INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE

N^o attribué par la bibliothèque
|_|_|_|_|_|_|_|_|_|_|

THÈSE

pour obtenir le grade de

DOCTEUR DE L'INPG

Spécialité : *Imagerie, Vision, Robotique*

préparée au Laboratoire d'Informatique de Grenoble et à l'INRIA Rhône-Alpes, dans le cadre de l'École doctorale *Mathématiques, Sciences et Technologies de l'Information, Informatique*

présentée et soutenue publiquement par

Pierre-Charles DANGAUTHIER

le 18 Décembre 2007

Titre :

Fondations, méthode et applications de l'apprentissage bayésien.

Directeurs de thèse :

Pierre BESSIÈRE

Composition du jury :

M.	Augustin LUX	Président
M.	Roderick EDWARDS	Rapporteur
M.	Philippe LERAY	Rapporteur
M.	Pierre BESSIÈRE	Directeur de thèse
M.	Guillaume BOUCHARD	Examinateur

Remerciements

Je souhaite tout d'abord remercier mon directeur de thèse, Pierre Bessière, pour la confiance et la sympathie qu'il m'a accordées. Merci de m'avoir fait découvrir les probabilités subjectives ; merci de m'avoir guidé, encouragé et conseillé tout au long de ce travail scientifique.

Je remercie aussi chaleureusement les membres du jury pour m'avoir fait l'honneur d'assister à ma soutenance. Merci Rod d'être venu de si loin pour partager votre expertise des classements échiquéens, merci Philippe pour vos intéressants commentaires sur le manuscrit, et merci Guillaume pour avoir relu en détails certains passages mathématiques et pour avoir partagé avec moi votre connaissance du domaine de l'apprentissage statistique.

D'autre part, je tiens tout particulièrement à remercier Ralf Herbrich et Thore Graepel pour m'avoir accueilli au sein de leur équipe de Microsoft Research durant trois mois à Cambridge. Ce fut un séjour extrêmement stimulant au cours duquel j'ai pu acquérir des connaissances nouvelles sur les modèles graphiques et l'inférence approchée. Le moteur de classement TrueChess est issu de cette rencontre ; c'est une extension du modèle TrueSkill développé pour le X-Box Live. Merci à Tom Minka pour EP et merci à Don Syme pour m'avoir initié à son langage F#.

Je n'oublie pas tous les membres de l'équipe E-Motion de Grenoble, qui m'ont accueilli, et stimulé pendant ce travail de thèse. Merci à Christian Laugier pour l'encadrement de cette équipe et à tous les doctorants pour les discussions enrichissantes que nous avons pu partager. Je pense aussi particulièrement à Anne Spalanzani pour sa participation au projet de sélection de capteurs.

Merci à Pau pour les discussions nocturnes sur la métaphysique en général et sur les mathématiques en particulier. Merci pour toutes ses heures de sommeil manquées, spécialement lors de notre travail sur la compétition Netflix.

Merci à mes relecteurs : Maud, Stéphanie, Thomas, Christophe, Amaury et Christopher.

Enfin, j'adresse un grand merci à ma famille et mes amis qui m'ont soutenu durant tout ce temps. Thérèse et Jean-Philippe, cette thèse vous est dédiée.

Résumé

Le domaine de l'apprentissage automatique a pour but la création d'agents synthétiques améliorant leurs performances avec l'expérience. Pour pouvoir se perfectionner, ces agents extraient des régularités statistiques de données incertaines et mettent à jour leur modèle du monde. Les probabilités bayésiennes sont un outil rationnel pour répondre à la problématique de l'apprentissage. Cependant, comme ce problème est souvent difficile, des solutions proposant un compromis entre précision et rapidité doivent être mises en œuvre. Ce travail présente la méthode d'apprentissage bayésien, ses fondations philosophiques et plusieurs applications innovantes.

Nous nous intéressons d'abord à des questions d'apprentissage de paramètres. Dans ce cadre nous étudions deux problèmes d'analyse de données à variables cachées. Nous proposons d'abord une méthode bayésienne pour classer les joueurs d'échecs qui améliore sensiblement le système Elo. Le classement produit permet de répondre à des questions intéressantes comme celle de savoir qui fut le meilleur joueur d'échecs de tous les temps. Nous étudions aussi un système de filtrage collaboratif dont le but est de prévoir les goûts cinématographiques d'utilisateurs en fonction de leurs préférences passées.

La deuxième partie de notre travail concerne l'apprentissage de modèles. D'abord nous nous intéressons à la sélection de variables pertinentes dans le cadre d'une application robotique. D'un point de vue cognitif, cette sélection permet au robot de transférer ses connaissances d'un domaine sensorimoteur vers un autre. Finalement, nous proposons une méthode permettant de découvrir automatiquement une nouvelle variable cachée afin de mieux modéliser l'environnement d'un robot.

Table des matières

Notations

Symboles

$:=$	Par définition.
$\leftarrow$	Écrasement de la valeur d'une variable dans un algorithme.
$\sim$	Déclaration de la distribution d'une variable aléatoire.
$\propto$	Proportionnalité.
$\approx$	Approximation.

Variables

$\{x \in \Omega \mid Q(x)\}$	Ensemble des éléments de Ω vérifiant la propriété Q.
$\complement_\Omega^A$	Complémentaire de A dans Ω : $\{x \in \Omega \mid x \notin A\}$
$x \in \mathbb{R}$	Scalaire ou variable réelle.
$\mathbf{x} \in \mathbb{R}^d$	Vecteur colonne de dimension finie d.
$\mathbf{x}^{\setminus k} \in \mathbb{R}^{d-1}$	Vecteur $\mathbf{x}$ privé de sa $k^{\text{ème}}$ composante.
$\mathbf{x}_a$	Sous-ensemble de variables de $\mathbf{x}$ dépendant de l'indice a.
$\mathbf{x}^\top$	Vecteur ligne, transposé de $\mathbf{x}$.
x_i	Composante i du vecteur $\mathbf{x}$ ou élément d'indexe i de $\{x_j \mid j \in I\}$.
$\{x_i\} = \{x_i\}_{i \in I}$	$\{x_i \mid i \in I\}$.
X	Variable aléatoire X.
$\mathbf{X}$	Vecteur de d variables aléatoires.

Graphes

$G := (V, E)$	Graphe de nœuds V et d'arêtes E.
$N_x^G = N_x$	Voisins du nœud x dans un graphe G.
$N_x^G \setminus y = N_x \setminus y$	Voisins du nœud x différents de y.
$f_a(\mathbf{x}_a)$	Nœud de type facteur dans un graphe de facteurs. Les variables liées à f_a sont les $\mathbf{x}_a \subset \mathbf{x}$.
x_k	Nœud de type variable dans un graphe de facteurs.
$m_{x_k \to f_a}(x_k) = m_{k \to a}(x_k)$	Message de la variable x_k vers le facteur f_a.
$m_{f_a \to x_k}(x_k) = m_{a \to k}(x_k)$	Message du facteur f_a vers la variable x_k.

Probabilités

$P(\mathbf{x}) := P(\mathbf{X} = \mathbf{x})$	Probabilité de la réalisation $\mathbf{X} = \mathbf{x}$.
$p(\mathbf{x})$	Densité de probabilité de $\mathbf{X}$ en $\mathbf{x}$, distribution de $\mathbf{X}$.
$p^{\backslash k}(\mathbf{x}^{\backslash k})$	Distribution des variables $\mathbf{X}^{\backslash k}$.
$\hat{p}, q$	Approximation d'une distribution p.
$E_p[f(\mathbf{X})]$	Espérance de $f(\mathbf{X})$ sous la distribution p.
	$E_p[f(\mathbf{X})] := \int_\Omega p(\mathbf{x})\, f(\mathbf{x})\, d\mathbf{x}$.
$\mathrm{VAR}_p[\mathbf{X}]$	Variance.
	$\mathrm{VAR}_p[\mathbf{X}] := E_p[\boldsymbol{X}^2] - E_p^2[\mathbf{X}]$.
$\mathcal{N}(x; \mu, \sigma^2)$	Distribution gaussienne (dimension 1).
$\mathcal{N}(x)$	Distribution gaussienne centrée réduite.
$\mathcal{G}(x; \tau, \pi) = [\tau, \pi]$	Distribution gaussienne, paramétrisation canonique ($d = 1$).
$\delta_A(x)$	Distribution de Dirac : $\forall x \in A, p(x) = 1;\ \forall x \in \complement_\Omega^A, p(x) = 0$.
$D_{\mathrm{KL}}(\,.\,\|\|\,.\,)$	Divergence de Kullback-Leibler.
$D_\alpha(\,.\,\|\|\,.\,)$	α-divergence entre deux distributions.

Modèles

Π	Variable représentant toutes les connaissances préliminaires.
$\mathcal{M}$	Modèle.
$\Delta = \{\delta_i\}$	Variable représentant les données observées.
$\Theta = \{\theta_i\}$	Variable représentant un vecteur de paramètres.
$\mathcal{H}$	Variables cachées.
$S \subset \mathcal{H}$	Variables cachées intéressantes.
$M \subset \mathcal{H}$	Variables cachées non intéressantes (*Marginalisées*).

Conventions

Tout au long de ce document nous opterons ainsi pour les conventions suivantes : x est une valeur numérique observée (réelle ou discrète), X une variable aléatoire, $\mathbf{x}$ un vecteur de valeurs et $\mathbf{X}$ un vecteur de variables aléatoires. Cependant, pour améliorer la lisibilité des formules et être cohérent avec la littérature existante, nous dérogerons parfois à ces définitions. Le contexte levera d'éventuelles ambiguïtés.

Introduction

Contexte : Apprentissage

Le domaine de l'*apprentissage automatique* a pour but la création d'agents synthétiques ou de programmes informatiques qui améliorent leurs performances avec l'expérience. Pour pouvoir se perfectionner, ces programmes doivent être capables de trouver des schémas récurrents (*patterns*), des régularités statistiques dans les données à leur disposition.

Ce problème de recherche et d'extraction de régularités est fondamental, car il est au cœur même de la démarche scientifique. Par exemple, J. Kepler (1571-1630) a découvert les lois empiriques décrivant les mouvements célestes à partir des nombreuses observations astronomiques de Tycho Brahe (1546-1601). Plus tard, Newton (1643-1727), en proposant sa théorie de la gravitation, a permis d'unifier le mouvement des astres et la chute des corps sous un même principe. De même, la découverte de régularités dans les spectres atomiques au début du vingtième siècle participa au développement de la mécanique quantique. La recherche de symétries, d'invariances, d'analogies, de redondances est une composante fondamentale du progrès scientifique. Une bonne compréhension, une bonne modélisation du monde permet de prédire le futur, donc de prendre de bonnes décisions et d'agir de façon pertinente.

L'apprentissage automatique est la traduction expérimentale de ce principe de recherche de schémas. Son but est de rendre un agent plus performant dans son environnement et d'extraire de la connaissance à partir de données brutes. Ces dernières décennies, les techniques d'apprentissage ont conduit à des avancées décisives dans de nombreux domaines. Nous pouvons citer par exemple la vision par ordinateur, la reconnaissance d'écriture, la traduction automatique, la robotique autonome, l'intelligence artificielle, la recherche d'information sur internet, la bioinformatique et la médecine expérimentale.

Problématique : Modèle et incertitude

Dans tous ces domaines, les méthodes d'apprentissage doivent répondre à deux difficultés principales : la nécessité d'un *modèle* préalable et l'*incertitude* présente dans les données.

En effet nous devons d'abord noter qu'un agent ne peut pas apprendre s'il ne possède pas une connaissance minimale du phénomène qu'il étudie. Trivialement, il faut qu'il soit

capable de donner une sémantique aux données numériques reçues, ne serait-ce que pour comprendre qu'elles concernent toutes le même phénomène. Si ces données concernent plusieurs aspects d'un même phénomène, l'agent doit les représenter avec autant de variables différentes. Il doit de plus être capable de structurer ces variables. L'agent utilise alors ce modèle pour intégrer les nouvelles données et mettre à jour ses connaissances.

La seconde difficulté provient de l'imperfection de l'interface entre l'agent et le monde : il est extêmement rare que les données récoltées ne soient pas bruitées. Dans la plupart des cas, les données sont une représentation approximative du réel : soit elles sont acquises par l'intermédiaire de capteurs bruités qui donnent une image déformée du phénomène ; soit certaines caractéristiques de ce phénomène ne sont pas directement observables et doivent être inférées de façon approchée.

Ainsi, tout mécanisme d'apprentissage, qu'il soit biologique ou artificiel, doit répondre à ces deux difficultés : il doit intégrer un dispositif de modélisation du problème et un mécanisme de traitement des incertitudes.

Approche : Méthode bayésienne

La méthode d'apprentissage bayésien, que nous étudions dans ce document, permet de répondre d'une façon rationnelle à ces deux difficultés. Pour cela, elle utilise la notion de probabilité, non pas comme une limite de fréquences objectives, mais pour modéliser les degrés de croyance subjectifs de l'agent.

Durant les vingt dernières années, cette approche de l'apprentissage a pris beaucoup d'importance du point de vue théorique et du point de vue des applications pratiques. Ceci peut être expliqué par trois raisons principales (Bis06) :

- ses qualités fondamentales de rigueur et de cohérence,
- l'apparition de modèles graphiques agréables et pratiques à manipuler,
- et la découverte de nouveaux algorithmes d'inférence approchée efficaces et précis.

Pour illustrer cette méthode, considérons que le problème soit de prévoir un futur lancer d'une pièce de monnaie.

Exemple 1. *Vous lancez cinq fois une pièce de monnaie et, surprise, vous obtenez cinq "Piles". Que ce passera t-il au sixième lancer ?*

En amateur de casino vous pourriez vous dire que "pour se rattraper" la pièce fera certainement un "Face". Ou peut-être que vous avez un peu étudié la théorie des probabilités et, comme les lancers sont indépendants, vous concluez qu'il y a une chance sur deux de refaire "Pile".

Mais n'est-il pas étonnant qu'une pièce donne cinq fois de suite la même chose ? Ne serait elle pas truquée ? Elle pourrait par exemple avoir deux cotés "Pile". A priori, vous pensiez que la pièce était équilibrée, mais ces données étonnantes devraient vous faire changer d'avis, du moins en partie...

La méthode d'apprentissage bayésien est particulièrement bien adaptée à ce genre de problème. Elle préconise une approche en trois étapes.

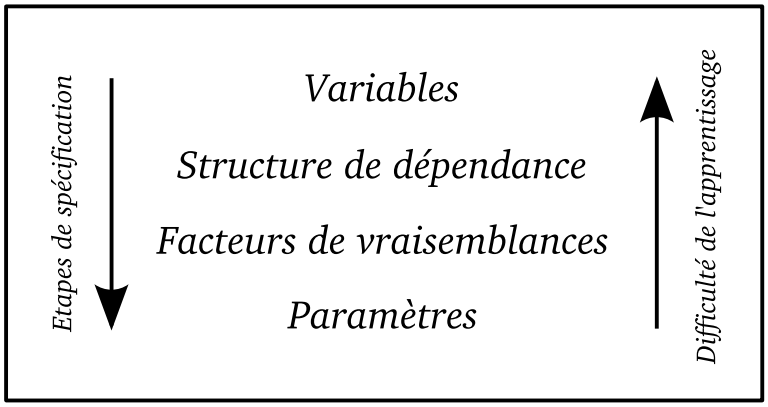


FIG. 1 – Évolutions opposées des ordres de la spécification manuelle des parties d'un modèle et de la difficulté des problèmes d'apprentissages correspondants.

1. Expression de nos connaissances préliminaires par la définition d'un modèle de la génération des données. Ce modèle est composé de trois sous-parties : un ensemble de *variables* (le biais intrinsèque à la pièce et une variable par lancer), une *structure de dépendance* entre ces variables (conditionnellement au biais, les lancers sont indépendants entre eux) et un ensemble de *facteurs* numériques donnant la vraisemblance des données sachant le biais.

2. Intégration des données observées pour la mise à jour de ce modèle (calcul des *paramètres* des probabilités *a posteriori* sur le biais).

3. Utilisation du nouveau modèle pour proposer une prédiction, accompagnée d'un indicateur de confiance sur sa justesse.

Ce problème d'apprentissage est très simple car il comporte peu de variables, car les calculs sont analytiques et car le but n'est que de calculer la distribution prédictive sur une des variables du modèle (le prochain lancer). D'autres problèmes d'apprentissage sont plus difficiles, en raison d'un nombre de variables beaucoup plus grand, de la présence de variables cachées "de nuisance", ou de l'impossibilité de faire des inférence exactes.

Le cadre bayésien permet aussi de définir des problèmes d'inférence de niveau supérieur : il ne s'agit plus d'apprendre un paramètre d'un modèle, mais d'apprendre le modèle lui-même, ou certaines de ses sous-parties, à partir des données.

La figure 1 classe ces différents problèmes d'apprentissage. La flèche de gauche indique l'ordre des étapes lors de la définition à la main d'un modèle. La flèche de droite représente la difficulté des problèmes d'apprentissage correspondants : il est plus facile d'apprendre un paramètre que d'apprendre la structure de dépendance, et il est encore plus difficile de trouver les variables pertinentes. Dans ce document, nous présentons des méthodes tentant d'apprendre des paramètres, des dépendances et des variables.

Plan de lecture

Notre document est organisé dans le sens de la flèche de droite de la figure 1. Nous commençons par présenter des algorithmes d'apprentissage de paramètre pour finir par des méthodes de découvertes de variables. Nous utilisons ce plan pour suivre la progression logique de la difficulté des problèmes. Ce manuscrit se divise en trois grandes parties.

Partie 1

Pour commencer, nous revenons sur les fondations mathématiques, philosophiques et sémantiques de l'approche bayésienne. Nous présentons les détails de cette méthode et précisons ses avantages et inconvénients lors de son application à l'apprentissage.

Partie 2

Dans une seconde partie, nous nous intéressons à l'apprentissage de paramètres lorsque le modèle est bien défini. Nous étudions pour cela deux problèmes d'analyse de données avec des variables cachées. Nous en donnons les détails dans la table ci-dessous.

	Chapitre 3	Chapitre 4
Problématique	Cotation, classement	Filtrage collaboratif
Application	Joueurs d'échecs	Goûts cinématographiques
Taille des données	3.5 millions de parties	100 millions de notes
Modèle utilisé	Variables cachées explicites	Variables cachées introduites
Inférence	Passage de message EP	EM + *Mean Field*
Temps de calcul	20 minutes	Une semaine
Résultats produits	Classement historiques	*Clustering* + prédictions

Partie 3

La dernière partie concerne le problème plus délicat de l'apprentissage de sous-parties du modèle, en particulier des variables. Nous considérons au chapitre 5 le problème de la sélection de variables pertinentes pour créer un modèle utile pour une tâche robotique. Enfin le chapitre 6 propose une méthode permettant de découvrir automatiquement une nouvelle variable cachée, afin de mieux modéliser l'environnement d'un robot virtuel.

	Chapitre 5	Chapitre 6
Problématique	Sélection de variables	Découverte de variable
Application	Robotique réelle	Robotique simulée
Taille des données	100 000	10 000
Modèle utilisé	Comparaisons d'algo.	Comparaison entropique de structures
Temps de calcul	10 minutes	20 minutes
Résultats produits	Changement de modalité	*Clustering* spatial

Première partie

Méthodologie de l'apprentissage bayésien

Chapitre 1

Méthodologie de l'apprentissage bayésien

> The Bayesian revolution in the sciences is fueled, not only by more and more cognitive scientists suddenly noticing that mental phenomena have Bayesian structure in them ; not only by scientists in every field learning to judge their statistical methods by comparison with the Bayesian method ; but also by the idea that science itself is a special case of Bayes Theorem ; experimental evidence is Bayesian evidence.
> - Eliezer Yudkowsky -

Cette première partie présente l'apprentissage bayésien. Nous décrivons d'abord la méthode statistique bayésienne dans sa généralité puis nous étudions son application à l'apprentissage.

1.1 Introduction

La méthode bayésienne est un ensemble de techniques statistiques utilisées pour modéliser des problèmes, extraire de l'information de données brutes et prendre des décisions de façon cohérente et rationnelle. Son cadre d'application est général, mais ses avantages sont déterminants lorsque l'information disponible est incertaine ou incomplète.

Bien que les premiers travaux d'inspiration bayésienne datent du XVII$^{\text{ème}}$ siècle, cette méthode connaît un regain de popularité depuis quelques décennies. Ce renouveau est sensible dans des domaines très variés, en partie grâce à la disponibilité de calculateurs puissants, mais aussi à cause d'une évolution de la pensée statistique et des problèmes abordés.

Cette méthode est devenue un outil important pour l'analyse de données expérimentales et la présentation de résultats scientifiques. Un exemple récent est celui de la *Food and Drug Administration* des États-Unis, une autorité en matière de méthodologie statistique. Cette agence a récemment publié des recommandations concernant l'introduction de procédures bayésiennes pour l'étude de l'efficacité de traitements médicaux (Cam06). Les sciences

sociales et politiques en sont aussi un domaine d'application, car les données y sont rares et coûteuses à collecter. Le livre de Gelman (GCSR03) fait référence dans ce domaine. Les sciences exactes ne sont pas non plus épargnées (Dos03), par exemple la physique des particules (Cou95 ; Dem06), la thermodynamique (CKHO98), la mécanique statistique (Jay57), la chimie (VEW93 ; PD06), la génétique (Smy04 ; CAH06) et la bioinformatique (Wil07).

La mécanique quantique joue un rôle particulier. En tant que science expérimentale, elle peut se prêter à une analyse bayésienne des résultats. Elle pose cependant des questions plus fondamentales sur la notion de probabilité, car selon l'interprétation de Copenhague, le monde quantique est intrinsèquement créateur d'aléa objectif. Un courant récent, initié par Jaynes (Jay90) et suivi par, entre autres, Baez, Caves et Fuchs (Bae03 ; CFS06), propose un point de vue subjectif sur ce hasard quantique : la superposition d'états ne serait que le reflet des degrés de croyance du scientifique[1]. Certains auteurs vont même jusqu'à proposer qu'une mécanique quantique bayésienne "augmentée" ne violerait pas les inégalités de Bell (You01). Cette approche, qui introduit des probabilités complexes, ne fait cependant pas l'unanimité.

La méthode bayésienne est aussi appliquée aux sciences cognitives, pour modéliser les comportement animaux et humains comme des prises de décisions rationnelles (Kor04 ; YSX94). Les neurosciences computationnelles ont pour but de comprendre le fonctionnement des neurones en tant que systèmes de traitement de l'information optimaux. L'approche bayésienne y est aussi prometteuse (PDZ03 ; WCNiA03 ; Den05).

En intelligence artificielle, la proposition de Bessière d'une théorie probabiliste des systèmes cognitifs sensori-moteurs (BDL+98a ; BDL+98b) a conduit à une méthode de programmation bayésienne des robots (LBDM03). Cette méthode a de nombreuses applications en robotique autonome (PHK+05 ; Koi05 ; CPL+06 ; DBM05 ; DBS05) et pour la programmation de personnages virtuels (LHABL04). De nombreux autres travaux utilisent les probabilités pour représenter explicitement l'incertitude en robotique, en particulier ceux de Fox, Burgard et Thrun (TBF05). Nous pouvons aussi citer des recherches en reconnaissance de forme et traitement des langues naturelles (Man99), en vision par ordinateur (ORP00) et en reconstruction 3D (GS05).

Tous ces travaux reposent sur la contribution fondamentale de Jaynes résumée dans son livre posthume *Probability Theory : The Logic of Science* (Jay03). Jaynes y étend la logique booléenne à des valeurs de vérités non entières grâce à l'approche subjective des probabilités.

En effet, la méthode bayésienne se distingue des autres méthodes statistiques par la sémantique qu'elle donne à la notion de probabilité. Dans cette thèse nous considérons qu'une méthode statistique est d'inspiration bayésienne si les probabilités qu'elle manipule sont *subjectives*. À l'opposé, les outils des statistiques classiques (dites orthodoxes ou fréquencistes), comme les estimateurs sans biais et les tests d'hypothèses, considèrent une

[1] Les liens mathématiques entre probabilité et mécanique quantique sont très étroits car, dans leur formulations mathématiques, la théorie des probabilités *est* un cas particulier de la théorie quantique, quand l'algèbre des observables est commutative.

interprétation *objective* des probabilités.

Dans la littérature et parmi les scientifiques, nous trouvons plusieurs interprétations du terme "bayésien" accompagnées d'interrogations récurrentes. Quelles sont les différences entre les approches objectives et subjectives ? Les deux approches ne sont-elles pas les mêmes quand la quantité de données tend vers l'infini ? L'objectivisme n'est-il pas un subjectivisme avec des *a priori* uniformes ? Un *a priori* est-il nécessairement subjectif ? Suffit-il d'utiliser des *a priori* pour être bayésien ? Appliquées à un même problème, ces méthodes conduisent-elles à des conclusions différentes ?

Parfois ces questions ne sont que le reflet de problèmes de définition, en particulier du terme "bayésien". Comme nous l'avons précisé plus haut, notre étude de la littérature nous a conduit à conclure que la substantifique mœlle des approches bayésiennes est de type sémantique. Nous proposons ainsi la définition suivante :

Définition 1. *une méthode d'analyse statistique est dite bayésienne si ses procédures découlent d'une interprétation sémantique subjective de la notion de probabilité.*

Nous verrons que des différences importantes existent en théorie comme en pratique entre subjectivisme et fréquencisme et nous espérons que revenir à un niveau sémantique permettra de clarifier le débat. Dans cette optique, ce chapitre débutera par un bref rappel de la définition mathématique d'une probabilité. Ensuite nous verrons les différentes interprétations possibles de cette notion, en particulier l'interprétation subjective. Puis nous étudierons plus en détails les fondations de la méthode bayésienne, en prenant pour cadre un *agent*, immergé dans un monde incertain. La deuxième partie de ce chapitre décrira l'approche bayésienne en elle-même, ses avantages philosophiques, sa méthodologie, ses difficultés et son utilisation pratique.

1.2 Fondations

> Nothing in Nature is random. ... A thing appears random only through the incompleteness of our knowledge.
> - Benedict Spinoza -

L'outil fondamental utilisé par la méthode bayésienne pour traiter l'information est la notion de probabilité. Curieusement, l'étude des probabilités est assez récente en regard de son omniprésence dans les sciences et techniques modernes. Jusqu'au milieu du XVII$^{\text{ème}}$ siècle, le terme *probable*, du latin *probare* (prouver, tester) qualifiait une assertion démontrable, justifiable et n'impliquait pas de notion d'indéterminisme ou d'incertitude.

Les premières études d'évènements non certains furent motivées par les jeux de hasard et de paris. Durant l'été 1654, Pascal utilise des méthodes de dénombrement et la notion d'espérance mathématique pour résoudre le problème de la "partie interrompue" (Pas65) proposé par le Chevalier Méré. À travers leur correspondance, Pascal et Fermat introduisent la notion de probabilité par le dénombrement et popularisent cette approche chez les joueurs de dés et de cartes. Huyguens (1629-1695) puis De Moivre (1667-1754) poursuivent ces travaux et, en 1713, Bernoulli établi la loi des grands nombres (Ber13) qui peut être

interprétée comme un lien entre fréquence empirique et probabilité. Un peu plus tard, le révérend Thomas Bayes (1702-1761) introduit une version (pour un cas particulier) du fameux *théorème de Bayes*, léguant ainsi son nom à une branche entière des statistiques modernes. Il n'est cependant pas certain que T. Bayes se considèrerait aujourd'hui comme bayésien, car son essai (Bay63) ne rentre pas dans des considérations sur l'interprétation des probabilités. C'est Laplace (1749-1827) qui propose en 1814 un énoncé général de ce théorème (Lap14), afin d'établir des raisonnements inductifs.

Entre 1750 et 1820 se met en place la théorie de erreurs visant à produire des méthodes standardisées de traitement d'observation bruitées. Laplace, s'inspirant des travaux de De Moivre, y contribue avec l'introduction en 1810 du théorème central limite et de la densité normale (ou gaussienne). Ces résultats sont liés à la méthode des moindres carrés, publiée indépendamment par Legendre (1805), Robert Adrain (1808) et Carl Friedrich Gauss (1809).

La notion de probabilité était introduite et peu de résultats fondamentaux nouveaux furent obtenus durant le siècle suivant, jusqu'à ce que Ronald Aylmer Fisher publie en 1925 son livre séminal *Statistical Methods for Research Workers* (Fis25). Ce travail, l'un des plus influents du XX$^{\text{ème}}$ siècle, a pour but d'apporter un maximum d'objectivité dans les procédures statistiques, en introduisant les notions de p-valeurs et de tests d'hypothèses. Cette œuvre, complétée par celles de Pearson (régression linéaire, corrélation), de Cramér (borne de Cramér-Rao, 1946) et de Neyman (intervalle de confiance, 1934), constitue la base de la majorité des statistiques utilisées de nos jours.

Cependant, cette approche objective commence à recevoir des critiques dès les années 1950 : un groupe de statisticiens de l'université du Michigan, dirigée Leonard Savage, montre que les p-valeurs peuvent assez facilement surestimer la significativité de résultats et conseillent de revenir à une approche subjective des probabilités. L'expression moderne des statistiques subjectives fut en fait amorcée par Ramsey en 1931 (Ram31) et Bruno de Finetti en 1937 (Fin74). Savage reprit et étendit ces travaux dans son livre de 1954 *The Foundations of Statistics* (Sav54). Plus tard, cette approche subjectiviste fut complétée par Jaynes (1922-1998), Harold Jeffrey (1891-1989), Richard T. Cox (1898-1991) et I. J. Good (1916-). Une étude détaillée de l'histoire du terme "bayésien" a récemment été publiée par Stephen E. Fienberg (Fie05).

1.2.1 Probabilité : Axiomatique

En dépit de tous ces progrès, les probabilités ne bénéficient pas d'une axiomatisation précise avant 1933. Des travaux initiés par Borel (1871-1956) conduiront Andrei Nikolaevich Kolmogorov (1903-1987) à proposer un telle axiomatisation (Kol33). En effet, avant cette contribution, les probabilités étaient définies plus ou moins rigoureusement, souvent sur des ensembles discrets, comme des nombre réels compris entre zéro et un et dont la somme totale est un. Or ce manque de rigueur et de généralité posait des problèmes et pouvait conduire à des paradoxes pour l'étude d'objets plus complexes sur des espaces non discrets ou non bornés, comme par exemple les variables continues ou les fonctions aléatoires.

Nous voulons souligner ici que l'axiomatique décrite ci-dessous ne préjuge pas d'une in-

terprétation sémantique. Une probabilité y est définie d'un façon mathématique abstraite, sans liens avec sa signification pour le monde réel, quelle soit objective ou subjective. Le grand apport de cette définition est de présenter des conditions conduisant à une définition cohérente et générale, respectant en particulier la stabilité par union dénombrable nécessaire pour travailler sur des ensembles infinis. Cette axiomatique et sa reformulation moderne en terme de théorie de la mesure, permettent en outre d'étudier des objets probabilistes mixtes, mêlant variables discrètes et variables continues.

Axiomatique de Kolmogorov

Définition 2. *Soit $\Omega = \{\omega_i \,|\, i \in I\}$ un ensemble, appelé univers des possibles. Les w_i sont des éventualités et une union $A = \bigcup_{j\in J} \{\omega_j\}$ est un évènement. Une probabilité peut être définie sur une famille d'évènements $\mathcal{A}$ si :*

- *$\mathcal{A}$ contient Ω,*
- *$\mathcal{A}$ est une σ-algèbre, c'est à dire que c'est une famille de sous-ensembles de Ω contenant l'ensemble vide, stable par prise du complémentaire et par union dénombrable.*

Définition 3. *Alors une probabilité P sur $\mathcal{A}$ est une fonction associant à chaque $A \in \mathcal{A}$ un nombre réel et vérifiant les propriétés de :*

- ***Normalisation** : $P(\Omega) = 1$,*
- ***Positivité** : $\forall A \in \mathcal{A} \quad P(A) \geq 0$,*
- ***Additivité** : si $\{A_i\}$ est une famille d'événements 2 à 2 incompatibles, alors $P(\cup_i A_i) = \sum_i P(A_i)$.*

En termes modernes, cette axiomatique se place dans le cadre de la théorie de la mesure et une probabilité est le troisième élément d'un espace probabilisé $(\Omega, \mathcal{A}, P)$ avec $\mathcal{A}$ une σ-algèbre contenant Ω et P une mesure telle que $P(\Omega) = 1$.

Dans ce cadre, le concept de *variable aléatoire* est introduit, qui n'a pour l'instant d'aléatoire que le nom, afin de pouvoir parler de probabilité sur des espaces T autres que l'espace abstrait Ω.

Définition 4. *Une variable aléatoire X est une fonction d'un espace probabilisé $(\Omega, \mathcal{A}, P)$ vers un espace T muni d'une algèbre d'évènement $\mathcal{B}$ telle que*

$$\forall B \in \mathcal{B} \quad X^{-1}(B) \in \mathcal{A}. \tag{1.1}$$

Par exemple, si $T = \mathbb{R}$, la signification de $P(a \leq X \leq b)$, avec $a, b \in \mathbb{R}$ est :

$$P(a \leq X \leq b) := P(X^{-1}([a, b])) = P(\ \{\omega \in \Omega \,|\, X(\omega) \in [a, b]\}\ ). \tag{1.2}$$

Ainsi, pour chaque probabilité sur Ω, une variable aléatoire X définit une probabilité sur T. Il est alors usuel d'oublier Ω et de ne parler que la probabilité induite sur T, qui est simplement une mesure normalisée. Une variable aléatoire est alors un outil mathématique simplifiant les notations, permettant de parler de $P(X = x)$ comme d'une fonction de $x \in T$, souvent simplement notée $P(x)$. Il est aussi possible de considérer un vecteur de variables aléatoires $\mathbf{X} = \{X_1, \cdots, X_d\}$ et la probabilité qu'il induit sur un T : $P(\mathbf{X} = \mathbf{x})$.

Quand T est un espace "continu" vérifiant certaines conditions (théorème de Radon-Nikodym), il est possible de définir une densité de probabilité $p(\mathbf{x})$. Cette densité représente la masse infinitésimale de probabilité présente autour de $\mathbf{x}$.

Définition 5. *La fonction de répartition de $P(\mathbf{X})$ est $F(\mathbf{t}) := P(X_1 \leq t_1, \cdots, X_d \leq t_d)$.*

Définition 6. *Si elle existe, la densité $p(\mathbf{x})$ est la dérivée de la fonction de répartition selon toutes ses coordonnées :*

$$p(\mathbf{t}) := \frac{\partial^d}{\partial t_1 \cdots \partial t_d} F(\mathbf{t}). \tag{1.3}$$

Par exemple si $T = \mathbb{R}$, $F(t) := P(X \leq t)$ et $p := F'$.

Il est alors usuel de travailler avec p au lieu de P, mais il est important de se souvenir que ce sont deux objets différents. En particulier une densité est une distribution, au sens de Schwartz. Par exemple la densité de Dirac n'est pas régulière et ne peut pas être associée à une fonction. D'autre part, ce qui nous intéresse au final est la probabilité d'un évènement, intégration de la distribution sur un sous-ensemble de T. De façon générale, la maximisation d'une densité n'a pas de sens et peut conduire à de mauvais résultats (sur-apprentissage).

Relation avec *Probabilty as Logic*

Jaynes ne base pas sa théorie des probabilités sur cette axiomatique, mais considère les probabilités comme une extension de la logique. En supposant qu'un agent représente ses degré de croyance par des nombres réels, il formule un certain nombre de desiderata assurant la *consistance* des raisonnements de l'agent et en déduit les propriétés nécessairement vérifiées par ces nombres réels. Or, il se trouve que ces propriétés sont très similaires à celles de Kolmogorov :

> [Kolmogorov] approach could hardly be more different from ours in general viewpoint and motivation ; yet the final results are identical in several respects.[...]
>
> For all practical purposes, then, our system will agree with [Kolmogorov's] if we are applying it in the set-theory context.[...]
>
> In summary, we see no substential conflict between our system of probability and Kolmogorov's as far as it goes ; rather, we have sought a deeper conceptual fondation which allows it to be extented to a wider calls of applications [...]

Cette classe plus large d'application n'est pas très importante pour notre travail, car dans la majorité des cas et en particulier dans cette thèse, les problèmes envisagés seront explicitement dotés d'une structure ensembliste, avec un T bien défini. De plus, Jaynes, suivant les recommandations de De Finetti, prend beaucoup de précautions avec les ensembles infinis. Sa théorie est basée sur des ensembles dénombrables finis et l'introduction d'infinis y est traitée par un attentif passage à la limite, sans quoi des inconsistances et paradoxes peuvent apparaître.

> Our self imposed inhibition of considering only finite sets and their well behaved limits enables us to avoid all of the useless and unnecessary paradoxing that has appeared in the recent statistical literature.

Kolmogorov ne fait pas de distinction entre ensembles finis et infinis car ces deux types sont unis par la théorie de la mesure, mais lors d'une interprétation bayésienne de l'axiomatique, il faudra aussi prendre garde aux cas pathologiques entraînant ces paradoxes. Néanmoins nous ne rencontrerons pas de tels cas dans cette thèse. De plus, Jaynes écrit à propos de ces paradoxes :

> But even here, when we consider the so-called "Borel-Kolmogorov Paradox", we found ourselves in agreement with Kolmogorov's resolution of it[...]
>
> But on this issue, too, we are not fanatics. We recognize that the language of set and measure theory was a useful development in terminology [...]

La différence principale entre ces deux constructions est le fait que Jaynes ancre sa théorie dans le monde, il donne une sémantique aux probabilités, alors que l'axiomatique de Kolmogorov est abstraite et ne propose pas de liens avec le réel. Ce point nous pousse à considérer l'axiomatique de Kolmogorov pour pouvoir étudier plus sereinement les différentes interprétations de la notion de probabilité (Haj03).

1.2.2 Probabilité : Sémantique

L'histoire des probabilités nous a déjà permis de dégager deux écoles différentes : l'objectivisme à la Fisher et le subjectivisme de Ramsey et Jaynes. Nous commentons succinctement ces courants ainsi que ceux des approches classique, logique et de propension, en nous basant sur le travail d'Alan Hájek (Haj03).

Probabilités classiques

Par interprétation classique, nous entendons celle associée aux travaux de Laplace, Pascal, Bernoulli, Huygens et Leibniz dans le contexte des jeux de hasard. Elle est basée sur le principe d'indifférence, proposant qu'en l'absence d'information, les probabilités se partagent équitablement entre les évènements : $P(A) = \frac{\text{Card(A)}}{\text{Card}(\Omega)}$. Ce principe a ensuite été étendu à des ensembles infinis non énumérables grâce au principe de maximum d'entropie[2] (Jay57). De plus, pour établir un pont avec la notion de fréquence empirique, Laplace introduit sa loi de succession $P(\text{succès au tirage } N+1 \mid N \text{ succès consécutifs}) = \frac{N+1}{N+2}$, rendant l'inférence inductive possible.

Cependant, plusieurs reproches peuvent être faits à cette interprétation. D'abord les probabilités classiques ne peuvent être que des nombres rationnels, ce qui pose un problème pour les sciences et en particulier la mécanique quantique, grande amatrice de $\sqrt{2}$. Plus fondamentalement, attribuer des probabilités égales à des évènements "également

[2]Le principe du maximum d'entropie a aussi sa place dans l'approche subjective.

vraisemblables" semble être une procédure circulaire. De plus, en considérant deux paramétrisations distinctes, le principe d'indifférence peut être auto-contradictoire (paradoxe de Bertand (Ber07 ; Bor09)). Finalement, ce principe extrait en quelque sorte de l'information d'un état d'ignorance : si nous ne connaissons rien d'un dé, pourquoi affirmer qu'il est équilibré ?

Probabilités logiques

Cette approche moins répandue, proposée par Johnson, Keynes, Jeffrey et surtout Carnap, étend l'interprétation classique pour permettre une induction rigoureuse, en introduisant une notion de "degré d'implication" apporté par de l'information sur une hypothèse. Elle repose sur un langage logique formel, composé de prédicats, variables, connecteurs et elle autorise des distributions non uniformes. Quelques difficultés apparaissent alors, comme la nécessité de choix arbitraires (comme celui du langage) et la nature purement syntaxique de cette approche.

Probabilités fréquencistes

C'est l'approche dite objective ou orthodoxe, base des travaux de Fisher et enseignée dans toutes les classes de statistiques. Cette interprétation basée sur les fréquences remonte à Venn (1876) qui affirme que la probabilité d'un évènement *est* sa fréquence empirique. Elle est similaire à l'approche classique sauf que l'on compte les réalisations réelles au lieu des réalisations potentielles. Comme cette interprétation fréquenciste stricte échoue pour des évènements non répétables (elle donne $P = 0$ ou $P = 1$), elle a été étendue en considérant que la probabilité est la *limite* de la fréquence quand on répète une infinité de fois *la même* expérience. Immédiatement, cette considération pose des problèmes fondamentaux. Déjà, deux expériences ne peuvent jamais être exactement identiques. De plus, il est toujours possible de réordonner une séquence infinie de résultats afin de faire converger la fréquence vers un nombre rationnel arbitraire entre 0 et 1. Pourquoi un ordre devrait-il être privilégié ? D'autre part, les limites peuvent violer les axiomes de Kolmogorov, en particulier l'additivité dénombrable. Finalement le lien entre probabilité et fréquence est peut-être trop fort. Pour nous, une séquence d'observations nous renseigne sur une séquence de phénomènes et devrait être traitée comme telle, au lieu de conduire à une conclusion définitive sur la probabilité d'un seul phénomène.

Probabilités propensionnistes

Le cadre de la propension (Popper (1959), Hacking (1965), Giere (1973), Gillies (2000)) propose que la probabilité soit une caractéristique intrinsèque du monde. Elle représente une *tendance*, une *disposition* du monde à se comporter d'une certaine façon. Ceci justifie que les fréquences convergent vers les probabilités. Le problème est que la propension n'est pas fondamentalement bien définie, c'est un concept plus métaphysique que scientifique, car on ne voit pas quelle expérience mettre en place pour mesurer la propension d'un évènement et il est donc difficile de vérifier qu'elle respecte l'axiomatique. De plus, la règle

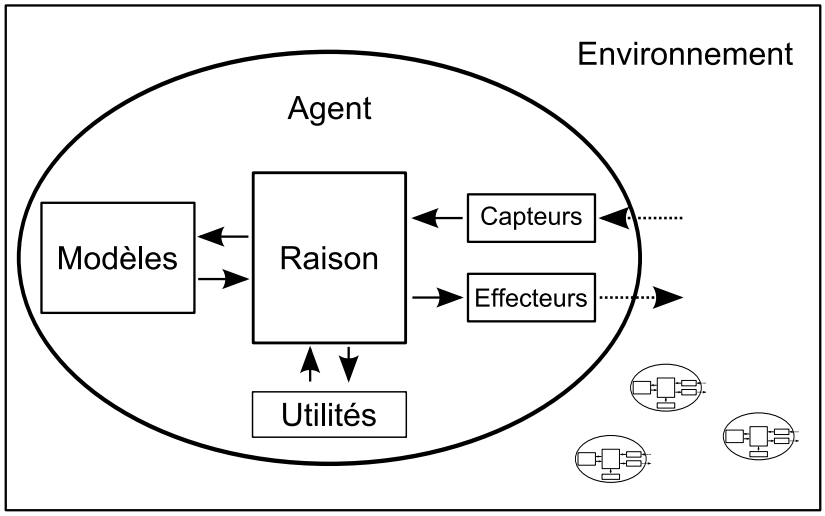


FIG. 1.1 – Représentation schématique d'un agent sensori-moteur inspirée de (Jak04). Cette figure représente l'environnement extérieur comme si il existait d'une façon objective, position qualifiée d'externaliste. Mais même si l'on adopte un point de vue internaliste, l'agent possède capteurs et actionneurs et finalement ce choix philosophique n'a pas d'influence sur son fonctionnement interne.

de Bayes étant symétrique, elle n'est pas compatible avec les propensions car ces dernières impliquent une asymétrie causale forte.

Probabilités subjectives

Comme nous l'avons entrevu précédemment, une probabilité subjective mesure le *degré de croyance d'un agent approprié* dans une proposition, comme par exemple dans la réalisation d'un évènement, futur ou passé. Ce n'est pas une caractéristique du monde mais de sa façon d'interpréter le monde. Notons que, contrairement à l'objectivisme qui postule l'existence de phénomènes aléatoires, le point de vue subjectif ne préjuge pas du déterminisme ou de l'indéterminisme du monde, qui reste une question ouverte.

La figure 1.1 présente un agent immergé dans un environnement. C'est un être qui acquiert des informations à travers des capteurs, fait des modèles et agit en fonction de ce qu'il considère comme utile. L'agent est autonome et éventuellement non réactif : il peut ignorer certaines de ces perceptions et agir ou non. Un modèle est sa représentation interne de l'environnement et de l'agent lui même, c'est ce qu'il pense être vrai. En revanche, l'utilité est une fonction qui permet de juger la valeur d'une action, elle sera nécessaire à la prise de décision. L'utilité, qui peut éventuellement varier au cours du temps, dépend du type d'agent et peut être basée sur des notions de curiosité, de survie, ou de recherche de vérité, d'esthétisme, de simplicité, de nourriture, d'argent...

L'approche subjective postule que l'agent n'est pas certain que le monde soit conforme à ce qu'il en perçoit, que ses capteurs lui en donnent une image incomplète et parfois

entachée d'erreurs. Il représente alors cette incertitude explicitement dans ses modèles, par des *degrés de croyance* numériques. Le *théorème de Cox-Jaynes* (Cox61 ; Jay03) démontre que si un agent respecte quelques desiderata élémentaires de consistance, alors ses degrés de croyance vérifient les axiomes de Kolmogorov. En d'autres termes, ce théorème montre que tout mécanisme d'induction est soit inconsistant, soit isomorphe aux probabilités. Les desiderata que doit respecter l'agent sont des axiomes de consistance. Ils peuvent se résumer informellement par les trois points suivants.

Définition 7. *(Prémisses de Cox-Jaynes)*
- *Représentation des degrés de croyance par des* ***nombres réels***
- ***Adéquation qualitative au sens commun****, si l'information de l'agent est totale, ses raisonnements doivent être isomorphes à la logique classique*
- ***Consistance*** *se traduisant par :*
  - *une consistance interne : si une conclusion peut être obtenue de plusieurs façons, alors chacune de ces façons doit conduire au même résultat,*
  - *une honnêteté intellectuelle : l'agent ne doit pas délibérément ignorer d'informations pertinentes,*
  - *une consistance dite de Jaynes : deux états mentaux identiques doivent être représentés par deux degrés de croyance identiques. Ce dernier point est une généralisation du principe d'indifférence laplacien.*

Le théorème de Cox-Jaynes nous dit donc que pour être consistant, un agent doit manipuler ses degrés de croyance comme des probabilités. Alors son raisonnement ne souffrira pas d'incohérence. *A contrario*, quand des degrés de croyance ne vérifient *pas* les règles de probabilités, il est toujours possible de proposer à un agent un *Dutch book* qu'il acceptera. Un *Dutch book* est un ensemble de paris individuellement favorables, mais dont l'issue globale est fatalement défavorable (Fin74).

Cette approche bayésienne, bien que philosophiquement confortable, présente aussi des difficultés. Le premier problème est la mesure du degré de croyance d'un agent. Une solution opérationnelle serait de lui présenter un évènement non certain et de voir combien il parierait sur telle ou telle réalisation. Son degré de croyance serait alors le prix qui rend le pari équilibré à ses yeux. Mais cette procédure ne nous donne en fait pas un accès direct au degré de croyance, car la prise de décision dépend aussi de l'utilité perçue par l'agent. Nous n'avons donc accès qu'au produit croyance-utilité. Ramsey, Savage et Jeffrey proposent des méthodes pour dériver à la fois croyance et utilité, mais ces méthodes sont sujettes à controverses. Dans cette thèse, les agents sont les programmes que nous écrivons, nous en maîtrisons donc la fonction d'utilité, ce qui nous affranchit de cette limitation. En revanche, d'autres difficultés de la méthode subjective se présenteront à nous, comme la détermination des *a priori*, la traduction de ces *a priori* en distribution de probabilités et surtout, la difficulté des calculs impliqués.

Ayant ainsi présenté les fondations philosophiques et sémantiques de l'approche bayésienne, nous allons détailler cette méthode et son application pratique dans la suite de ce chapitre.

1.3 Méthode bayésienne

> By inference we mean simply : deductive reasoning whenever enough information is at hand to permit it ; inductive or probabilistic reasoning when - as is almost invariably the case in real problems - all the necessary information is not available.
> - E.T. Jaynes -

Nous avons vu que, pour être consistant, un agent possédant une connaissance imparfaite de son environnement doit manipuler ses degrés de croyances comme des probabilités. Le reste de ce chapitre s'attache à détailler la procédure pratique qu'il devra suivre.

