interprétation Gaussienne de la fonction de covariance carré-exponentielle, la fonction $k_{CE}(\mathbf{x}_i, \mathbf{x}_j)$ peut se réécrire comme une distribution Gaussienne normalisée $c\mathcal{N}_{\mathbf{x}_i}(\mathbf{x}_j, \mathbf{W})$, où $c = \sigma_f^2(2\pi)^{D/2}|\mathbf{W}|^{1/2}$ est la constante de normalisation :

$$K_{ij} = c\iint \mathcal{N}_{\mathbf{x}_i}(\mathbf{x}_j, \mathbf{W})\mathcal{N}_{\mathbf{x}_i}(\mathbf{u}_i, \mathbf{\Sigma}_i)\mathcal{N}_{\mathbf{x}_j}(\mathbf{u}_j, \mathbf{\Sigma}_j)d\mathbf{x}_i d\mathbf{x}_j$$

D'après la règle du produit de Gaussienne, le résultat est à nouveau une Gaussienne, mais cette fois non-normalisée dû au facteur Z^{-1} :

$$K_{ij} = c \iint Z^{-1} \mathcal{N}_{\mathbf{x}_i}(\mathbf{a}, \mathbf{B}) \mathcal{N}_{\mathbf{x}_j}(\mathbf{u}_j, \mathbf{\Sigma}_j) d\mathbf{x}_i d\mathbf{x}_j$$

où la Gaussienne de moyenne $\mathbf{a}$ et de matrice de covariance $\mathbf{B}$ sera éliminée par intégration. Il se trouve que le facteur multiplicatif Z^{-1} de cette Gaussienne est de forme Gaussienne tel que $Z^{-1} = \mathcal{N}_{\mathbf{x}_j}(\mathbf{u}_i, \mathbf{W} + \mathbf{\Sigma}_i)$. Par conséquent, l'intégration sur la variable $\mathbf{x}_i$ donne :

$$K_{ij} = c \int \mathcal{N}_{\mathbf{x}_j}(\mathbf{u}_i, \mathbf{W} + \mathbf{\Sigma}_i) \mathcal{N}_{\mathbf{x}_j}(\mathbf{u}_j, \mathbf{\Sigma}_j) d\mathbf{x}_j$$

Pour ce qui est de la seconde variable, c-à-d. $\mathbf{x}_j$, nous utilisons le même raisonnement. Ainsi, en appliquant successivement la règle du produit de Gaussiennes et en intégrant cette fois sur $\mathbf{x}_j$, nous obtenons l'espérance de la covariance :

$$\begin{aligned} K_{ij} &= c\mathcal{N}_{\mathbf{u}_i}(\mathbf{u}_j, \mathbf{W} + \mathbf{\Sigma}_i + \mathbf{\Sigma}_j) \\ &= \sigma_f^2 (2\pi)^{D/2} |\mathbf{W}|^{1/2} \mathcal{N}_{\mathbf{u}_i}(\mathbf{u}_j, \mathbf{W} + \mathbf{\Sigma}_i + \mathbf{\Sigma}_j) \\ &= \frac{\sigma_f^2 \exp((\mathbf{u}_i - \mathbf{u}_j)^\top (W + \mathbf{\Sigma}_i + \mathbf{\Sigma}_j)^{-1} (\mathbf{u}_i - \mathbf{u}_j))}{|I + W^{-1}(\mathbf{\Sigma}_i + \mathbf{\Sigma}_j)|^{\frac{1}{2}}} \end{aligned}$$

Pour ce qui est de l'espérance de la variance, celle-ci est obtenue aisément. D'abord, nous avons :

$$K_{ii} = \int k_{CE}(\mathbf{x}_i, \mathbf{x}_i) \mathcal{N}_{\mathbf{x}_i}(\mathbf{u}_i, \mathbf{\Sigma}_i) d\mathbf{x}_i$$

Par définition d'une fonction de covariance stationnaire, $k(\mathbf{x}, \mathbf{x})$ est une constante, ce qui mène directement à :

$$K_{ii} = k_{CE}(\mathbf{x}_i, \mathbf{x}_i)$$

Par ailleurs, d'après la définition de fonction de covariance carré-exponentielle, nous savons que $k_{CE}(\mathbf{x}_i, \mathbf{x}_i) = \sigma_f^2$, et donc que la variance espérée est simplement σ_f^2.

Finalement, il s'agit de combiner les équations de variance et covariance en une seule fonction en introduisant le complément du delta de Kronecker :

$$\begin{aligned} &k_{ECE}((\mathbf{u}_i, \mathbf{\Sigma}_i), (\mathbf{u}_j, \mathbf{\Sigma}_j)) = \\ &\quad \frac{\sigma_f^2 \exp(-\frac{1}{2}(\mathbf{u}_i - \mathbf{u}_j)^\top (W + \mathbf{\Sigma}_i + \mathbf{\Sigma}_j)^{-1} (\mathbf{u}_i - \mathbf{u}_j))}{|I + W^{-1}(\mathbf{\Sigma}_i + \mathbf{\Sigma}_j)(1 - \delta_{ij})|^{\frac{1}{2}}} \end{aligned}$$