Dans le reste de ce document, nous nous intéressons à des problèmes d'analyse de données. Dans ce cadre, l'ontologie de l'agent s'applique de la façon suivante : les données sont collectées par un scientifique, ou statisticien, qui sera modélisé comme un agent. Ainsi c'est notre raisonnement qui devra respecter les règles de consistance bayésiennes.

1.3.1 Règles et méthode

Dans l'approche subjective, notons d'abord que toute probabilité est implicitement *conditionnelle*.

Définition 8. *La probabilité conditionnelle de l'évènement A sachant $B \neq \emptyset$ est :*

$$P(A \mid B) := \frac{P(A \cap B)}{P(B)}. \tag{1.4}$$

En effet, notre degré de croyance est forcément relatif aux informations en notre possession et même si nous l'omettons dans nos notations, toutes les probabilités de cette thèse seront conditionnées par une variable Π : $P(A) := P(A \mid \Pi)$. Cette variable Π représente nos connaissances *a priori* du domaine.

Avec cet outil, les manipulations des degrés de croyances se résument à trois règles de raisonnement, nécessaires et suffisantes pour manipuler tout degré de croyance de façon consistante.

Théorème 1 (De Cox-Jaynes)**.** *Les règles de manipulation des degrés de croyances découlant des prémisses de Cox-Jaynes (def. 7) sont :*

- ***Règle de normalisation*** *:*

$$\sum_{\mathbf{x}} P(\mathbf{x} \mid \Pi) = 1^{3} \tag{1.5}$$

- ***Règle de la somme*** *: (ou de marginalisation)*

$$\sum_{\mathbf{x}} P(\mathbf{x}, \mathbf{y} \mid \Pi) = P(\mathbf{y} \mid \Pi)^{4} \tag{1.6}$$

[3] $\sum_x P(x) := \sum_{i=0}^{n} P(X = x_i)$ et $\int p(x)\, dx := \int_{x=x_{\min}}^{x_{\max}} p(X = x)\, dx$.
[4] $P(\mathbf{x}, \mathbf{y}) = P(\mathbf{y}, \mathbf{x}) := P(\mathbf{X}^{-1}(\mathbf{x}) \cap \mathbf{Y}^{-1}(\mathbf{y}))$.

– ***Règle du produit*** *: (ou du conditionnement)*

$$P(\mathbf{x}, \mathbf{y} \mid \Pi) = P(\mathbf{x} \mid \mathbf{y}, \Pi)\, P(\mathbf{y} \mid \Pi) \tag{1.7}$$

Théorème 2 (De Bayes)**.** *Nous en déduisons le théorème de Bayes, ou de probabilité des causes, pour tout* $\mathbf{y}$ *de probabilité non nulle :*

$$P(\mathbf{x} \mid \mathbf{y}) = \frac{P(\mathbf{y} \mid \mathbf{x})\, P(\mathbf{x})}{P(\mathbf{y})} \tag{1.8}$$

Ces règles définissent la façon de manipuler des probabilités. Elles sont aussi valables pour des densités p. Voyons maintenant comment elles prennent place dans la méthode d'analyse bayésienne.

Méthode 1. *La méthode bayésienne pour résoudre un problème d'analyse de données est décrite informellement par les cinq étapes successives suivantes.*

1. *Formuler notre connaissance a priori du problème à partir de* Π.
   (a) *Établir un modèle* $\mathcal{M}$ *probabiliste de la génération des données.*
   (b) *Si ce modèle possède des paramètres libres* Θ*, traduire nos connaissances en une distribution de probabilités sur ces paramètres.*
2. *Collecter des données* Δ
3. *Utiliser la formule de Bayes (th. 2) pour mettre à jour nos connaissances.*
4. *Utiliser ces nouvelles connaissances et les règles (th. 1) pour*
   (a) *Tirer des conclusions*
   (b) *Faire des prédictions*
   (c) *Prendre des décisions maximisant une utilité espérée*
   (d) *Commencer une autre analyse : cette connaissance a posteriori devient notre nouvelle connaissance a priori.*
5. *Vérifier la pertinence des résultats. Si la vraisemblance du modèle est trop faible, si les performances prédictives sont mauvaises ou si les conclusions sont manifestement fausses, retourner au point 1.*

Nous insistons sur le fait qu'en toute rigueur, le modèle et les distributions *a priori* doivent être établis indépendamment des données et que les *a priori* doivent vraiment représenter un état de connaissances réel de l'agent. À ce stade cette remarque peut paraître superflue, mais de nombreuses approches ne respectent pas cette contrainte, même dans la communauté bayésienne. Par exemple, il est souvent recommandé d'adapter la complexité du modèle à la taille du jeu de données : un modèle avec trop de paramètres libres est considéré inadapté si peu de données sont disponibles. Cette recommandation n'a pourtant pas de justification dans les principes bayésiens. Il n'y a pas de raisons de modifier ses

a priori ou de réduire la complexité du modèle en fonction de la quantité de données collectées. De plus, pour respecter strictement la méthode, la sortie de l'algorithme doit être une distribution *a posteriori* et non une estimation ponctuelle qui, bien que présentant de nombreux intérêts, peut conduire à des mauvaises conclusions. En pratique il ne sera pas toujours possible de respecter scrupuleusement ces contraintes. Il faudra alors prendre garde à ce que les libertés et approximations décidées n'influent pas trop sur les résultats.

1.3.2 Instanciation et vocabulaire

Considérons que nos connaissances préliminaires Π à propos d'un problème nous poussent à considérer un modèle $\mathcal{M}$ avec des paramètres Θ. Supposons que nous soyons sûrs de $\mathcal{M}$ mais que nous voulions obtenir de l'information sur Θ à partir de données observées Δ. La méthode 1 nous recommande de commencer par transcrire notre information *a priori* sur les valeurs possibles de Θ par $P(\Theta \mid \mathcal{M}, \Pi)$. La mise à jour de ces connaissances découle alors automatiquement de la formule de Bayes (en omettant les Π) :

$$P(\Theta \mid \Delta, \mathcal{M}) = \frac{P(\Delta \mid \Theta, \mathcal{M})\ P(\Theta \mid \mathcal{M})}{P(\Delta \mid \mathcal{M})} \tag{1.9}$$

qui s'écrit informellement par

$$a\ posteriori = \frac{\text{vraisemblance} \times a\ priori}{\text{preuve}}. \tag{1.10}$$

L'*a priori* et l'*a posteriori* sont des probabilités sur les variables qui nous intéressent, c'est-à-dire sur Θ. La distribution sur toutes les variables $P(\Theta, \Delta \mid \mathcal{M})$ est appelée la *jointe*.

La fonction de *vraisemblance* $f_{\mathcal{M},\Delta}(\Theta) := P(\Delta \mid \Theta, \mathcal{M})$ n'est pas une probabilité de Θ, mais c'est la probabilité des données sachant Θ. La maximiser donne Θ_{ML}, paramètres de maximum de vraisemblance, les plus vraisemblables selon les données. Cette fonction joue un rôle centrale dans la méthode bayésienne[5]. En effet, l'étape 1.(a) de la méthode 1 correspond à son écriture. Le modèle probabiliste la détermine car il définit la procédure de génération des données quand les paramètres sont connus. C'est en fait par cette fonction que les variables non observées de $\mathcal{M}$ acquièrent leur sémantique, leur lien avec le réel, c'est-à-dire avec Δ.

Le dernier élément, que faute de traduction conventionnelle nous appellerons "preuve" (*evidence*), peut être vu comme une constante de normalisation, assurant $\int P(\Theta|\Delta, \mathcal{M}) d\Theta = 1$. Mais c'est aussi la probabilité des données sous les hypothèses du modèle. En ce sens, c'est la vraisemblance de $\mathcal{M}$ et elle peut servir à comparer des modèles entre eux. La preuve joue aussi un rôle important en physique statistique (*fonction de partition*) car elle contient toute l'information sur tous les états du système. En effet, elle s'exprime

$$Z := P(\Delta \mid \mathcal{M}) = \int P(\Delta \mid \Theta, \mathcal{M})\ P(\Theta \mid \mathcal{M})\, d\Theta \tag{1.11}$$

[5] $f_{\mathcal{M},\Delta}(\Theta)$ est aussi une brique fondamentale de l'analyse fréquencistes.

ce qui rend son calcul pratique aussi dur que n'importe quelle autre inférence dans $\mathcal{M}$. La valeur $P(\Delta \mid \mathcal{M})$ est aussi appelée *vraisemblance marginale* de $\mathcal{M}$, car nous marginalisons tous les Θ. Cette notion de marginalisation, d'intégration sur l'espace des paramètres est un élément clef donnant à la méthode bayésienne des avantages déterminants. Nous y reviendrons plus tard, mais avant, nous allons donner quelques détails sur les différents éléments introduits, en les illustrant par l'exemple du lancer d'une pièce.

1.4 Connaissances *a priori*

La traduction des connaissances préliminaires Π en un modèle $\mathcal{M}$ et des *a priori* est la première étape de la méthode bayésienne. $\mathcal{M}$ et les *a priori* sur Θ constituent alors un modèle probabiliste génératif complet du problème. C'est une étape subjective, ce qui lui est souvent reproché, mais *non arbitraire*, car deux agents possédant le même Π devront en déduire les mêmes *a priori*. En revanche, s'ils ne possèdent pas les mêmes informations, il nous parait cohérent qu'ils ne tirent pas les mêmes conclusions.

La transcription de Π en un modèle devrait être déterministe, mais pour l'instant aucune méthode automatique générale n'a été proposée. En effet les connaissances préliminaires sont informelles et la détermination de $\mathcal{M}$ est un choix parmi un ensemble de modèles acceptables. Ce choix introduit alors une dose d'arbitraire et un besoin d'imagination de la part de l'agent. La détermination du modèle et des *a priori* est une des grandes difficultés de l'approche bayésienne, d'autant plus que la sensibilité des résultats aux choix d'*a priori* peut être grande. Il est donc recommandé de faire ces choix avec soin, d'évaluer la pertinence des résultats qui en découlent et de réviser ces choix s'ils ne représentent pas bien nos *a priori* informels. La procédure de modélisation devient alors itérative. Il est aussi possible d'utiliser les données pour choisir $\mathcal{M}$ et les *a priori*, dans des procédures dites *bayésiennes empiriques* ou de maximum de vraisemblance de *type II*. Notons que ces méthodes violent alors la méthode 1 car elles utilisent les données pour établir les *a priori*.

1.4.1 Choix du modèle $\mathcal{M}$

La transcription de Π commence par la définition d'un modèle $\mathcal{M}$. Un tel modèle peut être divisé en trois parties $\mathcal{M} = \mathcal{M}_v \cup \mathcal{M}_i \cup \mathcal{M}_f$.

$\mathcal{M}_v$: Choix des variables. Il s'agit de choisir les variables aini que leur espace de définition. Les variables seront soit observées (Δ), soit cachées ($\mathcal{H}$), soit des paramètres (Θ). En bayésien, il n'y a pas de différence fondamentale entre variable cachée et paramètre : il s'agit dans les deux cas de variables aléatoires et il est possible de parler de leur probabilité. Au contraire, l'approche fréquenciste se basant sur la notion d'expériences répétées, ne peut pas assigner de probabilité aux paramètres. Ils ont une valeur unique bien définie, mais inconnue. La méthode fréquenciste se base alors sur des estimateurs *ad hoc* et des intervalles de confiance, qui sont eux aléatoires.

$\mathcal{M}_i$ **: Choix des indépendances conditionnelles entre ces variables.** Deux variables sont indépendantes si et seulement si $P(x, y) = P(x)\,P(y)$, c'est-à-dire si $P(x \mid y) = P(x)$, si la connaissance de y n'influe pas sur notre connaissance de x. C'est une hypothèse forte, en pratique comme en théorie, car l'ensemble des distributions jointes factorisables est de mesure nulle dans l'ensemble des distributions[6]. La plupart du temps les indépendances considérées sont des indépendances conditionnelles : $P(x, y) \neq P(x)\ P(y)$ mais $P(x, y \mid z) = P(x \mid z)\,P(y \mid z)$. Pour un problème particulier, l'ensemble des indépendances supposées fait partie du modèle. Nous verrons dans la dernière partie de cette thèse qu'il est aussi possible de l'apprendre.

$\mathcal{M}_f$ **: Choix des facteurs.** $\mathcal{M}$ est un modèle probabiliste représenté par une distribution jointe sur les variables de $\mathcal{M}_v$ et factorisée selon les indépendances de $\mathcal{M}_i$. Chacun de ces facteurs est une distribution de probabilités conditionnelle locale qu'il faut spécifier. Certaines de ces distributions ne seront conditionnées que par Π : ce sont les distributions *a priori*.

Exemple 2. *Considérons l'exemple d'une pièce dont nous ne savons pas si elle est équilibrée ou non, pour laquelle nous allons collecter des résultats de lancers "pile" et "face", que nous exploiterons ensuite. Dans ce cas, les variables correspondant à ce problème sont* $(\mathcal{M}_v)$ *:*

- θ *le biais de la pièce, sa probabilité de faire un pile* $\theta = P(pile)$*. Si la pièce était équilibrée nous aurions par exemple* $\theta = 0.5$*,*
- *une variable* δ_i *(pile ou face) associée à chaque lancer,* $\Delta = \{\delta_1, \cdots, \delta_d\}$*.*

De part notre connaissance usuelle des expériences de lancers de pièces, il est naturel de considérer qu'ils sont indépendants. De plus, les résultats de chacun des lancers n'est influencé que par la caractéristique θ *de la pièce. Ceci nous donne la structure de dépendance* $(\mathcal{M}_i)$ *:*

$$p(\theta, \Delta) = p(\Delta \mid \theta)\,p(\theta) = p(\theta) \prod_{k=1}^{d} p(\delta_k \mid \theta) \tag{1.12}$$

De plus, nos connaissances préalables nous indiquent que tous les lancers sont identiques, c'est à dire que les $p(\delta_k \mid \theta)$ *ont toutes la même forme. Pour traduire* $\mathcal{M}_f$*, nous posons que les facteurs de vraisemblance sont ceux de* d *lois de Bernoulli* $P(\delta = pile \mid \theta) = \theta$ *et* $P(\delta = face \mid \theta) = 1 - \theta$*, donc*

$$p(\theta, \Delta) = p(\theta)\ \theta^{\#pile}\ (1 - \theta)^{\#face}. \tag{1.13}$$

Il ne reste qu'à spécifier l'a priori $p(\theta)$ *pour compléter le modèle.*

1.4.2 Modèles graphiques

Les parties $\mathcal{M}_v$ et $\mathcal{M}_i$ de $\mathcal{M}$ peuvent se représenter par des graphes qui donnent un vision plus claire du modèle, qui permettent d'en déduire des propriétés intéressantes comme

[6] Du moins lorsqu'une telle mesure existe comme dans le cas paramétrique.

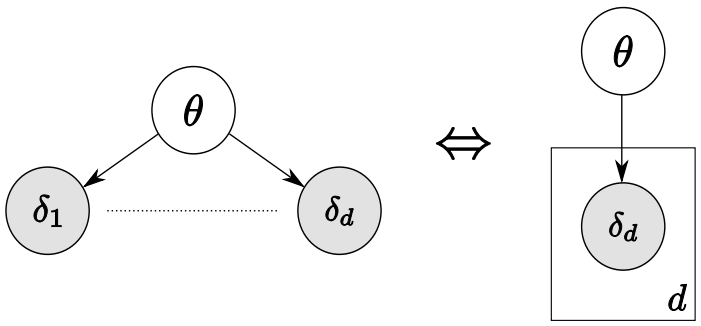


FIG. 1.2 – Réseau bayésien représentant la décomposition de la jointe pour le modèle d'une pièce. Il est courant de griser les variables observées. À droite la représentation équivalente avec une assiette (*plate*) est possible car toutes les distributions conditionnelles (sur les arcs) sont identiques. Ce modèle en étoile est appelé modèle de dépendance *bayésien naïf*, il est souvent employé pour ses propriétés calculatoires.

les indépendances conditionnelles et qui sont une source d'inspiration pour inventer des algorithmes d'inférence en terme de manipulations graphiques. Trois types de graphes sont employés : les réseaux bayésiens (Pea88 ; Jen96), les champs de Markov (KS80 ; GG84) et les graphes de facteurs (KFL01). Ces représentations sont équivalentes dans le sens où il est possible de traduire un graphe d'une représentation dans une autre (YFW03), en incorporant éventuellement des variables auxiliaires. Un modèle graphique est simplement la représentation de la probabilité jointe sur toutes les variables et de ses propriétés de factorisation. Notons que pour chaque $\mathcal{M}$, plusieurs représentations existent pour chaque type de graphe, représentant un sous-ensemble des indépendances conditionnelles supposées. Nous présentons le matériel minimal pour aborder cette thèse et renvoyons le lecteur au livre de Bishop (Bis06), chapitre 8, pour une présentation plus détaillée.

Réseaux bayésien Supposons que toutes les variables du modèle soient notées $\mathbf{x} = \{x_1, \cdots, x_d\}$. En appliquant successivement la règle du produit 1.7, la jointe se factorise en toute généralité en :

$$p(\mathbf{x}) = p(x_d|x_{d-1}, \cdots, x_1)\, p(x_{d-1}|x_{d-2}, \cdots, x_1)\, \cdots\, p(x_2|x_1)\, p(x_1). \tag{1.14}$$

De part nos connaissances préliminaires, si nous savons que x_2 est indépendant de x_1, $p(x_2|x_1)$ devient $p(x_2)$. Ces hypothèses d'indépendances conditionnelles définissent alors un jeu de *parents* pour chaque variable et la jointe devient :

$$p(\mathbf{x}) = \prod_k p(x_k \mid \text{parents}(x_k)). \tag{1.15}$$

Un réseau bayésien (Pea88) est un graphe sans circuit avec un nœud pour chaque x_k et un arc orienté provenant de chacun de ses parents. Le figure 1.2 présente un réseau bayésien pour l'exemple de la pièce.

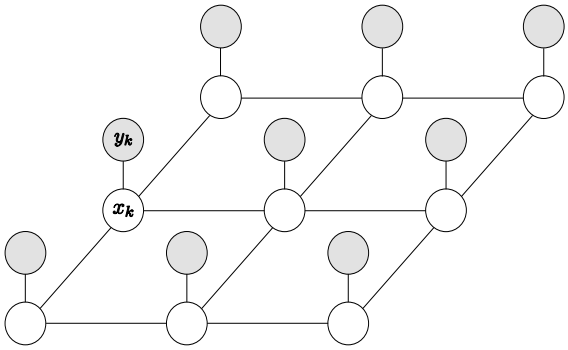


FIG. 1.3 – Un champ de Markov (MRF) pour un problème d'imagerie.

Champs de Markov Les champs de Markov (KS80) sont des graphes non orientés pour lesquels un nœud représente une variable ou un groupe de variables et les arcs une fonction de compatibilité entre ces nœuds. Ces graphes sont souvent utilisés en vision par ordinateur, où une grille de nœuds x_k non observés représente les pixels de l'image réelle non observée et une autre grille de y_k les pixels observés (fig. 1.3). Dans ces modèles, les liens probabilistes sont exprimés en terme d'énergie. Trouver les x_k de probabilité *a posteriori* maximale est équivalent à une minimisation de l'énergie du graphe. Cette énergie comporte souvent un terme exprimant que l'image réelle doit être "lisse" (x_k est similaire à ses voisins) et un terme imposant une relation entre chaque x_k et son observation y_k. En physique statistique, le modèle de magnétisation d'un matériau ferromagnétique (modèle de Potts et d'Ising (Bax82)) s'exprime naturellement dans ce formalisme, avec les y_k représentant le champ local.

$$p(\mathbf{x}, \mathbf{y}) = Z^{-1}\ e^{-E(\mathbf{x},\mathbf{y})} \tag{1.16}$$

$$E(\mathbf{x}, \mathbf{y}) = \sum_{i,j} f(x_i, x_j) + \sum_k g(x_k, y_k) \tag{1.17}$$

Graphe de facteurs Les graphes de facteurs (KFL01) (fig. 1.4), introduits plus récemment, sont plus généraux car ils permettent de représenter toute fonction factorisée de plusieurs variables, dont les distributions de probabilités :

$$p(\mathbf{x}) = \prod_a f_a(\mathbf{x}_a) \tag{1.18}$$

avec $\mathbf{x}_a$ le sous-ensemble de $\mathbf{x}$ connecté au facteur a. Dans le cas d'un modèle génératif comme celui de la pièce, les f_a sont les distributions conditionnelles et les *a priori*. Les graphes de facteurs rendent l'expression d'algorithme d'inférence particulièrement élégante et permettent d'exprimer facilement une plus grande classe d'indépendances conditionnelles que les réseaux bayésiens.

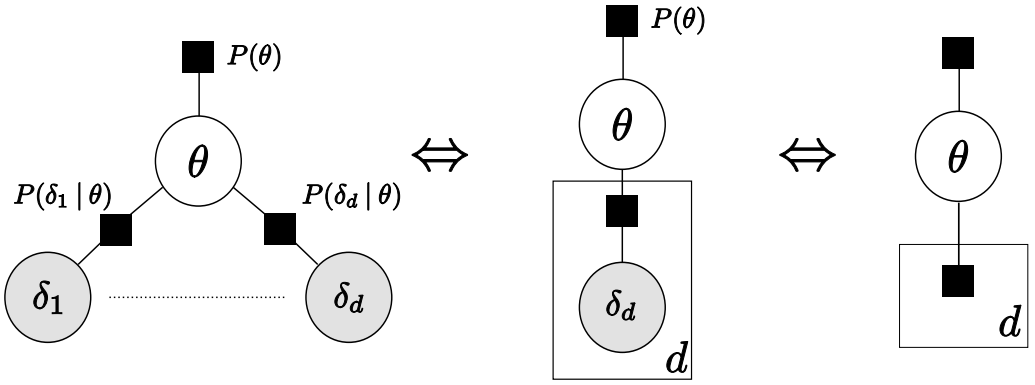


FIG. 1.4 – Représentation par un graphe de facteurs du modèle de la pièce. Comme les δ_k sont observés, ils peuvent ne pas être représentés et être intégrés dans le facteur qui leur est attaché.

1.4.3 Choix des *a priori*

> One thing which must remain clear ; prescribing the prior, by maximum entropy or otherwise, is NOT Bayesian. It may look formally as such, but the input here comes exclusively for the user. It may be a fair, or even good, approximation in some cases, but that needs to be proved in the specific cases. So-called objectivity is not possible ; the contradictions are well known.
> - Pr. Herman Rubin, Purdue University Department of Statistics -

Une fois le modèle $\mathcal{M}$ décidé, il faut mettre en place nos *a priori*, comme $p(\theta)$. La méthode bayésienne rigoureuse nous demande de traduire nos connaissances préalables en une distribution sur θ, mais ce n'est pas toujours facile et d'autres méthodes ont été proposées. Nous les présentons succinctement.

a priori informatifs

Ce sont ceux de l'approche bayésienne rigoureuse, l'*a priori* est directement déterminé par notre Π. Pour un problème d'analyse de données, il faut collecter et traduire les connaissances d'experts du problème. Cette collecte conduit à des *a priori* dits *elicited*. Le lecteur intéressé peut trouver dans (GW05) un exemple rigoureux de leur construction dans le domaine des sciences politiques quantitatives. Pour la pièce, imaginons que nous soyons sûr qu'elle soit équilibrée (car, par exemple, nous l'avons fabriquée nous même), nous aurions alors $P(\text{pile}) = 0.5$ et donc $p(\theta) = \delta_{0.5}(\theta)$.

a priori non informatifs

À l'opposé certains auteurs ont tenté d'introduire des *a priori non subjectifs*, ne dépendant pas de l'état de connaissance d'un agent mais déduits de règles formelles. C'est

l'école des *bayésiens objectifs*, en contradiction philosophique avec la méthode subjective que nous présentons dans cette thèse. Il est désormais largement admis qu'il n'existe pas d'*a priori* absolument non informatifs (KW96). Cependant ces *a priori* qui se veulent "non informatifs" ont les avantages (1) d'être faciles à obtenir, (2) de donner l'apparence de l'objectivité, (3) d'éviter de travailler avec des *a priori* subjectifs mal formulés (surtout en haute dimension), (4) d'exhiber des propriétés analytiques agréables et (5) d'avoir de bonnes propriétés fréquencistes. Ces méthodes ont comme point commun de n'utiliser comme source d'information que la *forme* de la fonction de vraisemblance $f_{\mathcal{M},\Delta}(\Theta)$ définie par $\mathcal{M}$. Ces *a priori* non subjectifs peuvent être vus comme une alternative par *défaut* quand des *a priori* subjectifs ne sont pas disponibles ; mais, comme ils violent la méthode bayésienne, ils peuvent conduire à des incohérences (Syv98) dont il faudra se méfier. Nous catégorisons ces *a priori* en deux groupes, bien que d'autres approches, basées sur la géométrie de l'information (SMD03) ou la similitude avec les résultats fréquencistes (WP63) aient aussi été proposées.

Règles d'invariance L'idée remonte au principe d'indifférence de Laplace qui choisit systématiquement une distribution uniforme. Dans notre exemple de la pièce, ce principe donne $p(\theta) = 1$ car $\theta \in [0, 1]$. Ce principe pose problème quand il conduit à des *a priori impropres* ($\int_\Theta p(\Theta) \neq 1$) par exemple sur $\mathbb{R}$, bien que souvent le *a posteriori* reste propre[7]. En outre, une absence d'information et l'équiprobabilité sont des états de connaissance assez différents : un *a priori* plat peut être *très* informatif. Un autre problème, sa non invariance par reparamétrisation, à conduit Jeffrey a introduire son *a priori* : $p(\Theta) := I(\Theta)^{-1/2}$ avec $I(\Theta)$ l'information de Fisher :

$$I(\Theta) := E_{\Delta|\Theta}\left[\frac{\partial \log f_\Delta(\Theta)}{\partial \Theta}\right]^2. \tag{1.19}$$

L'*a priori* de Jeffrey est invariant par changement de coordonnées et donne pour la pièce $p(\theta) = [\theta\,(1-\theta)]^{-1/2}$ (fig. 1.5(a)). En général, il conduit à des distributions impropres et ne fonctionne bien qu'avec des Θ de dimension un ou deux.

Une autre voie, dont nous pouvons faire remonter l'histoire à Poincarré (Poi12), est de rechercher des *a priori* invariants sous l'action d'un certain groupe de transformations. L'idée est de donner à l'*a priori* la même structure d'invariances que celle du modèle, c'est à dire de la vraisemblance. Par exemple si θ est un paramètre de localisation comme la moyenne d'une gaussienne, $f_\Delta(\theta)$ est invariante par translation, donc $p(\theta)$ doit l'être aussi, ce qui conduit à $p(\theta) \propto 1$. Si θ est un paramètre d'échelle comme l'écart type, $p(\theta) \propto 1/\theta$. Pour la pièce, $f_\Delta(\theta)$ est invariante par symétrie $\theta \leftrightarrow 1-\theta$, ce qui conduit à une famille de $p(\theta)$ symétrique par rapport à 0.5.

[7] Mais l'interprétation d'un *a priori* impropre en terme de degrés de croyance pose problème :

> One should not interpret any non-subjective prior as a probability distribution.
> – Bernardo (Ber97)–

Règles de faible influence L'idée développée par Berger et Bernardo (BB89) est de trouver l'*a priori* ayant le moins d'influence possible sur l'*a posteriori*, relativement aux données. Ils définissent une distance entre l'*a priori* et l'*a posteriori* et appellent "*a priori de référence*" celui qui la maximise. Les *a priori* de référence ont de nombreuses propriétés intéressantes et sont certainement les meilleurs candidats non subjectifs. Mais cette approche diffère de la méthode bayésienne, en particulier parce qu'elle impose une intégration sur l'espace des données, comme le font les méthodes fréquencistes. Par conséquent elle viole le principe de vraisemblance, affirmant que deux expériences produisant des vraisemblances proportionnelles doivent produire les mêmes inférences (BW88). Berger et Bernardo en sont conscients et conseillent d'utiliser un *a priori* subjectif quand c'est possible, mais, si ce n'est pas le cas, ils considèrent que les *a priori* de référence peuvent jouer le rôle d'*a priori* par défaut. Ils représentent en quelque sorte une convention parmi les statisticiens, un outil technique, comme les unités de longueur et de poids le sont pour les physiciens. Pour la pièce, le résultat est le même que celui obtenu par la méthode de Jeffrey.

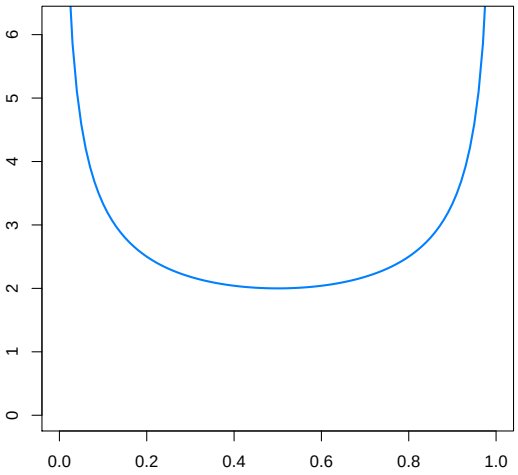
a priori partiellement informatifs

Cette catégorie est la plus fréquemment employée, lorsque nos connaissances subjectives nous donnent des idées sur la forme des *a priori*, mais ne nous permettent pas de les déduire complètement. La principale méthode est de considérer une famille paramétrique $\mathcal{F}_\alpha$ de distributions et de choisir les α correspondant à nos connaissances. Une telle famille ayant des propriétés analytiques intéressantes est celle des *a priori* conjugués, conduisant à des modèles hiérarchiques parfois déterminés par une méthode empirique. Nous décrivons aussi l'approche entropique.

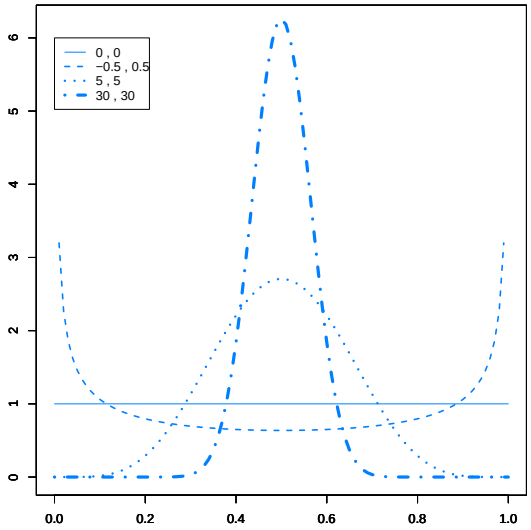
a priori conjugués Un *a priori* conjugué d'une vraisemblance est un *a priori* tel que l'*a posteriori* ait la même forme analytique que l'*a priori*. Il n'est pas toujours garanti qu'il en existe sauf pour la famille exponentielle (annexe A.1). La seule justification pour utiliser de tels *a priori* est qu'ils facilitent les calculs. Cependant ils sont souvent assez souples pour ne pas rentrer en contradiction avec Π. Un *a priori* conjugué s'interprète comme le résumé d'un certain nombre d'expériences virtuelles et ses paramètres reflètent cette analogie. Par exemple, le conjugué d'une loi multinomiale est une loi de Dirichlet avec comme paramètres le nombre d'observations virtuelles de chacune des éventualités de la multinomiale. Celui de la moyenne d'une gaussienne est aussi une gaussienne, celui de sa matrice de covariance est une Wishart inverse, celui d'une uniforme est une loi de Pareto. La vraisemblance, dans l'exemple de la pièce, est une loi de Bernoulli et son conjugué est une loi beta.

$$\theta \sim \text{Beta}(\theta|\alpha_1, \alpha_2) = \frac{\Gamma(\alpha_1 + \alpha_2 + 2)}{\Gamma(\alpha_1 + 1)\Gamma(\alpha_2 + 1)}\, \theta^{\alpha_1}(1-\theta)^{\alpha_2} \tag{1.20}$$

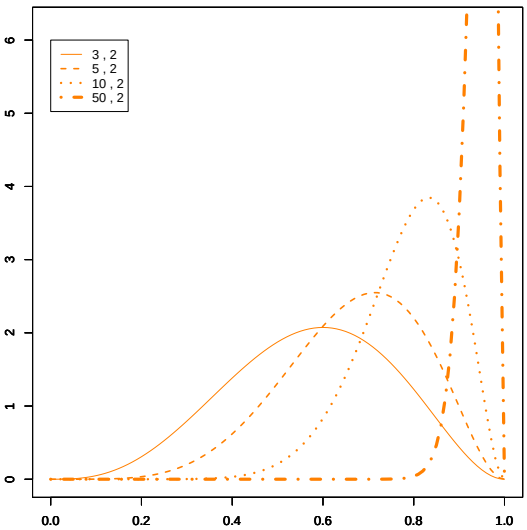
Les paramètres α_1 et α_2 sont des *hyper-paramètres* car ils contrôlent la distribution sur le paramètre θ. Nous avons alors un modèle hiérarchique car chaque (α_1, α_2) définit une distribution sur θ donc une distribution sur la distribution de δ_d = pile. Il est possible de poursuivre cette hiérarchie sur plusieurs niveaux en définissant un distribution paramétrée



(a) *a priori* de Jeffrey



(b) Beta symétrique ($\alpha_1 = \alpha_2$)



(c) Beta non symétrique ($\alpha_1 \neq \alpha_2$)

FIG. 1.5 – Différents *a priori* sur le paramètre θ une loi de Bernoulli.

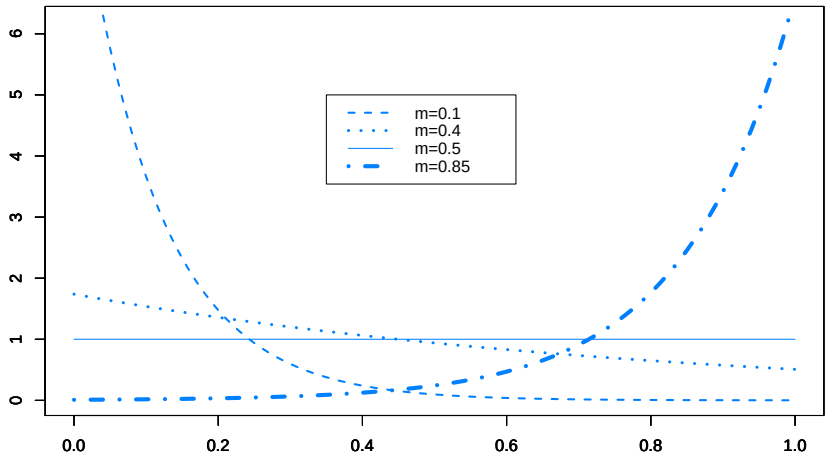


FIG. 1.6 – Distribution $p(\theta)$ d'entropie maximale sur $[0, 1]$ tels que $E_p[\theta] = m$.

sur les α, puis une autre sur ces paramètres... Le but d'un modèle hiérarchique est d'établir des liens entre plusieurs expériences, en les faisant partager un niveau supérieur du modèle. En revanche, ajouter indéfiniment des étages sur la même distribution ne changera pas les inférences produites et pour la pièce, il ne sert à rien d'aller plus loin que la beta. Interpréter les α comme le nombre de piles et de faces virtuels permet de traduire notre connaissance d'un biais éventuel ($\alpha_1 \neq \alpha_2$) et de sa force ($\alpha_1 + \alpha_2$) (fig. 1.5).

a priori **empiriques** Les *a priori* empiriques (*empirical bayes*) sont déterminés en utilisant les données. Par exemple, si nous avons choisi une famille d'*a priori* conjugué d'hyperparamètre α, l'*a priori* empirique correspond à la valeur de α maximisant la vraisemblance :

$$\alpha^* = \text{ArgMax}_\alpha \; p(\Delta \mid \alpha) = \text{ArgMax}_\alpha \int p(\Delta \mid \Theta) \, p(\Theta \mid \alpha) \, d\Theta. \tag{1.21}$$

Cette procédure, dite de *maximum de vraisemblance de type II*, viole aussi la méthode bayésienne et peut conduire à du sur-apprentissage car les données sont utilisées deux fois, mais elle est pratique et robuste à de mauvaises spécifications de $\mathcal{F}_\alpha$.

a priori **entropiques** Les entropies de distributions discrète P et continues p sont :

$$H[P] = -\sum p(x_i) \ln p(x_i) \geq 0 \tag{1.22}$$

$$H[p] = -\int p(\mathbf{x}) \ln p(\mathbf{x}) \, d\mathbf{x} \gtrless 0. \tag{1.23}$$

En fait, une entropie n'est définie que relativement à une mesure de base, qui est la mesure de comptage pour le cas discret et la mesure de Lebesgue pour le cas continu. Si p est proportionnelle à cette mesure, alors son entropie est maximale. Pour 1.22 et 1.23, c'est le cas quand p est la distribution uniforme, il est alors dit que l'entropie mesure l'incertitude,

l'absence d'information véhiculée par p. Une distribution de grande entropie est plate, elle est peu informative. Une distribution de faible entropie est piquée, comme par exemple un dirac, et transcrit une grande connaissance de la variable. Selon cette interprétation, il est possible de chercher un *a priori* contenant le moins d'information possible tout en vérifiant certaines contraintes imposées par nos connaissances préliminaires. Ce principe de maximum d'entropie, développé par Jaynes (Jay03) a donné de nombreuses applications pratiques, comme une reformulation de la mécanique statistique (Jay57). Dans le cadre de l'inférence subjective, il s'applique quand les contraintes *a priori* s'expriment par des espérances. Pour la pièce, supposons que nous sachions la moyenne $\int_\theta \theta p(\theta)\, d\theta = m$. Les *a priori* entropiques correspondants, obtenus par calcul variationnel et grâce aux multiplicateurs de Lagrange (Dan07) sont donnés par la figure 1.6. Un autre exemple standard est que la distribution gaussienne maximise l'entropie sous contrainte d'espérance et de variance fixées. Dans le cadre de notre approche, ce principe ne s'impose pas, mais s'il est cohérent avec nos connaissances préliminaires, alors il peut être utile de l'employer. Dans d'autres travaux, le maximum d'entropie est utilisé différemment, comme un principe d'induction concurrent de la règle de Bayes (Ski88 ; CP04). Comme la question de savoir s'il entre en conflit, cœxiste pacifiquement, ou bien étend la méthode bayésienne n'est pas aisée (Uff95), nous restreindrons son utilisation à l'obtention d'*a priori* entropiques.

1.5 Données

Un fois modèle et *a priori* spécifiés, nous pouvons regarder les données collectées. Un théorème fondamental, le théorème d'échangeabilité, aussi appelé théorème général de la représentation (Fin37), est une motivation forte pour l'utilisation de modèles paramétriques et d'*a priori*. Il affirme que si (1) nous utilisons une description probabiliste des données (comme dans la méthode bayésienne) et si (2) les données sont *échangeables*, alors il existe un modèle paramétrique sous-jacent les générant. Des données sont échangeables si leur ordre n'importe pas, c'est-à-dire si leur distribution jointe $p(x_1, \ldots, x_d)$ est invariante par permutation d'indices.

Théorème 3. *Si* $\mathbf{x} = (x_1, \cdots, x_d)$ *sont infiniment échangeables, alors leur jointe possède au moins une représentation de mixture pour un objet aléatoire* Θ *:*

$$p(\mathbf{x}) = \int \left(\prod_{i=1}^{d} p(x_i \mid \Theta) \right) dP(\Theta) \tag{1.24}$$

Donc, si les données sont échangeables, alors il est raisonnable de considérer qu'il existe des paramètres, des *a priori* sur ces paramètres et que les données sont *i.i.d.* conditionnellement à ces paramètres.

Des données i.i.d. sont des données *indépendantes et identiquement distribuées*. Dans ce cas la vraisemblance se factorise et les dépendances conditionnelles entre les variables de $\mathcal{M}$ sont simples, ce qui facilite les traitements analytiques et les inférences. Pour l'exemple

de la pièce, les résultats de lancers ne sont pas i.i.d. dans l'absolu, mais ils le deviennent conditionnellement à θ.

L'échangeabilité est une propriété différente de celle d'i.i.d.. Des données i.i.d. sont échangeables, mais la réciproque n'est pas toujours vraie. Par exemple, les coordonnées d'une donnée tirée selon une gaussienne 2D dont les axes propres sont les diagonales sont échangeables, mais elles ne sont pas indépendantes. Notons de plus que ce théorème n'est pas vrai si nous limitons Θ à des espaces vectoriels euclidiens. Il faut que Θ puisse être une mesure et donc que $P(\theta)$ soit une distribution sur des mesures. Cette catégorie d'objets est à la base des modèles non paramétriques infinis comme les processus de Dirichlet (Fer73) et les processus gaussiens (RW05) qui ne seront pas abordés dans cette thèse, bien que la méthode bayésienne s'y applique de la même façon.

Pour l'exemple de la pièce, supposons que nous l'ayons lancée 100 fois et que nous n'ayons observé que des piles : $\Delta =$ (pile-pile-. . . -pile). Nous choissions cet exemple étrange pour répondre à une question classique : quelle sera le résultat du lancer suivant ? Le premier réflexe d'une personne peu habituée aux statistiques est de répondre "face", car une séquence de 100 piles est déjà tellement improbable, que "pour se rattraper", la pièce devra faire de nombreux faces. Un étudiant qui débute l'étude des probabilités répondra plutôt qu'il y une chance sur deux de faire pile, car les lancers passés n'influencent pas les lancers futurs. Mais, intuitivement, un tel jeu de données doit nous faire penser qu'en réalité la pièce est fortement biaisée en faveur des piles et donc que le prochain lancer donnera pile. Nous verrons que l'inférence bayésienne permet de formaliser numériquement ce raisonnement.

1.6 Inférence

Après avoir collecté les données, l'inférence consiste en l'application automatique de la règle de Bayes pour calculer et exploiter des distributions *a posteriori*. Ces distributions peuvent être utilisées pour différentes tâches, comme l'estimation ponctuelle, l'estimation ensembliste, pour faire des prédictions ou comparer des modèles.

1.6.1 Un calcul d'intégrales

De façon générale, nous cherchons une distribution sur les variables qui nous intéressent, conditionnellement aux données et en marginalisant des variables cachées. Dans quelques cas simples, il n'y a pas de variables cachées et l'application de la règle de Bayes est immédiate, mais en général, l'inférence consiste en un calcul d'intégrale :

$$\begin{aligned} p(\Theta \mid \Delta, \mathcal{M}) &= Z^{-1} \int p(\Theta, \mathcal{H}, \Delta|\mathcal{M})\, d\mathcal{H} \\ \text{avec} \quad Z &= \int p(\Theta, \mathcal{H}, \Delta|\mathcal{M})\, d\mathcal{H}\, d\Theta. \end{aligned} \tag{1.25}$$

La distribution ainsi calculée représente toute l'information rationnellement dérivable sur Θ à partir des *a priori* et des données et uniquement cette information. Cependant, une

distribution peut être délicate à manipuler et nous avons parfois besoin d'une représentation ponctuelle, nécessairement sous optimale. Par exemple, il est courant de chercher le maximum *a posteriori* (MAP) :

$$\Theta^*_{\text{MAP}} = \text{ArgMax}_\Theta \; p(\Theta|\Delta, \mathcal{M}) \tag{1.26}$$

qui, à la différence du maximum de vraisemblance $\Theta^*_{\text{ML}} = \text{ArgMax}_\Theta \; p(\Delta|\Theta, \mathcal{M})$, intègre les informations *a priori*.

L'utilisation du MAP est un choix mais d'autres alternatives peuvent être plus pertinentes comme :

- faire un ou plusieurs tirages, pour s'autoriser à explorer des zones moins probables,
- utiliser l'espérance $E_{\Theta|\Delta,\mathcal{M}}[\Theta]$, qui a l'avantage de tenir compte de l'*a posteriori* entier et d'éviter les maxima locaux, mais qui a aussi ses limites en cas de multimodalité. De façon générale, il est possible de calculer l'espérance d'une fonction $g(\Theta)$ quelconque pour obtenir, par exemple, les autres moments de $p(\Theta|\Delta, \mathcal{M})$, ou, si g est une fonction indicatrice, la probabilité d'une zone de l'espace des Θ,
- utiliser des quantiles, par exemple la médiane qui est une meilleure représentation que la moyenne pour les distributions monomodales très asymétriques,
- déterminer le Θ^* minimisant une fonction de coût. Une telle fonction L donne le coût de décider Θ, alors que sa vraie valeur est Θ_{vrai}. Par exemple, L est souvent quadratique pour des modèles de régressions linéaires et dans un contexte de classification binaire, L définit les coûts des erreurs de type I, de type II et les gains réalisés lors d'une classification correcte. Comme Θ_{vrai} est inconnu, il s'agit de minimiser L'espérance de L sous la distribution *a posteriori* :

$$\Theta^* = \text{ArgMin}_\Theta \int L(\Theta, \Theta_{vrai}) \; p(\Theta_{vrai}|\Delta, \mathcal{M}) \, d\Theta_{vrai}, \tag{1.27}$$

- nous pouvons aussi calculer un intervalle de *croyance* défini comme une zone de l'espace contenant, par exemple 95% de la probabilité et centrée sur le MAP.

Notons que l'interprétation des intervalles de *croyance* bayésiens est différente de celle des intervalle de *confiance* fréquencistes. Par exemple, si pour un θ réel, $I = [13, 21]$ est un intervalle de croyance à 95%, alors par définition $P(\theta \in [13, 21]) = 95\%$.

En revanche, pour les fréquencistes, un intervalle de confiance est une variable aléatoire dépendante du jeu de données et θ est un paramètre non aléatoire inconnu. Pour chaque jeu de données de l'espace de toutes les données possibles, un intervalle numérique est produit. Si, pour le Δ effectivement observé, l'intervalle est $[13, 21]$, alors $P(\theta \in [13, 21])$ n'est pas 95%, mais 0 ou 1, car rien n'est aléatoire dans cette proposition. Soit la "vraie" valeur de θ est dans $[13, 21]$, soit elle n'y est pas. Il est donc incorrect de dire $P(\theta \in [13, 21]) = 95\%$. La bonne interprétation, plus délicate, est la suivante : si nous avions répété 100 fois la même expérience, alors nous aurions 100 intervalles numériques différents et en moyenne, 95 d'entre eux auraient contenu le vrai θ. Les fréquencistes résument cette information par : "nous avons une *confiance* de 95% que $\theta \in [13, 21]$"

Nous retrouvons là une différence fondamentale entre les approches fréquencistes et subjectives : ne voulant pas considérer une probabilité dans l'espace des paramètres, les

fréquencistes doivent s'appuyer sur un espace d'expériences répétées imaginaires. Les intégrations requises par ces méthodes ont lieu dans cet espace des données (*sampler espace*), alors que les intégrations bayésiennes ont lieu dans l'espace des paramètres (*hypothesis space*, ou *model space*).

En plus des différences philosophiques et sémantiques que cela implique, cette distinction a des conséquences en terme de machinerie calculatoire (Lor99). Les intégrales fréquencistes ne peuvent être que rarement déterminées analytiquement, mais comme les données sont souvent i.i.d., l'intégrande se factorise selon toutes les dimensions δ_i du *sample space* et des méthodes de Monte-Carlo simples donnent de bonnes approximations.

Cependant cette propriété de factorisation n'est, en général, plus vraie dans l'espace des paramètres, ce qui rend le calcul des intégrales bayésiennes bien plus délicat que celui des intégrales fréquencistes. De plus, l'*a posteriori* a une forme complexe, souvent multimodale. Lorsque la dimension de Θ est petite, les méthodes de quadratures usuelles restent performantes, mais elles ne sont plus utilisables dès que la dimension dépasse 3 ou 4, à cause d'une croissance exponentielle de la taille de l'espace (malédiction de la dimension). En effet, le problème de l'inférence bayésienne exacte a été montré NP-dur (Coo90).

Ainsi les calculs requis par l'inférence bayésienne sont par nature difficiles, mais depuis une vingtaine d'années, de nouvelles méthodes d'approximation ont permis leur dissémination. Avant de lister ces différentes techniques de calcul, notons que d'autres quantités peuvent être déduites de l'*a posteriori*, éventuellement au prix de nouveaux calculs. Par exemple nous pouvons être intéressés par les *marginales a posteriori* d'un vecteur Θ :

$$p(\theta_i|\Delta,\mathcal{M}) = \int p(\Theta|\Delta,\mathcal{M})\, d\Theta^{\setminus i}. \tag{1.28}$$

La distribution *prédictive* d'une future observation se calcule en moyennant les prédictions associées à chaque valeur de Θ :

$$p(\delta_{d+1}|\Delta,\mathcal{M}) = \int p(\delta_{d+1}|\Theta,\mathcal{M})\; p(\Theta|\Delta,\mathcal{M})\, d\Theta. \tag{1.29}$$

Cette marginalisation est un avantage intrinsèque à la méthode bayésienne, car elle permet de combiner les prédictions relatives à chaque Θ par leurs probabilités.

Le formalisme bayésien permet aussi de considérer le modèle $\mathcal{M}$ comme une variable et ainsi de comparer des modèles ou de les combiner. La comparaison de deux modèles se base sur leurs probabilités *a posteriori*, en marginalisant leurs paramètres respectifs :

$$\frac{p(\mathcal{M}_1|\Delta)}{p(\mathcal{M}_2|\Delta)} = \frac{p(\Delta|\mathcal{M}_1)\, p(\mathcal{M}_1)}{p(\Delta|\mathcal{M}_2)\, p(\mathcal{M}_2)} = \frac{\int p(\Delta|\Theta_1,\mathcal{M}_1)\, p(\Theta_1|\mathcal{M}_1)\, d\Theta_1}{\int p(\Delta|\Theta_2,\mathcal{M}_2)\, p(\Theta_2|\mathcal{M}_2)\, d\Theta_2} \cdot \frac{p(\mathcal{M}_1)}{p(\mathcal{M}_2)}. \tag{1.30}$$

Le *ratio* des preuves des modèles est appelé facteur de Bayes (*bayes factor*). Pour combiner des modèles, il suffit d'appliquer la règle de la somme. Par exemple, pour trouver la distribution *a posteriori* d'un paramètre θ :

$$p(\theta|\Delta) = \sum_i p(\theta|\Delta,\mathcal{M}_i)\, p(\mathcal{M}_i|\Delta). \tag{1.31}$$

Cette méthode dite de BMA (*bayesian model averaging*), étend le principe d'explications multiples d'Épicure (342-270 av. J.-C.) : si plusieurs théories sont compatibles avec les données, il faut toutes les conserver. La procédure ajoute une évaluation numérique de leur "compatibilité" par leurs probabilités *a posteriori*.

Nous proposons de passer en revue les différentes méthodes d'inférence, en fonction des types d'intégrations à mener.

1.6.2 Inférence exacte

Inférence exacte exhaustive

Lorsque les variables sont discrètes, les intégrales se réduisent à des sommes calculables en énumérant toutes les possibilités, en explorant exhaustivement l'espace des configurations. Dans ce cas, les distributions conditionnelles sont données par des tables de valeurs qu'il suffit de parcourir. À cause du fléau de la dimension, cette méthode n'est envisageable qu'en très faible dimension.

Ces calculs peuvent cependant être optimisés par la technique d'élimination de variables et être ainsi appliqués à des problèmes plus complexes. Comme les indépendances conditionnelles du modèle bayésien confèrent à l'intégrande des propriétés de factorisation, il est possible de réorganiser les intégrales, en "poussant" les sommes dans les produits. Par exemple :

$$p(x_1) = \int_{x_2} \int_{x_3} p(x_1|x_2)p(x_2|x_3)p(x_3)\, dx_3\, dx_2 \tag{1.32}$$

$$= \int_{x_2} p(x_1|x_2) \left[\int_{x_3} p(x_2|x_3)p(x_3)\, dx_3 \right] dx_2. \tag{1.33}$$

Cette propriété est due à la distributivité des opérations de somme et de produit et est généralisable à tout semi-anneau commutatif (MA00). Son application aux modèles de Markov cachés (HMM) est l'algorithme de Viterbi (Vit67), elle est aussi à l'œuvre dans la transformation de Fourrier rapide (FFT (Tuk65)).

Pearl (Pea88) l'introduit pour les réseaux bayésiens, en utilisant la programmation dynamique pour déterminer toutes les marginales simultanément. Avec la représentation graphique de la structure de dépendance, les calculs deviennent séquentiels et peuvent être interprétés comme une succession de passages de messages locaux. Cet algorithme, *belief propagation* (BP), a une complexité polynomiale mais ne peut être appliqué qu'à des structures sans boucles (cycles non dirigés) ou poly-arbres. L'application de BP aux HMM est l'algorithme *forward-backward* et si l'opérateur somme est remplacé par un max, nous avons l'algorithme de Viterbi (Vit67). Dans le formalisme des graphes de facteurs, l'inférence consiste à définir des messages de facteurs à variables, de variables à facteurs et des marginales approchées pour chaque variable (YFW03). Une variable est ensuite choisie comme racine et les messages sont propagés le long des branches jusqu'aux feuilles, puis

retro-propagés vers la racine selon les équations d'invariants :

$$m_{x\to f}(x) := \prod_{g\in N_x\setminus f} m_{g\to x}(x) \tag{1.34}$$

$$m_{f\to x}(x) := \int_{y\in N_f\setminus x} f(N_f) \prod_{y\in N_f\setminus x} m_{y\to f}(y)\; d(N_f\setminus x) \tag{1.35}$$

$$p(x) = \prod_{f\in N_x} m_{f\to x}(x). \tag{1.36}$$

Après ce transfert d'information, les $p(x)$ obtenus sont les vraies marginales de la jointe.

Lorsque le réseau présente des boucles, il est possible de regrouper certaines variables pour former un arbre de jonction (JLO90), puis de passer des messages dans cet arbre. Cette méthode, bien qu'optimale parmi les méthodes exactes (JJ94), a tout de même un coût exponentielle en la taille de la plus grande clique du graphe.

Inférence exacte analytique

Lorsque les variables sont continues et que les distributions conditionnelles ont une forme convenable, l'inférence peut être réalisée analytiquement, en particulier quand il est possible d'exprimer nos *a priori* de façon conjuguée.

Pour l'exemple de la pièce, considérons quatre modèles différents *a priori* équiprobables :
- $\theta|\mathcal{M}_1 \sim \delta_{0.5}(\theta)$: "la pièce est parfaitement équilibrée",
- $\theta|\mathcal{M}_2 \sim \delta_1(\theta)$: "la pièce a deux cotés pile",
- $\theta|\mathcal{M}_3 \sim \text{Beta}(\theta, 30, 30)$: "la pièce a éventuellement un léger biais",
- $\theta|\mathcal{M}_4 \sim 1$: "la pièce est truquée, mais nous ne savons comment".

Après l'observation du jeu de données Δ constitué de 100 piles, les *a posteriori* sont :

$$p(\theta|\Delta,\mathcal{M}_1) = \delta_{0.5}(\theta) \tag{1.37}$$

$$p(\theta|\Delta,\mathcal{M}_2) = \delta_1(\theta) \tag{1.38}$$

$$p(\theta|\Delta,\mathcal{M}_3) = \frac{161!}{130!\,30!}\,\theta^{130}\,(1-\theta)^{30} \tag{1.39}$$

$$p(\theta|\Delta,\mathcal{M}_4) = 101\,\theta^{100}. \tag{1.40}$$

Notons que les données n'ont pas pu modifier les certitudes exprimées par les modèles 1 et 2 ; il faut être attentif lors de l'emploi de probabilités *a priori* nulles. Les preuves $p(\Delta|\mathcal{M}_i) = \int_\theta \theta^{100}\, p(\theta|\mathcal{M}_i)\, d\theta$ des modèles sont respectivement :

$$p(\Delta|\mathcal{M}_1) = 2^{-100} \approx 7.8\,10^{-31} \tag{1.41}$$

$$p(\Delta|\mathcal{M}_2) = 1 \tag{1.42}$$

$$p(\Delta|\mathcal{M}_3) \approx 1.6\,10^{-16} \tag{1.43}$$

$$p(\Delta|\mathcal{M}_4) = 1/101 \approx 10^{-2}. \tag{1.44}$$

Nous voyons ainsi, que la vraisemblance du modèle 1, "la pièce est équilibrée", est extrêmement faible ; les données montrent qu'il est extrêmement peu probable que la pièce

soit équilibrée et donc que la prédiction "50-50" pour le prochain lancer n'est pas pertinente. Dans ce cas, le dirac en 1 est le meilleur modèle, mais remarquons que sa preuve aurait été 0 si il y avait eu le moindre face dans Δ. Les distributions prédictives sont $p(\delta_{d+1} = \text{pile}|\Delta, \mathcal{M}_i) = E_{\theta|\Delta,\mathcal{M}_i}[\theta]$:

$$p(\delta_{d+1} = \text{pile}|\Delta, \mathcal{M}_1) = 0.5 \tag{1.45}$$
$$p(\delta_{d+1} = \text{pile}|\Delta, \mathcal{M}_2) = 1 \tag{1.46}$$
$$p(\delta_{d+1} = \text{pile}|\Delta, \mathcal{M}_3) = 131/162 \approx 0.8 \tag{1.47}$$
$$p(\delta_{d+1} = \text{pile}|\Delta, \mathcal{M}_4) = 101/102 \approx 0.990. \tag{1.48}$$

Notons que la prédiction du modèle 4 est celle de la loi de succession de Laplace. En effet cette loi de Laplace est la distribution prédictive associée à une vraisemblance multinomiale avec un *a priori* conjugué de Dirichlet uniforme (hyper paramètres tous égaux à 0). Pour établir une prédiction bayésienne, utilisons la méthode BMA :

$$p(\delta_{d+1} = \text{pile}|\Delta) = \sum_i p(\delta_{d+1} = \text{pile}|\Delta, \mathcal{M}_i)\, p(\mathcal{M}_i|\Delta) \tag{1.49}$$
$$= \frac{\sum_i p(\delta_{d+1} = \text{pile}|\Delta, \mathcal{M}_i)\, d\theta \times 0.25\, p(\Delta|\mathcal{M}_i)}{\sum_i 0.25\, p(\Delta|\mathcal{M}_i)} \tag{1.50}$$
$$= 0.999903 \tag{1.51}$$

Le dénominateur est la preuve du modèle global, composition des 4 sous modèles. Nous obtenons ainsi une prédiction très proche de un, nous avons une grande confiance que "pile" se réalise à nouveau. Cette prédiction est obtenue automatiquement, alors qu'*a priori*, les quatre modèles étaient équiprobables, même celui d'une pièce équilibrée : l'influence des modèles non pertinents est automatiquement limitée par les données effectivement observées.

1.6.3 Inférence approchée stochastique

Les succès récents de l'approche bayésienne reposent en partie sur l'apparition de méthodes d'approximations stochastiques efficaces en grandes dimensions. Ces méthodes se basent sur la loi des grands nombres : l'intégrale est approximée par la somme des valeurs de l'intégrande en des points tirés aléatoirement. Par exemple si $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ est un ensemble de points tirés uniformément sur une zone A, alors

$$\frac{|A|}{n} \sum_{i=1}^{n} p(\mathbf{x}_i) \underset{n\to\infty}{\longrightarrow} \int_A p(\mathbf{x})\, d\mathbf{x} \tag{1.52}$$

indépendamment de la dimension et de la complexité de p, mais éventuellement très lentement. Pour accélérer cette convergence, il faut tirer les $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ avec plus de soin, par exemple selon une distribution q qui ressemble à p mais plus facilement échantillonnable, puis d'ajuster la formule 1.52 correctement (*importance sampling*). Un exemple séquentiel

très populaire en robotique est le filtre à particules (DDFG01), qui utilise la nature temporelle du modèle : p_{t-1} est la distribution d'importance échantillonnée pour approximer p_t.

Les méthodes MCMC (*Monte Carlo Markov Chain* (Nea93)) perfectionnent cette idée en simulant une chaîne de Markov dont la distribution stationnaire est p. Il est en général difficile de trouver une telle chaîne, mais différentes méthodes génériques existent, comme la méthode de Metropolis-Hastings (Has70) qui génère une chaîne d'échantillons en acceptant ou rejetant des points potentiellement intéressants. Par exemple, la méthode de Gibbs (*Gibbs sampling* (GG84)) s'applique lorsque $\mathbf{x} = \{x_1, \ldots, x_d\}$ est multidimensionnel et consiste à tirer successivement sur toutes les distributions conditionnelles $p(x_k|\mathbf{x}^{\setminus k})$ en instanciant les $\mathbf{x}^{\setminus k}$ avec les valeurs trouvées lors des précédents tirages. La généricité de cette approche à permis la création d'un moteur d'inférence universel, Winbugs (LTBS00). Des méthodes plus sophistiquées, comme RJMCMC (Reversible Jump MCMC (Gre95)) permettent de simuler un *a posteriori* définit dans un espace dont la dimension est aussi un paramètre, comme par exemple un modèle de mixture dont le nombre de composantes n'est pas connu.

Les méthodes de Monte-Carlo garantissent une convergence vers le vrai p avec une quantité de calcul infinie. Elles nécessitent cependant de vérifier que la chaîne à bien convergé, ce qui est assez délicat.

1.6.4 Inférence approchée déterministe

Nous avons ainsi des méthodes exactes et rapides en petite dimension et des méthodes stochastiques précises mais parfois lentes en grande dimension. Les méthodes approchées déterministes ont pour but d'être rapides en grande dimension, au prix d'une perte de précision.

Une méthode approchée est dite déterministe si elle ne repose pas sur une génération de nombres aléatoires pour trouver les *a posteriori*. Une autre différence avec les méthodes de Monte-Carlo est qu'en général, les approximations déterministes ne seront pas égales au vrai *a posteriori*, même avec une quantité de calcul infini. En effet, leur utilisation correspond à la recherche d'un compromis entre précision et temps de calcul : si l'inférence exacte est trop lente, nous cherchons une approximation raisonnable calculable en un temps raisonnable.

Approximations asymptotiques

Les approximations asymptotiques sont basées sur le théorème central limite : en petite dimension, avec beaucoup de données, un *a posteriori* monomodale peut être approché par une gaussienne multidimensionnelle centrée sur le MAP et dont la matrice de covariance est déterminée par la courbure locale autour du MAP. L'approximation de Laplace, est :

$$p(\Theta|\Delta, \mathcal{M}) \approx (2\pi)^{-\frac{d}{2}}|A|^{\frac{1}{2}} \exp\left(-\frac{1}{2}(\Theta - \Theta_{\text{MAP}})^\top A(\Theta - \Theta_{\text{MAP}})\right) \tag{1.53}$$

avec $-A$ la hessienne du log *a posteriori* en Θ_{MAP}. La log-preuve approchée,

$$\log p(\Delta|\mathcal{M}) \approx \log p(\Theta_{\text{MAP}}|\mathcal{M}) + \log p(\Delta|\Theta_{\text{MAP}}, \mathcal{M}) + \frac{d}{2}\log 2\pi - \frac{1}{2}\log|A|, \tag{1.54}$$

devient, lorsque le nombre de données n croît :

$$\log p(\Delta|\mathcal{M}) \approx \log p(\Delta|\Theta_{\text{MAP}}, \mathcal{M}) + \frac{d}{2}\log n. \tag{1.55}$$

Ce critère (BIC *Bayesian Information Criteria* (Sch78)) est fréquemment utilisé pour comparer des modèles, souvent avec Θ_{ML} au lieu de Θ_{MAP}. La dimension d de Θ est le nombre de paramètres du modèle et peut être délicate à déterminer si les paramètres sont liés. Utiliser le BIC évite le calcul de la marginalisation des paramètres, tout en conservant une pénalité pour les modèles complexes. Des travaux plus détaillés (CH97) ont montré que la méthode de Laplace était mieux adaptée pour sélectionner un modèle que pour les marginaliser, lorsque la structure est celle d'un réseau bayésien naïf.

Approximations numériques

La façon la plus directe pour calculer une intégrale est d'employer une méthode de quadrature (DR84). Ces méthodes permettent d'estimer numériquement des intégrales en petites dimensions. L'intégrande est évaluée en un certain nombre de points, qui sont ensuite interpolés par une fonction analytiquement intégrable. Une précision arbitraire puisse être atteinte en augmentant le nombre de points. Cependant ce nombre devient vite trop grand quand la dimension augmente, quand l'intégrande ne s'interpole pas facilement ou quand le support de l'intégrande est petit devant la taille de l'espace d'intégration. Comme ces trois cas sont courants lors d'inférences bayésiennes, ces méthodes sont rarement applicables en pratique, bien qu'elles soient presque imbattables en dimension un et deux.

Méthodes variationnelles : Variational Bayes

En toute généralité, une méthode variationnelle cherche à approximer un *a posteriori* p exact par une distribution q appartenant à une famille $\mathcal{F}$ de distributions agréables à manipuler. Il s'agit de trouver le q qui minimise une certaine "distance" avec p, c'est-à-dire qui minimise une fonctionnelle sur l'espace $\mathcal{F}$.

Souvent, $\mathcal{F}$ est une famille exponentielle (annexe A.1) très factorisée, comme les gaussiennes de covariance diagonale. Bien qu'en général p ne soit pas aussi simple, cette approximation peut être efficace pour modéliser une zone où p est très concentrée. La moyenne de q est une estimation du MAP de p et la variance de q mesure la confiance que nous pouvons avoir dans cette estimation.

L'approche *Variational Bayes* (VB, (Bea03)), anciennement appelées *ensemble learning* (Mac95 ; Die00), approxime p **et** fournit une borne sur la preuve. L'intégration est ainsi remplacée par une tâche plus simple : maximiser la borne, en la rendant aussi proche que possible de la vraie preuve.

Cette méthode s'applique en présence de variables cachées $\mathcal{H}$. La convexité du logarithme donne l'inégalité de Jensen pour toute fonction q :

$$\log p(\Delta|\mathcal{M}) \geq \int q(\mathcal{H},\Theta)\ \log \frac{p(\Delta,\Theta,\mathcal{H}|\mathcal{M})}{q(\mathcal{H},\Theta)}\, d\mathcal{H}\, d\Theta. \tag{1.56}$$

Le terme de droite est donc une borne inférieure à la preuve de la vraie jointe. L'idée est de choisir une fonction q simple rendant cette borne la plus grande possible. Dans l'approche *Variational Bayes*, q se factorise comme le produit d'une fonction des variables cachées $\mathcal{H}$ et d'une fonction des paramètres Θ :

$$q(\mathcal{H},\Theta) := q_{\mathcal{H}}(\mathcal{H})\ q_{\Theta}(\Theta). \tag{1.57}$$

Alors la borne inférieure est une fonctionnelle que nous pouvons noter :

$$B(q_{\mathcal{H}}, q_{\Theta}, \Delta) := \int q_{\mathcal{H}}(\mathcal{H})\ q_{\Theta}(\Theta)\ \log \frac{p(\Delta,\Theta,\mathcal{H}|\mathcal{M})}{q_{\mathcal{H}}(\mathcal{H})\ q_{\Theta}(\Theta)}\, d\mathcal{H}\, d\Theta. \tag{1.58}$$

La méthode consiste à optimiser cette borne itérativement par rapport à $q_{\mathcal{H}}$ et à q_{Θ}. Une fois l'optimisation (locale) terminée, q sera une bonne approximation de p car elle minimisera la divergence de Kullback-Leibler[8] $D_{\mathrm{KL}}(\, q \,||\, p\,)$. En effet :

$$\log p(\Delta|\mathcal{M}) - B(q_{\mathcal{H}}, q_{\Theta}, \Delta) = \int q_{\mathcal{H}}(\mathcal{H})\ q_{\Theta}(\Theta)\ \log \frac{q_{\mathcal{H}}(\mathcal{H})\ q_{\Theta}(\Theta)}{p(\Theta,\mathcal{H}|\Delta,\mathcal{M})}\, d\mathcal{H}\, d\Theta = D_{\mathrm{KL}}(\, q \,||\, p\,). \tag{1.59}$$

Au final la maximisation de la borne B est séquentielle et alterne itérativement deux étapes :

$$q_{\mathcal{H}}^{t+1}(\mathcal{H}) :\propto \exp\left[\int \log p(\mathcal{H},\Delta \mid \Theta,\mathcal{M})\ q_{\Theta}^{t}(\Theta)\, d\Theta\right] \tag{1.60}$$

$$q_{\Theta}^{t+1}(\Theta) :\propto p(\Theta \mid \mathcal{M})\ \exp\left[\int \log p(\mathcal{H},\Delta \mid \Theta,\mathcal{M})\ q_{\mathcal{H}}^{t}(\mathcal{H})\, d\mathcal{H}\right]. \tag{1.61}$$

Nous pouvons remarquer des similarités avec l'algorithme EM (DLR77). En effet l'étape 1.60 correspond à sa phase E (*Expectation*) et l'étape 1.61 à sa phase M (*Maximisation*). En fait, EM est un cas particulier de VB lorsque nous ne gardons pas une distribution entière $q_{\Theta}(\Theta)$ sur les paramètres, mais seulement une estimation ponctuelle Θ^*. Dans ce cas $q_{\Theta}(\Theta) = \delta(\Theta - \Theta^*)$ avec Θ^* une valeur optimale déterminée en phase M. L'algorithme EM, dont le but est de maximiser $p(\Theta \mid \Delta,\mathcal{M})$ par rapport à Θ est alors une itération des étape E et M suivantes :

$$q_{\mathcal{H}}^{t+1}(\mathcal{H}) := p(\mathcal{H} \mid \Delta, \Theta^t) \tag{1.62}$$

$$\Theta^{t+1} := \mathrm{ArgMax}_{\Theta} \int q_{\mathcal{H}}^{t}(\mathcal{H})\ \log p(\mathcal{H},\Delta,\Theta) d\mathcal{H}. \tag{1.63}$$

[8] Nous donnerons plus de détails sur cette divergence au chapitre 3.

Comme pour EM, les itérations de VB garantissent la croissance de la borne et donc la convergence de l'algorithme. Cependant, cette convergence sera locale et il est rarement possible de connaître la valeur du maximum global.

Dans notre travail, nous mettrons en place au chapitre 4 une inférence de type variationnelle pour le filtrage collaboratif. Il s'agira en fait d'un algorithme EM car nous maximiserons les paramètres Θ en phase M. De plus, pour rendre possible les calculs nécessités par la phase E, nous devrons considérer une distribution sur les variables cachées $q^t_{\mathcal{H}}(\mathcal{H})$ complètement factorisée. Cette factorisation complète est souvent appelée méthode de *Mean Field* en rapport à son introduction initiale en physique statistique (Cha87).

Notons finalement que lorsque $q_{\mathcal{H}}$ et q_{Θ} sont complètement factorisées et lorsque la famille $\mathcal{F}$ est exponentielle, l'approche VB peut se réduire à un algorithme de passage de messages locaux (WB05).

Méthodes variationnelles : Expectation Propagation

Expectation Propagation (EP, (Min01c)) est une méthode variationnelle déterministe dont la différence avec VB est la fonctionnelle minimisée. En première approximation, VB minimise $D_{\mathrm{KL}}(\, q \,||\, p \,)$ alors que EP minimise $D_{\mathrm{KL}}(\, p \,||\, q \,)$. Cette différence a plusieurs conséquences en terme de précision de l'approximation, de choix de la famille $\mathcal{F}$ et de temps de calcul. Nous détaillerons le fonctionnement de EP au chapitre 3 où nous l'utiliserons pour classer des joueurs d'échecs.

1.7 Apprentissage

Dans cette section, nous nous intéressons plus précisément à la notion d'apprentissage. Nous commençons par lister les méthodes les plus courantes (Mit97) en les triant en trois catégories.

1.7.1 Méthodes non probabilistes

Les méthodes non probabilistes sont les plus anciennes, et datent souvent de la première vague de la cybernétique (1950-1975). Nous pouvons citer les systèmes experts (Jac86), les systèmes de *classifiers* (Hol75) et les arbres de décision comme C4.5 (Qui93).

Nous plaçons aussi dans cette catégorie les réseaux de neurones artificiels (Bis95) et les méta-heuristiques comme les méthodes évolutionnistes (Koz92) et le recuit simulé (KGV83), bien que ces dernieres soient plus des algorithmes d'optimisation que d'apprentissage.

D'autres méthodes simples comme la prédiction par les K plus proches voisins (SDI06), ou la classification par l'algorithme du *K-mean* (Mac67) peuvent être très efficaces dans certains cas. Elles peuvent aussi servir à initialiser d'autres algorithmes plus complexes. Notons d'autre part que l'algorithme du K-mean peut être vu comme un EM dégénéré.

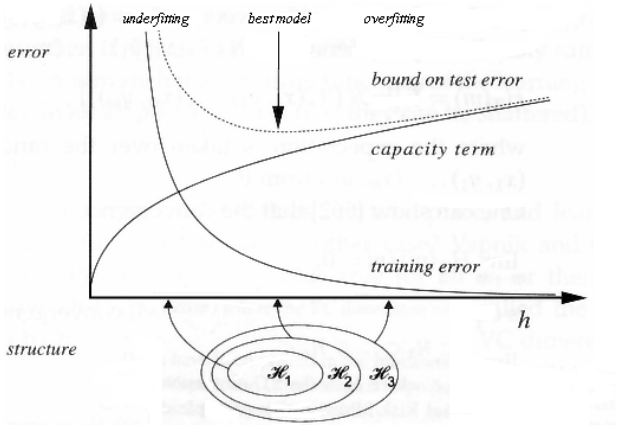


FIG. 1.7 – Minimisation du risque structurel. Un modèle (h) trop complexe (à droite) aura un risque empirique (*training error* sur le graphe) faible, mais sa VC-dimension (*capacity term*) sera si grande que le risque structurel (*test error*) sera trop fort. Il y aura *over-fitting*. À l'opposé un modèle de trop faible complexité aura un risque empirique trop grand. L'idéal est donc de minimiser la somme de ces deux termes (image de http://www.svms.org/).

1.7.2 Méthodes probabilistes objectives

Tout d'abord nous pouvons considérer que les méthodes de statistiques fréquencistes (comme les estimateurs sans biais) sont aussi des méthodes d'apprentissage, dans la mesure où elles permettent de résumer l'information contenue dans un grand nombre de données.

Mais nous plaçons surtout dans cette catégorie les travaux sur les propriétés calculatoires des algorithmes d'apprentissage comme la théorie de l'apprentissage statistique de Vapnik (Vap99) et le cadre PAC (*Probably Approximately Correct learning*) de Valiant (Val84).

L'approche de Vapnik est basée sur le principe de minimisation du risque structurel. C'est un principe inductif permettant de sélectionner un modèle après n'avoir vu qu'une quantité limitée de données. Le risque structurel est une fonction convexe de la flexibilité du modèle. Un modèle flexible est un modèle complexe, avec beaucoup de paramètres, qui peut donc s'adapter fortement aux données. Un modèle trop rigide n'est pas intéressant, car il ne va rien apprendre des données (*under-fitting*).

À l'opposé une trop grande flexibilité est aussi un vrai problème. En effet, si un modèle est trop souple, il est possible qu'un phénomène de sur-apprentissage apparaisse. Ce phénomène (*over-fitting*) est un problème récurrent en apprentissage : si nous entraînons un modèle avec beaucoup de paramètres sur un petit jeu de données, le modèle va tellement bien s'y adapter qu'il n'aura pas un bon pouvoir de généralisation à d'autres données. Le modèle aura appris par cœur ce jeu de données avec son bruit, sans réellement en extraire le signal utile. Ce phénomène est aussi appelé dilemme biais-variance en statistique.

Pour éviter le sur-apprentissage, le modèle devra donc être un compromis entre adéquation aux données et pouvoir de généralisation. Vapnik propose une mesure de complexité pour détecter le sur-apprentissage : la VC-dimension (ou terme de capacité). Si cette me-

sure est grande, alors le modèle est très flexible. Le risque structurel à minimiser est alors la somme du risque empirique (sur le jeu de données d'apprentissage) et de la VC-dimension (fig. 1.7). Cette approche théorique a produit un algorithme de classification très performant : les SVMs (*Support Vector Machine*) (BGV92). Les SVMs sont des classificateurs linéaires, mais en introduisant une fonction de noyau (*kernel trick*) (ABR64), ils peuvent s'appliquer à de nombreux problèmes.

Le problème du sur-apprentissage est traité de différentes façons dans les autres approches. Par exemple, il est recommandé d'arrêter l'apprentissage des poids d'un réseau de neurones avant la convergence (*early stopping*). Il est aussi possible d'introduire explicitement une contrainte sur la complexité d'un modèle en ajoutant un terme de régularisation à la fonction de coût. Le choix du terme de régularisation est toujours assez délicat. Nous avons aussi vu que le critère BIC comportait un terme pénalisant les modèles les plus complexes. Toutes ces méthodes peuvent être vues comme une instanciation du principe du rasoir d'Occam (1288-1349). Appliqué à l'apprentissage, ce principe de parcimonie affirme qu'entre deux explications également vraisemblables, il ne faut en conserver qu'une : la plus simple.

L'autre grande branche des méthodes probabilistes objectives est le cadre PAC qui a conduit à l'introduction des méthodes de *boosting* comme AdaBoost (FS95) permettant de combiner de médiocres algorithmes d'apprentissage afin de former un algorithme plus performant.

1.7.3 Méthodes probabilistes subjectives : Apprentissage bayésien

Nous pouvons déjà noter que plusieurs méthodes précédemment citées peuvent être décrites en termes bayésiens, ou que des extensions bayésiennes en ont été proposées. Par exemple, en appliquant la méthode bayésienne aux réseaux de neurones, Neal (Nea96) a montré qu'ils étaient équivalent à des processus gaussiens (RW05) quand le nombre de neurones de la couche cachée tend vers l'infini. D'autre part, les RVMs (*Relevance Vector Machine*) (Tip00) ou les BPMs (*Bayes Point Machines*) (HGC01) sont ses équivalents des SVMs qui égalent leurs performances sur certains jeux de données, tout en apportant le confort théorique du cadre subjectif. De nombreuses autres méthodes de traitement de données, comme les régressions aux moindres carrés, l'analyse en composantes principales, peuvent être interprétées en termes bayésiens (Bis06).

L'apprentissage bayésien consiste simplement à appliquer la règle de Bayes pour calculer des distributions *a posteriori* sur des quantités qui nous intéressent en marginalisant les autres variables. Comme il n'y a pas de différence fondamentale entre variables et paramètres, il est aisé de formuler un problème d'apprentissage en terme bayésiens : il ne s'agit que d'inférence. De plus, le contexte temporel de l'apprentissage se reflète dans la notion d'*a priori* et d'*a posteriori*. L'apprentissage consiste à extraire de l'information de données afin de mettre à jour sa connaissance, en passant d'un *a priori* à un *a posteriori*.

L'apprentissage bayésien est une méthode avec de solides fondations théoriques, incré-

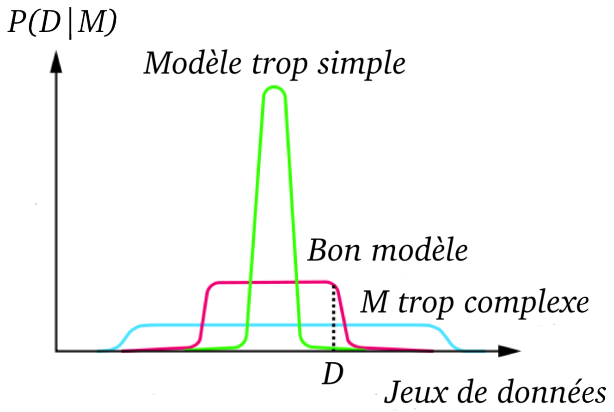


FIG. 1.8 – Rasoir d'Occam automatique. Pour un certain jeu de données $D := \Delta$, certains modèles seront trop complexes, d'autres trop simples. Cette représentation est schématique car l'axe des abscisses correspond à l'ensemble des jeux de données possibles (source : Ghahramani).

mentale, efficace, et hiérarchique. Surtout elle apporte une réponse élégante au problème du sur-apprentissage en fournissant une sorte de rasoir d'Occam automatique, sans que nous ayons besoin d'invoquer un quelconque principe extérieur.

En effet, si nous devons choisir entre deux modèles de complexité différente[9], nous devons considérer le ratio :

$$\frac{p(\mathcal{M}_1|\Delta)}{p(\mathcal{M}_2|\Delta)} = \frac{\int p(\Delta|\Theta_1,\mathcal{M}_1)\, p(\Theta_1|\mathcal{M}_1)\, d\Theta_1}{\int p(\Delta|\Theta_2,\mathcal{M}_2)\, p(\Theta_2|\mathcal{M}_2)\, d\Theta_2} \cdot \frac{p(\mathcal{M}_1)}{p(\mathcal{M}_2)}. \tag{1.64}$$

Le second terme est le ratio des *a priori*, et il est parfois intéressant d'assigner une probabilité plus faible au modèle plus complexe. Mais ceci ne doit être fait que si nos connaissances préliminaires nous y poussent. Il n'y a pas de raison de limiter *a priori* la complexité des modèles dans le cadre bayésien. En effet, une méthode bayésienne bien menée, sans approximation ni estimation ponctuelle, ne pose pas de problème de sur-apprentissage en raison de la marginalisation présente dans le premier terme.

> Bayesian methods don't overfit, because they don't fit anything !
> - Zoubin Ghahramani -

En effet, en considérant la vraisemblance *moyenne* sur l'espace des paramètres et non la vraisemblance *maximale*, ce terme va automatiquement pénaliser les modèles dont l'espace des paramètres est très grand, c'est à dire les modèles complexes, avec beaucoup de paramètres. En d'autres termes, comme nous ne considérons pas qu'un seul modèle à chaque fois, mais une multitude (un par paramètre), il est fort probable que pour certains paramètres un modèle complexe collera bien aux données et aura une grande vraisemblance.

[9]Nous considérons que ce choix est imposé, car en général la méthode conseille de garder les deux modèles avec leur probabilités *a posteriori*.

Mais pour de très nombreuses autres configurations des paramètres cette vraisemblance sera très faible et donc la vraisemblance marginale (moyenne) sera petite. Une autre façon de comprendre ce phénomène est schématisée par la figure 1.8 : comme la preuve est normalisée, un modèle simple, c'est-à-dire peu flexible, ne donnera une vraisemblance non nulle qu'à un petit ensemble de jeux de données Δ. Mais dans ce cas, elle sera plus importante que celle donnée par un modèle complexe qui "disperse" sa masse.

1.8 Conclusion

Dans ce premier chapitre, nous avons décrit la méthode bayésienne et ses fondations philosophiques. Bien que parmi la communauté des statisticiens, le terme "bayésien" recouvre différentes significations, nous considérons que ce qui la caractérise le plus fondamentalement est l'emploi de probabilités subjectives comme représentation de degrés de croyances. Le théorème de Cox-Jaynes définit alors un ensemble de règles claires et cohérentes pour manipuler ces probabilités. À la différence des statistiques fréquencistes qui utilisent des méthodes personnelles pour mettre à jour des probabilités impersonnelles, la méthode bayésienne utilise des méthodes impersonnelles pour mettre à jour des probabilités personnelles. Cette approche présente des avantages et des inconvénients que nous résumons ci-dessous.

1.8.1 Avantages

Le premier point fort de la méthode bayésienne est son élégance philosophique : une probabilité est une mesure d'incertitude subjective et ceci ne préjuge pas de l'existence éventuelle d'un hasard fondamental du monde physique. Il n'est pas nécessaire de supposer l'existence d'expériences aléatoires répétables pour réaliser des inférences inductives.

De plus, les règles de manipulation des probabilités fournissent une méthode automatique, cohérente, exempte de paradoxes et qui ne nécessite pas d'introduction d'autres principes *ad-hoc*. Les mêmes règles s'appliquent que nous possédions beaucoup ou peu de données et la loi de Bayes fournit une méthode élégante pour combiner nos connaissances *a priori* avec les informations provenant de ces données.

Un autre avantage est la facilité à communiquer et à comprendre les résultats subjectifs. Par exemple il est beaucoup plus intuitif de considérer des intervalles de crédibilité (concentrant 95% de la probabilité de trouver la vraie valeur) que des intervalles de confiance fréquencistes. Il est d'ailleurs notable que ces intervalles de confiance (et les p-valeurs) sont souvent interprétés à tort comme des intervalles de crédibilité bayésiens.

La règle de Bayes permet aussi des approches incrémentales (l'*a posteriori* à l'issue d'une expérience peut servir d'*a priori* pour la prochaine expérience) et hiérarchiques. Les modèles hiérarchiques, considérant les paramètres comme des variables aléatoires, permettent de partager de l'information entre différents sous-modèles. Cette hiérarchisation, qui peut s'étendre sur plusieurs niveaux, est très utilisée en sciences sociales pour modéliser des populations à différentes échelles. Par exemple, pour inférer la taille moyenne des hommes de différentes villes, il est intéressant d'introduire un paramètre de taille pour chaque ville

et de les lier par un hyper-paramètre au niveau du pays. Ainsi l'information récoltée en différents lieux peut servir à estimer la taille des hommes d'une ville pour laquelle nous n'avons que peu de données.

Plus techniquement, la règle de la somme permet de traiter aisément les paramètres de nuisance, en les marginalisant. Il n'existe pas de méthode équivalente pleinement satisfaisante avec les statistiques classiques (Lor99). Lors de la comparaison de modèles, cette marginalisation introduit aussi une pénalité pour les modèles les plus complexes. Cette régularisation est appelée rasoir d'Occam automatique.

1.8.2 Inconvénients

Une des critiques les plus courantes à l'encontre de la méthode bayésienne est justement sa subjectivité ; le but de la science étant d'obtenir des résultats indépendants du scientifique. Nous pensons que cette subjectivité a néanmoins l'avantage d'être explicite, obligeant l'utilisateur à établir clairement quels sont ses *a priori*. Une fois ces connaissances exprimées, l'application des règles d'inférence est automatique. En revanche, l'approche bayésienne est difficile à suivre d'une façon parfaitement rigoureuse pour deux raisons.

D'abord les connaissances *a priori* d'un agent sont souvent floues, mal formulées et leur traduction en un modèle probabiliste numérique est très difficile. Les hypothèses posées sont souvent fausses, il est nécessaire d'affiner les modèles après avoir vu leurs conséquences. Ce problème est spécialement critique lorsqu'il faut formuler des *a priori* en grande dimension, car l'intuition y est souvent mise en défaut. C'est pourquoi il est recommandé de mettre en place un cycle modélisation-inférence-évaluation permettant d'améliorer progressivement les modèles.

Il existe aussi le problème dit du "monde fermé" (*close world asumption*). Si le bon modèle n'a pas été considéré dès le départ, les inférences ne l'inventeront pas. Pour être parfaitement rigoureux, il faudrait considérer tous les modèles possibles *avant* de voir les données, afin d'être certain de ne pas en avoir oublié. Car oublier de considérer un modèle est équivalent à lui assigner une probabilité *a priori* nulle et donc une probabilité *a posteriori* tout aussi nulle, quelle que soit sa vraisemblance.

Le troisième grand problème de l'approche bayésienne est la difficulté calculatoire des inférences. Il est très tentant de définir un beau modèle génératif pour un problème, mais si il n'existe pas d'algorithme efficace suffisamment précis, ce modèle n'a pas beaucoup de valeur. Tout l'art est de trouver un compromis entre biais du modèle et faisabilité des inférences. Pour des problèmes difficiles, il faudrait en fait penser à la méthode d'inférence dès la phase de modélisation.

Deuxième partie

Apprentissage de paramètres

Chapitre 2

Introduction

Comme nous l'avons vu dans l'introduction, les problèmes d'apprentissage peuvent être classés selon leurs difficultés. Dans ce chapitre nous étudions les problèmes les plus aisés selon cette classification : ceux concernant un apprentissage de paramètres, en particulier lors de la présence de variables cachées.

2.1 Apprentissage paramétrique simple

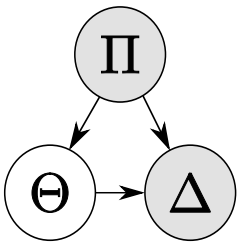


Fig. 2.1 – Réseau bayésien d'un problème paramétrique simple. Les variables observées sont Δ, Π représente le modèle et les *a priori*, enfin Θ est un jeu de paramètres.

Par apprentissage paramétrique simple, nous entendons la recherche d'un *a posteriori* et son utilisation, pour un jeu de variables appelées paramètres, lorsqu'il n'y a pas de variables cachées à marginaliser (fig. 2.1).

L'inférence se ramène à l'application de la règle de Bayes :

$$p(\Theta \mid \Delta, \mathcal{M}) = \frac{p(\Delta \mid \Theta, \mathcal{M})\, p(\Theta \mid \mathcal{M})}{p(\Delta \mid \mathcal{M})}. \tag{2.1}$$

Les calculs peuvent parfois être menés de façon analytique, comme pour l'exemple de l'inférence du biais d'une pièce de monnaie détaillé au chapitre 1.

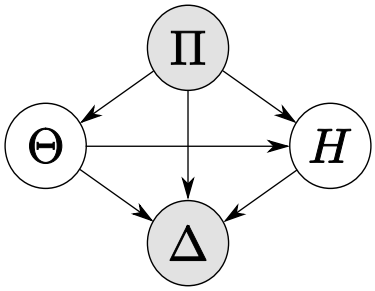


FIG. 2.2 – Réseau bayésien d'un problème paramétrique à variables cachées. Les variables observées sont Δ, les variables cachées $\mathcal{H}$, Π représente le modèle et les *a priori*, enfin Θ est un jeu de paramètres.

2.2 Apprentissage paramétrique à variables cachées

De façon générale, les problèmes à variables cachées se modélisent avec les ensembles (figure 2.2) :

- $\Delta = \{\delta_i\}$: l'ensemble des données observées,
- $\mathcal{H}$: l'ensemble des variables cachées non observées,
- Π : représente le modèle, ainsi que toutes les connaissances *a priori* utilisées.
- $\Theta = \{\theta_i\}$: un jeu de paramètres, souvent numériques. Par exemple, on peut avoir une variable δ_i gaussienne indépendante de $\mathcal{H}$: $P(\delta_i|\Pi, \Theta, \mathcal{H}) = \mathcal{N}(\delta_i; \mu, \sigma)$. Alors $\Theta = \{\mu, \sigma\}$

Nous pouvons alors classer ces problèmes en deux catégories :

Variables cachées explicites Les variables cachées ont une sémantique claire présente dans la formulation du problème concret, avant même sa modélisation par le statisticien.

Variables cachées introduites Les variables cachées sont introduites par le statisticien afin de faciliter les phases d'inférences ou de prédiction.

Notons que cette classification est quelque peu artificielle dans la mesure où elle repose sur la séparation des tâches entre le scientifique et le statisticien. Il est de plus fréquent de pouvoir associer une sémantique concrète aux variables artificiellement introduites.

En pratique, ces deux classes se distinguent par l'utilisation des résultats de l'inférence. Dans le premier cas, le but est d'obtenir de l'information sur les variables cachées en elles mêmes, dans le second nous sommes intéressés par une utilisation prédictive du modèle, en marginalisant les variables cachées.

Ces chapitres constituent une partie importante de nos contributions. Nous y détaillons un exemple concret pour ces deux types de problèmes.

Variables cachées explicites Pour les problèmes concernant des variables cachées explicites, le but est d'estimer un classement global d'entités à partir de comparaisons observées entre paires. Notre algorithme sera appliqué au classement de joueurs

d'échecs. Les inférences seront réalisées par un algorithme déterministe qui produira comme sortie une approximation des marginales des variables cachées.

Variables cachées introduites Pour les problèmes introduisant artificiellement des variables cachées, nous proposons deux modèles génératifs pour le filtrage collaboratif. Il s'agit de prévoir les goûts cinématographiques d'utilisateurs à partir de leurs appréciations passées. Les utilisateurs et les films seront regroupés en clusters et l'inférence sera réalisée par une méthode de type Espérance Maximisation (EM).

Chapitre 3

Variables cachées explicites : TrueChess

3.1 Introduction

3.1.1 Variables cachées explicites

Un problème à variables cachées explicites est un problème pour lequel l'existence et la sémantique de variables non observées sont introduites très tôt lors de la modélisation. Le but de l'inférence est d'obtenir de l'information sur ces variables par le calcul de leurs distributions de probabilités *a posteriori*. Dans de nombreux cas, la sémantique des variables cachées et des relations qu'elles entretiennent avec les variables observées est d'ordre causale. La règle de Bayes permet alors d'obtenir de l'information sur les causes à partir de la connaissance de leurs conséquences. C'est pour cette raison que cette règle est historiquement connue comme la "loi de probabilité des causes".

Cependant la règle de Bayes étant symétrique en ses arguments, elle s'applique même lorsqu'il n'y a pas de notion de causalité.

Pour modéliser ces problèmes, le scientifique fait l'hypothèse d'un ensemble de variables cachées expliquant les données observées. Ces variables ont une sémantique claire et l'inférence permet de mettre à jour leur connaissance après avoir observé des données. Pour exprimer formellement la sémantique de ces variables, le modélisateur décrit leur relation probabiliste avec les variables observées, en formulant leur fonction de vraisemblance, pouvant dépendre d'un jeu de paramètres $\mathbf{\Theta}$.

Certains problèmes, avec peu de variables discrètes, se résolvent assez facilement car l'inférence exhaustive y est possible. Ce n'est pas toujours le cas, et de nombreuses problèmes réels sont plus délicats. Nous présentons ici la résolution d'un problème de plus grande taille à l'aide d'inférences approchées.

3.1.2 Un modèle génératif pour le classement

Le problème à variables cachées étudié ici est celui du classement d'objets à partir de résultats de comparaisons entre paires. Plus concrètement, nous nous intéressons à l'évaluation et au classement des compétiteurs lors de tournois d'échecs. Nous proposons un modèle bayésien génératif pour estimer les cotations des meilleurs joueurs mondiaux lors des 150 dernières années. Nous montrerons que les systèmes actuellement utilisés et en particulier le système Elo peuvent être perfectionnés.

Nous présenterons d'abord le problème du classement et les différentes méthodes employées pour le résoudre dans le cadre des échecs. Nous verrons ensuite que ce problème peut être traité par une approche bayésienne. Nous présenterons alors notre modèle, son algorithme d'apprentissage associé, ses hypothèses et résultats.

3.2 Le problème du classement

3.2.1 Motivations

Dans le cadre de compétions, comme les sports d'équipe et individuels, les jeux de plateaux, les jeux vidéo, les joueurs désirent ardemment comparer leur niveaux relatifs. De nombreuses possibilités s'offrent pour estimer ces niveaux et ainsi classer les joueurs. Par exemple, il est possible d'attribuer des points lors d'un championnat où toutes les équipes s'affrontent deux à deux. Une autre possibilité est d'organiser une coupe, qui ne nécessitera qu'un nombre logarithmique de matchs pour permettre de définir un classement, qui ne sera cependant que partiel. Mais que faire lorsque l'organisateur ne contrôle pas l'établissement des matchs ? Comment établir un classement sur la base de résultats partiels ? Comment prendre en compte les matchs nuls ? Comment apprécier la variabilité du niveau d'un joueur ? En résumé la question est : **comment estimer de façon optimale un ordre global sous-jacents à partir d'un petit nombre de comparaisons locales non déterministes[1] et non transitives[2] ?**

L'obtention d'un tel ordre, ou mieux d'une mesure quantitative (cotation) des habilités des participants est utile pour différentes raisons. D'abord une telle estimation permet de proposer aux joueurs des adversaires de niveau similaire et ainsi d'organiser des matches équilibrés, d'issue incertaine, donc intéressants pour les joueurs et les spectateurs. Cet avantage devient décisif pour les jeux en ligne avec une grande communauté de joueurs. Dans le même esprit, les cotations permettent de présélectionner des participants pour des événements et tournois spéciaux. Finalement, les participants sont motivés pour faire progresser leur niveau, pour participer à des compétitions, pour rencontrer et apprendre de joueurs de niveaux supérieurs. Un retour précis sur l'évolution de son niveau permet en outre de mettre en place des stratégies d'entraînement plus efficaces. Pour toutes ces raisons, l'obtention d'une estimation précise du niveau des joueurs est nécessaire.

[1] A peut gagner contre B et perdre à la partie suivante.
[2] A bat B et B bat C ne signifie pas forcément que A va battre C

3.2.2 Classements historiques

Dans le cas particulier du monde des échecs, la précision de l'estimation est importante, spécialement lorsque nous cherchons à établir des comparaisons historiques. En effet, les meilleurs joueurs sont des professionnels qui consacrent leur vie à faire progresser leur niveau. D'interminables discussions ont lieu dans cette communauté pour répondre à des questions telles que :

- Qui est le meilleur joueur mondial à l'heure actuelle ?
- Qui était le meilleur joueur en 1991 ? Garry Kasparov ou Anatoly Karpov ?
- Emanuel Lasker (1868-1841), second champion du monde officiel, a gardé son titre durant 27 ans. Était ce justifié ? Dominait-il ses contemporains ?
- Qui aurait gagné si Kasparov avait rencontré Paul Morphy, meilleur joueur du XIX siècle ?
- Quelle est l'évolution typique d'un joueur durant sa carrière ? Termine-t-il toujours à son meilleur niveau ?
- Bobby Fischer était-il au paroxysme de son jeu lorsqu'il renonça aux compétions officielles en 1975 ?
- Les cotations Elo font apparaître une inflation moyenne des cotations pendant le XX siècle (figure 3.2). Cette tendance est elle réelle ou est-ce la conséquence d'un biais de la méthode ?
- Kasparov est-il vraiment le premier à avoir franchi la barrière mythique des 2800 points Elo ?
- Et finalement, qui est le meilleur joueur d'échecs de tous les temps ?

Toutes ces questions difficiles ne peuvent être étudiées que si l'on dispose d'une méthode rigoureuse de cotation. D'autres raisons nous incitent à étudier ce problème :

- des jeux de données de bonne qualité sont disponibles,
- une partie d'échecs se joue à un contre un, il n'y a pas d'équipe, ce qui simplifie la formulation du problème
- alors que ce problème se prête bien à l'approche statistique, étonnamment peu de travaux ont été réalisés,
- le système officiel (Elo) présente des faiblesses reconnues.

3.3 État de l'art

Le système actuellement utilisé dans la plupart des compétitions d'échecs est le système Elo, mis au point par Árpád Élo (1903-1992) dans les années 50 (Elo86). Avant cette contribution, la Fédération des États-Unis (USCF, United States Chess Federation) utilisait le système Harkness, un système de récompense assez juste, mais dont de certains résultats étaient vus comme inexacts par les experts. Le système Elo fut adopté par l'USCF en 1960 et par la Fédération Internationale des Échecs (FIDE) en 1970. Les différentes fédérations nationales, des communautés de jeux en ligne, des organisation sportives comme la FIFA

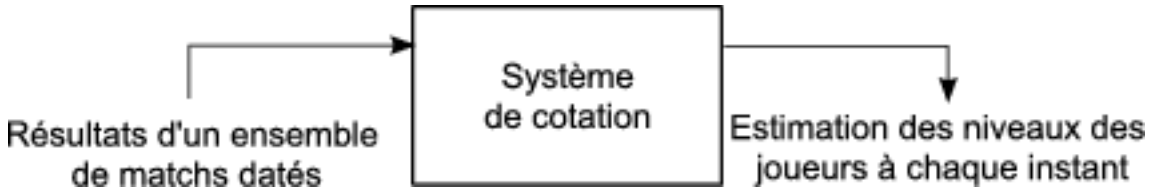


FIG. 3.1 – Les systèmes de cotation prennent en entrée les résultats de matchs et produisent en sortie une cotation réelle pour chacun des joueurs impliqués.

(Fédération Internationale de Football Association)[3] ont mis en place diverses variantes de ce système.

3.3.1 Généralités

Quelles sont les caractéristiques désirables d'un système de classement ? Un critère important est l'adéquation de ses estimations (et prédictions éventuelles) avec les données observées. Le système doit en outre être mathématiquement et théoriquement justifié. Cependant d'autres objectifs parfois contradictoires entrent en jeu. Par exemple, nous aimerions avoir un système compréhensible par les joueurs et d'utilisation aisée pour les organisateurs de tournois. La charge de calcul pour mettre à jour les estimations doit rester raisonnable. D'autre part le système doit être robuste aux tentatives de fraudes et doit motiver les joueurs à participer aux tournois. Les systèmes courants et en particulier Elo, proposent des compromis entre ces *desiderata*.

Nous croyons cependant que les besoins de rigueur mathématique et de pouvoir prédictif deviennent de plus en plus importants pour plusieurs raisons :

- grâce aux récents progrès du matériel informatique et des méthodes statistiques, des estimations précises sont accessibles avec des temps de calcul raisonnables,
- sans un modèle clair et rigoureux, il est très difficile de répondre à certaines questions comme celle de la cohérence temporelle des estimations,
- pour les joueurs et plus dramatiquement pour les meilleurs joueurs mondiaux qui consacrent leur vie aux échecs, les niveaux doivent être estimés le plus précisément possible et ne doivent pas dépendre de choix arbitraires. Des biais dus à des simplifications excessives ou à des modèles peu réalistes doivent être évités, éventuellement au prix d'un système plus complexe.

Tous les systèmes de cotation existants représentent le niveau d'un joueur par un nombre réel. Bien que le pouvoir d'expression d'une représentation unidimensionnelle soit manifestement trop faible pour refléter toutes les caractéristiques du jeu d'un participant, elle possède l'avantage d'être simple et de permettre des comparaisons directes. Un joueur possédant 2800 points Elo peut être considéré meilleur qu'un joueur de 2000 points Elo.

[3] Plus précisément, la FIFA utilise Elo pour classer les équipes nationales féminines mais pas les équipes masculines. Le système masculin ayant reçu de nombreuses critiques, des alternatives comme le World Football Elo Ratings (Run97) ont été proposées.

Nous pouvons supposer qu'une représentation 1D est assez puissante pour faire des prédictions précises, car la relation "a gagné contre" est presque transitive pour des joueurs "assez différents".

En d'autre termes, nous considérons raisonnable l'hypothèse que l'ensemble des joueurs soit localisé sur une variété quasi unidimensionnelle dans l'espace de toutes leurs caractéristiques.

Les différents systèmes de classement partagent aussi le type de leurs entrées, un ensemble de résultats de matchs avec leurs dates. Ni les informations sur le déroulement de la partie, ni la couleur jouée (blanc ou noir) ne sont prises en compte, bien qu'elles auraient permis de construire de meilleurs modèles. En effet, ces données ne sont souvent pas disponibles pour les anciennes parties. Ne pas en tenir compte permet par ailleurs d'obtenir un système général pouvant s'appliquer à d'autres contexte, comme le tennis, le football, les jeux de cartes et jeux vidéos.

Alors le but est d'utiliser l'information fournie en entrée le plus efficacement possible pour dériver une estimation du niveau de chaque joueur à chaque instant (figure 3.1).

Avant de décrire notre approche, nous présentons quatre méthodes utilisées par la communauté échiquéenne : Elo, Chessmetrics, Glicko et Edo.

3.3.2 Le système Elo

Elo : Modèle

Contrairement à ses prédécesseurs et à d'autres méthodes *ad hoc*, le système Elo est fondé sur une approche statistique. Formulé en terme bayésien, le niveau[4] de chaque joueur J_i est représenté par un variable cachée s_i (s pour *skill*). La performance effective d'un joueur de niveau s_i est une autre variable cachée p_i. Son introduction permet de modéliser la variabilité du comportement du joueur d'un match à l'autre et traduit ainsi l'incomplétude de la modélisation en une incertitude probabiliste. La performance est alors le résultat d'une perturbation gaussienne appliquée au niveau,

$$p_i|s_i \sim \mathcal{N}\left(p_i; s_i, \beta^2\right).$$

Bien que la variabilité β soit en réalité différente pour chaque joueur, Elo la considère comme constante.

En connaissant les performances de deux joueurs, Elo suppose que le gagnant est celui qui possède la plus grande performance. En notant $J_1 > J_2$ le fait que le joueur J_1 batte le joueur J_2, nous obtenons :

$$P(J_1 > J_2|s_1, s_2) = P(p_1 > p_2|s_1, s_2) = \Phi\left(\frac{s_1 - s_2}{\sqrt{2}\beta}\right)$$

[4]Dans le reste du document, nous emploierons indifféremment les termes *niveau* et *cotation* pour parler de la variable s_i.

où Φ est la fonction de répartition d'une gaussienne centrée réduite. En fait la plupart des variantes modernes remplacent Φ par une courbe logistique, car elle entraîne une meilleur adéquation avec les données réelles et simplifie les calculs.

Finalement les niveaux peuvent et doivent varier au cours du temps, mais la dynamique de cette évolution n'est pas explicitement décrite par le modèle Elo, elle résulte de la façon de mettre à jour les estimations et est contrôlée par un paramètre nommé facteur K.

La variante gaussienne de Elo est un modèle de type *Thurstone Case V* et sa version logistique un modèle de *Bradley-Terry* (Dav88).

Elo : Apprentissage

Le système Elo est essentiellement utilisé en mode *en ligne*, c'est-à-dire pour mettre à jour l'estimation des niveaux du temps $t-1$ vers une estimation au temps t à partir des résultats de nouveaux matchs. Après un match entre J_1 et J_2, les niveaux s_1 et s_2 sont mis à jour de façon à rendre le résultat du match plus probable. De plus, la contrainte $s_1 + s_2 = const$ est maintenue.

Soit $g = +1$ si J_1 gagne, $g = -1$ c'est J_2 et $g = 0$ si le match est nul. Alors la mise à jour Elo (linéarisée) est :

$$\begin{aligned} \delta &\leftarrow \alpha\beta\sqrt{\pi}\left(\frac{g+1}{2} - \Phi\left(\frac{s_1 - s_2}{\sqrt{2}\beta}\right)\right) \\ s_1 &\leftarrow s_1 + g\,\delta \\ s_2 &\leftarrow s_2 - g\,\delta \end{aligned} \tag{3.1}$$

où $O < \alpha < 1$ détermine le poids de la nouvelle donnée face à l'ancienne estimation. La valeur $\alpha\beta\sqrt{\pi}$ (facteur K) contrôle ainsi l'inertie des estimations, leur sensibilité aux nouvelle données, c'est-à-dire la vitesse d'apprentissage de Elo. C'est un paramètre libre du modèle qui est choisi empiriquement.

Cette formule de mise à jour est habituellement présentée sous une forme plus simple équivalente :

$$s_1 \leftarrow s_1 + K\,(R - Q)$$

avec R le nombre total de points gagnés par J_1 lors du tournoi[5] et Q son score attendu selon les anciennes cotations. Ce score attendu est la somme des scores attendus $Q = \sum Q_k$ pour chacun des matchs de J_1. Les Q_k sont soit donnés par la fonction Φ, soit par la fonction logistique :

$$Q_k = \frac{1}{1 + 10^{\frac{s_i - s_k}{400}}}.$$

Ainsi une cotation sera d'autant plus corrigée qu'elle est en contradiction avec le résultat des matchs. Si un grand maître bat un débutant, leurs cotations ne seront donc quasiment

[5] Deux points pour chaque victoire, un point pour les matchs nuls et zéro pour chaque défaite.

pas modifiées. Mais si le débutant l'emporte, les deux cotations varieront beaucoup car elles sont manifestement en contradiction avec l'expérience. Ainsi les cotations se rapprochent naturellement vers une meilleure image du vrai niveau des joueurs.

Elo : Avantages et Limitations

Les avantages du système Elo reposent sur ses fondations statistiques et sur l'aisance calculatoire de ses mises à jour. Le modèle est intuitif car un joueur gagnera d'autant plus de points que son match était difficile. La formule de mise à jour peut être appliquée avec une calculatrice de poche car la fonction logistique a fait l'objet d'une tabulation officielle. Cette facilité est très appréciée des joueurs.

Cependant, ce système possède plusieurs limitations de diverses natures, décrites ici par ordre croissant d'importance :

- Grâce aux calculateurs modernes, la linéarisation et la tabulation peuvent être évitées.
- Le modèle d'évolution étant implicite, le facteur K est choisi d'une façon assez arbitraire. À l'heure actuelle la FIDE assigne un facteur K différent selon que le joueur est débutant, confirmé ou grand maître[6]. La FIDE impose ainsi artificiellement une inertie des cotations croissante avec le niveau des joueurs.
- Pour ne pas perturber le système, les vrais débutants ne peuvent pas entrer dans le classement global avant la fin d'une période provisionnelle. Le but est de collecter assez de données afin d'avoir une estimation fiable de leur niveau avant de les introduire dans l'ensemble des joueurs cotés. En effet Elo est instable et sensible à l'introduction de joueurs manifestement mal cotés.
- L'évolution globale des niveaux au cours du temps n'est pas d'interprétation facile. En effet les cotations sont relatives à l'ensemble des joueurs courants. En raison de l'arrivée permanente de nouveaux joueurs (faible cotation) et du départ d'anciens joueurs (cotations élevées), la somme des cotations ne se conserve pas au niveau global et diminue au cours du temps. Lorsque cette déflation à été vérifiée par la FIDE et l'USCF, différentes corrections *ad hoc* ont été tentées. Durant les années 70 des points de bonus ont été donnés à certains joueurs et à la fin des années 80, l'USCF à décidé de mettre une borne inférieure à la cotation des joueurs[7], même s'ils perdaient toutes leurs parties.
- La déflation a aussi été expliquée par l'incapacité de Elo à suivre assez vite les jeunes joueurs progressant rapidement ainsi que par des effets pervers dus à l'introduction de ces bornes inférieures.
- Indépendamment, depuis le milieu des années 80, nous pouvons constater une inflation nette des cotations FIDE pour les meilleurs joueurs mondiaux (figure 3.2). Aucune explication consensuelle de cette inflation n'a été apportée (Edw06b).
- Elo ne produit pas de mesure de confiance dans la qualité de ces estimations. Ceci est la source de plusieurs problèmes et la justification de l'ajout de plusieurs règles

[6] Il est de 25 pendant une période provisionnelle, 15 jusqu'à 2400 points Elo et 10 ensuite.

[7] Ce palier fut fixé à 100 puis 200 points sous le meilleur niveau atteint.

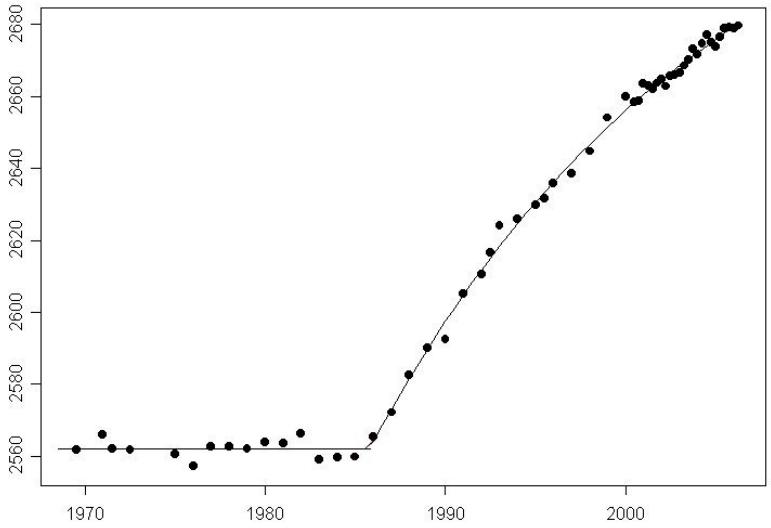


FIG. 3.2 – Moyenne des classements Elo de la FIDE du $11^{\text{ème}}$ au $50^{\text{ème}}$ meilleur joueur constatée par R. Edwards (Edw06b). Diverses explications ont été apportées : un accroissement réel du niveau des joueurs, 100 points bonus accordés aux femmes en 1986, l'arrivée de bons joueurs sous cotés de l'ex-URSS dans le classement mondial, l'augmentation du nombre de joueurs et donc de points à partager entre eux, un changement de méthode de calcul non publié. Aucune de ces explications n'est l'objet d'un consensus (Edw06b).

additionnelles correctives.

– La précision des prédictions faites à partir des cotations estimées est assez faible. En premier lieu, Elo ne peut pas prédire des matchs nuls, car il ne modélise pas cette éventualité. De plus, en étudiant les données, nous nous apercevons que les joueurs faiblement cotés gagnent plus souvent que prévu. La constance de la variabilité β peut être incriminée (GJ99).

Dans le but de minimiser ces problèmes, de faciliter l'emploi du système et d'éviter des estimations manifestement erronées, chaque fédération applique un jeu de règles additionnelles. Par exemple le règlement officiel de la FIDE (FID05) impose les choix suivants :

– Pour être valide, un tournoi doit apporter suffisamment d'informations (peu de joueurs non classés et peu de débutants),
– les cotations officielles ne sont publiées que quatre fois par an,
– les joueurs inactifs ne sont plus cotés,
– une mise à jour ne doit pas être calculée après chaque match, mais après chaque tournoi. Dans ce cas seulement le nombre de parties gagnées et le niveau moyen des adversaires sont pris en compte,
– la fonction Φ est discrétisée,

– les cotations sont arrondies à l'entier le plus proche,
– si le résultat d'un match est très surprenant (la cotation du perdant était supérieur de plus de 350 points à celle du gagnant), alors la mise à jour est artificiellement amoindrie.

3.3.3 Chessmetrics

J. Sonas propose Chessmetrics (Son05), un système "pondéré et protecteur" d'estimation de performances[8]. Désirant à la fois motiver les participants à jouer fréquemment et obtenir de bonnes performances prédictives, il met au point de nouvelles formules de mise à jour. Les paramètres introduits sont ajustés pour minimiser l'erreur de prédiction. Le but de ce système est de présenter des classements historiques et Sonas publie une quantité considérable de données brutes et prétraitées. Cependant Chessmetrics n'est pas fondé sur un modèle génératif. Par exemple la cotation d'un bon joueur devenant inactif va automatiquement revenir vers la moyennes des cotations. Pourtant, *a priori*, ce niveau n'a pas de raison particulière de décroître, mais Sonas veut encourager la participation aux tournois. Par ailleurs, il suggère une modification du facteur K et de la fréquence de mises à jour. Sonas propose aussi de remplacer la fonction Φ par une droite, échangeant ainsi la gaussienne de $P(p_i|s_i)$ par une fonction porte. Cette dernière modification ne nous parrait pas pertinente d'un point de vue modélisation, mais Sonas plaide argue d'une meilleure adéquation aux données, sous les hypothèses de son modèle.

3.3.4 Glicko 1&2

Comme Elo ne gère pas l'incertitude de ses évaluations, la FIDE à mis en place une période provisionnelle durant laquelle les débutants ne sont pas intégrés au système global. Avec Glicko et Glicko 2 (Gli99), Mark Glickman propose de modéliser explicitement cette incertitude, faisant de la cotation une variable aléatoire.

L'inférence bayésienne permet alors de calculer la distribution *a posteriori* sur la cotation à partir de son *a priori* et de sa vraisemblance. Glickman distribue la croyance sur une cotation de façon gaussienne :

$$s_i \sim \mathcal{N}\left(s_i; \mu_i, \sigma_i^2\right).$$

Comme σ_i^2 dépend de chaque joueur, les débutants peuvent être incorporés en leur assignant *a priori* une grande variance, notre connaissance de leur niveau étant très vague. Chaque match apportant de l'information, cette incertitude aura tendance à décroître. Glickman introduit aussi un modèle dynamique probabiliste liant la cotation d'un joueur aux instants t_1 et t_2,

$$s_i^{t_2}|s_i^{t_1} \sim \mathcal{N}\left(s_i^{t_2}; s_i^{t_1}, \gamma^2\right)$$

où γ mesure l'amplitude des variations possibles au cours de cet intervalle de temps. Ce modèle dynamique étant symétrique, il autorise *a priori* tout autant une progression qu'une

[8] *"A weighted and padded simultaneous performance rating"*(Son05).

régression entre deux instants. Comme pour Elo, Glickman utilise un modèle d'observation de *Bradley-Terry*, ne modélisant donc pas explicitement les matchs nuls. La méthode d'apprentissage est résumée par l'algorithme 1.

Algorithme 1 : Principales étapes de l'apprentissage de Glicko.

Entrées : Résultats pour I joueurs et N matchs
Sorties : Paramètres $\{\mu_i^t, \sigma_i^t\}$ des croyances *a posteriori* sur toutes les cotations

```
Assigner un a priori peu informatif sur les cotations à t = 0;
Discrétiser uniformément l'axe du temps;
Grouper les parties d'un même intervalle de temps en un tournoi;
pour chaque tournoi faire
    Calculer les {μ_i^t, σ_i^t} des joueurs impliqués dans le tournoi par une approximation
    de la vraisemblance de Bradley-Terry;
    Les modifier de façon à incorporer l'influence du modèle dynamique;
fin
Lisser les estimations passées en propageant l'information vers l'arrière, selon la
méthode du filtrage arrière des filtres de Kalman;
```

Ainsi Glicko améliore plusieurs aspects de Elo, en exprimant une incertitude sur les cotations, en modélisant la dynamique temporelle des niveaux et en propageant de l'information pour améliorer les cotations passées. Mais il ne constitue toujours pas un modèle complètement génératif, ne modélise pas correctement les matchs nuls, impose une discrétisation du temps et traite différemment les phases de propagation de l'information vers le futur et vers le passé.

3.3.5 Edo

Rod Edwards propose le système statistique Edo (Edw06a) pour classer les joueurs sur une base historique. Utilisant le modèle de *Bradley-Terry*, son approche se particularise par la modélisation de la dynamique des cotations. Au lieu de l'exprimer directement de façon probabiliste, il considère un même joueur à différentes époques[9] comme autant de joueurs différents, mais dont les cotations sont liées. Ces liens sont décrits comme un certain nombre de parties virtuelles ayant abouti à des matchs nuls. Ces parties imaginaires donnent une dynamique symétrique aux cotations, comme le bruit gaussien de Glicko. Une fois ces parties virtuelles introduites, le problème se réduit à celui d'un seul grand tournoi, sans dynamique, solvable par la méthode de *Bradley-Terry* (Dav88). Cette méthode ne calcule que les cotations de maximum de vraisemblance selon le modèle logistique mais Edwards a développé un algorithme efficace pour en estimer les variances.

De plus, pour éviter que des cotations extrêmes n'apparaissent, une correction est apportée afin de respecter une distribution satisfaisante des niveaux.

[9] Edo prend l'année comme intervalle de temps élémentaire.

La méthode Edo peut être interprétée d'un point de vue bayésien. Le modèle logistique est utilisé pour la vraisemblance, les parties virtuelles peuvent être vues comme des pseudo-données (à la manière d'une distribution conjuguée) et la correction est l'équivalent d'un *a priori* sur le domaine des possibles cotations. Edo a les avantages de présenter un modèle dynamique (bien que formulé étrangement), de ne pas dépendre de l'ordre temporel des matchs (symétrie par rapport à la direction de l'axe du temps) et de produire une mesure d'incertitude. En revanche il n'est pas capable de générer des parties nulles et le choix du nombre de parties virtuelles est délicat. De plus, la recherche du maximum de vraisemblance pour le tournoi géant nécessite une quantité de calculs importante. Rod Edwards applique sa méthode sur des données concernant des tournois de 1809 à 1902.

3.4 TrueChess : Modèle

Bien qu'indépendamment développé, notre modèle, TrueChess est proche du modèle Edo. Cependant, il est exprimé dans le formalisme général des modèles probabilistes bayésiens et l'algorithme d'apprentissage associé est différent. Le modèle et l'algorithme TrueChess sont une extension du modèle TrueSkill développé par R. Herbrich et T. Graepel (HG06).

Notre modèle considère que le vrai niveau des joueurs est inconnu et cherche à l'estimer par une distribution *a posteriori* gaussienne. Il produit donc une estimation de son incertitude, rendant inutile les périodes provisionnelles. C'est un modèle complètement génératif modélisant explicitement les parties nulles, pouvant ainsi être utilisé pour faire des prédictions. Il comporte un modèle dynamique gaussien symétrique comme celui de Glicko et son modèle de vraisemblance est plus proche du *Thurstone Case V* (performance gaussienne) que du cas logistique.

Notre méthode ne nécessite pas de discrétisation particulière de l'axe du temps et permet l'introduction de tournois (plusieurs parties à un même instant). Le traitement ne dépend pas de l'ordre des données et comme Edo, il permet que des matchs futurs modifient une estimation d'une cotation ancienne.

3.4.1 Un modèle bayésien

Avec la convention $\{x_i\} := \{x_i \, , i \in [\![1, I]\!]\}$, nous définissons les ensembles suivants pour N matchs, T tournois et I joueurs :

- ensemble des I joueurs $\{J_i\}$,
- ensemble des T dates de tournois $\{t_t\}$ (un et un seul tournoi par date),
- ensemble des N résultats de matchs $\{g_n\}$ (plusieurs matchs par tournoi, 2 joueurs par match),
- ensemble des $I \times T$ cotations $\{s_i^t\}$. Une seule cotation par joueur et par date. Une méthode d'estimation des cotations aux autres instants sera proposée dans la partie 3.5.8.

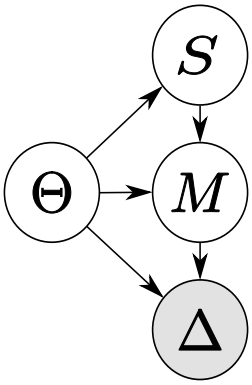


FIG. 3.3 – Dépendances de haut niveau dans le modèle TrueChess. Les variables observées sont Δ, les paramètres Θ, les variables cachées intéressantes S et les variables cachées intermédiaires M. La variable Π représentant le modèle choisi et tous les *a priori* spécifiés n'est pas représentée.

- ensemble de $I \times N$ performances $\{p_i^n\}$. Il y a une performance par joueur et par match l'ayant impliqué. Comme les joueurs ne participent pas à tous les matchs, $I \times N$ est une borne supérieure au cardinal de $\{p_i^n\}$.
- ensemble des N différences de performances $\{d_n\}$. Une variable d_n est introduite pour chaque match.

Les variables considérées par le modèle TrueChess sont donc $\{s_i^t, p_i^n, d_n, g_n\}$. Ce sont toutes des variables à valeur dans $\mathbb{R}$, sauf les $\{g_n\}$ qui sont à valeur dans $\{-1, 0, +1\}$. De plus, leurs interdépendances conditionnelles dépendront d'un jeu de paramètre Θ.

Les variables observées sont $\Delta = \{g_n\}$, résultats des N matchs. Elles prennent comme valeur 1,-1 ou 0 selon l'issue du match.

Parmi les variables cachées $H = \{s_i^t, p_i^n, d_n\}$, seul l'ensemble $S = \{s_i^t\}$ nous intéresse. Les autres variables cachées $M = \{p_i^n, d_n\}$ seront marginalisées. La figure 3.3 présente ce modèle.

La sémantique des variables est la suivante. Chaque s_i^t est le niveau du joueur J_i à l'instant t (pendant le tournoi). Les $\{p_i^n\}$ sont leurs performances au cours du match n. Comme il y a plusieurs matchs par tournoi, chaque joueur sera associé à plusieurs performances par tournoi. Cependant chaque joueur ne possède qu'une cotation par tournoi. Un match n'impliquant que deux joueurs J_i et J_j, nous introduisons la variable intermédiaire $d_n = p_i - p_j$. De même que précédemment, $g_n = +1$ si J_i gagne, -1 si c'est J_j et 0 en cas de match nul.

3.4.2 Modèle génératif d'une partie

Considérons une unique partie entre deux joueurs J_1 et J_2. Leurs cotations sont *a priori* indépendantes :

$$p(s_1, s_2) = p(s_1)\, p(s_2).$$

De plus, comme notre connaissance *a priori* sur ces valeurs est faible, nous leur assignons une distribution peu informative sous la forme d'une gaussienne à grande variance σ_0^2 :

$$s_i \sim \mathcal{N}(s_i; \mu_0, \sigma_0^2).$$

Remarquons que les valeurs précises de μ_0 et σ_0 sont arbitraires car elles définissent l'échelle et la valeur centrale de notre système de cotation. Nous choisissons $\mu_0 = 2700$ et $\sigma_0 = 100$ afin d'obtenir une échelle similaire à celle d'Elo.

Une fois ces *a priori* définis, la relation entre cotation et performance est gaussienne :

$$p_i|s_i \sim \mathcal{N}(p_i; s_i, \beta^2).$$

Le paramètre β, variabilité de la performance, est supposé identique pour tous les joueurs et tous les matchs.

Alors nous définissons la variable de différence entre les deux performances :

$$d = p_1 - p_2.$$

Le résultat g du match est alors :

$$g = \begin{cases} -1 & \text{si } d < -\epsilon \text{ } (J_2 \text{ gagne}) \\ 0 & \text{si } |d| \leq \epsilon \\ +1 & \text{si } d > \epsilon. \end{cases} \tag{3.2}$$

Ainsi nous avons défini un modèle génératif (Fig. 3.4) complet d'une partie, intégrant explicitement la possibilité de matchs nuls. Les paramètres de ce modèle sont $\Theta = \{\mu_0, \sigma_0, \beta, \epsilon\}$. Le graphe de facteur représente la décomposition de la jointe :

$$p(s_1, s_2, p_1, p_2, d, g) = p(s_1)\, p(s_2)\, p(p_1|s_1)\, p(p_2|s_2)\, p(d|p_1, p_2)\, p(g|d)$$

3.4.3 Modèle génératif d'un tournoi

Un tournoi est un ensemble de parties ayant lieu à la même date t. En pratique il peut s'agir de vrais tournois, mais cette notion peut être étendue. Par exemple si les dates fournies dans les données ne précisent que l'année de chaque partie, nous considérerons que tous les matchs d'une même année appartiennent au même tournoi.

Dans un tournoi chaque joueur n'est représenté que par une seule cotation s_i^t, mais il possède autant de performances p_i^n que de matchs joués. Ainsi nous obtenons un graphe bouclé du type de celui de la figure 3.5.

3.4.4 Modèle de plusieurs tournois

Pour modéliser l'évolution des niveaux de joueurs dans le temps, nous proposons deux modèles. Un modèle à variabilité constante :

$$s_i^{t_2}|s_i^{t_1} \sim \mathcal{N}(s_i^{t_2}; s_i^{t_1}, \gamma^2)$$

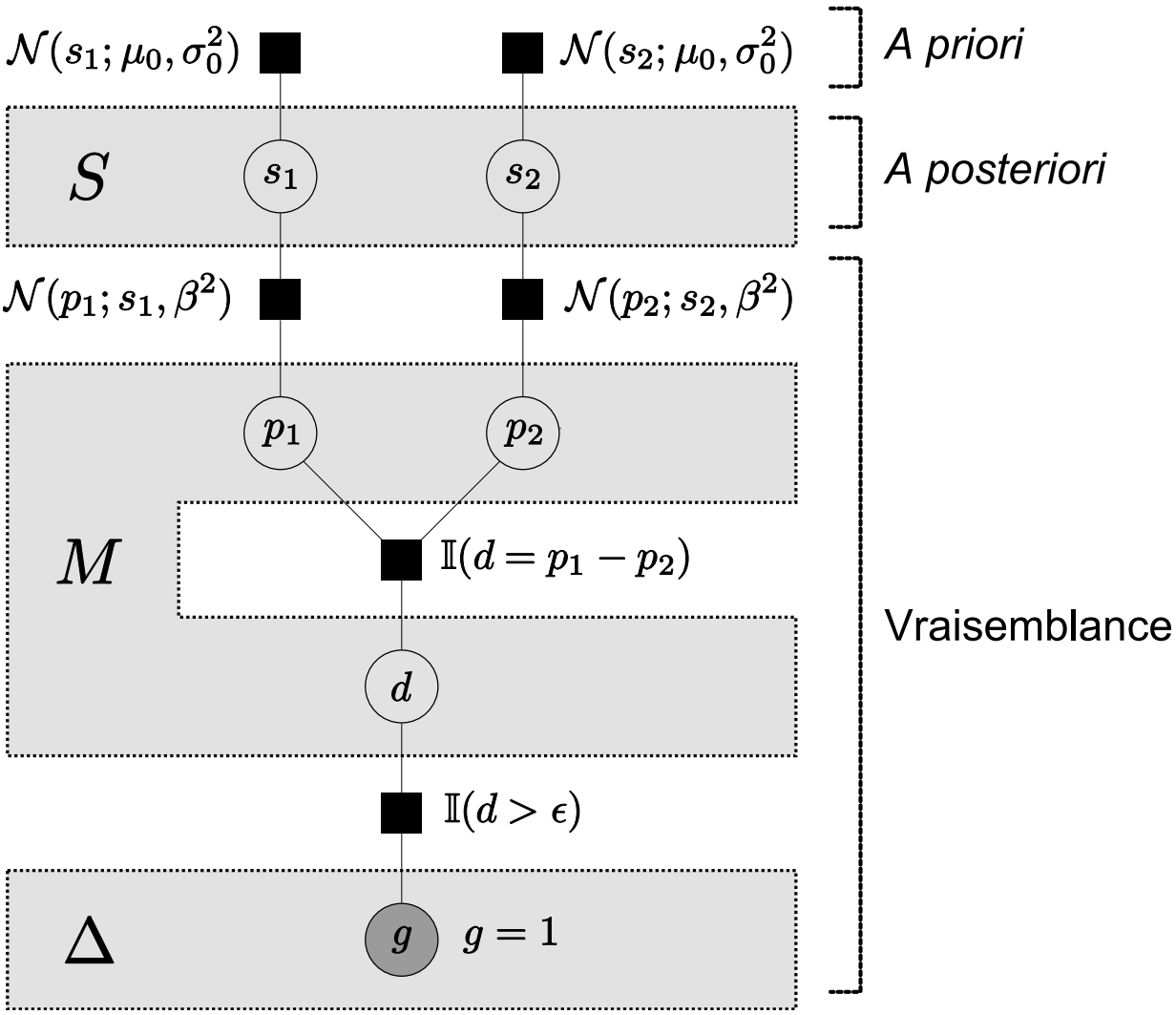


FIG. 3.4 – Graphe de facteur pour un match entre les joueurs J_1 et J_2. Nous avons observé que J_1 a gagné ($g = 1$) donc le facteur attaché à g impose $d > \epsilon$. Les *a priori* sur les cotations s_1 et s_2 sont gaussiens de paramètres μ_0 et σ_0. Le modèle de vraisemblance correspond à la partie inférieure du graphe. Les variables cachées intermédiaires (M) sont les performances et leur différence. Le but de l'inférence sera d'obtenir l'*a posteriori* sur s_1 et s_2. Comme g est observée et ne varie pas, nous pouvons nous passer de sa représentation et nous contenter du facteur qui lui est attaché. Finalement les paramètres de ce modèle sont $\Theta = \{\mu_0, \sigma_0, \beta, \epsilon\}$.

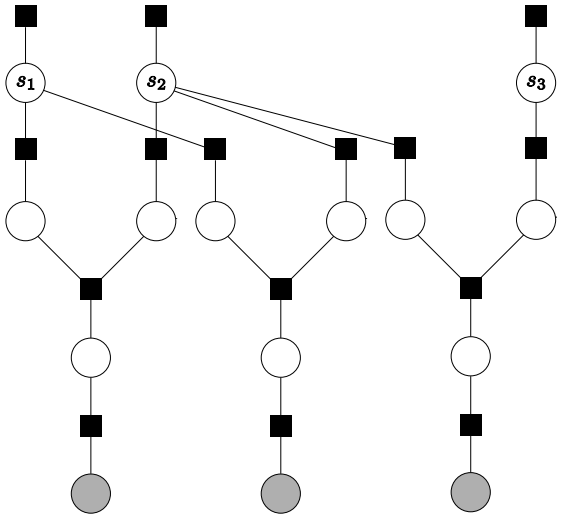


FIG. 3.5 – Graphe d'un tournoi à trois joueurs et trois matchs. Les joueurs J_1 et J_2 ont joué deux fois ensemble et J_2 a joué une fois contre J_3. Nous remarquons que ce graphe possède une boucle due aux deux matchs entre J_1 et J_2.

et un modèle brownien :

$$s_i^{t_2} | s_i^{t_1} \sim \mathcal{N}(s_i^{t_2}; s_i^{t_1}, \gamma^2 (t_2 - t_1)).$$

Le deuxième modèle, plus réaliste, autorise une variation de niveau proportionnelle au temps passé. Les deux modèles sont symétriques et locaux : ils ne privilégient *a priori* ni une augmentation, ni une diminution de cotation et suivent l'hypothèse de Markov d'ordre un. Dans les deux cas, γ est un nouveau paramètre (exemple figure 3.6).

3.4.5 Modèle complet

La figure 3.7 présente le modèle graphique complet pour un jeu de données constitué de 15 664 parties des 88 meilleurs joueurs de la période 1857-1991. Nous définissons autant de tournois qu'il y a d'années dans cette période. Cette discrétisation du temps n'est pas requise par le modèle ni par l'algorithme, mais par le jeu de données qui ne précise pas le jour exact des parties.

L'algorithme d'apprentissage devra estimer les distributions *a posteriori* de toutes les cotations. Cette inférence est difficile du fait des nombreuses boucles introduites par les participations multiples de chaque joueur et par les liens dynamiques.

3.5 TrueChess : Apprentissage

Une fois le graphe construit et les facteurs définis, le but de l'inférence est de calculer les distributions *a posteriori* de toutes les cotations $p(s_i^t|\Delta)$ en marginalisant sur toutes les autres variables. Il aurait aussi été possible de chercher à calculer la distribution jointe

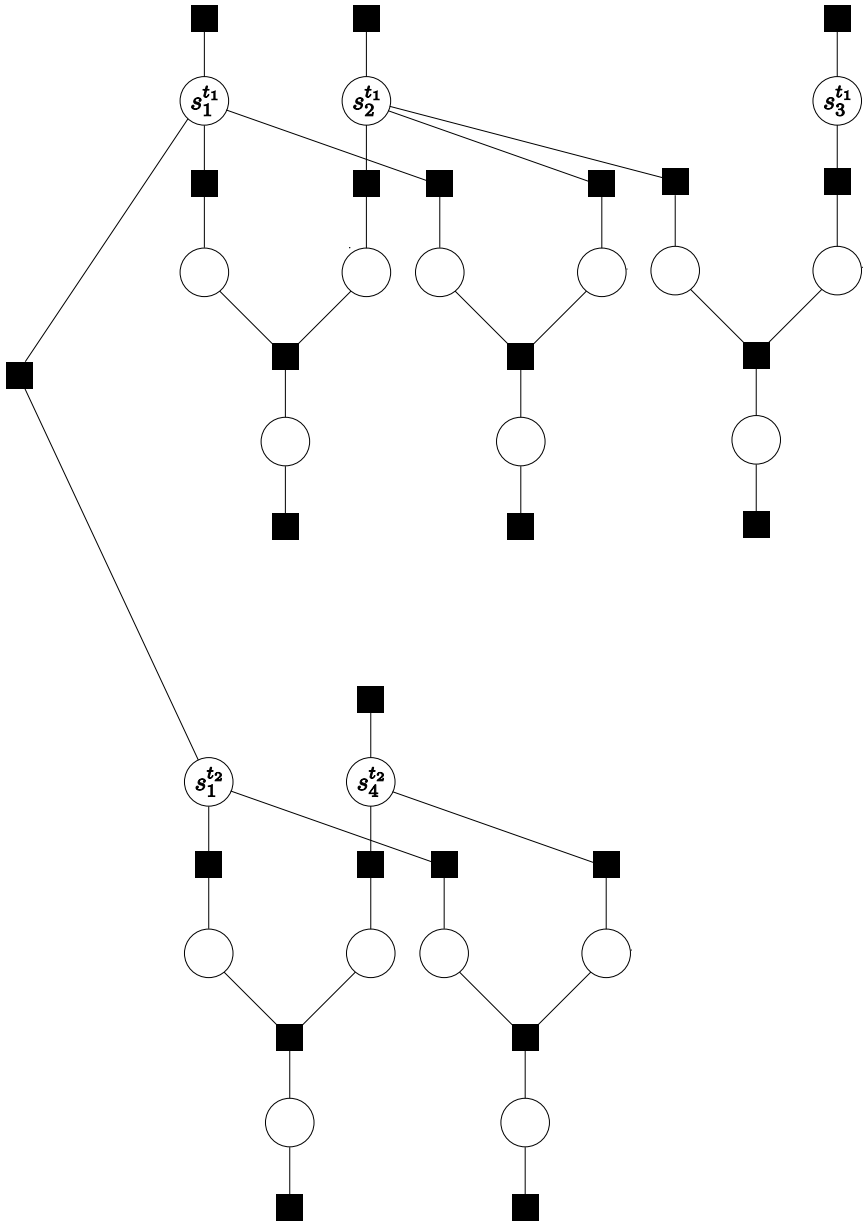


FIG. 3.6 – Graphe représentant deux tournois impliquant quatre joueurs. Le facteur liant $s_1^{t_1}$ et $s_1^{t_2}$ modélise la dynamique de l'évolution du niveau du joueur J_1. Les variables observées g_n ne sont pas représentées, mais leur valeurs déterminent les facteurs qui leur étaient liés (en bas de chaque tournoi).

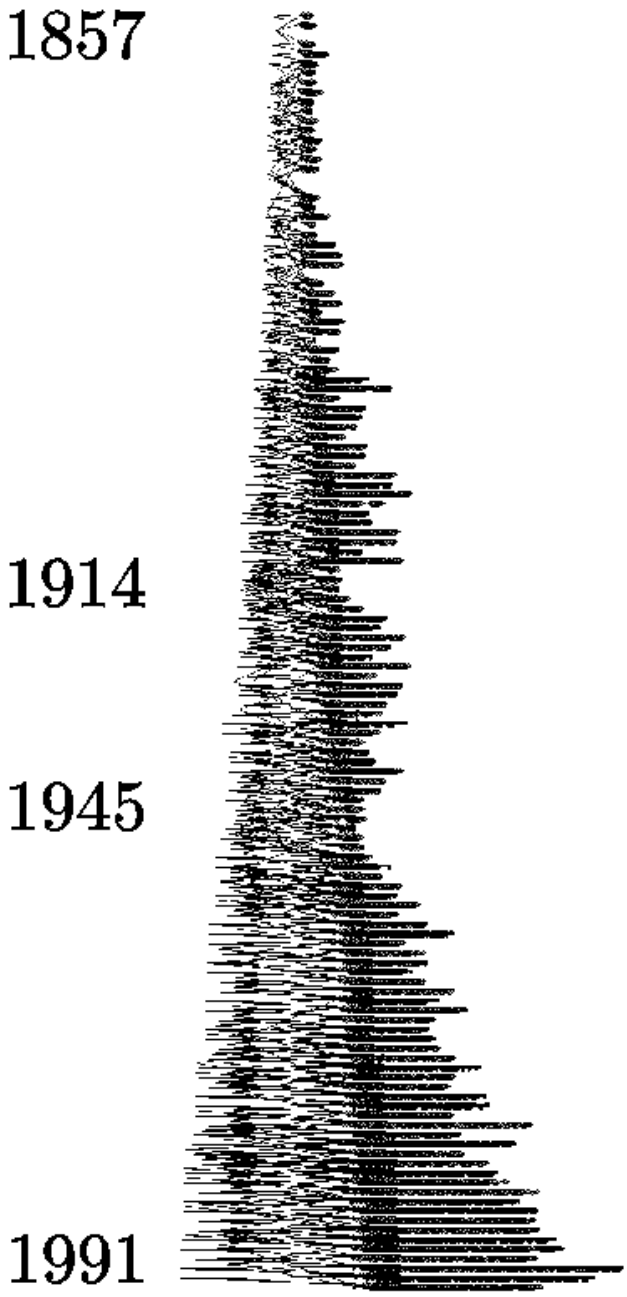


FIG. 3.7 – Modèle graphique complet. Tous les matchs d'une même année sont groupés en un seul tournoi. Chaque ligne correspond à un tournoi et les matchs les plus anciens sont en haut. Ce jeu de données présente 15 664 matchs de 88 joueurs. Le graphe possède environ 60 000 variables cachées et 150 000 arêtes. Nous notons que le nombre de matchs croit avec le temps. Les années correspondant aux deux guerres mondiales sont visibles.

$p(\{s_i^t\}|\Delta)$, mais son calcul exact et sa représentation sont délicats. De plus, dans le cadre des échecs, nous sommes intéressés par la connaissance de la cotation marginale de chaque joueur, indépendamment de celle des autres participants. Nous utilisons pour cela l'algorithme *expectation propagation* (*EP*) généralisant, entre autres, *loopy belief propagation* (*LBP*).

3.5.1 Expectation Propagation

EP (Min01b) est une méthode générale permettant d'approximer un produit incalculable

$$p(\mathbf{x}) = t_1(\mathbf{x}) \cdots t_n(\mathbf{x}) \tag{3.3}$$

par un produit calculable de termes approchés

$$\hat{p}(\mathbf{x}) = \hat{t}_1(\mathbf{x}) \cdots \hat{t}_n(\mathbf{x}), \tag{3.4}$$

les termes $\hat{t}_a(\mathbf{x})$ étant membres d'une famille $\mathcal{F}$ de fonctions plus simples. Cet algorithme initialise d'abord tous les termes et les réajuste itérativement et plusieurs fois jusqu'à convergence. Comme il approxime chaque terme $t_a(\mathbf{x})$ par un $\hat{t}_a(\mathbf{x})$, EP trouve une approximation globale de p par une multitude d'approximations locales. En toute généralité, une succession d'approximations locales n'implique pas une bonne approximation globale, mais sous certaines conditions, la qualité du résultat est suffisante.

Une approche naïve et peu précise pour réaliser les approximations locales est d'approximer chaque terme indépendamment, de sorte que $\forall a\ \hat{t}_a(\mathbf{x}) \approx t_a(\mathbf{x})$. Il est cependant plus précis de considérer le contexte de $t_a(\mathbf{x})$, son "voisinage".

Par exemple, les méthodes de filtrage de type *Assumed Density Filtering* (ADF (May79)), affine l'estimation de $\hat{t}_a(\mathbf{x})$ en utilisant le "passé", c'est-à-dire les $\hat{t}_b(\mathbf{x})$ $\forall b < a$. Les méthodes ADF sont issues de la théorie de l'automatique et un exemple classique est le filtre de Kalman (Kal60). Les filtres bayésiens en général fonctionnent sur ce principe.

EP généralise ADF, car il ne suppose pas d'ordre "temporel" et affine chaque terme en fonction de son contexte complet, "passé" et "futur". EP n'est pas un filtre, contrairement à ADF.

EP commence par initialiser tous les $\hat{t}_a(\mathbf{x})$ à une fonction arbitraire, par exemple la fonction constante 1. La suite est itérative. Nous choisissons un terme approché $\hat{t}_a^{\text{old}}(\mathbf{x})$, annulons son influence du produit global, le remplaçons par le vrai terme correspondant $t_a(\mathbf{x})$, calculons le nouveaux produit et finalement nous trouvons le nouveau terme approché $\hat{t}_a^{\text{new}}(\mathbf{x})$ qui aurait conduit à ce nouveau produit global, en minimisant une fonction de coût. Traditionnellement, la fonction de coût est une mesure de "divergence" entre distributions.

L'algorithme 2 présente EP dans toute sa généralité. Dans notre application, plusieurs contraintes facilitent cette procédure. D'abord $p(\mathbf{x})$ est une distribution de probabilités, c'est donc une fonction normalisée (ou du moins normalisable) positive. De plus, le vecteur $\mathbf{x}$ est le vecteur de toutes les variables aléatoires de notre modèle (cotations, performances, différences de performance et données observées). En notant $\mathbf{x} = (x_1, \ldots, x_k)$, les x_k sont

Algorithme 2 : Algorithme EP général.

Entrées : $p(\mathbf{x}) = t_1(\mathbf{x}) \cdots t_n(\mathbf{x})$
Sorties : $\hat{p}(\mathbf{x}) = \hat{t}_1(\mathbf{x}) \cdots \hat{t}_n(\mathbf{x})$

Initialisation : $\forall a\ \hat{t}_a(\mathbf{x}) \leftarrow 1$;
tant que *Critère de convergence non vérifié* **faire**
  Choisir un terme $\hat{t}_a^{\text{old}}$;
  Annuler son influence dans l'approximation courante $\hat{p}^{\text{old}}$;
    $\hat{p}^{\setminus a}(\mathbf{x}) := \frac{\hat{p}^{\text{old}}(\mathbf{x})}{\hat{t}_a^{\text{old}}(\mathbf{x})}$;
  Incorporer à sa place le terme exact t_a et trouver le nouveau terme approximé ;
    $\hat{t}_a^{\text{new}}(\mathbf{x}) \leftarrow \text{ArgMin}_{q \in \mathcal{F}}\ D\Big(\hat{p}^{\setminus a}(\mathbf{x})\ t_a(\mathbf{x}) \,||\, \hat{p}^{\setminus a}(\mathbf{x})\ q(\mathbf{x}) \Big)$;
fin

des variables réelles unidimensionnelles. De plus, $p(\mathbf{x})$ se factorise selon le modèle probabiliste que nous avons décrit et les termes exacts $t_a(\mathbf{x})$ sont les facteurs de notre graphe de facteur, ce sont les distributions conditionnelles.

En effet, comme p est un distribution jointe sur plusieurs variables, sa factorisation est déterminée par le modèle probabiliste, comme c'est le cas pour un réseau bayésien :

$$p(\mathbf{x}) = \prod_k p(x_k|\text{parents}(x_k)). \tag{3.5}$$

Pour TrueChess, comme nous utilisons le formalisme des graphes de facteurs, nous avons :

$$p(\mathbf{x}) = \prod_a t_a(\mathbf{x}) = \prod_a f_a(\mathbf{x}_a) \tag{3.6}$$

où $\mathbf{x}_a$ est le sous-ensemble de $\mathbf{x}$ connecté au facteur f_a.

De plus, nous choisissons pour $\mathcal{F}$ la famille des gaussiennes complètement factorisées :

$$\hat{p}(\mathbf{x}) = \prod_{x_k \in \mathbf{x}} \hat{p}(x_k) = \prod_k \hat{p}(x_k) \tag{3.7}$$

où chaque $\hat{p}(x_k)$ est une gaussienne unidimensionnelle. Nous avons ainsi deux factorisations différentes de $\hat{p}$ qu'il ne faut pas confondre : la factorisation due au modèle probabiliste $\hat{p}(\mathbf{x}) = \prod_a \hat{f}_a(\mathbf{x}_a)$ et la factorisation due aux choix de $\mathcal{F}$: $\hat{p}(\mathbf{x}) = \prod_k \hat{p}(x_k)$.

Avec ces hypothèses, nous pouvons définir des *messages* selon le schéma :

$$\hat{p}(\mathbf{x}) = \prod_a \hat{f}_a(\mathbf{x}_a) \tag{3.8}$$

$$= \prod_a \prod_{x_k \in \mathbf{x}_a} \hat{f}_{a,k}(x_k) \tag{3.9}$$

$$= \prod_{x_k \in \mathbf{x}} \prod_{a \in N_{x_k}} \hat{f}_{a,k}(x_k) \tag{3.10}$$

$$= \prod_{x_k \in \mathbf{x}} \prod_{a \in N_{x_k}} m_{f_a \to x_k}(x_k). \tag{3.11}$$

L'équation 3.8 provient de la factorisation de la jointe, 3.9 de l'hypothèse de factorisation complète, 3.10 est une réorganisation et 3.11 la définition des messages de facteur à variable. L'identification entre 3.11 et 3.7 prouve l'équivalence entre les *belief* locaux de LBP (produits des messages arrivant à x_k) et les marginales de $\hat{p}$.

On définit alors les messages de variable à facteur par

$$m_{x_k \to f_a}(x_k) := \prod_{b \neq a} m_{f_b \to x_k}(x_k) \tag{3.12}$$

en collectant les messages arrivant à x_k.

Grâce à l'hypothèse de factorisation, l'étape de minimisation de l'algorithme 2 devient locale à chaque facteur. Chaque itération de EP est alors équivalente au passage de tous les messages issus du facteur choisi (Min01c). Dans cette terminologie, "passer une message" signifie calculer le nouveau facteur $\hat{f}_a(\mathbf{x}_a)$ par la minimisation décrite. Ce nouveau facteur est un produit des $\hat{f}_{a,k}(x_k)$, c'est à dire un produit des messages sortant du facteur. Pour finir de "passer" ces messages, il faut mettre à jour les distributions marginales $\hat{p}(x_k)$ des variables liées au facteur.

D'autre part, comme EP ne nécessite pas d'ordre particulier pour le réajustement des termes et comme les facteurs approximés sont factorisés, la méthode n'impose pas l'ordre de passage des messages, même au sein d'un même facteur. Cet ordre, ou *agenda*, qui reste à définir, influence la vitesse de convergence et détermine le point stationnaire atteint. Les points stationnaire de EP sont les extrema d'une fonction d'énergie libre non convexe (HOW+05 ; Min01a), mais les approximations produites sont en général de bonne qualité, surtout lorsque les membres de $\mathcal{F}$ forment de bonnes approximations des vrais termes (MWJ99 ; KFL01 ; BG96 ; Gal63).

Appliqué au modèle TrueChess, EP devient donc l'algorithme 3, dont les étapes sont détaillées dans les paragraphes suivants.

Algorithme 3 : Principales étapes de l'inférence par passage de messages pour TrueChess.

Entrées : Résultats des matchs $\Delta = \{g_n\}$
Sorties : Marginales approchées $\forall (i^*, t^*)\ \hat{p}(s_{i^*}^{t^*} | \Delta)$

```
Construire le graphe de facteurs correspondant à Δ;
Initialiser tous les messages à la distribution uniforme;
Initialiser toutes les marginales à la distribution uniforme;
Déterminer un agenda pour le passage des messages;
tant que Critère de convergence non vérifié faire
    Passer le message courant de l'agenda;
        en mettant à jour ce message et sa marginale associée;
        (si nécessaire, utiliser une approximation gaussienne);
fin
```

Justification du choix de EP

EP est particulièrement bien adapté au modèle TrueChess. Étant donné le nombre de variables cachées et la taille des jeux de données traités, une méthode stochastique ou exhaustive serait trop coûteuse. En général, EP permet un excellent compromis entre précision et temps de calcul. De plus, la structure éparse du graphe de dépendance rend adéquate la délocalisation des calculs apportée par le passage de messages, qui exploite localement les sous structures non cycliques. Dans cette optique, l'utilisation du classique LBP n'est pas possible car tous les termes n'appartiennent pas à une même famille paramétrique close par passage de message. Grâce à ses phases de projection, EP est en revanche capable de gérer ce problème. Finalement EP est *a priori* mieux adaptée que des méthodes de passage de messages variationnelles de type *Mean Field*, pour deux raisons. D'abord nous verrons que nous devons approximer des *gaussiennes tronquée rectifiées* (*GTR*, sec. A.2.3) par des gaussiennes, ce qui est assez facile car les deux types de distributions sont unimodales. D'autre part, EP et méthodes variationnelles peuvent être unifiées en tant que minimisation d'α-divergences, avec $\alpha = 1$ pour EP et $\alpha = 0$ pour *Mean Field* (Min05). Détaillons cette différence et ses conséquences.

Définition 9 (Divergence de Kullback-Leibler)**.** *Si $\boldsymbol{X}$ est une variable aléatoire de $\mathbb{R}^d$ et*

$$\forall \mathbf{x} \in \mathbb{R}^d, q(\mathbf{x}) = 0 \Rightarrow p(\mathbf{x}) = 0,$$

la divergence de Kullback-Leibler de q par rapport p est définie par

$$D_{KL}(\, p \,||\, q \,) := E_p \left[\log \frac{p(\boldsymbol{X})}{q(\boldsymbol{X})} \right] = \int_{\mathbb{R}^d} p(\mathbf{x}) \, \log \frac{p(\mathbf{x})}{q(\mathbf{x})} \, d\mathbf{x}. \tag{3.13}$$

Remarques :
- Si p et q ne sont pas normalisées, le terme $\int (q(\mathbf{x}) - p(\mathbf{x}))\ d\mathbf{x}$ doit être ajouté.
- En théorie des codes, cette valeur mesure le nombre de bits supplémentaires pour transmettre des messages distribués selon p mais codés selon q. Intuitivement, c'est notre surprise moyenne à la réception d'un message produit sous p alors que nous nous attendions à le voir produit sous q.
- Cette divergence n'étant pas symétrique et ne vérifiant pas l'inégalité triangulaire, elle n'est pas une distance. Cependant, $D_{\mathrm{KL}}(\, p \,||\, q \,) \geq 0$ et $p = q$ presque partout équivaut à $D_{\mathrm{KL}}(\, p \,||\, q \,) = 0$.
- La KL-divergence est un cas particulier de la famille des α-divergences (ZR95), inspirées de la théorie de la géométrie de l'information (iAN00).

Définition 10 (α-divergences)**.** *L'α-divergences de q par rapport p est définie par :*

$$D_\alpha(\, p \,||\, q \,) := \frac{1}{\alpha \, (1 - \alpha)} \int_{\mathbb{R}^d} \alpha \, p(\mathbf{x}) + (1 - \alpha) \, q(\mathbf{x}) - p(\mathbf{x})^\alpha \, q(\mathbf{x})^{(1-\alpha)} \, d\mathbf{x}. \tag{3.14}$$

Remarques :
- Cette définition est valable même si p et q ne sont pas normalisées.

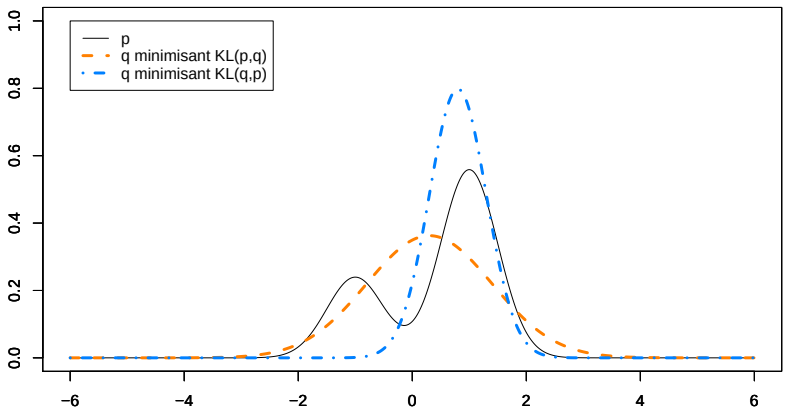


FIG. 3.8 – Distributions minimisant les divergences inclusive $D_{\mathrm{KL}}(\, p \,||\, q \,)$ et exclusive $D_{\mathrm{KL}}(\, q \,||\, p \,)$, avec q gaussienne et p bimodale. La première englobe toute la masse de p, surestimant sa variance. La seconde se concentre sur le plus gros mode et est donc trop confiante dans son approximation de p. Le choix d'une divergence doit dépendre de sa simplicité calculatoire, du but du problème et du modèle probabiliste envisagé.

- $\lim_{\alpha \to 1} D_\alpha(\, p \,||\, q \,) = D_{\mathrm{KL}}(\, p \,||\, q \,)$.
- $\lim_{\alpha \to 0} D_\alpha(\, p \,||\, q \,) = D_{\mathrm{KL}}(\, q \,||\, p \,)$.
- $D_{0.5}(\, p \,||\, q \,)$ et $D_2(\, p \,||\, q \,)$ sont de vraies distances (resp. de Hellinger et du χ^2).
- Les α-divergences se justifient par leur origine géométrique : elles ne dépendent pas de la mesure de base de l'espace et sont donc invariantes par reparamétrisation de p et q.

Divergence inclusive et divergence exclusive Les cas particuliers $\alpha = 0$ et $\alpha = 1$ sont fréquemment utilisés pour définir la distribution $\mathrm{ArgMin}_{q \in \mathcal{F}} D_\alpha(\, p \,||\, q \,)$. Souvent $\mathcal{F}$ est une famille exponentielle complètement factorisée sur les variables de $\mathbf{x}$.

Pour $\alpha = 0$, $D_{\mathrm{KL}}(\, q \,||\, p \,)$ est dite *exclusive* ou *forçant les zéros* car $p(\mathbf{x}) = 0 \Rightarrow q(\mathbf{x}) = 0$. La densité q a tendance à ne modéliser que le plus gros mode de p (Fig. 3.8), sous-estimant donc sa variance. Les algorithmes variationnels comme *Mean Field* utilisent cette divergence et sont ainsi souvent trop confiants. Utilisée comme minimisation locale par un algorithme de passage de messages, $D_{\mathrm{KL}}(\, q \,||\, p \,)$ a cependant l'avantage d'induire une équivalence entre minimisation locale et minimisation globale. De plus, elle est appropriée à certains types de modèles, pour lesquels l'*a posteriori* présente de nombreux modes équivalents. Par exemple, pour des modèles de mixture, le nommage de chaque composante n'est pas pertinent et l'*a posteriori* aura un mode pour chacun de ces étiquetages. Trouver un de ces modes symétriques est suffisant.

Pour $\alpha = 1$, $D_{\mathrm{KL}}(\, p \,||\, q \,)$ est dite *inclusive* ou *évitant les zéros* car $p(\mathbf{x}) > 0 \Rightarrow q(\mathbf{x}) > 0$.

La densité q tente d'englober toute la masse de p, surestimant sa variance. *Belief Propagation* et *Expectation Propagation* utilisent cette divergence et ne sont donc pas bien adaptés aux modèles de mixture. Ici la minimisation de divergences locales n'est pas strictement équivalente à une minimisation globale, bien que des expérimentations (Min05) montrent que cette approximation est assez bonne, surtout si $\mathcal{F}$ permet de bien représenter p. En revanche, lorsque $q(\mathbf{x})$ est complètement factorisée, ses marginales sont exactement celles de p (Théorème 4). Cette propriété est intéressante lorsque le but de l'inférence est de trouver ces marginales (comme dans l'application TrueChess), car elle garantit que les croyances sur les variables après la convergence de l'algorithme de passage de messages approximent les marginales de p. Ceci n'est pas vrai pour les méthodes variationnelles et c'est un argument fort en faveur de EP pour le modèle TrueChess.

Théorème 4 (Divergence inclusive et marginales)**.** *Si $q(\mathbf{x})$ est totalement factorisée et minimise $D_{KL}(\, p \,||\, q\,)$, alors ses marginales coïncident avec celles de p.*

Démonstration. On se restreint au cas où les variables $\mathbf{x}$ sont discrètes. Le cas continu est similaire en utilisant l'optimisation variationnelle. La distribution q étant factorisée, nous minimisons

$$\int_{\mathbf{x}} p(\mathbf{x}) \log \frac{p(\mathbf{x})}{\prod_k^K q_k(x_k)}\, d\mathbf{x}$$

sous les K contraintes $\int_{x_k} q_k(x_k) dx_k = 1$. Le lagrangien est alors

$$L(\lambda) \propto -\int_{\mathbf{x}} p(\mathbf{x}) \sum_k \log q_k(x_k)\, d\mathbf{x} + \sum_k \lambda_k \left(\int_{x_k} q_k(x_k) dx_k - 1 \right).$$

En supposant que chaque x_k prennent I_k valeurs possibles x_k^i, les dérivées partielles sont

$$\begin{aligned}
\frac{\partial L(\lambda)}{\partial q_k(x_k = x_k^i)} &= -\int_{\mathbf{x}} \frac{\partial \left[p(\mathbf{x}) \log q_k(x_k) \right]}{\partial q_k(x_k = x_k^i)}\, d\mathbf{x} + \lambda_k \frac{\partial}{\partial q_k(x_k = x_k^i)} \left(\int_{x_k} q_k(x_k) dx_k - 1 \right) \\
&= -\int_{\mathbf{x}^{\setminus k}} \int_{x_k} \frac{\partial \left[p(\mathbf{x}) \log q_k(x_k) \right]}{\partial q_k(x_k = x_k^i)}\, d\mathbf{x} + \lambda_k \int_{x_k} \frac{\partial q_k(x_k)}{\partial q_k(x_k = x_k^i)} dx_k \\
&= -\int_{\mathbf{x}^{\setminus k}} \frac{p(\mathbf{x}^{\setminus k}, x_k = x_k^i)}{q_k(x_k = x_k^i)} d\mathbf{x}^{\setminus k} + \lambda_k.
\end{aligned}$$

En combinant leur annulation pour tous les i, nous obtenons

$$\int_{\mathbf{x}^{\setminus k}} p(\mathbf{x})\, d\mathbf{x} = -\lambda_k q_k(x_k).$$

Comme q_k est normalisée, nous obtenons son égalité avec la $k^{\text{ième}}$ marginale de p. □

Ainsi nous choisissons EP comme algorithme d'inférence pour TrueChess. Détaillons les différentes phases de l'algorithme 3.

3.5.2 Initialisation

Tous les messages et marginales seront approximés par des gaussiennes unidimensionnelles initialisées à l'uniforme impropre $\mathcal{G}(.;0,0)$. Bien que EP soit un algorithme local, nous n'avons pas constaté de sensibilité au choix de l'initialisation, à condition qu'elle ne soit pas pathologique et qu'elle soit cohérente avec les équations des messages. Son effet est de plus atténué par le choix d'un "bon" agenda.

3.5.3 Invariants

Les invariants conservés lors des passages de messages déterminés par EP sont identiques à ceux de LBP. Nous rappelons les messages de variable à facteur, de facteur à variable et les marginales des variables.

$$m_{x\to f}(x) := \prod_{g\in N_x\setminus f} m_{g\to x}(x) \tag{3.15}$$

$$m_{f\to x}(x) := \int_{y\in N_f\setminus x} f(N_f) \prod_{y\in N_f\setminus x} m_{y\to f}(y)\; d(N_f\setminus x) \tag{3.16}$$

$$\hat{p}(x) = \prod_{f\in N_x} m_{f\to x}(x) \tag{3.17}$$

3.5.4 Passage des messages

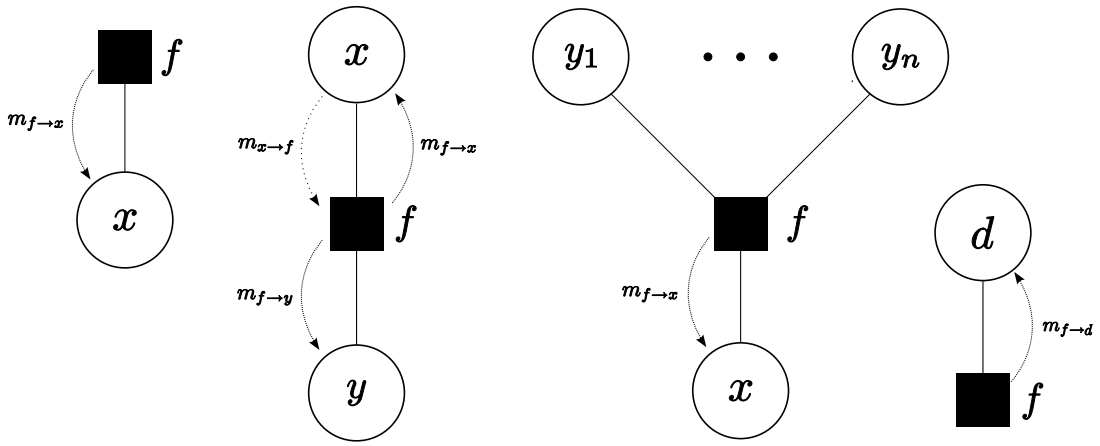


FIG. 3.9 – Les quatre types de facteurs du modèle et leurs messages. De gauche à droite : facteur d'*a priori*, de performance, de combinaison linéaire et de résultat. Les messages de variable à facteur ne sont pas explicites. Seul le facteur de résultat nécessite une approximation de type EP (projection).

Lors des itérations de EP, il faut mettre à jour séquentiellement messages et marginales grâce aux équations précédentes. Le calcul d'un nouveau message à partir des anciens mes-

sages s'appelle *mise à jour*, ou *passage* du message. Ce calcul dépend du facteur concerné. Nous dérivons les mises à jour pour tous les facteurs présents dans notre modèle (Fig. 3.9). Nous montrons qu'il suffit de maintenir en mémoire que les messages de *facteurs à variable* et les marginales. Les messages de *variable à facteur* sont implicitement calculés.

Facteur d'*a priori*

Lorsqu'un joueur apparaît pour la première fois dans le modèle, sa cotation est distribuée selon

$$s_i^t \sim \mathcal{N}\left(s_i^t; \mu_0, \sigma_0^2\right) = \mathcal{G}\left(s_i^t; \tau_0, \pi_0\right).$$

En nommant $x = s_i^t$ et f ce facteur, le nouveau message envoyé par le facteur est :

$$\begin{aligned} m_{f\to x}^{\text{new}}(x) &\leftarrow \int_{y\in N_f\setminus x} f(x) \prod_{y\in N_f\setminus x} m_{y\to f}(y)\ d(N_f\setminus x) \\ &= \int_{\emptyset} \mathcal{G}\left(x; \tau_0, \pi_0\right) \prod_{\emptyset} 1 \\ &= \mathcal{G}\left(x; \tau_0, \pi_0\right). \end{aligned} \tag{3.18}$$

Comme ce message ne dépend que des paramètres τ_0 et π_0, il sera constant au cours des itérations (une fois l'initialisation $\mathcal{G}\left(x; 0, 0\right)$ oubliée). La nouvelle marginale de x est alors

$$\hat{p}^{\text{new}}(x) = m_{f\to x}^{\text{new}}(x) \prod_{g\in N_x\setminus f} m_{g\to x}^{\text{old}}(x).$$

Comme avant la mise à jour de $m_{f\to x}^{\text{new}}$, nous avions

$$\hat{p}^{\text{old}}(x) = m_{f\to x}^{\text{old}}(x) \prod_{g\in N_x\setminus f} m_{g\to x}^{\text{old}}(x),$$

nous pouvons calculer la nouvelle marginale à partir du nouveau message, de l'ancien message et de l'ancienne marginale :

$$\begin{aligned} \hat{p}^{\text{new}}(x) &\leftarrow \frac{m_{f\to x}^{\text{new}}(x)\cdot \hat{p}^{\text{old}}(x)}{m_{f\to x}^{\text{old}}(x)} \\ &= \mathcal{G}\left(x; \tau_{f\to x}^{\text{new}} + \tau_x^{\text{old}} - \tau_{f\to x}^{\text{old}}, \pi_{f\to x}^{\text{new}} + \pi_x^{\text{old}} - \pi_{f\to x}^{\text{old}}\right) \end{aligned} \tag{3.19}$$

Ainsi, le calcul de la nouvelle marginale se réduit à additionner et soustraire des paramètres canoniques. Ce calcul de marginale est générique et nous l'utiliserons pour tous les types de facteurs.

Facteurs de performance

C'est un bruit gaussien entre $x := s_i^t$ et $y := p_i^t$.

$$y|x \sim f(x,y) = \mathcal{N}\left(y; x, \beta^2\right)$$

Déterminons le nouveau message de f vers y :

$$m_{f\to y}^{\text{new}}(y) \leftarrow \int_x \mathcal{N}\left(y; x, \beta^2\right) \cdot m_{x\to f}^{\text{old}}(x)\, dx \tag{3.20}$$

Cependant, le calcul et le maintient en mémoire des messages de *variables à facteurs* comme $m_{x\to f}^{\text{old}}$ n'est pas nécessaire. En effet les relations 3.15 et 3.16 donnent

$$\begin{aligned} \hat{p}^{\text{old}}(x) &= m_{f\to x}^{\text{old}}(x) \prod_{g\in N_x\setminus f} m_{g\to x}^{\text{old}}(x) \\ m_{x\to f}^{\text{old}}(x) &= \prod_{g\in N_x\setminus f} m_{g\to x}^{\text{old}}(x). \end{aligned} \tag{3.21}$$

Nous obtenons ainsi une expression fonction d'une marginale et d'un message de *facteur à variable* :

$$m_{x\to f}^{\text{old}}(x) = \frac{\hat{p}^{\text{old}}(x)}{m_{f\to x}^{\text{old}}(x)}. \tag{3.22}$$

De façon générale, les messages de *variable à facteurs* seront implicitement calculés de cette manière.

Dans le cas du facteur de performance le message cherché (Eq. 3.20) est

$$\begin{aligned} m_{f\to y}^{\text{new}}(y) &\leftarrow \int_x \mathcal{N}\left(y; x, \beta^2\right) \cdot \frac{\hat{p}^{\text{old}}(x)}{m_{f\to x}^{\text{old}}(x)}\, dx \\ &= \int_x \mathcal{N}\left(y; x, \beta^2\right) \cdot \mathcal{N}\left(x; \mu, \sigma^2\right)\, dx \\ &= \mathcal{N}\left(y; \mu, \sigma^2 + \beta^2\right) \\ \text{avec} \quad \sigma^2 &:= \frac{1}{\pi_x^{\text{old}} - \pi_{f\to x}^{\text{old}}} \\ \mu &:= \frac{\tau_x^{\text{old}} - \tau_{f\to x}^{\text{old}}}{\pi_x^{\text{old}} - \pi_{f\to x}^{\text{old}}}. \end{aligned}$$

En notation canonique, la mise à jour est alors :

$$\begin{aligned} a &:= \left(1 + \beta^2\ (\pi_x^{\text{old}} - \pi_{f\to x}^{\text{old}})\right)^{-1} \\ m_{f\to y}^{\text{new}}(y) &\leftarrow \mathcal{G}\left(y; a\ (\tau_x^{\text{old}} - \tau_{f\to x}^{\text{old}}), a\ (\pi_x^{\text{old}} - \pi_{f\to x}^{\text{old}})\right) \end{aligned} \tag{3.23}$$

La nouvelle marginale $\hat{p}^{\text{new}}(y)$ est calculée grâce à l'équation 3.19.

L'équation de mise à jour du message remontant vers x se déduit de 3.23 car le facteur de performance est symétrique $\mathcal{N}(x; y, \beta^2) = \mathcal{N}(y; x, \beta^2)$.

Facteurs d'évolution dynamique

Ce facteur lie une cotation entre deux instants :

$$f(s_i^{t_2}|s_i^{t_1}) = \mathcal{N}(s_i^{t_2}; s_i^{t_1}, \gamma^2\,(t_2 - t_1)). \tag{3.24}$$

Les équations de mise à jour sont similaires à celles des facteurs de performance.

Facteurs de différence

Les variables $\{d_n\}$ sont des différences de performances. Le facteur associé est déterministe :

$$d|p_i, p_j \sim f(d, p_i, p_j) = \delta_{d=p_j-p_i}\,(d)\,. \tag{3.25}$$

De façon générale, considérons un facteur liant linéairement x et n variables y_i :

$$f(x, y_1, \ldots, y_n) = \delta_{x=\boldsymbol{a}^\top \boldsymbol{y}}\,(x)\,. \tag{3.26}$$

En utilisant les propriétés arithmétiques des gaussiennes (section A.2) et en notant :

$$\begin{aligned}\pi_j &:= \pi_{y_j}^{\text{old}} - \pi_{f\to y_j}^{\text{old}}\\ \tau_j &:= \tau_{y_j}^{\text{old}} - \tau_{f\to y_j}^{\text{old}}\end{aligned}$$

le nouveau message de f vers x (Eq. 3.16) prend pour paramètres canoniques :

$$\begin{aligned}\pi_{f\to x}^{\text{new}} &\leftarrow \left(\sum_{j=1}^{n} \frac{a_j^2}{\pi_j}\right)^{-1}\\ \tau_{f\to x}^{\text{new}} &\leftarrow \pi_{f\to x}^{\text{new}} \cdot \left(\sum_{j=1}^{n} a_j \cdot \frac{\tau_j}{\pi_j}\right).\end{aligned} \tag{3.27}$$

Dans notre cas, $\boldsymbol{a} = [-1, 1]$. La marginale découle de l'équation 3.19.

Pour déterminer l'expression des messages $m_{f\to y_j}^{\text{new}}(y_j)$, nous considérons la transformation linéaire duale en échangeant x et y_j et en ajustant $\boldsymbol{a}$. Dans notre cas, ceci est équivalent à supposer des facteurs $p_i = p_j - d$ ou $p_j = p_i + d$.

Facteurs de résultat

Selon la valeur observée de g, le facteur f peut prendre plusieurs expressions :

$$f(d) = \begin{cases} f_>(d) & = \delta_{d>\epsilon}\,(d)\\ f_{|.|}(d) & = \delta_{|d|<\epsilon}\,(d)\\ f_<(d) & = \delta_{d<-\epsilon}\,(d)\,.\end{cases} \tag{3.28}$$

Ces trois cas sont similaires car il conduisent à des messages exacts $m_{f\to d}^{\text{new}}(d) = f(d)$ non gaussiens. Si nous conservions de tels messages, leur propagation dans le graphe rendrait tous les autres messages non gaussiens, invalidant les calculs précédents.

Par exemple, à la première itération, la nouvelle marginale exacte $p^{\text{new}}(d)$ est le produit de $f(d)$ et du message gaussien provenant du facteur de différence $\hat{m}^{\text{new}}_{g\to d}(d)$. C'est donc une gaussienne tronquée rectifiée (GTR, voir section A.2.3).

Intuitivement l'idée est d'approximer cette distribution monomodale par une gaussienne afin de toujours travailler avec la famille des distributions gaussiennes. Cette méthode est l'instanciation des projections EP pour TrueChess.

Alors la démarche est d'approximer en premier lieu la GTR $p^{\text{new}}(d)$ par une gaussienne $\hat{p}^{\text{new}}(d)$, puis d'en déduire le message gaussien $\hat{m}^{\text{new}}_{f\to d}(d)$ qui aurait produit $\hat{p}^{\text{new}}(d)$ par multiplication avec $\hat{m}^{\text{new}}_{g\to d}(d)$.

Remarque 1 : Bien qu'il soit *a priori* impossible d'approximer convenablement une fonction de Heaviside par une gaussienne, cette méthode nous le permet en tirant partie du "contexte" dans lequel est utilisée cette Heaviside (Fig. 3.10).

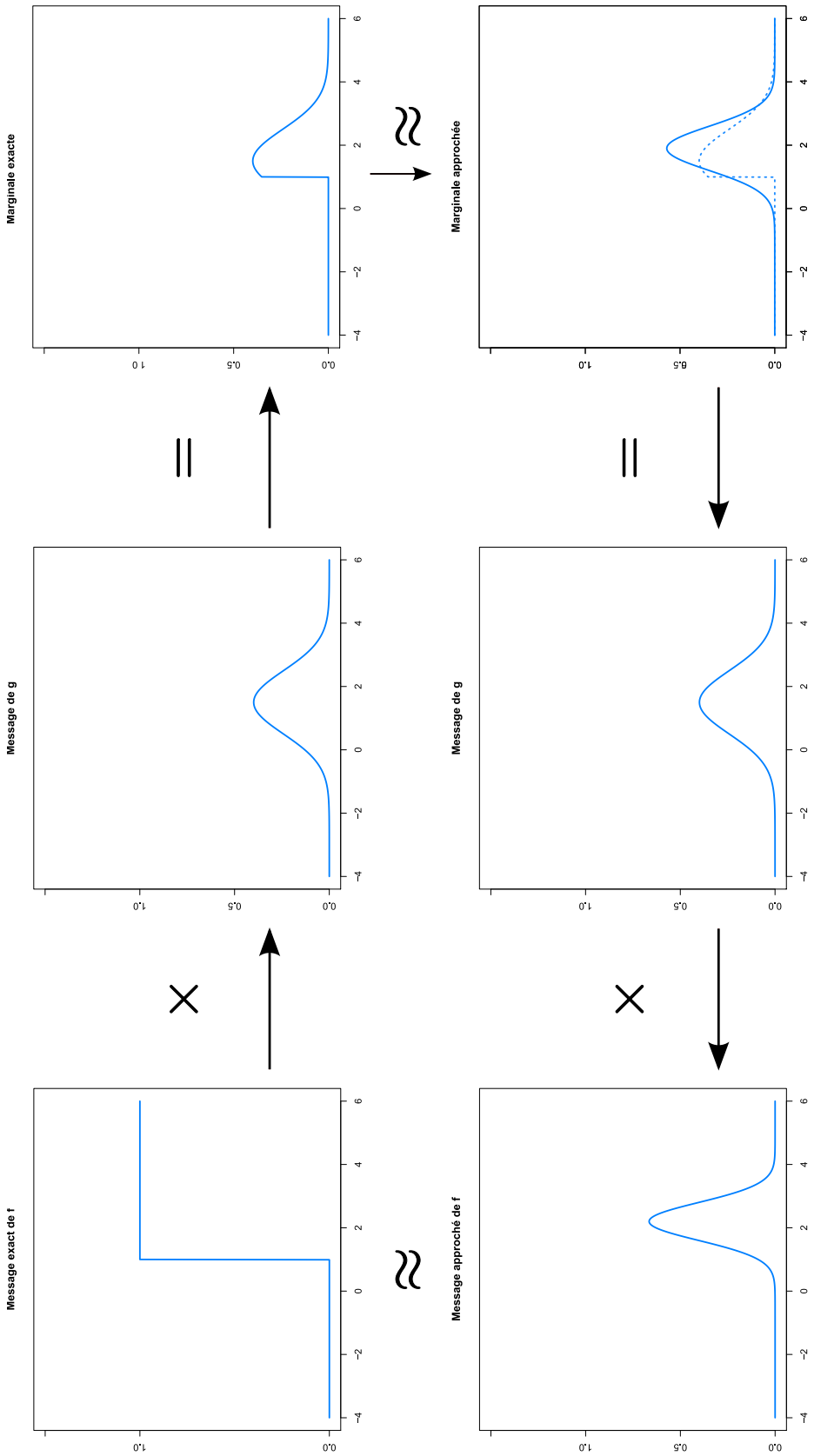


FIG. 3.10 – Illustration de l'approximation EP d'un facteur de résultat. Les figures se lisent dans le sens horaire. Le message exact du facteur de résultat (haut-gauche), multiplié par le message provenant de g (haut-centre) donne comme vraie marginale de g une GTR (haut-droite). Cette GTR est approximée par une gaussienne (bas-droite) par la méthode d'identification des moments. Alors le message approximé (bas-gauche) est défini comme celui qui aurait donné la marginale approximée (bas-doite) par multiplication avec le message de g (bas-centre). Ainsi nous avons réussi à approximer une fonction de Heaviside (haut-gauche) par une gaussienne (bas-gauche), ce qui paraît *a priori* impossible. Nous y sommes parvenus grâce au "contexte" définit par le message de g.

Remarque 2 : La méthode pour approximer la GTR $p^{\text{new}}(d)$ par une gaussienne $\hat{p}^{\text{new}}(d)$ découle de EP : $\hat{p}^{\text{new}}(d)$ est la gaussienne minimisant $D_{\text{KL}}(\, p^{\text{new}} \,||\, \hat{p}^{\text{new}} \,)$. Le théorème 5 montre que ceci est équivalent à identifier leurs deux premiers moments. En pratique, $\hat{p}^{\text{new}}(d)$ sera la gaussienne d'espérance $E_{p^{\text{new}}}[d]$ et de variance $\text{VAR}_{p^{\text{new}}}[d]$.

Théorème 5. ***Théorème de l'identification par les moments.***
Soit une distribution quelconque p sur $\mathbb{R}^d$. Soit une famille exponentielle $\mathcal{F}$ de $\mathbb{R}^d$ définie par son vecteur de statistiques $\boldsymbol{\psi}(\boldsymbol{X})$. Alors la distribution

$$p_{\boldsymbol{\theta}^*} = ArgMin_{q \in \mathcal{F}}\ D_{KL}(\, p \,||\, q \,)$$

est implicitement donnée par

$$E_{p_{\boldsymbol{\theta}^*}}\left[\boldsymbol{\psi}(\boldsymbol{X})\right] = E_p\left[\boldsymbol{\psi}(\boldsymbol{X})\right]. \tag{3.29}$$

Commentaires : Nous utilisons cette égalité d'espérances pour trouver $\boldsymbol{\theta}^*$, paramètre de la distribution de $\mathcal{F}$ la plus proche de p. Le système d'équations donné par l'égalité 3.29 a toujours une solution $\boldsymbol{\theta}^*$ et, dans le cas où $\mathcal{F}$ est la famille des gaussiennes, cette solution est simple. Ainsi, pour les gaussiennes unidimensionnelles paramétrées par (μ, σ^2), nous avons $\boldsymbol{\psi}(X) = [X, -X^2/2]^\top$. Alors 3.29 équivaut à

$$\begin{cases} \mu^* & = E_p[X] \\ \sigma^{2*} & = \text{VAR}_p[X]. \end{cases} \tag{3.30}$$

La distribution $p_{\boldsymbol{\theta}^*}$ est la *projection* de p sur la variété $\mathcal{F}$ et l'opération $p \mapsto p_{\boldsymbol{\theta}^*}$ est appelée *identification par les moments* (*moment matching*).

Démonstration. Comme la KL-divergence est log-convexe en son deuxième argument, il existe un minimum unique. Pour le trouver, étudions :

$$\begin{aligned} f(\boldsymbol{\theta}) &:= D_{\text{KL}}(\, p \,||\, p_{\boldsymbol{\theta}} \,) \\ &= \int p(\mathbf{x}) \left(\log p(\mathbf{x}) - \log p_{\boldsymbol{\theta}}(\mathbf{x})\right)\, d\mathbf{x} \\ &= E_p\left[\log p\right] - E_p\left[\boldsymbol{\theta}^\top \boldsymbol{\psi}(\mathbf{X})\right] + E_p\left[\log Z(\boldsymbol{\theta})\right] \\ &= E_p\left[\log p\right] - \boldsymbol{\theta}^\top E_p\left[\boldsymbol{\psi}(\mathbf{X})\right] + \log Z(\boldsymbol{\theta}). \end{aligned}$$

Annulons son gradient $\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta})$. Or

$$\begin{aligned} \nabla_{\boldsymbol{\theta}} \log Z(\boldsymbol{\theta}) = \frac{\nabla_{\boldsymbol{\theta}} Z(\boldsymbol{\theta})}{Z(\boldsymbol{\theta})} &= \frac{\nabla_{\boldsymbol{\theta}} \int \exp\left(\boldsymbol{\theta}^\top \boldsymbol{\psi}(\mathbf{x})\right)\, d\mathbf{x}}{Z(\boldsymbol{\theta})} \\ &= \frac{\int \nabla_{\boldsymbol{\theta}} \exp\left(\boldsymbol{\theta}^\top \boldsymbol{\psi}(\mathbf{x})\right)\, d\mathbf{x}}{Z(\boldsymbol{\theta})} = \frac{\int \boldsymbol{\psi}(\mathbf{x}) \exp\left(\boldsymbol{\theta}^\top \boldsymbol{\psi}(\mathbf{x})\right)\, d\mathbf{x}}{Z(\boldsymbol{\theta})} \\ &= E_{p_{\boldsymbol{\theta}}}\left[\boldsymbol{\psi}(\mathbf{X})\right]. \end{aligned}$$

Donc les solution $\boldsymbol{\theta}^*$ vérifient $E_p\left[\boldsymbol{\psi}(\mathbf{X})\right] = E_{p_{\boldsymbol{\theta}}}\left[\boldsymbol{\psi}(\mathbf{X})\right]$. Nous montrons que f atteint un minimum en $\boldsymbol{\theta}^*$ car la matrice de ses dérivées secondes est définie positive (car c'est la matrice de covariance de $\boldsymbol{\psi}(\mathbf{X})$). □

Dans le cas particulier du facteur de résultat, les étapes de mises à jour de la marginale approchée de d et du message approché de f vers d sont donc :

$$\begin{aligned}\hat{p}^{\text{new}}(d) &\leftarrow \text{Identification Moments}\left[m^{\text{new}}_{f\to d}(d)\cdot\frac{\hat{p}^{\text{old}}(d)}{\hat{m}^{\text{old}}_{f\to d}(d)}\right]\\ \hat{m}^{\text{new}}_{f\to d}(d) &\leftarrow \frac{\hat{m}^{\text{old}}_{f\to d}(d)}{\hat{p}^{\text{old}}(d)}\cdot\hat{p}^{\text{new}}(d).\end{aligned}$$

En utilisant l'expression des moments d'une GTR (Eq. A.2) et la définition A.3 de W_f, nous obtenons :

$$\begin{aligned}\tau_{\text{t}} &:= \hat{\tau}^{\text{old}}_d - \hat{\tau}^{\text{old}}_{f\to d}\\ \pi_{\text{t}} &:= \hat{\pi}^{\text{old}}_d - \hat{\pi}^{\text{old}}_{f\to d}\\ \hat{\tau}^{\text{new}}_d &\leftarrow \frac{\pi_{\text{t}}}{1-W_f(\tau_{\text{t}}/\sqrt{\pi_{\text{t}}},\epsilon\sqrt{\pi_{\text{t}}})}\\ \hat{\pi}^{\text{new}}_d &\leftarrow \frac{\tau_{\text{t}}+\sqrt{\pi_{\text{t}}}\cdot V_f(\tau_{\text{t}}/\sqrt{\pi_{\text{t}}},\epsilon\sqrt{\pi_{\text{t}}})}{1-W_f(\tau_{\text{t}}/\sqrt{\pi_{\text{t}}},\epsilon\sqrt{\pi_{\text{t}}})}\\ \hat{\tau}^{\text{new}}_{f\to d} &\leftarrow \hat{\tau}^{\text{old}}_{f\to d}+\hat{\tau}^{\text{new}}_d-\hat{\tau}^{\text{old}}_d\\ \hat{\pi}^{\text{new}}_{f\to d} &\leftarrow \hat{\pi}^{\text{old}}_{f\to d}+\hat{\pi}^{\text{new}}_d-\hat{\pi}^{\text{old}}_d.\end{aligned} \tag{3.31}$$

Ainsi les formules 3.18, 3.19, 3.23, 3.27 et 3.31 nous donnent toutes les mises à jour des messages et marginales de TrueChess.

3.5.5 Agenda

Le choix de l'ordre dans lequel les messages sont passés est libre et influence la vitesse de convergence de EP. Dans la littérature, deux types d'agendas sont proposés.

Agendas synchrones. Tous les messages du graphe sont passés simultanément et le processus est répété. C'est l'approche historique de Pearl (Pea88), facile à mettre en œuvre et adaptée au calculateurs parallèles.

Agendas asynchrones. Les messages sont passés successivement, en tenant compte des effets des messages précédents. Nous choisissons cette approche, car ces agendas apportent une convergence meilleure, plus fréquente et plus rapide. Cependant, les agendas sont souvent définis à la main pour chaque modèle, par exemple à partir des symétries du graphe (champs de Markov).

Nous proposons deux nouvelles méthodes de création d'agendas, un algorithme général et totalement automatique et un algorithme semi-automatique optimisant les flux d'information, plus performant pour les grands graphes.

Propriétés d'un agenda

Un graphe de facteurs défini deux messages pour chacune de ses arêtes. Un *agenda* est une liste ordonnée de ces messages, chaque message pouvant apparaître un nombre arbitraire de fois. Plus précisément un agenda est défini par une *grammaire* récursive : un agenda est composé de listes de messages, de listes de sous-agendas et de parties itérées jusqu'à convergence des distributions concernées (critère $< \delta$) :

$$\begin{aligned} \text{AGENDA} &\rightarrow \text{ELEMENTAIRE} \mid \text{ITERE} \mid \text{SEQUENCE} \\ \text{ELEMENTAIRE} &\rightarrow (\text{liste de messages}) \\ \text{ITERE} &\rightarrow (\text{AGENDA}, \delta) \\ \text{SEQUENCE} &\rightarrow (\text{liste d'AGENDA}). \end{aligned}$$

Un agenda est dit *complet* quand tous les messages du graphe y sont présents, *exact* quand son exécution converge vers les meilleures marginales possibles, *minimal* si c'est le plus petit agenda complet exact. Il est dit *optimal* si de plus sa vitesse de convergence est optimale. Un *ordonnanceur* est un algorithme générant un agenda pour un graphe. Un ordonnanceur est *statique* si l'agenda est déterminé avant tout passage de message et *dynamique* s'il décide de l'ordre en fonction de l'évolution de l'inférence. Comme cette définition conduit à qualifier tout agenda itératif de dynamique, nous réservons le terme *semi dynamique* pour ces derniers. Un agenda conduisant à des passages de message non convergent est dit *infini* (il n'est pas *convergent*). Finalement, un agenda est dit *calculable* s'il ne conduit pas à un échec des passages de messages (ce cas peut se produire à cause de messages impropres et de divisions par zéro).

Tous les agendas ne conduisant pas forcément aux mêmes marginales, l'optimalité d'un agenda n'est pas démontrable si nous ne connaissons pas le vrai *a posteriori* p. Alors une heuristique est de chercher un agenda complet, convergent et minimal qui soit *efficace*, c'est-à-dire qui optimise les flux d'information. Intuitivement, lorsqu'une variable x vient de recevoir un message, il est intéressant de propager cette information reçue en passant ensuite les messages sortant de x. Cette idée intuitive est justifiée par l'algorithme somme-produit de BP, qui définit l'agenda correct pour calculer les marginales d'un arbre.

Dans le cas d'un ordonnanceur dynamique, le choix de l'ajout d'un message dans l'agenda en construction est fait en ligne. Après chaque passage de message, l'ordonnanceur tente de passer tous les messages possibles et choisi de retenir le message qui a produit le plus de changement, qui a transmis le plus d'information. Ce changement est mesuré par la KL entre l'ancienne et la nouvelle marginale cible. En fait, en maintenant une liste des effets de chaque message, il n'est pas nécessaire de tester tous les messages, mais seulement une liste de candidats potentiels. Cette méthode qui implique un surcoût de calcul de l'ordre de l'arité moyenne des facteurs, donne de meilleurs résultats qu'un ordonnancement synchrone ou naïf (EMK06), mais ne permet pas la constructions d'agendas récursifs ou modulaires. De plus, ce surcoût en espace et en temps, bien que polynomial, devient prohibitif pour des grands graphes comme celui de la présente étude.

Pour éviter ces calculs, il est possible d'estimer *statiquement* les changements potentiels,

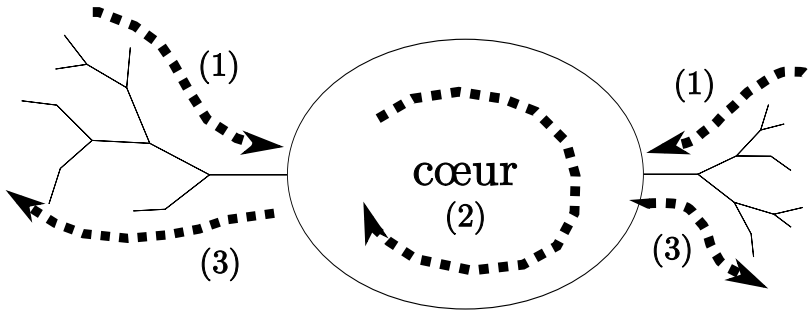


FIG. 3.11 – Trois phases d'un agenda produit par l'ordonnanceur automatique : (1) prétraitement des branches, (2) traitement du cœur, (3) post-traitement des branches. Les facteurs et variables ne sont pas représentés.

en définissant pour chaque message son ancienneté (`timestamp`) et le nombre de messages qu'il attend avant d'avoir reçu toutes l'information de ses sources (`pending`).

Ordonnanceur automatique semi dynamique

Notre ordonnanceur prend un graphe de facteurs en entrée, le divise en deux sous graphes (le *cœur* et les *branches*) et retourne un agenda composé de trois parties : le prétraitement des branches, le traitement du cœur et le post-traitement des branches. Le cœur est défini comme le sous graphe nécessitant une inférence itérée. C'est donc le sous graphe contenant les boucles, les facteurs approximés (nécessitant une projection EP) et tous les facteurs présents entre les boucles et les facteurs approximés. Par exemple un arbre à facteurs exacts ne contient pas de cœur et l'agenda produit par l'ordonnanceur sera celui de BP. L'agenda final comportera ainsi trois phases : (1) passage séquentiel des messages des feuilles des branches vers le cœur, (2) itérations dans le cœur jusqu'à convergence et (3) retro-propagation des messages dans les branches en sens inverse de (1) (Fig. 3.11.)

L'ordonnanceur est décrit par l'algorithme 4. Lors de l'initialisation (alg. 5), trois ensembles sont construits : l'ensemble des nœuds, des voisins de chaque nœud et des messages. Un nœud est une variable ou un facteur complémenté de l'attribut booléen `propre` qui vaut `vrai` si la marginale associée à la variable est propre et qui n'a pas de sémantique si le nœud est un facteur. Dans le troisième ensemble, chaque message est décrit par son nœud `source` x, son nœud `cible` y, le nombre de messages qu'il attend avant d'être transmissible (`pending`) et `timestamp`, date de la dernière l'élimination d'un message arrivant à x. Pour éviter de générer un agenda *incalculable*, nous ajoutons les booléens `exact` et `cibleProPreReq`. En effet certains facteurs projectifs requièrent que les variables cibles de leurs messages soient propres.

La phase d'élimination détermine simultanément la partition cœur/branches et les pre et post agendas des branches (alg. 6).

Pour le cœur, l'heuristique (alg. 7) est de passer les messages qui ont reçu le plus

Algorithme 4 : Grandes phases de l'ordonnanceur automatique.

Entrées : Graphe de facteurs G
Sorties : Agenda global

Phase d'initialisation des structures de données ;
  Création de l'ensemble des nœuds;
  Création de l'ensemble des voisins;
  Création de l'ensemble des messages;
Phase d'élimination ;
  Construction du cœur ;
  Construction du pré-agenda des branches;
  Construction du post-agenda des branches;
Construction de l'agenda du cœur ;
Réorganisation des sous-agendas;

Algorithme 5 : Phases d'initialisation de l'ordonnanceur automatique.

Entrées : Graphe de facteurs
Sorties : Trois ensembles

Construire l'ensemble de nœuds. Pour chaque nœud :;
  `propre` $\leftarrow$ propreté initiale pour une variable;
  `propre` $\leftarrow$ `vrai` pour les variables observées;
  `propre` $\leftarrow$ `vrai` pour les facteurs;
Construire l'ensemble des voisins N_x pour chaque nœud x;
Construire l'ensemble des messages $m_{x \to y}$ enregistrant :;
  `source` $\leftarrow x$;
  `cible` $\leftarrow y$;
  `pending` $\leftarrow \mathrm{Card}\,(\mathrm{N_x}) - 1$;
  `timestamp` $\leftarrow 0$;
  `exact` $\leftarrow$ `vrai` si x est une variable, l'exactitude du facteur sinon;
  `cibleProPreReq` $\leftarrow$ `faux` pour une variable, la valeur adéquate pour un facteur;
$t \leftarrow 0$;

Algorithme 6 : Phases d'élimination de l'ordonnanceur automatique.

Entrées : Graphe de facteurs, les 3 ensembles
Sorties : Pre et post agendas des branches, 3 ensembles modifiés

tant que *il existe une feuille dans la liste des nœuds* ($N_x = y$) **faire**
  si $m_{x\to y}$.`exact` = `vrai` **alors**
    $t \leftarrow t + 1$;
    Retirer $m_{x\to y}$ et $m_{y\to x}$ de l'ensemble des messages;
    Retirer x de l'ensemble des nœuds;
    Retirer x de N_y et y de N_x;
    y.`propre` ← `vrai`;
    pour tous les *messages* $m_{y\to z}$ *sortant de* y **faire**
      $m_{y\to z}$.`pending` ← $m_{y\to z}$.`pending` -1 ;
      $m_{y\to z}$.`timestamp` ← t;
    fin
    Ajouter $m_{x\to y}$ à la fin du pre-agenda;
    Ajouter $m_{y\to x}$ à la fin du post-agenda;
  fin
fin
Inverser l'ordre du post-agenda ;

d'information récente. En cas d'égalité, les messages exacts sont prioritaires, puis nous propageons localement l'information (`timestamp`). Ceci assure que nous ne passerons pas de messages approximés à des variables non initialisées, ce qui peut poser problème. Nous évitons aussi de calculer un message ayant des entrées impropres.

Les problèmes d'agendas non calculables proviennent de l'initialisation impropre des distributions ($\mathcal{G}(.; 0, 0)$). En effet, *certains* facteurs définissent un calcul de message qui n'est pas défini si ses entrées sont impropres. Si toutes les initialisations sont propres, notre ordonnanceur fonctionnera sur tout type de graphe. Sinon, il fonctionnera avec la majorité des graphes rencontrés (arbres, champs markoviens, TrueChess...) grâce à l'introduction des champs `propre` et `ciblePropreReq`. Mais même avec ces précautions, certains graphes restent pathologiques. Ces graphes qui conduisent à un agenda non calculable sont des graphes pour lesquels le passage d'un message pathologique ne peut être évité. Un message est non calculable sous l'une des deux conditions :

- il est issu d'un facteur (exact ou non) recevant au moins un message impropre et ne le supportant pas,
- il est issu d'un facteur approximé et pointe vers une variable dont la marginale courante est impropre.

S'il existe un agenda évitant ces conditions, l'ordonnanceur automatique le trouvera. La figure 3.12 présente un cas où il n'en existe pas.

Techniquement, trouver le prochain message est l'opération la plus coûteuse et une implémentation naïve avec une liste triée a un coût $O(n^2)$, si n est le nombre de nœuds.

Algorithme 7 : Construction de l'agenda du cœur.

Entrées : Graphe de facteurs, les 3 ensembles
Sorties : Agenda du cœur

Initialiser l'agenda du cœur à une liste vide ;
tant que *l'ensemble des messages n'est pas vide* **faire**
  $t \leftarrow t + 1$;
  Trouver le meilleur message $m_{x \to y}$, qui est ;
    celui de `pending` minimum;
    si égalité, préférer un message exact;
    si égalité, préférer un message de plus grand `timestamp` ;
    si $m_{x \to y}$.`cibblePropreReq` = `vrai` et y.`propre` = `faux`, **alors** choisir un autre message ;
    si $\exists\, z$ tel que $z \in N_x$ et $z \neq y$ et z.`propre`=`faux`, **alors** choisir un autre message ;
  Retirer $m_{x \to y}$ de l'ensemble des messages;
  Ajouter $m_{x \to y}$ à la fin de l'agenda du cœur ;
  Retirer y de N_x;
  y.`propre` $\leftarrow$ `vrai`;
  pour tous les *messages* $m \in \{m_{y \to z} \mid z \neq x\}$ **faire**
    m.`pending` $\leftarrow$ m.`pending` -1 ;
    m.`timestamp` $\leftarrow t$;
  fin
fin
Effacer les messages de variable à facteur de l'agenda du cœur ;

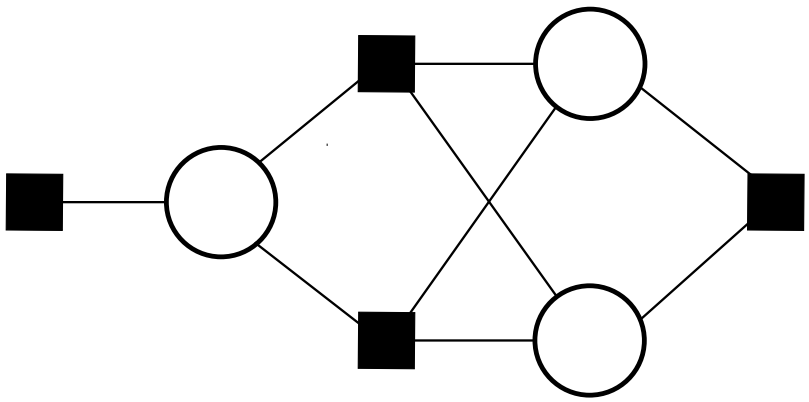


FIG. 3.12 – Ce graphe très connecté est pathologique car il n'admet pas d'agenda calculable si les marginales sont initialisées par des distributions impropres. Tous les facteurs ont une entrée impropre.

En utilisant le fait que tous les messages ne doivent pas être triés à chaque fois et avec une bonne structure de données, le coût est en $O(n \log n)$ (fig. 3.13). Les messages à reclasser dans la liste de priorités après le passage de $m_{x \to y}$ sont ceux arrivant à x, ceux partant de y et ceux partant de $z \in N_y$.

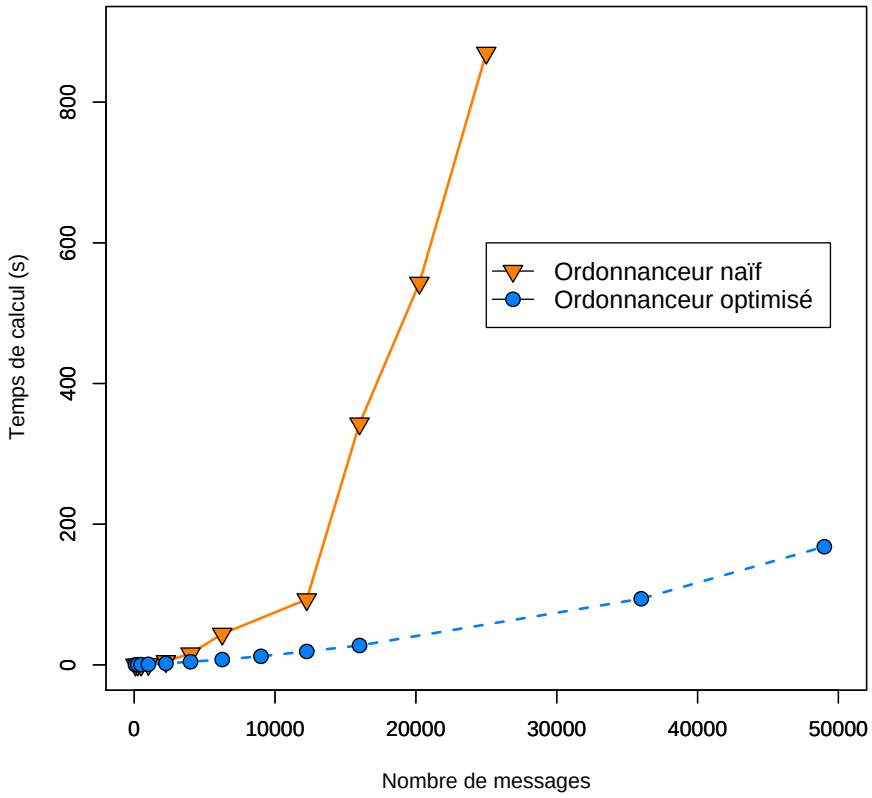


FIG. 3.13 – Temps de calculs expérimentaux de l'ordonnanceur pour des champ de Markov de taille croissant. Une approximation exponentielle avec et sans optimisation donne respectivement $O(n^{1.5})$ et $O(n^{2.5})$ sur ces données, passant de 800 à 50 secondes pour 25 000 messages. Même pour la version optimisée, un petit terme en $O(n^2)$ provient de la construction de la liste des voisins.

Ordonnanceur semi-automatique semi dynamique

L'ordonnanceur automatique est général et rapide, mais il n'est pas assez performant avec de grands jeux de données comme ceux de cette thèse. D'abord le temps de construction de l'agenda est trop important, mais surtout l'agenda produit ne converge pas assez vite, il n'est pas optimal. L'algorithme automatique optimise en effet *localement* les flux

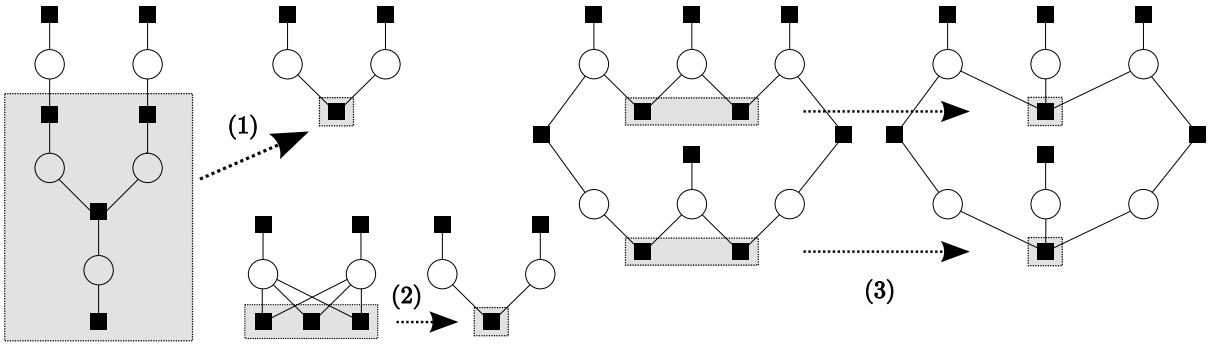


FIG. 3.14 – Ordonnancement semi-automatique. Le programmeur définit une structure hiérarchique du graphe, qui se retrouve dans la structure hiérarchique de l'agenda global. Chaque sous-agenda pour chaque sous structure à chaque niveau d'abstraction est calculé par l'ordonnanceur automatique. Pour le modèle TrueChess, les sous structures sont (1) les matchs, (2) tous les matchs entre deux mêmes joueurs dans le même tournoi et (3) les tournois.

d'information, mais n'a pas une vision *globale* du graphe et ne peut donc pas optimiser globalement les flux d'information.

Nous proposons une méthode d'ordonnancement semi-automatique hiérarchique, guidée par le programmeur, qui lui a une vision globale du graphe. Le programmeur définit des sous structures dans le graphe grâce à sa connaissance du modèle et de la structure de ses boucles. L'idée est de faire circuler l'information dans les petites boucles, de converger localement, avant de transmettre l'information aux structures voisines constituant de plus grandes boucles. Pour chacune des structures l'agenda est produit par l'ordonnanceur automatique décrit ci-dessus. Le processus est *bottom-up* et le formalisme des graphes de facteurs permet de remplacer une sous structure déjà traitée par un "gros" facteur. Comme cette organisation hiérarchique est aussi présente dans la grammaire des agendas, l'ordonnanceur automatique peut être appelé à un plus haut niveau et insérer les sous-agendas dans l'agenda global.

Pour le modèle TrueChess, les sous structures sont (1) le match, (2) tous les matchs entre deux mêmes joueurs dans le même tournoi et (3) les tournois (fig. 3.14). Comme ce graphe n'est quasiment qu'un cœur, les sous-agendas seront itérés hiérarchiquement. Le temps de calcul de l'agenda est très réduit (moins de 5 secondes pour 150 000 messages) car le graphe n'est jamais considéré dans sa totalité par l'ordonnanceur automatique. L'agenda produit A_global est décrit schématiquement ci-dessous. Il converge rapidement vers de bons résultats.

$$
\begin{aligned}
\text{A\_global} &:= \text{ITERE}(\text{A\_aller}, \text{A\_retour}) \\
\text{A\_aller} &:= \text{SEQUENCE}(\text{A\_tournoi}_1, \cdots, \text{A\_tournoi}_n) \\
\text{A\_retour} &:= \text{SEQUENCE}(\text{A\_tournoi}_n, \cdots, \text{A\_tournoi}_1) \\
\text{A\_tournoi}_n &:= \text{ITERE}(\text{A\_}J_1J_2, \cdots, \text{A\_}J_iJ_j) \\
\text{A\_}J_iJ_j &:= \text{ITERE}(\text{A\_match}_{i,j}^1, \cdots, \text{A\_match}_{i,j}^k) \\
\text{A\_match}_{i,j}^k &:= \text{SEQUENCE}(m_{a\ priori \to \text{résultat}}, m_{\text{résultat} \to a\ priori}))
\end{aligned}
$$

3.5.6 Critère d'arrêt

EP étant un algorithme d'optimisation locale distribuée, sa convergence n'est pas toujours garantie. Lorsqu'il y a convergence, la qualité de l'approximation n'est certaine que dans des cas particuliers (Wal06), mais en général elle est de bonne qualité. De plus, la non convergence de EP indique que la famille $\mathcal{F}$ est une mauvaise approximation de p. Les résultats de (Wal06) impliquent que EP converge pour notre graphe, car la distribution approchée est un gaussienne totalement factorisée.

Nous définissons alors un critère numérique d'arrêt pour les parties itératives de l'agenda : une itération est arrêtée quand les distributions concernées ne varient plus beaucoup. Plus précisément, le critère d'arrêt est vérifié quand toutes les variations entre nouvelles et anciennes distributions sont inférieures à un seuil δ. La variation d'une distribution est mesurée par la variation de ses paramètres canonique :

$$\max \left|\tau^{\text{new}} - \tau^{\text{old}}\right| + \left|\pi^{\text{new}} - \pi^{\text{old}}\right| \leq \delta. \tag{3.32}$$

Nous préférons cette formule à la KL entre les gaussiennes, d'abord car c'est une bonne heuristique de similarité, mais surtout car elle est beaucoup plus rapide à calculer, ce calcul étant répété un grand nombre de fois.

3.5.7 Optimisation des paramètres

Les paramètres de notre algorithme sont $\{\mu_0, \sigma_0, \beta, \gamma, \epsilon, \delta\}$. Le dernier ne fait pas partie du modèle probabiliste et sera ajusté séparément pour assurer la convergence (sec. 3.6.1). Les paramètres $\Theta = \{\mu_0, \sigma_0, \beta, \gamma, \epsilon\}$ sont optimisés par maximum de vraisemblance de type II (maximisation de la preuve).

Cette optimisation est réalisée en deux temps. D'abord nous procédons à une maximisation dans l'espace $\{\beta, \gamma\}$, puis nous exprimons une formule pour déduire ϵ à partir des données et de β. Comme $\{\mu_0, \sigma_0\}$ définissent le zéro et l'unité de l'échelle des cotations, ils ne font pas varier $P(\Delta|\Theta)$ et sont choisis pour que les cotations finales ressemblent aux cotations Elo.

Optimisation de β et γ

Nous pourrions obtenir une estimation ponctuelle de $\Theta = \{\beta,\ \gamma\}$ par son MAP :

$$\begin{aligned}\Theta^* &= \text{ArgMax}_\Theta\ P(\Theta|\Delta) \\ &= \text{ArgMax}_\Theta\ \frac{P(\Delta|\Theta)\ P(\Theta)}{P(\Delta)} \\ &= \text{ArgMax}_\Theta\ P(\Delta|\Theta)\ P(\Theta).\end{aligned}$$

Considérant $P(\Theta)$ comme uniforme, nous cherchons le maximum de la vraisemblance intégrée (type II) :

$$\begin{aligned}\Theta^* &= \text{ArgMax}_\Theta\ P(\Delta|\Theta) \\ &= \text{ArgMax}_\Theta\ \int_H P(H,\ \Delta|\Theta)\, dH \\ &=: \text{ArgMax}_\Theta\ Z(\Theta) \\ &=: \text{ArgMax}_\Theta\ \log Z\end{aligned}$$

où H est l'ensemble des variables cachées du modèle (niveaux, performances et différences de performances). Avec les notations de EP (3.5.1), les variables sont $\mathbf{x} := \{H, \Delta\}$ et la jointe $p(\mathbf{x}) := P(H, \Delta|\Theta)$ est non normalisée car les données Δ ont été observées. Lors de l'inférence, nous ne nous soucions pas de la normalisation des messages, en ne propageant que les deux premiers moments des gaussiennes. Nous cherchons donc à approximer Z une fois la convergence réalisée.

Une fois l'agenda terminé, EP retourne une distribution $\hat{p}$ non normalisée minimisant approximativement $D_{\text{KL}}(\, p \,||\, \hat{p}\,)$. L'approximation de Z est alors $\hat{Z} := \int \hat{p}(\mathbf{x})\, d\mathbf{x}$, constante de normalisation de $\hat{p}$.

La dérivation permettant de calculer $\hat{Z}$ n'est pas triviale et les indications de Minka dans (Min01b ; Min01c) sont assez succinctes. Nous donnons ici les démonstrations manquantes et la dérivation du calcul de la preuve pour le cas particulier d'un graphe de facteur et d'une famille de distributions factorisées $\mathcal{F}$.

D'abord nous montrons que l'approximation $Z \approx \hat{Z}$ est justifiée. Pour cela, considérons d'abord que $\hat{p}$ minimise la divergence globale $D_{\text{KL}}(\, p \,||\, \hat{p}\,)$. Alors le théorème 6 nous assure que $Z = \hat{Z}$. Le problème est que EP ne minimise pas la divergence globale, mais, comme nous l'avons vu, une succession de divergences locales. Il n'est donc pas garantit que le $\hat{Z}$ final soit strictement égal à Z cependant l'approximation est de bonne qualité (Min05).

Théorème 6. *Soit p une distribution non normalisée et $\hat{p} = ArgMin_{q\in\mathcal{F}}\ D_{KL}(\, p \,||\, q\,)$ avec $\mathcal{F}$ une famille de fonctions normalisables. Alors $\int \hat{p}(\mathbf{x})\, d\mathbf{x} = \int p(\mathbf{x})\, d\mathbf{x}$.*

Démonstration. Comme $q \in \mathcal{F}$, posons $q(\mathbf{x}) = K\ \overline{q}(\mathbf{x})$ avec $\int \overline{q}(\mathbf{x})\, d\mathbf{x} = 1$. Alors

$$\begin{aligned}\hat{p} &= \mathrm{ArgMin}_{q\in\mathcal{F}}\ \int p(\mathbf{x}) \log \frac{p(\mathbf{x})}{q(\mathbf{x})}\, d\mathbf{x} - \int (q(\mathbf{x}) - p(\mathbf{x}))\ d\mathbf{x} \\ &= \mathrm{ArgMin}_{K,\,\overline{q}}\ -\int p(\mathbf{x}) \log\left(K\ \overline{q}(\mathbf{x})\right)\ d\mathbf{x} + K \\ &= \mathrm{ArgMin}_{K,\,\overline{q}}\ K - \log K \int p(\mathbf{x})\, d\mathbf{x} - \int p(\mathbf{x}) \log \overline{q}(\mathbf{x})\, d\mathbf{x}.\end{aligned}$$

Les choix de K et $\overline{q}$ sont indépendants, donc $K^* = \mathrm{ArgMin}_{K\in\mathbb{R}}\ K - \log K \int p(\mathbf{x})\, d\mathbf{x}$. En annulant la dérivée par rapport à K, il vient $K^* = \int p(\mathbf{x})\, d\mathbf{x}$. □

Il suffit donc de calculer $\hat{Z} = \int \hat{p}(\mathbf{x}) d\mathbf{x}$. Ce calcul serait assez facile si nous prenions garde à ajuster correctement les constantes de normalisation des messages durant l'inférence (EP *rigoureux*). Or nous ne nous préoccupons pas de ces constantes, car nous ne propageons que les premier et deuxième moments de gaussiennes, alors que le moment d'ordre zéro devrait aussi être calculé lors des projections. La raison de ce choix est double : d'abord nous allons voir qu'il est possible de retrouver le bon $\hat{Z}$ et ensuite, pour des raisons de stabilité numérique, il est conseillé de normaliser les messages lors de l'inférence, ce que nous avons implicitement fait. En pratique, nous réalisons donc un EP *normalisé*.

Dans un souci de simplicité, renommons $\hat{p}' = \prod_a \hat{f}'_a$ la production d'un EP rigoureux, gérant convenablement les moments d'ordre 0 et $\hat{p} = \prod_a \hat{f}_a$ la distribution que nous avons en pratique obtenue, en renormalisant à chaque fois les messages calculés (*i.e.* $\forall a\ \int \hat{f}_a = 1$). Nous voulons retrouver $\hat{Z}'$ ($\hat{Z}' \approx Z$) à partir des résultats de l'inférence. Nous montrons alors que $\hat{Z}'$ est un produit de constantes locales à chaque facteur et de constantes locales à chaque variable.

Théorème 7. *La constante de normalisation $\hat{Z}'$ d'un EP rigoureux se calcul à partir des messages d'un EP normalisé par*

$$\hat{Z}' = \prod_k z_k \cdot \prod_a s_a, \tag{3.33}$$

avec pour chaque variable x_k

$$z_k = \int_{x_k} \prod_a m_{a\to k}(x_k)\, d\mathbf{x} \tag{3.34}$$

et pour chaque facteur f_a

$$s_a = \frac{1}{\int_{\mathbf{x}} \hat{p}(\mathbf{x})\, d\mathbf{x}} \cdot \int_{\mathbf{x}} \frac{f_a(\mathbf{x})}{\prod_k m_{a\to k}(x_k)} \cdot \hat{p}(\mathbf{x})\, d\mathbf{x}. \tag{3.35}$$

Démonstration. Définissons s_a comme la constante de normalisation de $\hat{f}'_a$ ($\hat{f}'_a = s_a \hat{f}_a$). Alors

$$\hat{Z}' = \int_{\mathbf{x}} \hat{p}'(\mathbf{x})\, d\mathbf{x} = \int_{\mathbf{x}} \prod_a \hat{f}'_a(\mathbf{x})\, d\mathbf{x} = \int_{\mathbf{x}} \prod_a s_a \hat{f}_a(\mathbf{x})\, d\mathbf{x} \tag{3.36}$$

$$= \left(\prod_a s_a\right) \int_{\mathbf{x}} \hat{p}(\mathbf{x})\, d\mathbf{x} \tag{3.37}$$

Grâce aux deux factorisations, le second terme devient

$$\int_{\mathbf{x}} \hat{p}(\mathbf{x})\, d\mathbf{x} = \int_{\mathbf{x}} \prod_a \hat{f}_a(\mathbf{x})\, d\mathbf{x} = \int_{\mathbf{x}} \prod_a \prod_k m_{a\to k}(x_k)\, d\mathbf{x} = \int_{\mathbf{x}} \prod_k \prod_a m_{a\to k}(x_k)\, d\mathbf{x} \tag{3.38}$$

$$= \prod_k \int_{x_k} \prod_a m_{a\to k}(x_k)\, d\mathbf{x} = \prod_k z_k \tag{3.39}$$

Nous avons donc $\hat{Z}' = \prod_a s_a \prod_k z_k$. Il reste à calculer les s_a. Lors de l'EP normalisé, chaque projection donne une fonction g^* :

$$g^* = \text{ArgMin}_g\ D_{\text{KL}}\Big(\hat{p}^{\setminus a}\, f_a \,||\, \hat{p}^{\setminus a}\, g \Big). \tag{3.40}$$

D'où nous tirons le nouveau facteur normalisé $\hat{f}_a(\mathbf{x})$:

$$g^*(\mathbf{x}) = s^*\ \hat{f}_a(\mathbf{x}). \tag{3.41}$$

Or, il s'avère que la fonction g^* est en fait $\hat{f}'_a$ et donc que $s^* = s_a$. En effet, les projections d'un EP rigoureux donnent :

$$\hat{f}'_a = \text{ArgMin}_g\ D_{\text{KL}}\Big(\hat{p}'^{\setminus a}\, f_a \,||\, \hat{p}'^{\setminus a}\, g \Big) \tag{3.42}$$

$$= \text{ArgMin}_g\ D_{\text{KL}}\left(f_a \prod_{b\neq a} \hat{f}'_b \,||\, g \prod_{b\neq a} \hat{f}'_b \right) \tag{3.43}$$

$$= \text{ArgMin}_g\ D_{\text{KL}}\left(f_a \prod_{b\neq a} s_b \prod_{b\neq a} \hat{f}_b \,||\, g \prod_{b\neq a} s_b \prod_{b\neq a} \hat{f}_b \right) \tag{3.44}$$

$$= \text{ArgMin}_g\ D_{\text{KL}}\left(f_a \prod_{b\neq a} \hat{f}_b \,||\, g \prod_{b\neq a} \hat{f}_b \right) \tag{3.45}$$

$$= \text{ArgMin}_g\ D_{\text{KL}}\Big(\hat{p}^{\setminus a}\, f_a \,||\, \hat{p}^{\setminus a}\, g \Big) \tag{3.46}$$

$$= g^*. \tag{3.47}$$

Ayant ainsi montré que $s_a = s^*$, une méthode similaire à celle de la démonstration du théorème 6, mais appliqué à la projection 3.40 nous permet de calculer s_a. En écrivant les fonctions $g(\mathbf{x}) = s \cdot \overline{g}(\mathbf{x})$ avec $\int \overline{g} = 1$ et en ouvrant la KL, il vient :

$$s^* = \text{ArgMin}_s\ -\log s \int_{\mathbf{x}} \hat{p}^{\setminus a}(\mathbf{x})\, f_a(\mathbf{x})\, d\mathbf{x} + s \int_{\mathbf{x}} \hat{p}^{\setminus a}(\mathbf{x}) \overline{g}^*\, d\mathbf{x}. \tag{3.48}$$

En annulant la dérivée et comme nous avons vu que l'optimal $\overline{g}^*$ est par définition $\hat{f}_a$ (eq. 3.41), nous avons finalement

$$s_a = \frac{\int_{\mathbf{x}} \hat{p}^{\setminus a}(\mathbf{x})\, f_a(\mathbf{x})\, d\mathbf{x}}{\int_{\mathbf{x}} \hat{p}^{\setminus a}(\mathbf{x}) \hat{f}_a\, d\mathbf{x}} \tag{3.49}$$

$$= \frac{1}{\int_{\mathbf{x}} \hat{p}(\mathbf{x})\, d\mathbf{x}} \cdot \int_{\mathbf{x}} \frac{f_a(\mathbf{x})}{\hat{f}_a(\mathbf{x})} \cdot \hat{p}(\mathbf{x})\, d\mathbf{x}. \tag{3.50}$$

Comme $\hat{f}_a(\mathbf{x}) = \prod_k m_{a\to k}(x_k)$, le théorème est démontré. □

Avec ce théorème, il est aisé de calculer les z_k, qui sont les constantes de normalisations des produits des messages arrivant à chaque variable. Par exemple, une variable de performance de TrueChess a deux messages entrants, qui sont deux gaussiennes implicitement normalisées. Leur produit n'est pas normalisé, mais sa constante est facile à calculer à partir des propriétés des gaussiennes (cf. annexe A.2.3).

Les s_a sont plus délicats à calculer car ils nécessitent d'intégrer les fonction f_a. Cependant, le s_a d'un facteur devient simple si ce facteur envoie un message exact. Nous rappelons que dans notre modèle, seul les facteurs de résultats ne sont pas exacts et nécessiterons donc une intégration. Pour simplifier les calculs de s_a, définissons les marginales normalisées et les marginales incomplètes normalisées :

$$\overline{q}(x_i) := \frac{1}{z_i} \prod_a m_{a \to i}(x_i) \tag{3.51}$$

$$z_i^{\backslash a} := \int_{x_i} \prod_{b \neq a} m_{b \to i}(x_i)\, dx_i \tag{3.52}$$

$$\overline{q}^{\backslash a}(x_i) := \frac{1}{z_i^{\backslash a}} \prod_{b \neq a} m_{b \to i}(x_i). \tag{3.53}$$

Alors nous obtenons

$$s_a = \int_{\mathbf{x}} \frac{f_a(\mathbf{x})}{\prod_i m_{a \to i}(x_i)} \prod_i \overline{q}(x_i)\, d\mathbf{x} \tag{3.54}$$

$$= \int_{\mathbf{x}} f_a(\mathbf{x}) \prod_i \frac{z_i^{\backslash a}}{z_i} \overline{q}^{\backslash a}(x_i)\, d\mathbf{x} \tag{3.55}$$

$$= \left(\prod_{x_i \in \mathbf{x}_a} \frac{z_i^{\backslash a}}{z_i} \right) \int_{\mathbf{x}_a} f_a(\mathbf{x}) \prod_{x_i \in \mathbf{x}_a} \overline{q}^{\backslash a}(x_i)\, d\mathbf{x}_a. \tag{3.56}$$

L'équation 3.56 est la formule à utiliser pour calculer le s_a d'un facteur non exact. Il faut pour cela calculer, pour chacune des variables attachées au facteur, les valeurs de z_i, de $z_i^{\backslash a}$ et de $\overline{q}^{\backslash a}(x_i)$, puis intégrer par rapport à f_a. Ces calculs sont relativement aisés car les nœuds du graphe ont en général peu de voisins et car les gaussiennes sont pratiques à manipuler. Les calculs sont en fait similaires à ceux de la dérivation des messages. Dans le cas TrueChess, les seuls messages non exacts sont envoyés par les facteurs de résultats. Prenons l'exemple d'un facteur $f_a(d) = \delta_{d>0}(d)$. Nous avons $\mathbf{x}_a = d$ et $s_a = \frac{z_d^{\backslash a}}{z_d} \int_d \delta_{d>0}(d)\ \overline{q}^{\backslash a}(d)$. Les calculs de $z_d^{\backslash a}$ et de z_d sont faciles et comme $\delta_{d>0}(d)\ \overline{q}^{\backslash a}(d)$ est une GTR, évaluer son aire est aisé avec la fonction Φ.

Pour les facteurs envoyant des messages exacts vers x_j, nous avons

$$m_{a \to j}(x_j) = \int_{\mathbf{x}_a^{\backslash j}} f_a(\mathbf{x}_a) \prod_{x_i \in \mathbf{x}_a, i \neq j} \overline{q}^{\backslash a}(x_i)\, dx_i \tag{3.57}$$

car la marginale de x_j est déjà dans la famille exponentielle $\mathcal{F}$, avant la projection. Alors

s_a devient

$$s_a = \left(\prod_{x_i \in \mathbf{x}_a} \frac{z_i^{\setminus a}}{z_i}\right) \int_{\mathbf{x}_a} f_a(\mathbf{x}) \prod_{x_i \in \mathbf{x}_a} \overline{q}^{\setminus a}(x_i)\, d\mathbf{x}_a \tag{3.58}$$

$$= \left(\prod_{x_i \in \mathbf{x}_a} \frac{z_i^{\setminus a}}{z_i}\right) \int_{x_j} \int_{\mathbf{x}_a^{\setminus j}} f_a(\mathbf{x}) \prod_{x_i \in \mathbf{x}_a, i \neq j} \overline{q}^{\setminus a}(x_i)\, dx_i\, \overline{q}^{\setminus a}(x_j)\, dx_j \tag{3.59}$$

$$= \left(\prod_{x_i \in \mathbf{x}_a} \frac{z_i^{\setminus a}}{z_i}\right) \frac{z_j}{z_j^{\setminus a}} \tag{3.60}$$

$$= \prod_{x_i \in \mathbf{x}_a, i \neq j} \frac{z_i^{\setminus a}}{z_i} \tag{3.61}$$

Ainsi nous avons tous les éléments nécessaires pour calculer $\hat{Z}$, approximation de Z. Pour des raisons numériques, nous calculons plutôt $\log \hat{Z}$. Comme $\hat{Z}$ est une fonction de Θ, sa maximisation nous permettra de choisir les paramètres Θ.

Optimisation de ϵ

La définition de ϵ (3.2) le lie directement avec la probabilité d'obtenir un match nul, en marginalisant cotations et performances :

$$\begin{aligned} P(\text{nul}) &= 2\Phi\left(\frac{\epsilon}{\sqrt{2\,\beta^2 + 2\,\sigma_0^2}}\right) - 1 \\ \text{soit} \qquad \epsilon &= \sqrt{2\,\beta^2 + 2\,\sigma_0^2}\;\Phi^{-1}\left(\frac{1 + P(\text{nul})}{2}\right). \end{aligned} \tag{3.62}$$

La probabilité $P(\text{nul})$ est estimée par maximum de vraisemblance comme la fréquence empirique des matchs nuls dans Δ. Nous présenterons section 3.6.6 un modèle plus complet évitant cette estimation.

3.5.8 Interpolation des cotations

Jusqu'ici nous n'avons calculé les marginales des cotations qu'aux instants où le joueur avait participé à un tournoi. Cependant, comme notre modèle dynamique est exprimé en temps continu, il est possible de trouver ces marginales quelque soit t. Une première méthode serait d'introduire une nouvelle variable s_i^t au modèle, connectée uniquement par des facteurs d'évolution dynamique, puis de relancer l'inférence sur le graphe complet. Mais il est possible d'éviter ce calcul en utilisant directement les marginales produites et le modèle dynamique. Notons $\pi_d = \frac{1}{\gamma^2(t_2 - t_1)}$ la précision du bruit dynamique.

Pour estimer la marginale s_i^t à partir de $s_i^{t_1}$ et $s_i^{t_2}$ avec $t_1 \leq t \leq t_2$, il faut d'abord estimer les facteurs de "vraisemblance" du modèle de la figure 3.15. Ces facteurs sont gaussiens

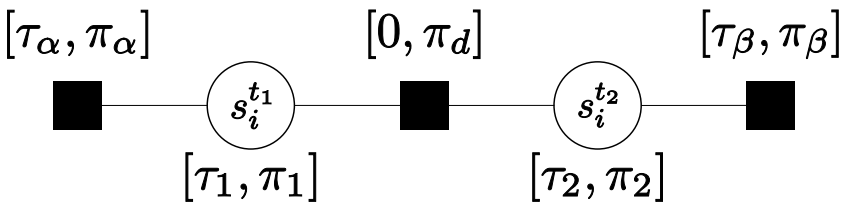


FIG. 3.15 – Après inférence, les marginales de deux cotations successives d'un joueur peuvent être vues comme résultantes du produit d'un facteur de lien dynamique et de deux facteurs de "vraisemblance". Il est possible d'estimer les paramètres de ces facteurs f_α et f_β à partir des marginales et de la précision du lien dynamique π_d.

et les équations de propagation des messages nous donnent le système d'équation suivant (paramètres canoniques) :

$$\begin{cases} \tau_1 &= \tau_\alpha + \frac{\frac{\tau_\beta}{\pi_\beta}}{\frac{1}{\pi_\beta}+\frac{1}{\pi_d}} \\ \tau_2 &= \tau_\beta + \frac{\frac{\tau_\alpha}{\pi_\alpha}}{\frac{1}{\pi_\alpha}+\frac{1}{\pi_d}} \\ \pi_1 &= \pi_\alpha + \frac{1}{\frac{1}{\pi_\beta}+\frac{1}{\pi_d}} \\ \pi_2 &= \pi_\beta + \frac{1}{\frac{1}{\pi_\alpha}+\frac{1}{\pi_d}}. \end{cases}$$

Considérant que les précisions sont des nombres positifs, la solution du système est donnée par :

$$\begin{cases} \pi_\beta &= \frac{-b+\sqrt{d}}{2a} \\ \pi_\alpha &= \pi_1 - \frac{\pi_\beta \pi_d}{\pi_\beta + \pi_d} \\ \tau_\beta &= \frac{\tau_2 - B\,\tau_1}{1 - A\,B} \\ \tau_\alpha &= \tau_1 - A\,\tau_\beta \end{cases} \quad \text{avec} \quad \begin{cases} a &:= \pi_1 \\ b &:= 2\,\pi_d \pi_1 - \pi_1 \pi_2 \\ c &:= \pi_d^2 \pi_1 - \pi^2 \pi_2 - \pi_d \pi_1 \pi_2 \\ d &:= b^2 - 4ac \\ A &:= \frac{\pi_d}{\pi_\beta + \pi_d} \\ B &:= \frac{\pi_d}{\pi_\alpha + \pi_d} \end{cases}$$

Une fois ces facteurs connus, le modèle de la figure 3.16 permet de trouver la marginale de s_i^t. Les formules de passage de messages donnent, pour $t_1 \leq t \leq t_2$:

$$\begin{cases} \tau_t &= \frac{\frac{\tau_\alpha}{\pi_\alpha}}{v_1 + \frac{1}{\pi_\alpha}} + \frac{\frac{\tau_\beta}{\pi_\beta}}{v_2 + \frac{1}{\pi_\beta}} \\ \pi_t &= \frac{1}{v_1 + \frac{1}{\pi_\alpha}} + \frac{1}{v_2 + \frac{1}{\pi_\beta}} \end{cases} \quad \text{avec} \quad \begin{cases} v_1 &:= \gamma^2\,(t - t_1) \\ v_2 &:= \gamma^2\,(t_2 - t) \end{cases}$$

La figure 3.17 présente l'interpolation de cotation dans un cas simple, avec deux niveaux de bruits dynamiques différents. Comme sous entendu par ce modèle de pont brownien, la moyenne des marginales évolue linéairement avec le temps. L'écart type suit quant à lui une courbe non rectiligne, représentant un accroissement d'incertitude entre deux points de données. La figure 3.18 présente cette interpolation de cotation pour le jeu de données des

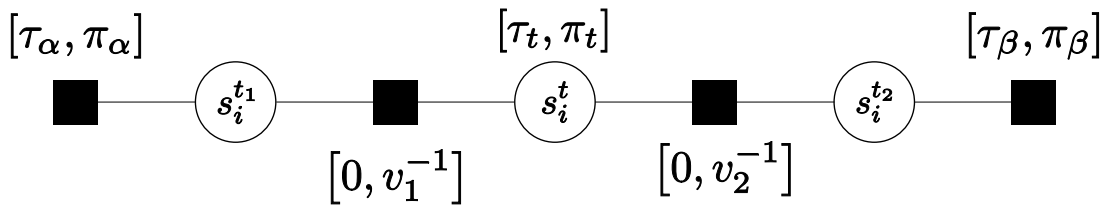


FIG. 3.16 – Modèle permettant l'estimation d'une cotation intermédiaire au temps t.

échecs, avec les paramètres maximisant la preuve. Nous remarquons que la variation des écarts types est presque linéaire (Fig. 3.19). Ceci est le symptôme d'une faiblesse relative du bruit dynamique par rapport à l'incertitude des vraisemblances. Nous en déduisons que la variabilité des niveaux des joueurs est plus la conséquence de leur fluctuation intra tournoi que de leur évolution temporelle.

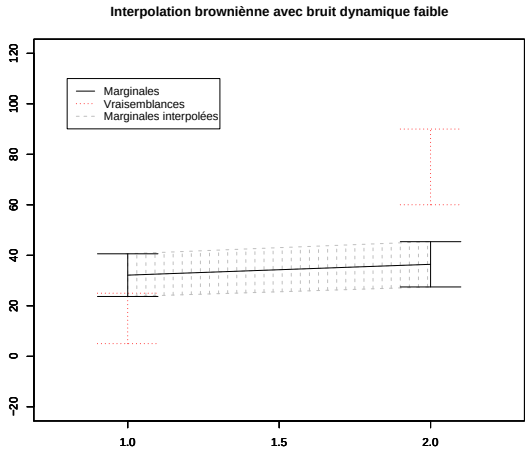
3.6 TrueChess : Résultats

Nous avons mis en place toutes les briques permettant de réaliser l'inférence des cotations : formules de messages, agenda et calcul de la preuve. Dans cette section, nous étudions d'abord la convergence de EP, puis nous nous intéressons à l'influence des paramètres sur les résultas produits. Après avoir choisis les meilleurs paramètres, nous appliquons notre algorithme à deux jeux de données (un "petit" et un "grand") et finalement nous validons nos résultats en les comparant à ceux de Elo.

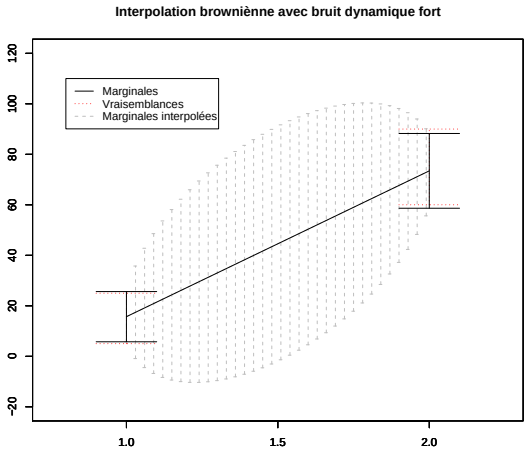
3.6.1 Convergence

Le plus haut niveau de l'agenda est une itération de passage de messages du passé vers le futur (passe *aller*), puis du futur vers le passé (passe *retour*). La figure 3.20(a) montre que EP a presque convergé dès la deuxième itération. Après deux *aller-retour*, la preuve ne varie quasiment plus, ni aucune des ≈60 000 distributions du modèle. Ainsi $\delta = 10^{-2}$ suffit pour assurer la convergence et 10^{-3} assure une estimation minutieuse. Le temps de calcul est alors de l'ordre de dix secondes sur un ordinateur de bureau standard ($\delta = 10^{-2}$). La construction du graphe et de l'agenda prend environ 45 secondes avec ce premier jeu de données.

Comme les cotations produites sont visuellement satisfaisantes et globalement en accord avec les experts du domaine, nous pensons que le minimum local atteint est de bonne qualité. EP étant un algorithme déterministe, il n'est pas possible de trouver d'autres minima, sauf éventuellement en changeant d'agenda. De tels essais ont conduit à des cotations numériquement semblables, au prix d'un temps de calcul accru. La figure 3.20 illustre aussi l'intérêt indéniable de rétro propager l'information pour obtenir un classement historique.



(a) Bruit dynamique faible ($\gamma^2 = 100$). Le lien entre deux cotations proches est fort, elles doivent être similaires. Les marginales et les vraisemblances sont donc assez différentes. Peu d'incertitude est ajoutée aux cotations intermédiaires.



(b) Bruit dynamique fort ($\gamma^2 = 33000$). Comme le lien est faible, les marginales sont proches des vraisemblances. Le fort bruit ajoute beaucoup d'incertitude entre t_1 et t_2.

FIG. 3.17 – Estimation des cotations intermédiaires à partir des cotations au instants t_1 et t_2 et du modèle dynamique brownien. Les marginales sont tracées en noir (trait plein), les cotations intermédiaires résultantes en pointillés gris. Les distribution représentées en rouge correspondent aux facteurs de vraisemblance, c.-à-d. aux marginales amputées du lien dynamique sur t_1 et t_2.

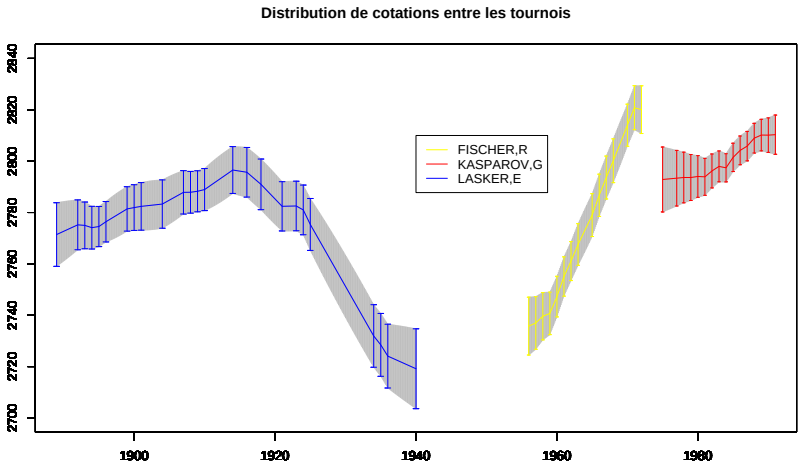


FIG. 3.18 – Interpolation des estimations des cotations pour Kasparov, Fischer et Lasker ($\mu \pm \sigma$).

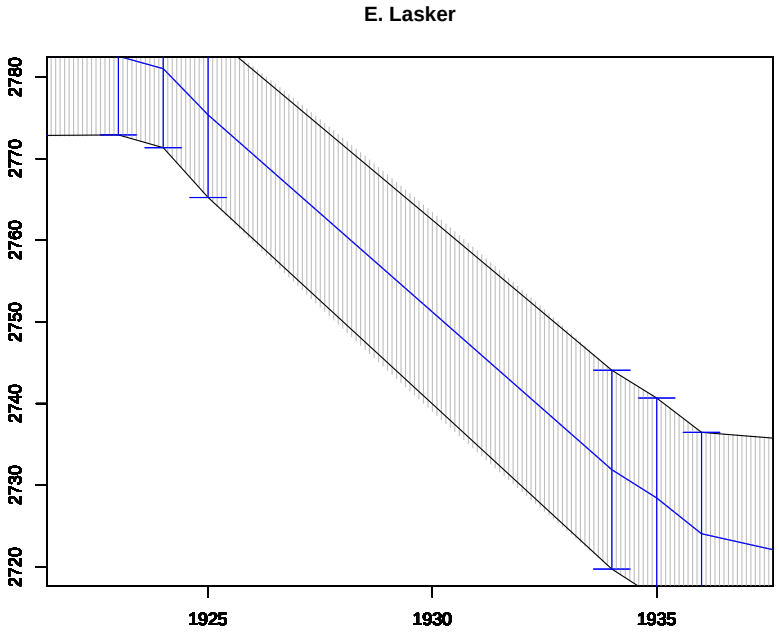
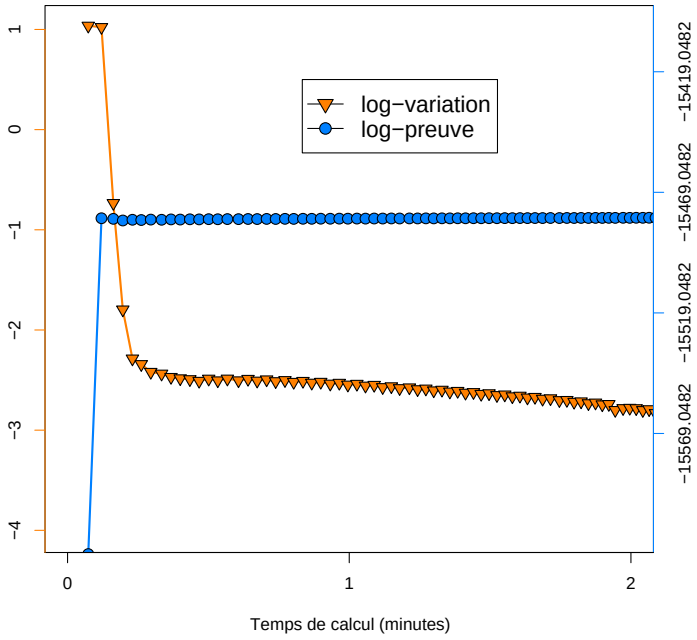
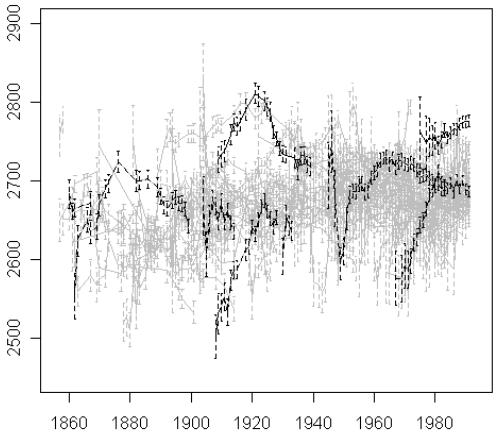


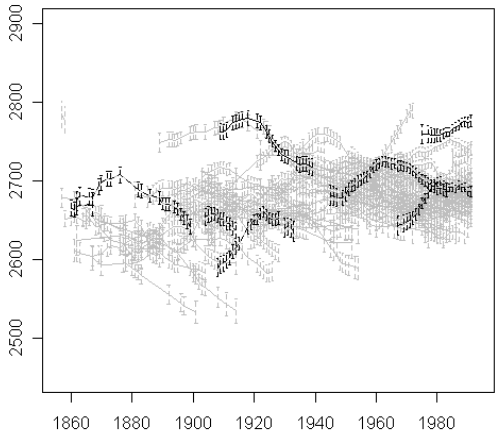
FIG. 3.19 – Zoom sur les cotations de Lasker. La variation temporelle de l'écart type σ est faiblement non linéaire (débordement des segments de droite noirs).



(a) Convergence



(b) Une passe *aller*.



(c) Un *aller-retour*.

FIG. 3.20 – Convergence sur un jeu de 15 664 parties. Fig (a) : une mesure de la preuve et de la variation des distributions est réalisée après chaque passe *aller* et chaque passe *retour* de l'agenda. Les 60 000 distributions varient beaucoup pendant le premier *aller* et le premier *retour* et se stabilisent ensuite assez vite. La stabilisation de la preuve confirme la convergence. Fig $(b) - (c)$: Cotations inférées de différents joueurs au cours du temps. La comparaison de ces figures montre l'effet de lissage et de réduction des incertitudes de la passe *retour*. D'après (a), le résultat de chaque match avec les cotations lissées (c) est 0.9% plus probable que son équivalent avec (b), rendant le jeu de données entier $e^{140} \approx 10^{60}$ fois plus probable. La retro-propagation des informations est donc essentielle pour établir des cotations historiques pertinentes.

3.6.2 Influence et choix des paramètres

Parmi les paramètres de notre algorithme, seuls β et γ doivent être choisis. En effet μ_0 et σ_0 ont pour seul effet de définir l'échelle verticale et nous avons vérifié qu'ils ne font pas varier la preuve. Nous choisissons $\mu_0 = 2704$ et $\sigma_0 = 100$ pour obtenir des cotations similaires à celle de Elo. Nous pouvons voir le domaine des cotations Elo comme notre *a priori* sur les cotations TrueChess. Une fois σ_0 fixé, seules les valeurs des ratios β/σ_0 et γ/σ_0 sont pertinentes.

Influence de β et γ

Les figures 3.21 et 3.22 présentent les cotations inférées pour différents β (variabilité de performance) et γ (variabilité temporelle des niveaux). Comme nous constatons que ces deux paramètres ont une influence sur les classements relatifs, ils seront donc choisis avec soin, en maximisant la preuve du modèle.

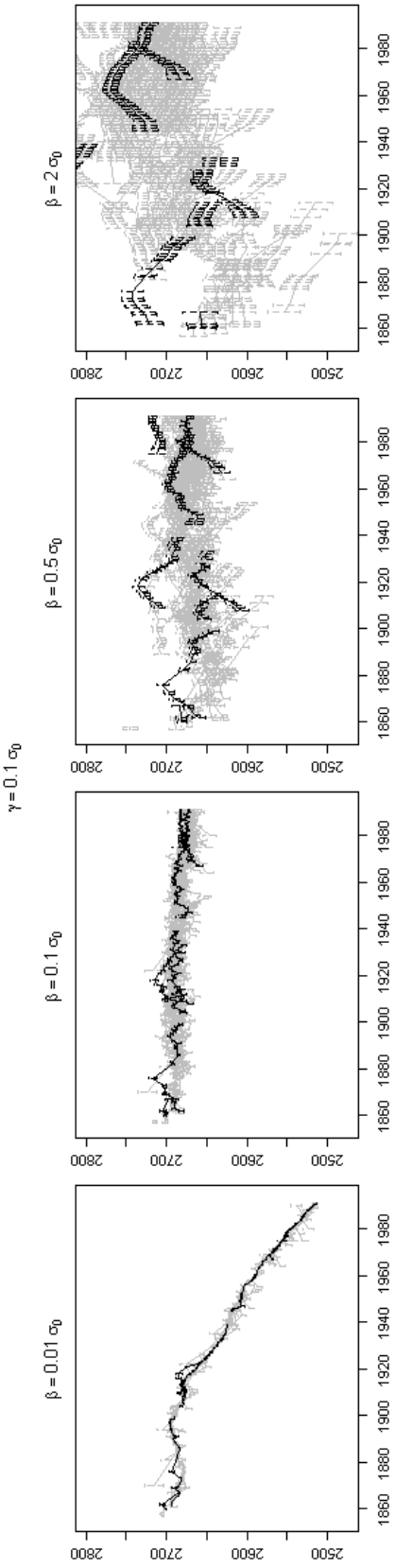


FIG. 3.21 – Influence de la variabilité de performance β sur les cotations ($\gamma = 0.1\sigma_0$). De grandes valeurs de β assouplissent les contraintes sur les cotations. Non seulement l'échelle verticale est dilatée, mais les positions relatives des courbes varient avec β. Pour β très faible, les performances sont égales aux niveaux et il est difficile de trouver un jeu de niveaux compatibles avec les résultats non transitifs des matchs. La meilleure solution est l'égalité des niveaux à chaque instant. L'allure est décroissante car les nouveaux joueurs battent en moyenne les plus anciens, mais ces derniers ont une incertitude plus faible et tirent ainsi les nouveaux vers le bas.

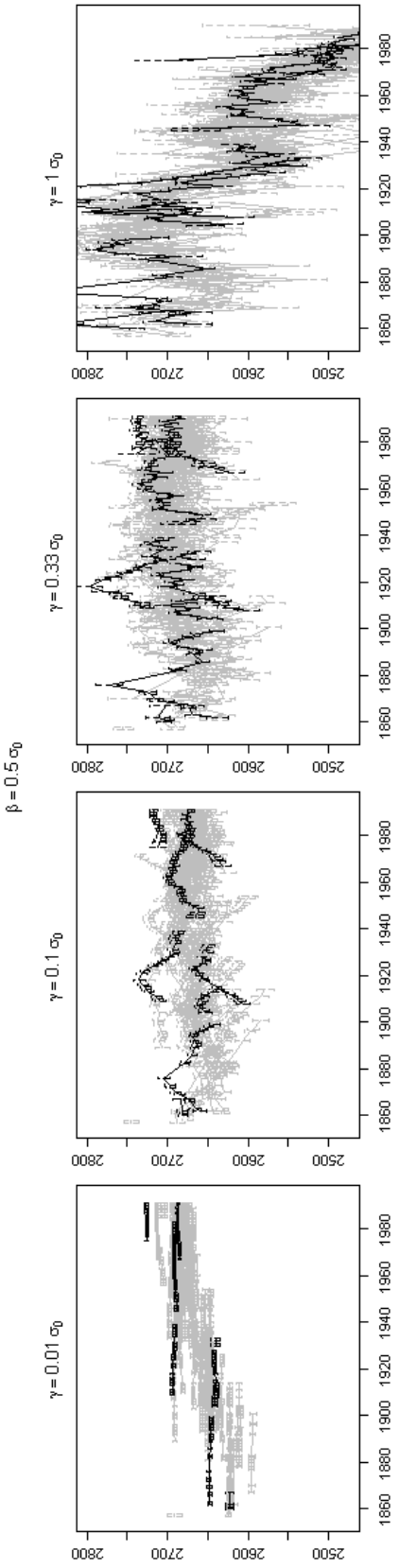
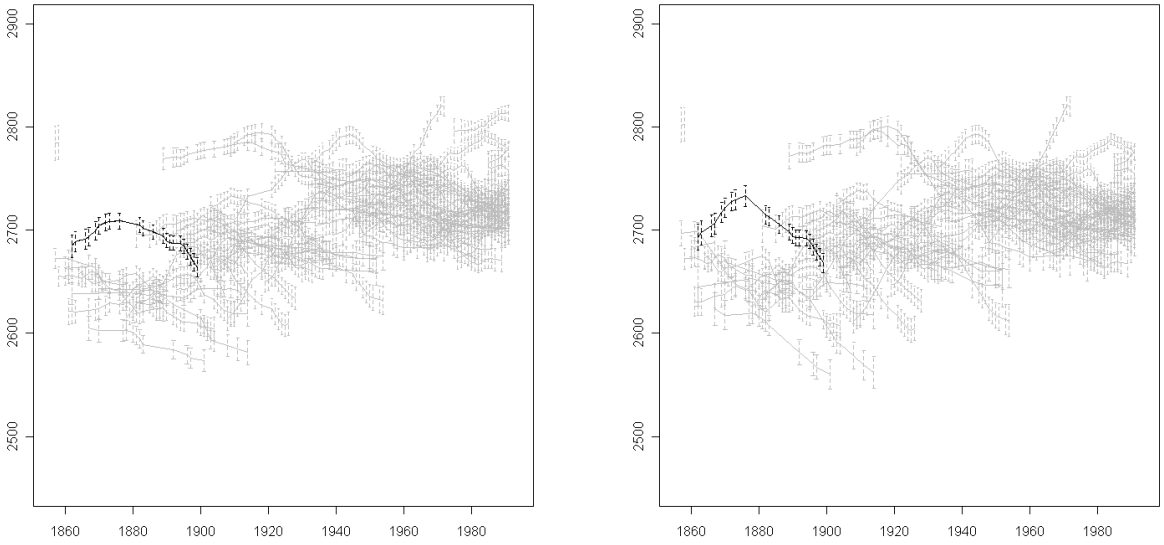


FIG. 3.22 – Influence de la variabilité dynamique γ. Une trop faible valeur impose la constance des niveaux alors qu'une trop forte annule le couplage temporel. Le paramètre γ agit comme une régularisation, un *a priori* de lissage. Différents γ conduisent à des classements relatifs différents.



(a) Modèle constant. $P(\Delta) = e^{-15485}$

(b) Modèle brownien. $P(\Delta) = e^{-15479}$

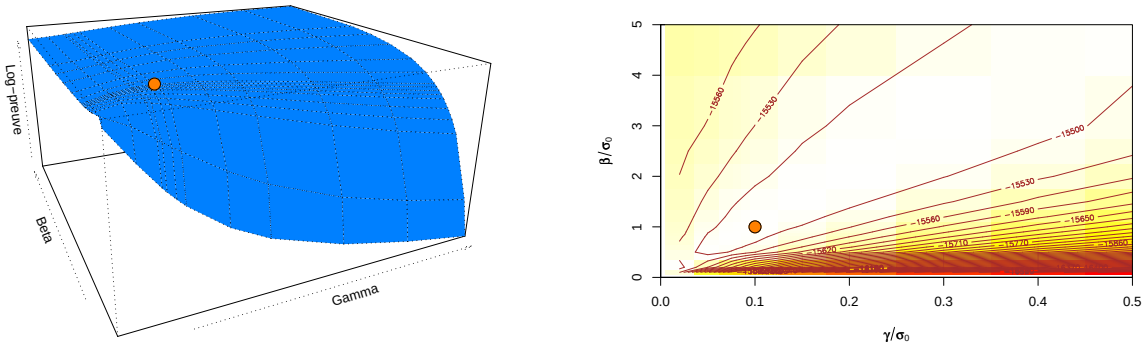
FIG. 3.23 – Comparaison des deux modèles dynamiques. Le modèle brownien (variabilité temporelle $= \gamma^2\,\Delta t$) est $e^6 \approx 10^{2.3}$ fois plus probable que le modèle à variabilité constante, soit 0.03% d'amélioration par match. Quand un joueur se retire des tournois plusieurs années, sa cotation peut varier davantage avec le modèle brownien (ex. tracé noir).

Influence du modèle dynamique

Les résultats précédents concernent le modèle dynamique brownien, où la variabilité dynamique est en fait proportionnelle au temps entre deux matchs, $\sigma_{\text{dyn}} = \sqrt{\gamma^2\,\Delta t}$. La figure 3.23 illustre la supériorité de ce modèle sur modèle constant $\sigma_{\text{dyn}} = \sqrt{\gamma^2}$: le modèle brownien étant deux cent fois plus probable. Le modèle brownien sera utilisé dans le reste de notre étude.

Choix de β et γ

Pour choisir les deux paramètres libres β et γ, nous maximisons la log-preuve (définition et formule section 3.5.7). La figure 3.24 présente le profile de $\log Z = f(\frac{\beta}{\sigma_0}, \frac{\gamma}{\sigma_0})$. Le maximum de f est atteint pour $\beta \approx \sigma_0$ et $\gamma \approx 0.1\sigma_0$. Comme la surface est globalement plane, notre système est assez robuste au choix des paramètres. La seule mauvaise région est la zone de faible β car alors les performances d'un joueur ne peuvent pas varier lors d'un tournoi et il est donc difficile d'expliquer que J_1 batte et soit battu par J_2, ce qui arrive pourtant fréquemment. Il est aussi intéressant de noter que le γ optimal à une valeur non extrême : il est donc avantageux de modéliser la dynamique des niveaux. Nous remarquons de plus une arête d'équation $\gamma = \beta/10$, suggérant que le modèle n'a (approximativement) qu'un seul degré de liberté. En effet, en restant sur cette arête, la valeur β/σ_0 définit la facilité de sortir du domaine de l'*a priori* et l'allure des cotations ne varie *pas beaucoup*. Mais elles



(a) Log-preuve (b) Vue de dessus

FIG. 3.24 – Profil et lignes de niveaux de la preuve en fonction de β et de γ. Le maximum est atteint pour $\beta = \sigma_0$ et $\gamma = 0.1\sigma_0$.

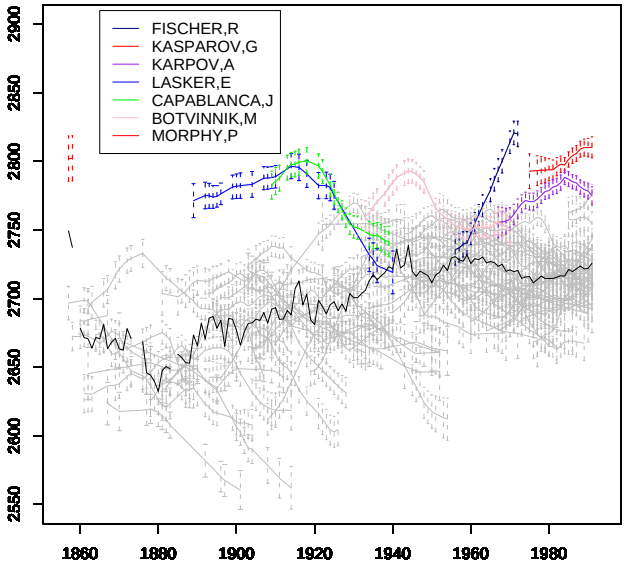
sont tout de même différentes et il est préférable de conserver les paramètres optimaux.

Finalement, nous constatons que la preuve optimale vaut e^{-15479}, ce qui en moyenne donne un probabilité de 0.37 pour chaque résultat de match g_i. Ce résultat nous mène à trois conclusions. En premier lieu $P(g_i|\Pi)$ est loin de 1, ce qui indique que nous n'avons pas sur apprit les données (pas d'*overfitting*). En effet un modèle Π_1 ayant énormément de paramètres et apprenant par cœur les données donnerait $P(g_i|\Pi_1) \approx 1$. Deuxièmement, comme un match a trois issues possibles, un modèle uniforme naïf Π_2 n'apprenant rien donnerait $P(g_i|\Pi_2) = 1/3$. Donc nous faisons mieux, nous avons bien appris quelque chose. Finalement comme $0.37 \approx 1/3$, nous pouvons conclure que les g_i sont tout de même difficiles à prévoir, ce qui sera confirmé dans la section 3.6.5. En effet les données sont issues de vrais tournois avec des participants globalement de même niveau. Il est assez rare de voir un match entre un grand maître et un débutant.

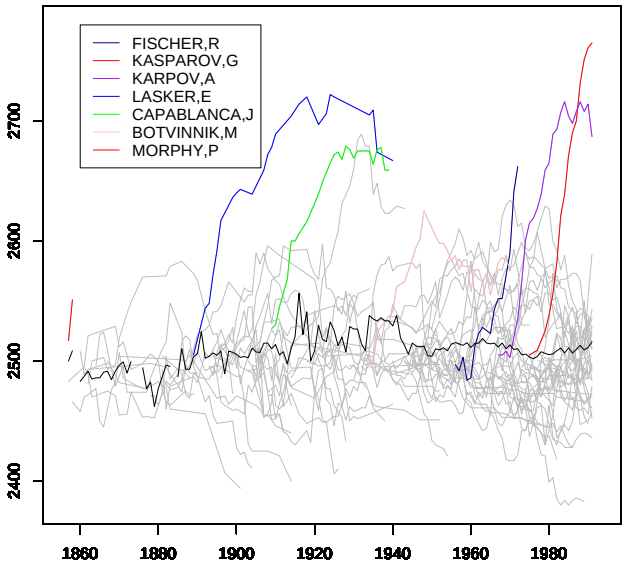
3.6.3 Classement historique

La figure 3.25(a) présente les cotations inférées avec les meilleurs paramètres pour le jeu de données de 15 664 parties entre 88 des meilleurs joueurs mondiaux au cours des XIX et XX$^{\text{ème}}$ siècles. Nous comparons les résultats de TrueChess avec ceux de Elo sans période provisionnelle, ni paliers et avec $K = 10$. Nous retrouvons dans les deux cas la domination de certains joueurs historiques, mais les résultats sont assez différents. Les cotations TrueChess semble intuitivement plus pertinentes grâce à leur allure lisse. De plus, TrueChess n'impose pas une valeur initiale pour la première cotation de chaque joueur. Au contraire notre implémentation d'Elo impose que cette première cotation soit μ_0. Finalement TrueChess donne l'incertitude sur le niveau du joueur, incertitude notablement plus grande en début/fin de carrière car non contrainte par des parties passées/futures.

Nous constatons que la carrière d'un joueur se décompose souvent en trois phases : progression, atteinte d'un maximum et stagnation, puis baisse de niveau, ce qui est intuitif et en accord avec les conclusions d'historiens des échecs. Elo reflète moins bien ce schéma,



(a) TrueChess ($\mu_0 = 2704, \sigma_0 = 100, \beta = \sigma_0, \gamma = 0.1\ \sigma_0, \delta = 10^{-3}, \epsilon = 102.17$)



(b) Elo ($\mu_0 = 2500, K = 10$)

FIG. 3.25 – Cotations comparées sur un jeu de 15 664 parties des 88 meilleurs joueurs mondiaux. Quelques joueurs célèbres sont mis en évidence. La ligne continue noire représente la moyenne des cotations. L'échelle verticale n'étant pas pertinente, seule l'allure et les positions relatives des courbes sont intéressantes.

surtout quand nous lui ajoutons des règles *ad hoc* prévenant une chute de niveau comme celle des paliers (règles FIDE (FID05)). Nous lisons d'autre part une légère augmentation de la moyenne des cotations, ce qui est possible car, contrairement à Elo, le modèle TrueChess ne définit pas un match comme un échange de points à somme nulle.

Concernant les cotations TrueChess individuelles, nous mettons en évidence Emanuel Lasker (1868-1941) (bleu), qui devint le second champion du monde en battant Steinitz par (10-4-5) en 1894 et qui régna pendant 27 ans sur ses contemporains. En 1921, il perdit officiellement son titre face à José Raúl Capablanca (1888-1942) (vert), ce qui est confirmé par notre analyse, bien que leurs niveaux se soient vraisemblablement croisés plus tôt, vers 1915. Il est à noter que le titre de champion du monde ne dépend pas des cotations, mais est décerné lors de rencontres spéciales. Capablanca est connu pour avoir été un génie des échecs dont l'ascension fut rapide et nous voyons qu'il était effectivement au plus haut niveau mondial dès ses 20 ans (1910). Il perdit son titre en 1927 contre Alexander Alekhine. En effet son niveau a beaucoup baissé au cours des années 1920.

Suite à la mort accidentelle d'Alekhine en 1946, Mikhaïl Botvinnik (1911-1995) (rose) fut le cinquième champion du monde et maintint son titre jusqu'à la période 1957-1963 durant laquelle il le perdit et le regagna deux fois. Nos cotations sont en accord avec ce fait historique, plaçant Botvinnik largement au dessus de ces contemporains entre 1935 et 1950. La victoire de Botvinnik marqua le début du règne des soviétiques sur le monde des échecs, à l'exception de la performance de l'américain Robert Fischer (1943-) (bleu marine). "Bobby" Fischer fut champion du monde de 1972 à 1975 en battant assez facilement le russe Boris Spassky. Son caractère excentrique et ses exigence en matière d'organisation des tournois ont fini par lui faire perdre son titre officiel : il refusa les conditions d'un match contre Karpov en 1975. Cependant, Fischer est souvent considéré comme l'un des plus grands joueurs de tous les temps et nos résultats laissent entrevoir qu'il avait encore un grand potentiel de progression lorsqu'il se retira. Nous reviendrons sur son cas avec notre deuxième jeu de données.

Suite au forfait de Fischer, Anatoli Karpov (1951-) (violet) ramena le titre en URSS. Toujours actif, il devient le joueur le plus victorieux de tous les temps en remportant son $161^{\text{ème}}$ tournoi en juillet 2005. Garry Kasparov (1963-) (rouge) lui reprit le titre en 1985, bien que nos résultats l'estiment meilleur plus tôt. Rappelons que toute estimation des cotations dépend cruellement du choix et de la qualité des données. Nous verrons qu'avec un jeu plus complet et plus récent, les cotations de Kasparov sont affinées et rendent compte plus fidèlement de la réalité historique.

Le dernier joueur remarquable de la figure 3.25(a) est l'américain Paul Morphy (1837-1884) (rouge). Il n'a jamais été classé officiellement, pourtant de nombreux auteurs le décrivent comme "éblouissant" et rapportent qu'il fut invaincu entre 1857 et 1884, date de sa mort.

> "Morphy, I think everyone agrees, was probably the greatest genius of them all."
> - R. Fischer -

Dès ses neuf ans, Morphy était considéré comme l'un des meilleurs joueurs de la Nou-

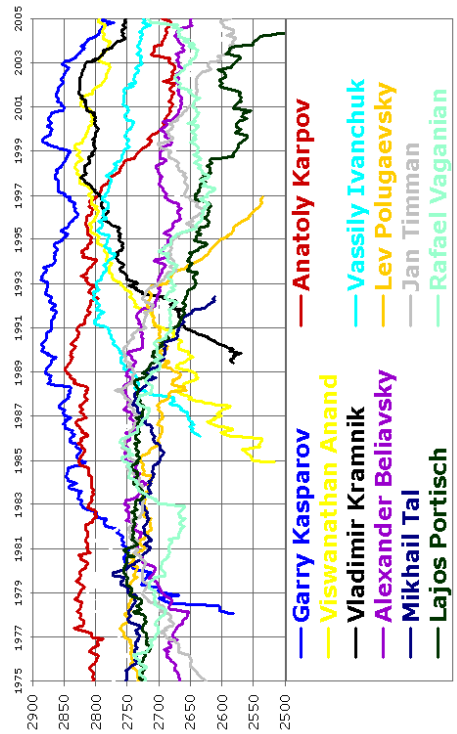
velle Orléans. Il entreprit d'étudier la loi, mais ayant obtenu son diplôme trop jeune pour pratiquer, il se mit plus sérieusement aux échecs et remporta le premier congrès américain des échecs à vingt ans. À cette époque, le haut niveau échiquéen était la propriété de l'Europe et Morphy se rendit sur le vieux continent pour y faire une tournée de dix-neuf mois. Il y battit tous les champions, y compris Adolf Anderssen, avec une aisance déconcertante. De retour en Amérique, Morphy, surnommé *The Pride and Sorrow of Chess*, décida de se retirer de la compétition après avoir vaincu quasiment tous ses opposants sérieux. Sa santé mentale se dégrada rapidement et il mourut d'une hydrocution à 47 ans, sans avoir jamais repris les échecs.

> "We also remember the brilliant flight of the American super-genius Paul Morphy, who in a couple of years conquered both the New and the Old Worlds. He revealed a thunderous brand of pragmatism, aggression, and accurate calculation to the world – qualities that enabled America to accomplish a powerful spurt in the second half of the 19th century...."
> - G. Kasparov -

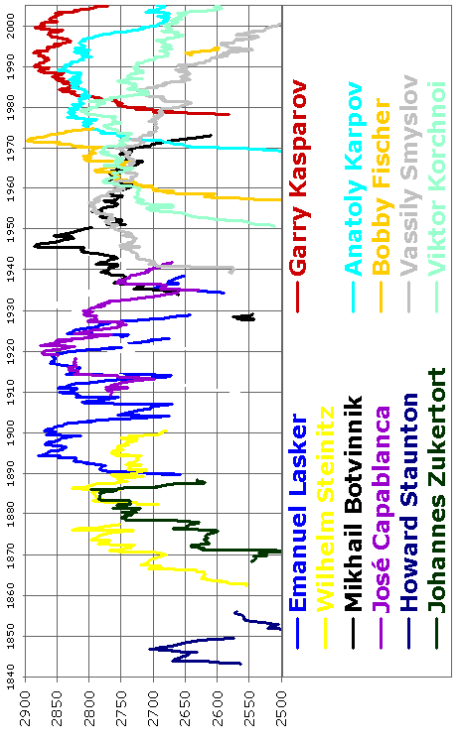
Il est souvent reporté que le jeu de Morphy a annoncé l'avènement des échecs modernes.

> "During our game's long history, the two most talked about and legendary players are, without a doubt, Paul Morphy and Robert Fischer. Their careers were oddly (and sadly) similar : both were prodigies, both dominated the other players of their time, both were American, both quit in their primes, and both suffered from mental abnormalities."
> - Jeremy Silman, Maître international -

3.6.4 Données ChessBase



(b) 1975-2004



(a) 1840-2004

FIG. 3.26 – Cotations Chessmetrics, d'après (Son05). Nous retrouvons approximativement les mêmes joueurs exceptionnels, mais nos résultats diffèrent sur de nombreux points. En particulier Anand est sous évalué et les cotations de Kasparov sont décroissantes.

Notre but étant d'établir un classement historique fiable, nous avons appliqué notre algorithme à un jeu de données plus conséquent. En effet la sélection des données est cruciale quand on prétend répondre à des questions aussi difficiles et controversée que "qui est le meilleur joueur de tous les temps". Doit-on par exemple intégrer les parties rapides, les *blitz* et les parties par correspondance ? Doit-on mélanger les différents types de tournois ? Comment évaluer la fiabilité de résultas collectés plus de deux siècles dans le passé ? Faut-il intégrer les tournois d'exhibition et les matchs amicaux ? Comment tenir compte des matchs joués avec handicap ?

J. Sonas (Chessmetrics (Son05)) et R. Edwards (Edo (Edw06a)) ont réalisé un travail impressionnant de collecte et de tri de données. Edwards ne s'intéresse qu'au XIX siècle et a soigneusement sélectionné ses données. Sonas s'est basé sur la collection ChessBase (Mat) en la complétant pour réduire certains biais de sélection. Les résultats de son algorithme non probabiliste sont présentés figure 3.26.

Nous n'avons pas effectué un tel travail historique et nous sommes basés sur l'hypothèse que la base ChessBase complète était un bon échantillon de toutes les parties jouées. Cette base est en effet la plus grande collection mondiale annotée et contient plus de 3.5 millions de parties de 1560 à 2006 entre ≈200 000 joueurs. Nous n'avons gardé que les matchs postérieurs à 1850, date de la naissance des échecs modernes. Ainsi nous avons travaillé avec 3 505 366 parties entre 206 059 joueurs uniques, la majorité d'entre elles ayant été collectées entre 1978 et 2006.

Le graphe de facteur correspondant contient alors près de douze millions de variables aléatoires reliées par plus de quinze millions de facteurs. L'algorithme devra ainsi propager environ 60 millions de messages différents.

Lors de l'inférence, l'agenda alloue environ 6 Go de mémoire et converge en moins de dix minutes sur un Pentium 4. La log-preuve optimale vaut alors -3 953 997 pour les paramètres $\mu_0 = 1200$, $\sigma_0 = 400$, $\beta = 1.2\ \sigma_0$, $\gamma = 0.15\ \sigma_0$, $\epsilon = 344.06$. La fréquence empirique de parties nulles dans la base est de 30.3%. Toutes les conclusions sur l'influence des paramètres, du modèle dynamique restent valables pour ce deuxième jeu de données.

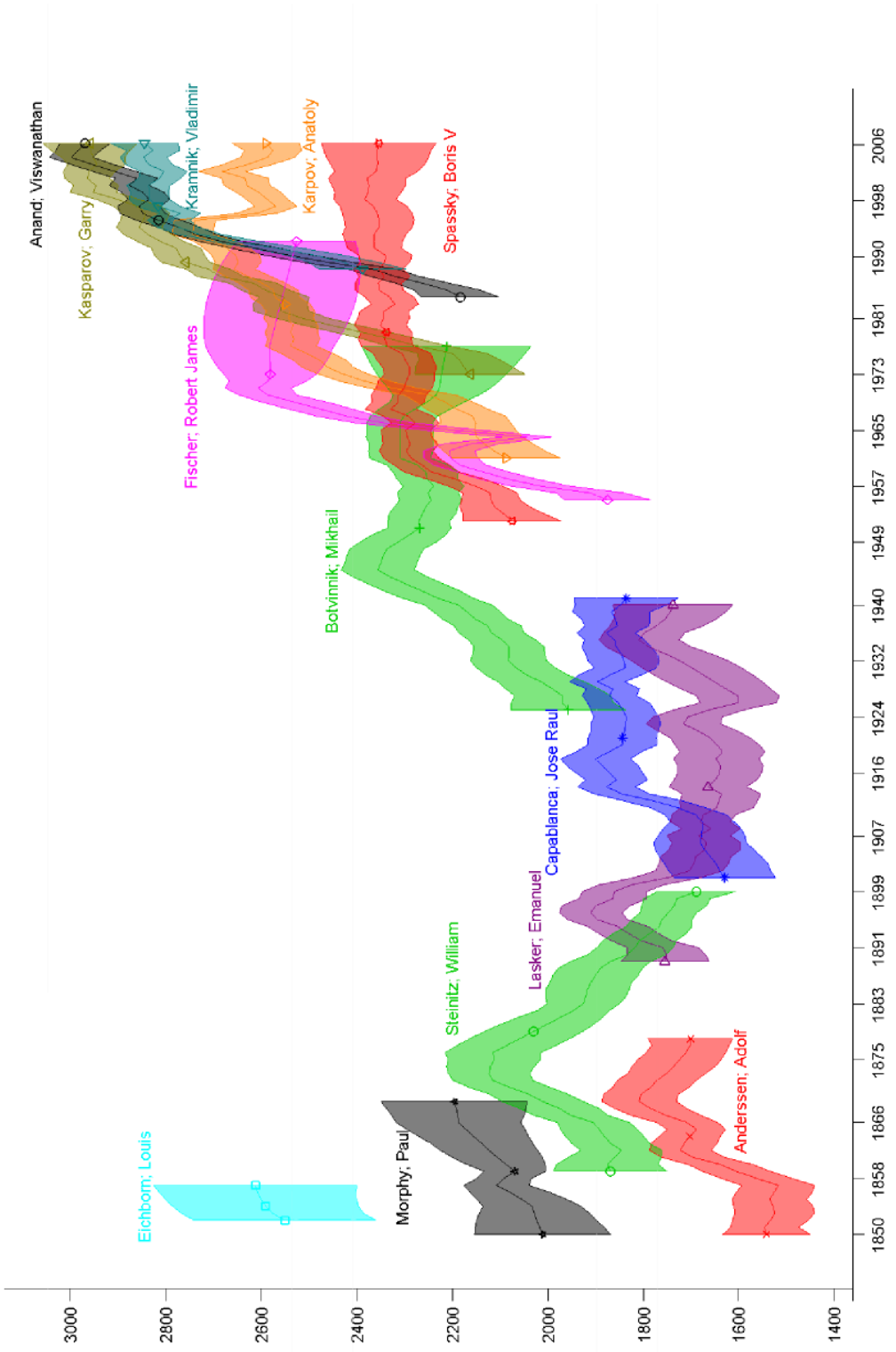


FIG. 3.27 – Cotations TrueChess pour 3.5 millions de parties et 200 000 joueurs. Bien que le modèle probabiliste soit fortement bouclé et comprenne plus de douze millions de variables aléatoires, l'inférence est réalisée en une dizaine de minutes seulement grâce à l'efficacité de la combinaison EP-ordonnanceur semi-automatique. Seuls quelques joueurs célèbres sont représentés.

Nous constatons que notre première étude avait vraisemblablement surestimé la force de Morphy par rapport aux compétiteurs d'aujourd'hui (figure 3.27). Néanmoins, il reste largement meilleur que ses contemporains. Notons aussi l'apparition à cette époque de Louis Eichborn (1812-1882), banquier allemand, à un très haut niveau. Cependant nous pensons que ses cotations sont surévaluées en raison d'un biais des données. En effet, Eichborn était un collègue d'Anderssen, lui-même un des plus grands de cette époque. Les seules parties connues d'Eichborn l'ont opposé à Anderssen et ont été relevées par Eichborn lui-même puis publiées en 1912 (Got12). Ses notes ne concernant que ses parties gagnantes et nulles, le nombre de ses défaites n'est pas connu. De plus, en analysant les parties jouées (Loy02), il semble qu'il s'agisse de parties amicales rapides avec probablement un important handicap pour Anderssen.

Ce nouveau jeu de données permet de compléter l'histoire de Bobby Fischer. Il réapparaît en 1992 pour un match revanche contre Spassky, contre qui il avait déjà gagné en 1972. En 1992, le champion du monde officiel est Kasparov, mais Fischer n'a jamais accepté la perte de son titre lors de son forfait contre Karpov en 75. Arguant que le titre ne peut se perdre que sur le plateau, il déclare que son match contre Spassky décidera du champion du monde. Fischer bat Spassky par 10 victoires, 15 nuls et 5 défaites. De nombreux grands maîtres estiment pourtant que son niveau de jeu a baissé durant ses 20 ans d'inactivité. Nos résultats confirment cette opinion : la courbe de Fischer, promis à un brillant avenir en 75, s'avère être constante, voir légèrement décroissante à cause de ce résultat mitigé face à un Spassky qui finalement, n'a pas beaucoup progressé dans les 20 dernières années. Il est d'ailleurs intéressant de noter la constance de Spassky, qui se maintient au même niveau depuis 50 ans. Spassky s'est progressivement retiré de la compétition à partir des années 80, participant occasionnellement à des tournois par équipes mais ne s'entraînant plus intensivement. Il est plaisant de constater que ses cotations TrueChess reflètent cette stagnation. Notons par ailleurs l'augmentation de l'incertitude sur la cotation de Fischer pendant ses années de retrait.

Finalement nous retrouvons la suprématie de Kasparov ces dernières années, il domine fortement (les écarts-types ne se chevauchant pas) Kramnik depuis 2002 et nous pouvons prédire avec confiance qu'un match les opposant verrait la victoire de Kasparov. Nous constatons aussi l'ascension rapide de Viswanathan Anand (1969-), devenu à 18 ans le premier grand maître international indien. Anand est aussi le premier champion du monde FIDE indien (2000). Selon notre analyse, il est probable que le niveau d'Anand soit supérieur à celui de Kasparov depuis 2004, mais les incertitudes associées rendent cette conclusion statistiquement non significatives.

Lors de l'introduction à ce chapitre, nous avons posé quelques questions difficiles sur l'histoire des échecs. Essayons d'y répondre avec les résultats de TrueChess sur ChessBase[10]

- *Qui est le meilleur joueur mondial à l'heure actuelle ?* Anand et Kasparov se battent pour la première place, avec un léger avantage pour Kasparov en 2006.
- *Qui était le meilleur joueur en 1991 ? Garry Kasparov ou Anatoly Karpov ?* Nous

[10]Rappelons que nous ne donnons que des pistes de réponses relatives à notre modèle et aux données ChessBase utilisées.

pouvons dire que c'est Kasparov, avec une bonne marge de confiance.
- *Emanuel Lasker (1968-1941), second champion du monde officiel, a gardé son titre durant 27 ans. Etait ce justifié ? Dominait-il ses contemporains ?* Oui, entre 1893 et 1900. En dehors de cette période, Steinitz et Capablanca étaient de sérieux concurrents.
- *Qui aurait gagné si Kasparov avait rencontré Paul Morphy, meilleur joueur du XIX^{ème} siècle ?* Kasparov à partir des années 80, aurait gagné.
- *Quelle est l'évolution typique d'un joueur durant sa carrière ? Termine-t-il toujours à son meilleur niveau ?* Non. La plupart des joueurs voient leur niveau baisser quand ils vieillissent.
- *Bobby Fischer était-il au paroxysme de son jeu lorsqu'il renonça aux compétions officielles en 1975 ?* Oui. Quand il rejoua en 1992 contre Spassky, il n'était plus aussi brillant qu'en 1975.
- *Les cotations Elo font apparaître une inflation moyenne des cotations pendant le XX siècle. Cette tendance est elle réelle ou est-ce la conséquence d'un biais de la méthode ?* L'hyper-inflation est certainement due aux faiblesses de Elo et le déboguage est difficile. Néanmoins nous constatons tout de même une augmentation des meilleurs niveaux depuis 15 ans, mais moins spectaculaire que celle de Elo.
- *Kasparov est-il vraiment le premier à avoir franchi la barrière mythique des 2800 points Elo ?* Kasparov fut 2800 Elo en 1992. Notre analyse confirme qu'il est le premier à avoir atteint ce niveau.
- *Et finalement, qui est le meilleur joueur d'échecs de tous les temps ?* Probablement Viswanathan Anand en 2004.

3.6.5 Performances prédictives

Pour valider nos résultats nous avons comparé les performances prédictives de TrueChess et de Elo par validation croisée à 10 groupes, sur le premier jeu de données. Les paramètres utilisée sont ceux maximisant la preuve pour TrueChess et $\{\mu_0 = 2500, K = 10\}$ pour Elo.

TrueChess atteint une taux de prédictions justes de 55.93% et Elo 27.12% (Fig. 3.28), mais cette comparaison n'est pas loyale dans la mesure où Elo ne sait pas prédire des matchs nuls. Nous avons alors soustrait les 52.21% de parties nulles des données et recommencé la validation. TrueChess reste alors plus performant que Elo (61.54% contre 58.4%).

Notons encore qu'il est normal que les performances soient assez éloignées de 100%, car les matchs présents dans la base sont par construction d'issue incertaine. En effet ce sont des matchs enregistrés lors de tournois souvent internationaux, pour lesquels les joueurs sont minutieusement sélectionnés afin d'augmenter le suspens. Il est très rare de voir des joueurs de niveaux très différents s'affronter et plus de la moitié des matchs finissent par un nul. Nous conjecturons que le taux de réussite optimal n'est pas très éloigné de celui de TrueChess.

La table 3.29 présente la matrice de confusion de TrueChess. La première ligne de 0 est due à l'organisation des données, où le joueur J_1 est défini comme le gagnant, ce qui

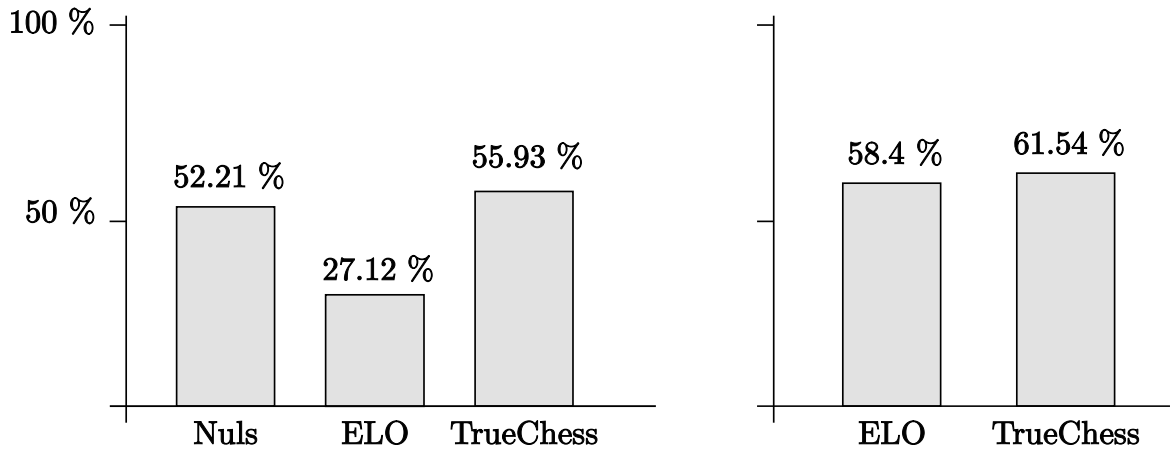


FIG. 3.28 – Performances prédictives mesurées par validation croisée (10 groupes). À gauche, TrueChess surpasse Elo car ce dernier n'est pas capable de prédire les 52.21% de matchs nuls. À droite, les matchs nuls ont été retirés et alors l'avantage de TrueChess est plus modeste.

Vrai \ Prédiction	−1	0	1
−1	0	0	0
0	210	8061	43
1	61	6573	678

FIG. 3.29 – Matrice de confusion de TrueChess en validation croisée.

ne fait pas perdre de généralité à l'étude. Nous constatons que, les matchs étant serrés, TrueChess prévoit majoritairement des nuls. En revanche, quand il prévoit une victoire, il ne se trompe de vainqueur que dans 10% des cas.

Finalement, nous présentons dans la figure 3.30 une analyse plus détaillée des prédictions de TrueChess. À chaque nouvelle prédiction, TrueChess produit trois probabilités, à savoir que J_1 gagne, qu'il perde ou que le match soit nul. La décision finale sera celle maximisant ce triplet, mais il est possible de tracer ces probabilités dans le 2-simplexe. Nous constatons alors que les 15 664 triplets (histogrammes) ne se répartissent pas aléatoirement dans cet espace, mais sur un croissant de lune. Cette forme est le résultat de la paramétrisation gaussienne du modèle et de la construction non aléatoire du jeu de données. Nous constatons que très peu de parties s'achèveront clairement par une victoire.

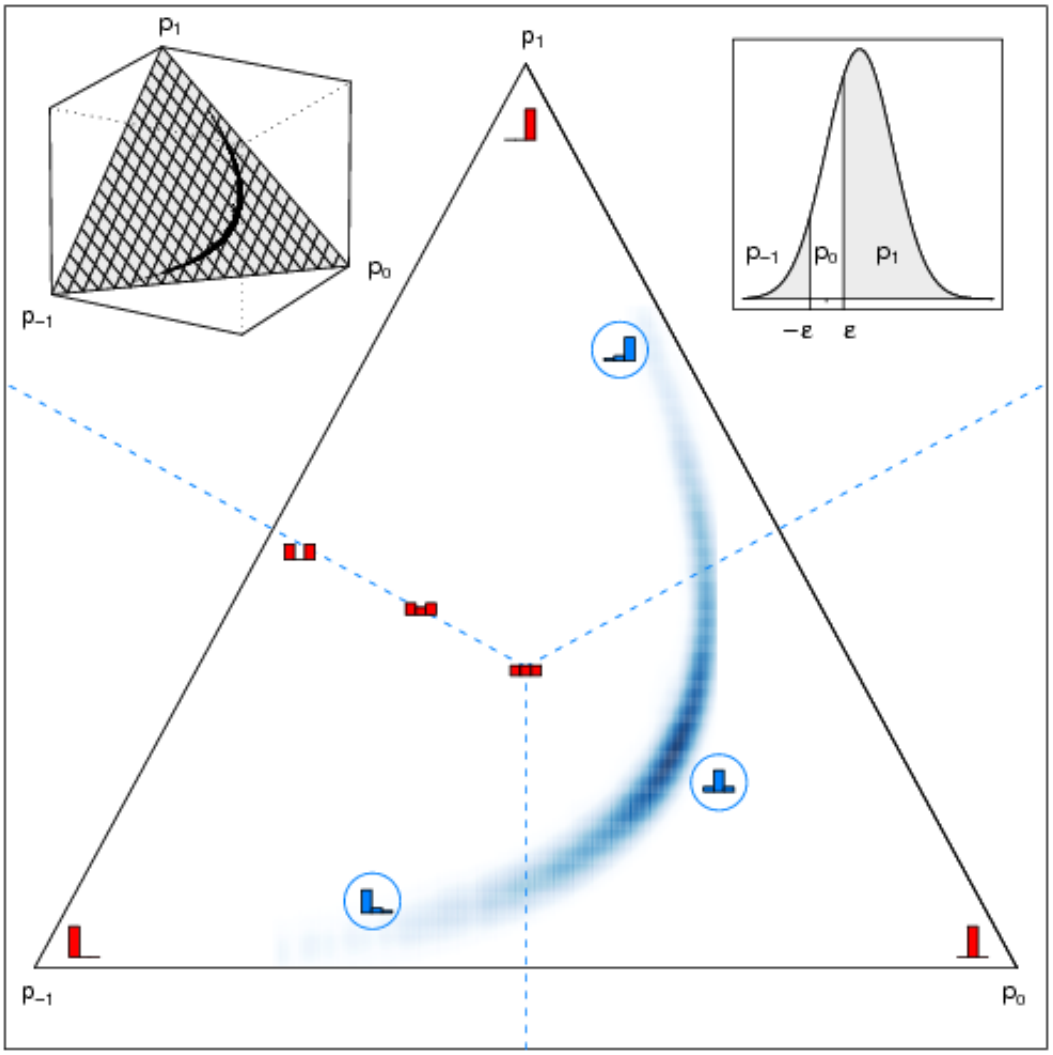


FIG. 3.30 – **Prédictions dans le simplexe**. Pour chacune des 15 664 parties, TrueChess produit p_{-1}, p_0 et p_1, probabilités de l'issue du match ($p_0 := P(\text{nul})$). Comme $p_{-1}+p_0+p_1 = 1$, chacun de ces triplets se représente par un point dans le 2-simplexe (fig. haut,gauche). Les petits histogrammes rouges illustrent des positions remarquables sur ce simplexe. Les 15 664 parties prédites lors de la validation croisée constituent alors le nuage de points bleu. Les trois histogrammes cerclés représentent des décisions faciles, quand une probabilité domine les deux autres. Les lignes pointillées sont les frontières de décision de TrueChess, délimitant trois zones, dans lesquels la prédiction sera constante (-1,0 ou 1). La forme particulière du nuage de points a deux causes distinctes, mais qui interfèrent. D'abord, le modèle génératif lui-même contraint les histogrammes possibles, car il se réduit à découper une gaussienne en trois parties selon $[-\infty, -\epsilon], [-\epsilon, \epsilon], [\epsilon, \infty]$ (fig. haut,droite). Ce partitionnement dépend des paramètres de la gaussienne qui sont fonction des μ,σ des joueurs et de β. Par exemple, comme $\beta \neq 0$, une variance irréductible borne la valeur maximale de p_0. D'autre part le choix du jeu de données intervient, car seuls les joueurs de niveaux similaires s'affrontent et beaucoup de parties sont nulles.

3.6.6 Extensions

Modélisation des marges

Rappelons notre modélisation d'une partie : si d est la différence de performances entre J_i et J_j, nous comparons d_n avec ϵ pour définir l'issue du match. La valeur de ϵ, identique pour tous les joueurs et constante dans le temps, est apprise à partir de la fréquence empirique de partie nulles sur le jeu de données entier (section 3.5.7). Nous proposons un autre modèle, plus complet, qui associe une valeur différente de ϵ pour chaque joueur et pour chaque instant.

En effet, nous constatons dans les données que plus un joueur est fort, plus il aura tendance à faire des matchs nuls. De plus, la fréquence de matchs nuls augmente avec le temps : 30% vers 1800 contre 60% aujourd'hui (53% en moyenne).

Alors, nous ajoutons aux variables cachées les marges individuelles pour chaque match $\{\epsilon_i^n\}$ en définissant que le joueur J_i gagne le match n ($g^n = +1$) si $p_i^n > p_j^n + \epsilon_j^n$, c'est à dire si sa performance dépasse la performance de J_j plus la marge de J_j. La marge représente l'habilité d'un joueur à forcer le match nul contre un joueur plus fort. Le modèle génératif est alors :

$$g^n = \begin{cases} +1 & \text{si } d^n := p_i^n - p_j^n > \epsilon_j^n \\ -1 & \text{si } d^n < -\epsilon_i^n \\ 0 & \text{si } -\epsilon_i^n < d^n < \epsilon_j^n. \end{cases} \tag{3.63}$$

Nous modélisons les marges comme des variables gaussiennes et leur assignons une dynamique brownienne similaire à celle des cotations.

$$\begin{aligned} \epsilon_i^0 &\sim \mathcal{N}\left(\epsilon_i^0; \nu_0, \zeta_0\right) \\ \epsilon_i^{t+1} &\sim \mathcal{N}\left(\epsilon_i^{t+1}; \epsilon_i^t, \zeta^2 \Delta t^2\right) \end{aligned} \tag{3.64}$$

Pour assurer la positivité des marges, nous introduisons des facteurs approximés $\delta_{>0}\left(\epsilon_i^t\right)$ du même type que les facteurs de résultats.

Nous avons de nouveaux paramètres (ν_0, ζ_0, ζ) et chaque joueur est caractérisé par quatre nombres à chaque instant, la moyenne et la variance de sa cotation ainsi que la moyenne et la variance de sa marge. Comme les nouveaux facteurs introduits dans le graphe sont similaires aux anciens, les équations de messages sont les mêmes et l'agenda est similaire.

Le graphe complet avec les données ChessBase comprend alors 18 500 000 de variables et 27 600 000 facteurs, la convergence prend 20 minutes et nécessite 11Go de mémoire. La log-preuve avec ce modèle est meilleur qu'avec le modèle à marge fixe ($\log Z$ = -3 661 813 contre -3 953 997), démontrant que ce modèle a plus de facilité à représenter les données.

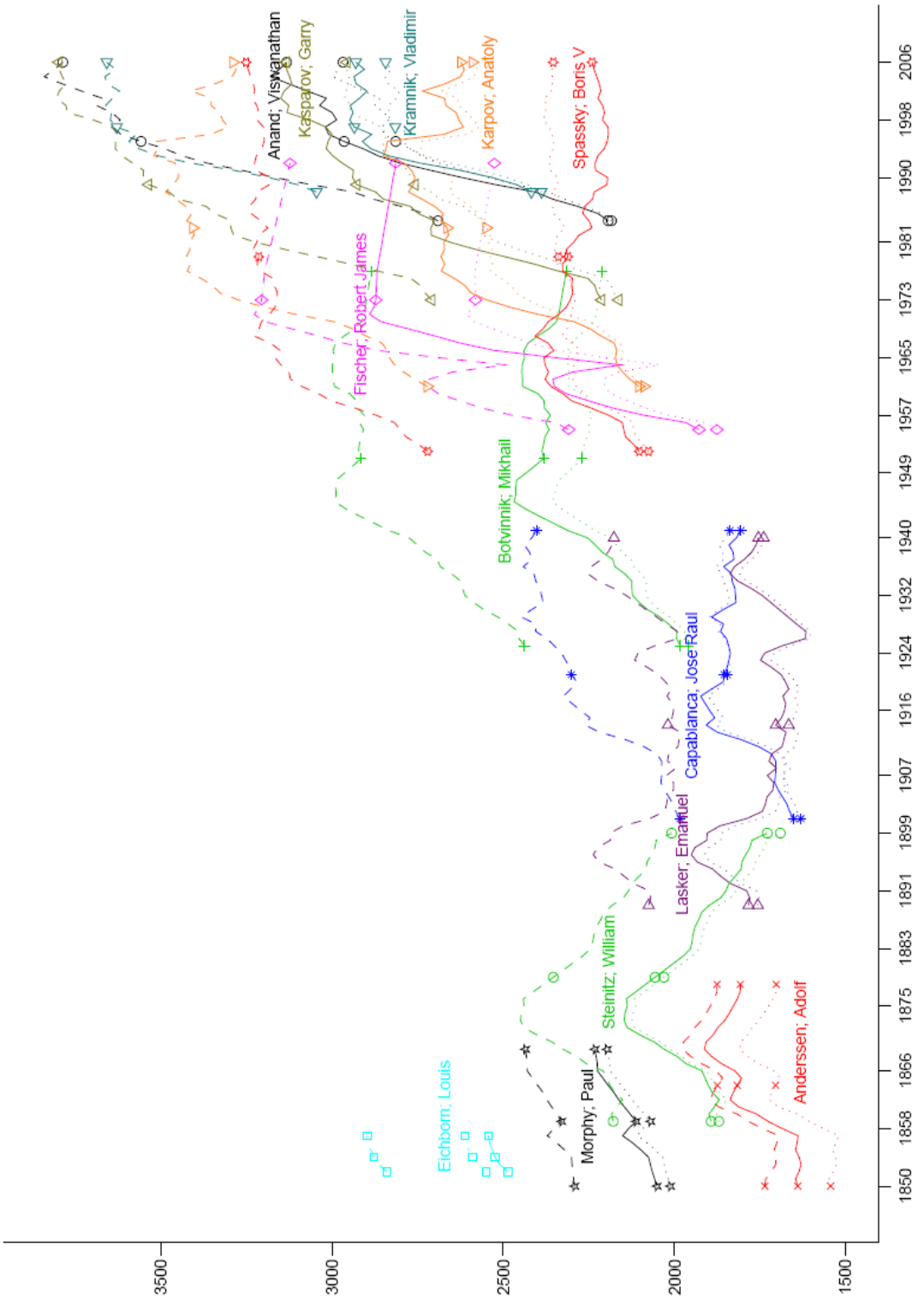


FIG. 3.31 – Résultats avec ChessBase et modélisation des marges. Les courbes pleines sont les moyennes des cotations μ_i^t, les traits interrompus sont $\mu_i^t + \epsilon_i^t$ et les pointillées sont les anciennes μ_i^t, inférées avec le modèle à marge fixe. Les incertitudes ne sont pas représentées pour plus de clarté.

Les résultats (figure 3.31) montrent que les joueurs possèdent des habiletés différentes pour forcer le nul, Spassky (courbe rouge) étant très fort dans ce domaine. Il a même accru cette faculté à partir de 1980, gagnant moins de matchs, mais obtenant plus de nuls. Sa cotation μ a donc baissé dans le nouveau modèle, alors qu'elle était constante dans l'ancien.

La valeur de la marge permet ainsi de mieux comprendre le style des joueurs. Par exemple celle de Fischer est la seule à être décroissante avec le temps et effectivement il est connu pour la marge avec laquelle il remportait ses matchs. Nous remarquons aussi que les marges sont beaucoup plus grandes en 2000 qu'en 1850. À cette époque il n'était pas beaucoup plus dur de battre un adversaire que d'obtenir un nul. Aujourd'hui, ce n'est plus le cas : gagner demande un niveau bien plus grand que celui assurant un nul.

TrueChess en ligne

TrueChess s'attaque au problème de trouver des cotations historiques : l'inférence est effectuée avec un jeu de données complet, en une seule fois. En revanche, Elo est une méthode incrémentale, qui permet une mise à jour facile et rapide des cotations d'un grand nombre de joueurs. Peut-on trouver un compromis entre la précision de TrueChess et la praticité de Elo ?

Il est possible de refaire une inférence TrueChess après chaque acquisition de résultat et d'écraser les anciennes cotations inférées. La fédération américaine enregistre environ 500 000 parties par an concernant 80 000 licenciés. Ce volume de données n'est pas très grand et il est parfaitement envisageable de recalculer des cotations TrueChess une ou deux fois par jour. D'autant plus que leur temps de calcul peut être réduit de deux façons.

Premièrement, il est possible de réutiliser les résultats de l'inférence précédente pour initialiser le graphe de facteur augmenté. Partir du minimum local précédent au lieu de l'uniforme économise en général les deux premières passes de l'agenda.

Peut être que TrueChess peut traiter toutes les parties d'échecs jouées au monde et mettre à jour les cotations de façon quotidienne. Si jamais ces données ne permettent pas l'inférence présentée, nous proposons une inférence plus rapide en modifiant l'agenda.

En effet, nous remarquons qu'une nouvelle information sur les cotations au temps t a une influence décroissante dans le temps. Un match en 2006 aura plus d'influence sur une cotation de 2005 que sur une cotation de 1850. Ainsi nous pouvons envisager un agenda différent, qui serait fondé sur une notion de fenêtre temporelle. La taille de la fenêtre déterminera un compromis entre précision et temps de calcul. L'agenda est identique à celui présenté précédemment, sauf qu'il ne s'applique qu'entre t et $t - \tau$: les cotations plus anciennes que τ sont considérées comme définitives et ne seront plus changées. Cet algorithme, *windows based EP* (QM03) permet une extension *online* de TrueChess. Cette approche TrueChess+fenêtre a aussi l'avantage de fixer les cotations. Il est possible que les utilisateurs du système n'apprécient pas vraiment de voir leurs cotations passées varier alors qu'il ne jouent plus, ce qui est pourtant meilleur en terme d'estimation statistique. Leur garantir que leurs cotations datant de plus de soient définitives aiderait peut-être à faire accepter le nouveau système.

3.7 Conclusion

3.7.1 Contributions

Nous avons proposé un modèle bayésien génératif d'un problème non trivial : la cotation d'objets à partir de comparaisons non transitives et non déterministes entre paires.

Alors que ce problème était auparavant résolu par des méthodes de maximum de vraisemblance, notre approche bayésienne permet de rationaliser et d'améliorer ces procédures. De plus, elle permet d'incorporer rigoureusement une contrainte supplémentaire : l'évolution temporelle de la qualité des objets.

Nous avons alors introduit toutes les briques nécessaires à l'inférence des cotations marginales. En particulier les ordonnanceurs automatique et semi automatique constituent des éléments nouveaux essentiels pour l'efficacité de la méthode.

Cette méthode a été appliquée à un jeu de données de plus de trois millions et demi de parties d'échecs, donnant lieu à une modèle graphique bouclé de près de vingt millions de variables aléatoires. Grâce au choix de l'agenda et au principe itératif de l'algorithme, la convergence vers une solution approchée est réalisée en une vingtaine de minutes. Ceci démontre l'efficacité des méthodes déterministes lorsque le graphe est convenablement structuré.

De plus, il est souvent reproché aux méthodes bayésiennes leur lenteur par rapport à des méthodes classiques. TrueChess est indéniablement plus lent qu'Elo, mais le succès de cette méthode montre qu'avec des approximations adéquates, nous pouvons attaquer de grands jeux de données, sans renoncer aux avantages d'une approche théoriquement fondée.

Le modèle TrueChess appliqué aux données ChessBase est probablement un des plus grands modèles graphiques bayésiens non triviaux jamais proposés.

Le classement obtenu apporte un nouveau regard sur l'histoire des échecs. Nos résultats sont globalement cohérents, par exemple les meilleurs joueurs trouvés ont étés sacrés champions du monde et sans surprise les personnes se détachant du lot sont Anand, Kramnik, Kasparov, Karpov et Fischer. Comme elles ne se basent que sur les résultats des matchs, nos cotations ne sont pas suffisantes pour étudier l'histoire des échecs dans son intégralité. Mais elles constituent certainement un outil quantitatif intéressant pour compléter des analyses plus complètes, se basant par exemple sur les parties jouées, sur la psychologie, l'histoire personnelle des joueurs et sur le contexte historique des tournois.

Nous remarquons finalement que le modèle TrueChess est très général et peut donc être appliqué à tous les problèmes de comparaison de paires. Nous pensons en particulier aux sports pour lesquels Elo est utilisé, comme le tennis et le classement de football (annexe A.3), mais aussi aux jeux en ligne et aux jeux vidéo.

3.7.2 Perspectives

Parmi les méthodes de cotation, TrueChess est une méthode claire, rigoureuse et statistiquement bien justifiée. Néanmoins, les cotations produites ne peuvent pas être considérées

comme absolues et définitives pour trois raisons principales.

D'abord elles sont relatives à un modèle, représentation par définition incomplète de la réalité. Notre modèle est perfectible sur plusieurs aspects. Par exemple le modèle d'évolution dynamique pourrait se passer de l'hypothèse de Markov d'ordre 1 pour permettre ainsi l'incorporation d'un *a priori* sur l'allure typique (croissante puis décroissante) des courbes de niveau. Techniquement, la piste d'une modélisation par des processus gaussiens (RW05) semble prometteuse. En effet notre modèle dynamique brownien est un processus gaussien particulier.

Nous pourrions aussi modéliser le fait de jouer avec les noirs ou les blancs et produire ainsi deux niveaux pour chaque joueur. Ceci améliorerait certainement le pouvoir prédictif. Nous remarquons aussi que, dans le modèle actuel, la cotation *a priori* d'un joueur inconnu ne dépend pas de l'époque. Relaxer cette hypothèse permettrait certainement d'améliorer la probabilité des données. Un dernier point : nous utilisons une vraisemblance gaussienne (*Thurstone Case V*) alors que les travaux précédents se basent sur une fonction logistique (*Bradley-Terry*) en arguant d'une meilleure adéquation aux données (et d'un calcul plus aisé). Nous avons envisagé l'emploi de la méthode *gaussian quadrature EP* (ZH05) pour utiliser la vraisemblance logistique. Cependant, même en écartant le surcoût calculatoire, nous n'aurions pas vu d'amélioration significative. En effet, la différence entre les modèles gaussien et logistique est le poids relatif des queues de leurs distributions. Or nous avons vu que la plupart des matchs étaient très serrés et donc que la distribution est presque centrée, minimisant ainsi l'influence de la queue sur la vraisemblance d'une partie.

La deuxième raison mettant les cotations en perspective est l'influence du choix des données. Nous avons vu que leur sélection jouait une rôle important et devait être réalisée avec soin au prix d'un travail historique conséquent.

La dernière interrogation posée par ce travail et c'est un point important, est la pertinence même de la notion de comparaison historique. Les cotations obtenues sont elles consistantes à travers le temps ? Cette cohérence est-elle seulement envisageable ?

Prenons l'exemple simple de deux joueurs d'exactement même niveau à chaque instant. Toutes leurs parties étant nulles, leurs cotations seront toujours égales à μ_0 car rien ne permet de les distinguer. Imaginons maintenant qu'il apprennent *simultanément* une meilleure façon de jouer. Après cela, ils continueront à faire des nuls et donc leurs cotations resteront μ_0. Ainsi une progression globale et subite du vrai niveau de tous les joueurs est indétectable, quelque soit la méthode employée.

Depuis deux cents ans, la théorie des échecs a fait d'énorme progrès, l'apparition des ordinateurs a permis aux joueurs de mieux s'entraîner et les bibliothèques d'ouvertures se sont enrichies. Si Morphy était téléporté en 2006, il est bien possible qu'il perde même face à un joueur moyen (de plus ce dernier pourrait avoir étudié les parties et le style du voyageur temporel).

Cette remarque sur la cohérence temporelle est aussi valide pour la cohérence spatiale. À un même instant, s'il y a deux sous groupes de joueurs séparés, les inter comparaisons n'ont pas de justification. Le problème sous-jacent est en fait la force des liens entre les joueurs. Deux joueurs sont fortement liés s'ils se sont souvent rencontrés, mais ils le sont aussi s'ils ont eu de nombreux partenaires en commun.

Ainsi l'objection ne signifie pas que toute comparaison est impossible si le graphe des rencontres est suffisamment dense. La cohérence locale des cotations étant assurée par TrueChess, nous pouvons supposer qu'elle se transmet de proche en proche pour assurer une cohérence globale la plus grande possible. Bien entendu, plus les joueurs sont éloignés dans le graphe des rencontres, plus une comparaison directe est risquée.

D'autre part nous pouvons supposer que des cas d'augmentation subite et générale des vrais niveaux n'arrivent pas en pratique, l'information et les techniques ayant en réalité une tendance à diffuser progressivement parmi les joueurs.

Nous voyons deux principales pistes pour approcher plus en détails le problème de cohérence globale. D'abord nous pourrions ajouter une variable latente à chaque année pour initialiser la cotation des nouveaux entrants. Cette variable permettrait de calibrer les cotations en modélisant les progrès de la science échiquéenne. D'autre part, une hypothèse de départ de ce travail à été de ne pas considérer les corrélations entre cotation, en traitant seulement les gaussiennes marginales et non pas la matrice de covariance complète. Une telle étude devra astucieusement gérer la taille de cette matrice.

Chapitre 4

Variables cachées introduites : Netflix

4.1 Introduction

4.1.1 Variables cachées introduites

Dans le chapitre précédent, nous avons étudié en détails un problème d'apprentissage en présence de variables cachées explicites, dont nous cherchions à approximer les marginales *a posteriori*. Dans ce chapitre, nous nous intéressons à un problème pour lequel nous avons introduit des variables cachées, non pas pour trouver leurs *a posteriori*, mais dans le but de faire de la prédiction sur de futures données. Nous proposons un modèle génératif de regroupement (*clustering*) parallèle de deux types d'entités différents afin de résoudre un problème de filtrage collaboratif. L'inférence exacte pour ce modèle n'étant pas réalisable en raison de la taille du jeu de données, nous proposons d'employer une méthode de type Espérance Maximisation (EM), qui approxime un *a posteriori* sur des paramètres du modèle par un maximum local de vraisemblance. Nous proposons deux modèles différents, l'un d'entre eux nécessitant de plus une approximation variationnelle.

4.2 Le problème du filtrage collaboratif

4.2.1 Motivations

Le filtrage collaboratif est une méthode permettant de prédire les intérêts d'un utilisateur à partir de l'ensemble de ses préférences passées et de celles d'autres utilisateurs. Si nous supposons que les préférences ne varient pas trop avec le temps, il est possible d'exploiter des similitudes de préférences : par exemple si les utilisateurs u_1 et u_2 ont par le passé aimé les même objets $\{m_i\}$ et que u_1 aime le nouvel objet m_{i+1}, alors il est probable que u_2 l'aimera aussi ; et nous pouvons lui recommander.

Le premier filtre collaboratif a été développé par Xerox (GONT92) pour un moteur de recherche de documents fondé sur des commentaires d'autres utilisateurs. Depuis, cette méthode a été appliquée dans de nombreux domaines, en particulier par des sites inter-

net commerciaux, comme Amazon et LibraryThing pour les livres ; ILike, MusicMatch et Last.fm pour la musique et GroupeLens et Netlix pour les films.

4.2.2 Données Netflix

Début octobre 2006, la société Netflix, qui loue des DVD sur internet en Amérique du Nord, a lancé une compétition de filtrage collaboratif, en diffusant une partie de ses données et en offrant un million de dollar à la première équipe qui produira des prédictions dont la justesse dépassera de 10% ses propres prédictions. La compétition est ouverte pour cinq années et tous les ans un prix intermédiaire est décerné à la meilleure équipe. À ce jour (septembre 2007) les meilleurs algorithmes ont atteint 8.43% d'amélioration, mais il est toujours incertain que la barre des 10% soit franchissable.

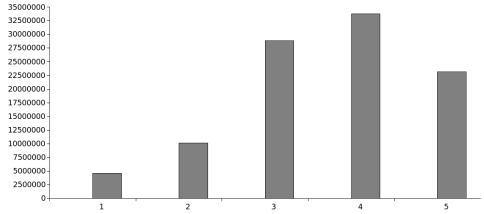
Les données diffusées sont un jeu de plus de 100 millions de notes, chacune étant un quadruplet (*user*, *movie*, *rating*, *date*). La taille de ce jeu de données dépasse de deux ordres de grandeur celle des jeux précédemment disponibles au public. Chaque note (*rating*) est une valeur entière entre 1 et 5 donnée par un utilisateur à un film. Les titres des films sont aussi fournis, mais la plupart des équipes et notre méthode, ne les prend pas en compte et approchent le problème en ne considérant que les identificateurs des films. Par exemple, nous n'exploitons pas l'information que deux films soient réalisés par la même personne, ni qu'ils présentent les mêmes acteurs. De plus, nous n'utilisons pas les dates. Ces informations externes devraient bien entendu être incorporées au système afin de le perfectionner, mais nous ne nous sommes intéressés qu'au problème de filtrage abstrait. Ces 100 millions de notes concernent plus de 480 000 utilisateurs et 18 000 films. Chaque utilisateur a noté en moyenne 200 films, mais certains en ont vu très peu, certains beaucoup (jusqu'à 17 000). La figure 4.1 présente certains aspects des données. Nous remarquons que la distribution des notes est décalée vers le haut (moyenne 3.8), indiquant que les utilisateurs ont tendance à noter des films qu'ils ont aimés. Nous retrouvons aussi un effet de "longue queue" : les notes sont concentrées sur un petit nombre de films à succès, alors que de très nombreux films ne reçoivent que très peu de notes. Le même effet est présent pour les utilisateurs. Nous voyons aussi que les personnes donnant en général des notes médiocres sont plus variables dans leurs évaluations que celle donnant souvent de bonnes notes.

Pour comparer les différentes méthodes, Netflix propose un jeu de 2,8 millions de couples (*user*,*movie*) pour lesquels il faut prévoir les *ratings* correspondants. L'évaluation est calculée par la racine de l'erreur quadratique moyenne (RMSE) sur ce jeu de test :

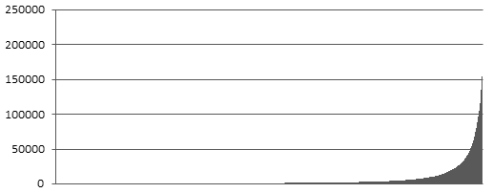
$$\mathrm{RMSE} := \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(r_i^{\mathrm{predicted}} - r_i^{\mathrm{true}}\right)^2}. \tag{4.1}$$

4.3 État de l'art

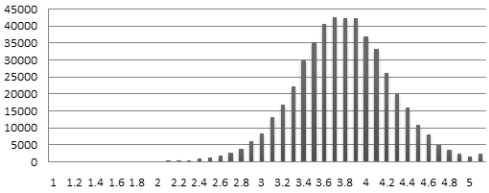
De nombreuses méthodes d'apprentissage automatique ont étés appliquées au problème du filtrage collaboratif. Nous pouvons les classer en deux catégories : les approches fondées



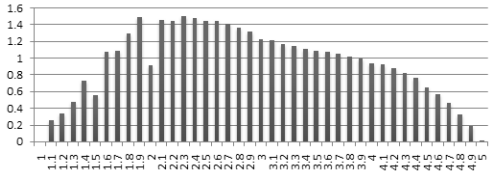
(a) Histogramme des notes.



(b) Nombre de données par films.



(c) Nombres d'utilisateurs dont la note moyenne est en abscisse.



(d) Variance des notes d'utilisateurs dont la note moyenne est en abscisse.

FIG. 4.1 – Données Netflix : $N_r = 100\,480\,507$ notes, $N_u = 480\,189$ utilisateurs et $N_m = 17\,770$ films.

sur une mémoire et celles fondées sur un modèle.

4.3.1 Approches fondées sur une mémoire

Ces approches considèrent le problème comme la complétion de valeurs manquantes d'une grande matrice. Considérons les données comme une matrice R de taille $N_u \times N_m$ avec les utilisateurs u_k en ligne et les films m_l en colonnes. Seules les cases (k, l) correspondant à une donnée auront une valeur connue $r_{k,l} \in [1, 5]$. La prédiction se résume à trouver les valeurs des cases vides de façon "cohérente" aux cases remplies. La matrice R est creuse à 98.8%.

Pour cela, les approches les plus anciennes brisent la symétrie entre utilisateurs et films (CMB07), donnant deux familles de méthodes : celles fondées sur les utilisateurs et celles fondées sur les films.

Considérons les approches fondées sur les utilisateurs initiées par GroupLens (RIS+94). La prédiction de note $r_{k,l}$ donnée par l'utilisateur u_k au film m_l est la moyenne des notes $\overline{r_k}$ données par u_k à tous les films qu'il a vu $\{m_j\}$, corrigée par une somme pondérée des notes mises par les autres utilisateurs $\{u_i\}$ à m_l. Cette somme est pondérée par une mesure de similarité entre utilisateurs $\mathrm{sim}(u_k, u_i)$:

$$r_{k,l} := \overline{r_k} + \frac{\sum_{u_i} \mathrm{sim}(u_k, u_i)\ (r_{i,l} - \overline{r_i})}{\sum_{u_i} |\mathrm{sim}(u_k, u_i)|}. \tag{4.2}$$

Cette procédure étant heuristique, il faut choisir une mesure de similarité qui devra être calculée et stockée pour tous les couples d'utilisateurs, soit $O(N_u^2)$ fois, justifiant l'appellation "fondée mémoire". Comme N_u est trop grand, il faut se restreindre à un ensemble

de K utilisateurs voisins, ce qui donne une complexité de $O(N_u^2 \times N_m \times K)$ pour l'apprentissage, $O(K)$ pour une prédiction et une complexité mémoire de $O(N_u \times K)$. Le choix de la fonction de similarité entre utilisateurs est crucial et différentes idées ont été proposées (SM95 ; WMG+06), comme la corrélation de leur notes sur les films qu'ils ont tous deux vus $\{m_n\}$, ou le cosinus de leur vecteur de notes :

$$\text{sim}(u_k, u_i) := \frac{\sum_{m_n} r_{k,n}\, r_{i,n}}{\sqrt{\sum_{m_n} r_{k,n}^2 \; \sum_{m_n} r_{i,n}^2}}. \tag{4.3}$$

Bien que complètement symétriques, les approches fondées sur les films (item-based) sont plus récentes et les chercheurs ont du développer des mesures de similarités différentes (Kar01), ce qui souligne la difficulté de ce choix heuristique.

Récemment, une autre façon de compléter R a été proposée et a donné de très bons résultats prédictifs. L'idée est que pour déterminer une note, nous n'avons pas besoin de connaître exactement quel est le film et quel est l'utilisateur, mais seulement certaines de leurs caractéristiques, par exemple savoir que c'est un film d'horreur et que l'utilisateur est un enfant permet de prédire une mauvaise note. Formellement, cela veut dire que R est une matrice compressible, son rang n'est pas aussi grand que $\text{Min}(N_u, N_m)$. Si R était complète, il suffirait de réaliser sa décomposition en valeurs singulières (SVD) et de ne garder que les K plus grande valeurs singulières afin d'obtenir une approximation R' de rang K de R. Les vecteurs des matrices unitaires produites donnent alors une caractérisation numérique de chaque film (et de chaque utilisateur) à K valeurs. Pour prédire une note, il suffit de réaliser le produit scalaire du vecteur de caractéristiques du film et de celui des caractéristiques de l'utilisateur.

Initialement appliquée dans le domaine du traitement des langues naturelles sous le nom de *latent semantic analysis* (LSA) (DDF+99), elle a été adaptée à des matrices R creuses et de très grande taille (VM05). SimonFunk (Web06) a étendu et appliqué cette méthode au problème de Netflix et propose une SVD régularisée en minimisant l'erreur quadratique de reconstruction de R, par une descente de gradient avec arrêt prématuré. En publiant ses algorithmes, il a engendré une multitude de variantes (Pat07). En effet, comme R est creuse, en plus des problèmes de gestion de mémoire, la SVD nécessite une régularisation importante pour ne pas sur-apprendre.

4.3.2 Approches fondées sur un modèle

Au lieu de tenter de reconstruire R, il est possible de définir un modèle paramétrique de la génération des données et d'estimer ces paramètres à partir de l'ensemble d'apprentissage (BHK98). De telles méthodes reposent souvent sur un groupement des utilisateurs en types d'utilisateurs, par des techniques de clustering dur comme le *K-Mean* (KS04), qui nécessitent tout de même une définition de distance entre utilisateurs. Des méthodes probabiliste plus modernes ont aussi été employées comme l'extension *probabilistic LSA* de LSA, qui a l'avantage de ne pas contraindre chaque utilisateur à un seul type (*soft clustering*).

Peu de travaux appliquent le clustering à la fois aux utilisateurs et aux films. Par exemple, le modèle de mixture de multinomiales de Marlin (Mar04) ne cherche à grouper que les utilisateurs entre eux, alors que dans sa formulation, rien n'empêche cette extension. Dans son modèle à classe cachées (HP99), Hofmann propose un double clustering, mais avec des notes binaires (+-1). Notre approche reprend les avantages de la formulation générative de Marlin, avec la double clusterisation de Hofmann, en l'appliquant à des mixtures de multinomiales à 5 valeurs.

4.4 Netflix : Modèles

Nous présentons ici une approche bayésienne pour le filtrage collaboratif, appliqué aux données Netflix. Nous proposons deux modèles de clustering différents. Tous deux établissent en parallèle des clusters d'utilisateurs et des clusters de films, mais ces deux modèles donnent une sémantique différente aux clusters. Dans les deux cas la méthode de prédiction est identique et l'inférence repose sur l'algorithme EM : les variables d'associations à des clusters étant les variables cachées (phase E) et les paramètres arg-maximisés sont les profils de notation associées à chaque couple (cluster d'utilisateur-cluster de film) (phase M). De part sa structure, un des modèles nécessite de plus une approximation variationnelle de type *Mean-Field* dans sa phase E. Nous décrivons ces modèles, les calculs d'inférence associés et présentons leurs résultats.

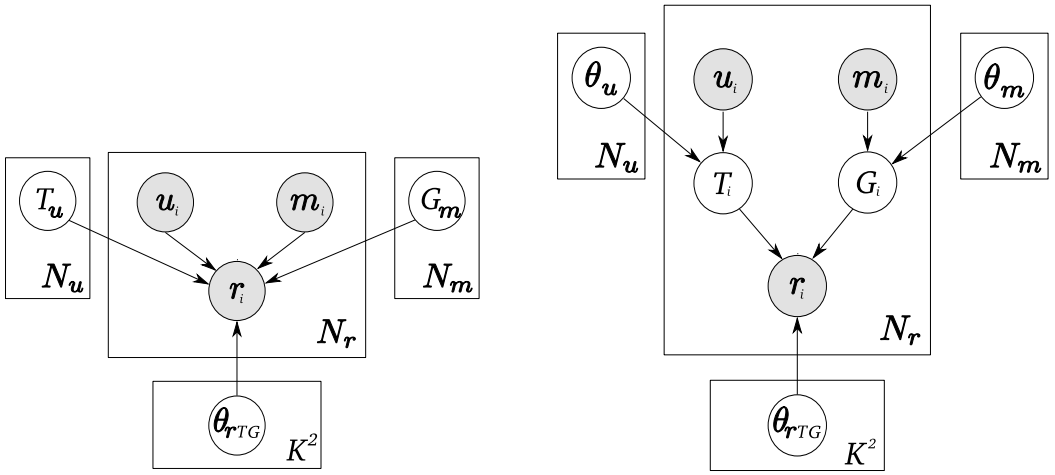
Pour clarifier les notations, nous indiçons les N_r notes du jeu d'apprentissage par i. Ainsi la $i^{\text{ième}}$ note du jeu de données est notée r_i. Elle a été donnée par l'utilisateur u_i au film m_i. Pour parcourir l'ensemble des N_u utilisateurs, nous utilisons l'indice u et respectivement l'indice m pour parcourir l'ensemble des N_m films.

4.4.1 Modèle A : un seul type par utilisateur

Dans ce modèle, nous groupons les utilisateurs en K types et les films en K genres. Pour chaque utilisateur u, nous introduisons une variable aléatoire T_u indiquant son *type*. Par exemple si l'utilisateur numéro 44 est du type $T = 88$, nous notons $T_{44} = 88$. Pour chaque utilisateur u, sa variable de type T_u est une variable cachée. Il y a donc autant de variables de type que d'utilisateurs, soit N_u. Le but de l'inférence dans ce modèle A est d'abord de trouver l'*a posteriori* sur tous les T_u (et sur tous les *genres* de films G_m). Une fois ces $N_u + N_m$ *a posteriori* calculés, ils seront utilisés pour marginaliser les variables de type et de genre afin de faire des prédictions.

Il est important de souligner que, dans ce modèle un utilisateur n'a qu'un seul type, mais comme nous ne savons pas lequel avec précision, nous devons considérer une distributions sur tous les types, et ce pour chacun des utilisateurs (et respectivement pour les films et les genres).

Pour bien expliquer ce modèle A, considérons que nous en ayons fixé tous les paramètres et que nous voulions générer des données. Pour générer une donnée r_i, nous choisissons uniformément un utilisateur, disons $u = 44$ et un film $m = 55$. Comme nous connaissons



(a) Modèle A comportant une seule variable d'association T_u par utilisateur u, signifiant que u a un seul type. Nous cherchons l'*a posteriori* sur tous les T_u et les G_m pour ensuite faire des prédictions.

(b) Modèle B de mixture, avec autant de variables d'associations que de données. Un utilisateur a une degré d'appartenance à chaque type. Nous cherchons la fonction d'appartenance la plus probable.

FIG. 4.2 – Deux modèles génératifs de clustering pour le filtrage collaboratif

le numéro de l'utilisateur, nous en connaissons aussi son type, disons $T_u = T_{44} = 88$. De même $G_m = G_{55} = 99$. Une fois le type et le genre connu, la note ne dépend plus ni du numéro de l'utilisateur, ni du numéro du film. Cette note r_i qui prend sa valeur entre 1 et 5 est tirée selon une distribution multinomiale dont les paramètres sont notés $\{\theta_{r,T,G}\}$. Par exemple $\theta_{r=3,T=88,G=99} = p(r = 3|T = 88, G = 99)$ est la probabilité que les utilisateurs du type $T = 88$ mette une note valant 3 aux films du genre $G = 99$.

Pour créer un jeu complet de données nous devrions répéter ce processus N_r fois pour générer les N_r données $\{r_i\}$. Imaginons que lors de ces répétitions l'utilisateur $u = 44$ soit, par hasard, tiré une seconde fois. Alors, le type T sera à nouveau $T_{44} = 88$: les distributions $p(T|u)$ sont des Diracs piqués en T_u. Si nous insistons sur ce point c'est que ce sera différent dans le modèle B. Les $5 \times K \times K$ valeurs $\theta_{r,T,G}$ sont les seuls paramètres du modèle A. Ces paramètres peuvent ainsi être stockés dans une matrice à trois dimensions (type, genre et valeur de la note) telle que $p(r \mid u, m, T_u, G_m) = \theta_{r,T_u,G_m}$.

La figure 4.2(a) représente la densité jointe en considérant que nous étudions une jeu de N_r données.

Une fois le modèle instancié avec les N_r données, nous obtenons théoriquement un graphe comprenant $(3N_r + N_u + N_m + K^2) \approx 300$ millions de variables aléatoires.

La jointe sur ces variables est :

$$p(\{u_i\},\{m_i\},\{r_i\},\{T_u\},\{G_m\},\{\theta_{r,T,G}\}) \\ = \prod_u p(T_u) \prod_m p(G_m) \prod_i p(u_i, m_i) \prod_{r,T,G} p(\theta_{r,T,G}) \prod_i p(r_i \mid u_i, m_i, T_{u_i}, G_{m_i}, \{\theta_{r,T,G}\}). \tag{4.4}$$

Avec ce modèle, les données observées sont les N_r triplets $\{u_i, m_i, r_i\}$, les variables cachées sont les N_u variables $\{T_u\}$ et les N_m variables $\{G_m\}$ et les paramètres sont les $\Theta := \{\theta_{r,T,G}\}$. Comme nous considérons que tous les *a priori* sont uniforme, la jointe est proportionnelle à la vraisemblance :

$$p(\{u_i\},\{m_i\},\{r_i\},\{T_u\},\{G_m\},\{\theta_{r,T,G}\}) \propto \prod_i \theta_{r_i,T_{u_i},G_{m_i}}. \tag{4.5}$$

L'algorithme EM s'attachera à trouver les paramètres Θ maximisant cette vraisemblance, en affinant successivement une approximation de l'*a posteriori* sur les variables cachées (phase E) et en optimisant les paramètres sous cet *a posteriori* (phase M). Comme à chaque fois qu'un même utilisateur u revient dans les données il est associé à la même variable T_u, de nombreuses boucles sont présentes dans le graphes de dépendances, bien que cela ne soit pas explicite dans la figure 4.2(a) (ambiguïté du formalisme graphique des assiettes). Ces nombreuses boucles rendront nécessaire une approximation variationnelle de la phase E.

4.4.2 Modèle B : modèle de mixture

Dans ce modèle nous considérons que chaque utilisateur est représenté par une mixture de différents types. Par exemple, un utilisateur peut être à 25% du type 1 et à 75% du type 3. D'un point de vue génératif, à chaque apparition du même utilisateur, un nouveau type est tiré selon une distribution $p(T|u,\Theta)$ spécifique à u. Nous voulons par là donner plus de flexibilité au modèle : pour chacune de ses notes, un utilisateur peut avoir agit selon le profil du type des enfants, puis pour une autre note selon le profil du type des filles. Comme il a plus de pouvoir d'expression, ce modèle demandera moins de clusters (K plus petit) que le modèle A pour représenter les mêmes données.

De plus, comme le type dépend de la donnée et non plus uniquement de l'utilisateur, la variable T entre dans l'assiette principale (fig. 4.2(b)), les boucles disparaissent et nous n'avons plus besoin de l'approximation variationnelle. La caractéristique d'un utilisateur n'est plus une variable de type dont nous chercherions l'*a posteriori* comme dans le modèle A. Dans le modèle B, c'est une distribution sur les types $p(T|u)$ qui nous intéresse. Cependant, nous ne chercherons pas l'*a posteriori* sur cette distribution (pas de niveau probabiliste supplémentaire), mais nous calculerons la distribution de maximum *a posteriori*.

Dans ce modèle, les variables observées sont les N_r triplets $\{u_i, m_i, r_i\}$, les variables cachées sont les N_r couples $\{T_i, G_i\}$. Les paramètres Θ sont les K^2 multinomiales $\theta_{r,T,G}$ et les distributions sur les types pour chaque utilisateur $p(T|u,\Theta) = \theta_u$ (et symétriquement

les θ_m). EM produisant les paramètres de maximum de vraisemblance, nous aurons bien les distributions θ_u et θ_m les plus probables. Avec des *a priori* uniformes sur $\{u_i, m_i\}$ et Θ, la jointe est :

$$p(\{u_i\},\{m_i\},\{r_i\},\{T_i\},\{G_i\},\Theta) = p(\Theta)\prod_i p(T_i \,|\, u_i,\Theta)\, p(G_i \,|\, m_i,\Theta)\, p(r_i \,|\, T_i, G_i,\Theta). \quad (4.6)$$

L'*a priori* $p(\Theta)$ est choisi uniforme.

4.5 NetFlix : Apprentissage

4.5.1 Modèle A

La procédure standard pour trouver des paramètres de maximum de vraisemblance avec variables cachées est l'algorithme EM (DLR77). EM alterne une phase E durant laquelle un *a posteriori* sur les variables cachées est calculé avec l'état courant des paramètres et une phase M durant laquelle les paramètres sont mis à jour afin de maximiser l'espérance du logarithme de leur vraisemblance sous cet *a posteriori*. Cette procédure itérative garantit la non décroissance de la vraisemblance et converge donc vers un maximum local de vraisemblance des paramètres.

Phase E

En notant $\{\Delta_i\}$ l'ensemble des triplets de données, nous devons calculer

$$p(\{T_u\},\{G_m\} \mid \{\Delta_i\},\Theta). \quad (4.7)$$

Nous utilisons pour cela une approximation de type *Mean Field* :

$$q(\{T_u\},\{G_m\}) = \prod_u q(T_u)\prod_m q(G_m) \quad (4.8)$$

complètement factorisée et en suivant la méthode variationnelle (Bea03), nous cherchons q minimisant $D_{\mathrm{KL}}(\, q \,||\, p\,)$. La divergence est alors :

$$D_{\mathrm{KL}}(\, q \,||\, p\,) = \sum_{\{T_u\},\{G_m\}} q(\{T_u\},\{G_m\}) \log \frac{q(\{T_u\},\{G_m\})}{p(\{T_u\},\{G_m\} \mid \{\Delta_i\},\Theta)} \quad (4.9)$$

$$= \sum_{\{T_u\},\{G_m\}} \prod_u q(T_u)\prod_m q(G_m) \log \frac{\prod_u q(T_u)\prod_m q(G_m)}{\frac{p(\{T_u\},\{G_m\},\{\Delta_i\},\Theta)}{p(\{\Delta_i\},\Theta)}}. \quad (4.10)$$

La preuve $p(\{\Delta_i\},\Theta)$ ne dépendant pas de q, la KL est proportionelle à :

$$D_{\mathrm{KL}}(\, q \,||\, p\,) \;\propto\; \sum_u \sum_{T_u} q(T_u)\log q(T_u) \quad (4.11)$$

$$+ \sum_m \sum_{G_m} q(G_m)\log q(G_m) \quad (4.12)$$

$$- \sum_{\{T_u\},\{G_m\}} \prod_u q(T_u)\prod_m q(G_m) \log \prod_i \theta_{r_i,T_{u_i},G_{m_i}}. \quad (4.13)$$

Le dernier terme se simplifie en sortant la somme sur i :

$$\sum_{\{T_u\},\{G_m\}} \prod_u q(T_u) \prod_m q(G_m) \log \prod_i \theta_{r_i,T_{u_i},G_{m_i}} \tag{4.14}$$

$$= \sum_i \sum_{\{T_u\},\{G_m\}} \prod_u q(T_u) \prod_m q(G_m) \log \theta_{r_i,T_{u_i},G_{m_i}} \tag{4.15}$$

$$= \sum_i \sum_{T_{u_i}} \sum_{G_{m_i}} q(T_{u_i}) q(G_{m_i}) \log \theta_{r_i,T_{u_i},G_{m_i}}. \tag{4.16}$$

Pour minimiser cette KL sous la contrainte que, pour toutes les variables cachées, les marginales variationnelles sont normalisées, nous introduisons $N_u + N_m$ multiplicateurs de Lagrange. Nous annulons donc les dérivées de

$$A := D_{\mathrm{KL}}(\, q \,||\, p \,) + \sum_u \lambda_u \left[\sum_{T_u} q(T_u) - 1 \right] + \sum_m \lambda_m \left[\sum_{G_m} q(G_m) - 1 \right] \tag{4.17}$$

par rapport aux $KN_u + KN_m$ valeurs de probabilités. Par exemple, pour un utilisateur u', nous avons l'équation :

$$0 = \frac{\partial A}{\partial q(T_{u'})} = \log q(T_{u'}) + 1 - \sum_i \delta(u' = u_i) \sum_{G_{m_i}} q(G_{m_i}) \log \theta_{r_i,T_{u_i},G_{m_i}}. \tag{4.18}$$

Ce qui nous donne les équations de mise à jour pour ces probabilités :

$$q(T_u) \propto \exp \left[\sum_i \delta(u = u_i) \sum_{G_{m_i}} q(G_{m_i}) \log \theta_{r_i,T_{u_i},G_{m_i}} \right] \tag{4.19}$$

$$q(G_m) \propto \exp \left[\sum_i \delta(m = m_i) \sum_{T_{u_i}} q(T_{u_i}) \log \theta_{r_i,T_{u_i},G_{m_i}} \right]. \tag{4.20}$$

Algorithmiquement, il suffit d'initialiser à 0 toutes les $q(T_u)$ et $q(G_m)$, puis de parcourir toutes les données et incrémenter les $q(T_u)$ et $q(G_m)$ au bons endroits définis par u_i et m_i. Une fois le parcours terminé, il faut renormaliser ces distributions.

Phase M

La phase M est plus simple, il faut maximiser

$$B := \sum_i \sum_{\{T_u\},\{G_m\}} q(\{T_u\},\{G_m\}) \log \theta_{r_i,T_{u_i},G_{m_i}} \tag{4.21}$$

$$= \sum_i \sum_{T_{u_i}} \sum_{G_{m_i}} q(T_{u_i}) q(G_{m_i}) \log \theta_{r_i,T_{u_i},G_{m_i}} \tag{4.22}$$

par rapport aux $5K^2$ paramètres Θ. En introduisant des multiplicateurs de Lagrange, il vient

$$0 = \frac{\partial B}{\partial \theta_{r,T,G}} = \sum_i q(T_{u_i})\, q(G_{m_i}) \frac{\delta(r_i = r)}{\theta_{r,T,G}} + \lambda. \tag{4.23}$$

Ce qui donne la mise à jour :

$$\theta_{r,T,G} \propto \sum_i \delta(r_i = r)\; q(T_{u_i})\; q(G_{m_i}). \tag{4.24}$$

Encore une fois, il suffit de parcourir les données, incrémenter la matrice Θ dans sa case définie par chaque donnée et finalement renormaliser selon l'axe des r.

Clustering hiérarchique

Lors de l'exécution de EM, nous constatons que les *a posteriori* sur les variables cachées convergent "assez vite" vers des diracs : l'algorithme devient rapidement très sûr du type de chaque utilisateur. Comme nous l'avons évoqué au chapitre précédent, cette trop grande confiance, cette perte d'incertitude, est une caractéristique dommageable des approximations variationnelles de type $D_{\mathrm{KL}}(\, q \,||\, p \,)$. Mais nous proposons ici d'exploiter cette faiblesse, en en profitant pour accroître le nombre K de types (et de genres) de façon hiérarchique. En effet, le modèle A a besoin d'un grand nombre de clusters pour bien représenter les données, car il ne porte pas de sémantique de mixture. Le problème est que l'occupation mémoire est quadratique en K et que le temps de calcul croît aussi avec K, ce qui est très restrictif avec 10^8 données.

Alors une fois que EM a assez bien convergé, pour chaque utilisateur, nous oublions les types dont la probabilité est devenue quasi nulle et réutilisons leur place mémoire pour découper le type le plus probable en autant de sous types. Nous faisons de même pour les genres de films et nous relançons EM en modifiant la matrice A de façon cohérente. Ainsi nous pouvons aller jusqu'à plusieurs centaines de clusters avec des temps de calculs raisonnables.

4.5.2 Modèle B

Phase E

Comme nous avons un couple (T_i, G_i) pour chaque donnée, nous devons estimer N_r matrices $K \times K$:

$$f_i(T,G) := p(\{T_i\},\{G_i\} \mid \{u_i\},\{m_i\},\{r_i\},\Theta) \tag{4.25}$$

$$\propto p(T_i \mid u_i,\Theta)\; p(G_i \mid m_i,\Theta)\; p(r_i \mid T_i, G_i, \Theta). \tag{4.26}$$

Mais nous n'avons pas besoin de stocker toutes ces matrices, car la phase M les utilise indépendamment en parcourant aussi séquentiellement les données i.

Phase M

La phase M doit calculer :

$$\mathrm{ArgMax}_\Theta \sum_i \sum_{T_i,G_i} f_i(T_i,G_i)\; \log p(T_i \mid u_i,\Theta) p(G_i \mid m_i,\Theta)\; p(r_i \mid T_i, G_i, \Theta). \tag{4.27}$$

Ce qui donne les mises à jour :

$$p(R \mid T, G, \Theta) \propto \sum_i \delta(r_i = r)\ f_i(T, G) \tag{4.28}$$

$$\theta_u = p(T \mid u, \Theta) \propto \sum_i \sum_G \delta(u_i = u)\ f_i(T, G) \tag{4.29}$$

$$\theta_m = p(G \mid m, \Theta) \propto \sum_i \sum_T \delta(m_i = m)\ f_i(T, G). \tag{4.30}$$

Nous parcourons donc une fois toutes les données pour réaliser simultanément les phases E et M, évitant ainsi d'avoir à stocker les N_r matrices $f_i(T, G)$.

4.6 Netflix : Résultats

4.6.1 Prédiction

Avec les deux modèles, les prédictions sont réalisées de la même façon. Pour un nouveau couple (u, m), nous calculons la distribution prédictive $p(r \mid u, m, \Theta_{\text{ML}})$ avec les paramètres de maximum de vraisemblance retournés par EM, en marginalisant sur la distribution des types :

$$p(r \mid u, m, \Theta_{\text{ML}}) = \sum_{T,G} p(r \mid T, G, \Theta_{\text{ML}})\ p(T \mid u, \Theta_{\text{ML}})\ p(G \mid m, \Theta_{\text{ML}}). \tag{4.31}$$

Ce calcul se traduit par des produits de matrices multidimensionnelles. En effet, pour chaque prédiction et pour chaque valeur de r, nous multiplions d'abord la matrice 2D $p(r \mid u, m, \Theta_{\text{ML}})$ par la matrice colonne $p(T \mid u, \Theta_{\text{ML}})$. Nous calculons ensuite le produit scalaire du vecteur résultant et de la matrice colonne $p(G \mid m, \Theta_{\text{ML}})$ pour marginaliser sur les genres. Cette opération devant être répétées pour les cinq valeurs de r et pour chaque donnée du fichier de test, nous pouvons le résumer par un produit d'une matrice 4D par deux matrices 2D.

La différence sémantique entre les deux modèles n'est pas pertinente pour la prédiction : soit nous marginalisons sur notre *a posteriori* sur le type, soit nous moyennons sur les degré d'appartenance de l'utilisateur aux différents types.

Comme la fonction de coût est la RMSE, la décision maximisant l'utilité est l'espérance de $p(r|u, m, \Theta_{\text{ML}})$. En effet, en notant $p(r) := p(r|u, m, \Theta_{\text{ML}})$ notre croyance *a posteriori* sur la note à prévoir, nous cherchons :

$$r^* = \text{ArgMin}_x\ E_p[L_{\text{RMSE}}(x, r)] \tag{4.32}$$

$$= \text{ArgMin}_x\ \int_r (x - r)^2\ p(r)\ dr \tag{4.33}$$

$$= \text{ArgMin}_x\ x^2 + E_p[r^2] - 2\ x\ E_p[r]. \tag{4.34}$$

En annulant la dérivée, nous obtenons bien $r^* = E_p[r]$.

4.6.2 Résultats

Avec le jeu de données de $N_r = 10^8$ données, $N_u = 480189$ utilisateurs et $N_m = 17770$ films et un nombres de clusters de l'ordre de $K = 10$, nos itérations EM prennent environ 10h sur une ordinateur de bureau standard (C++, gcc, 1GHz, 2Go RAM), après optimisations. Comme il faut au moins une dizaine d'itérations pour avoir des résultats pertinents, nous n'avons pas testé extensivement nos modèles.

Cependant, en novembre 2006, quand nous avons arrêté de travailler sur ce projet, nous étions dans la compétition avec une RMSE de **0.9519**. Ce score s'approche de celui de la société Netflix (0.9514), mais est très loin de celui des meneurs d'aujourd'hui (0.8712 fin septembre 2007). Un algorithme naïf qui prédit systématiquement la moyenne des notes pour chaque film donne une RMSE de 1.05.

Les meilleurs résultats (0.9519) ont étés obtenus avec le modèle A, avec $K = 15$ et deux niveaux hiérarchiques soit un nombre de clusters de 255×225. Les clusters de films obtenus sont intuitivement pertinents car, des films similaires sont regroupés ensemble (par exemple des DVDs d'une même série télévisée). Pour contrer la convergence prématurée due à l'approximation *Mean Field*, nous avons ajouté un *a priori* de Dirichlet symétrique de poids 5 sur les densités des variables d'affectation, ce qui n'a amélioré les résultats que de quelques centièmes. Nous avons aussi constaté qu'un clustering asymétrique ($K = 60$ pour les utilisateurs et $K = 1$ pour les films) faisait chuter les performances, confirmant l'intérêt du clustering parallèle. Un autre essai pour prévenir la convergence précoce fut l'incorporation d'un recuit simulé, afin de trouver un meilleur maximum local. Cet ajout n'a pas sensiblement amélioré les performances prédictives.

Pour le modèle B, la meilleure performance est une RMSE de 0.959 avec 10×10 clusters et une cinquantaine d'itérations. Les genres de films sont aussi intuitivement pertinents. Nous avons constaté qu'avec ce modèle, EM convergeait assez vite vers des minima locaux et était très sensible à une initialisation aléatoire des matrices de paramètres. L'ajout d'*a priori* de Dirichlet n'a pas beaucoup amélioré les résultats.

4.7 Conclusion

4.7.1 Contributions

Dans ce chapitre nous avons proposé deux modèles génératifs originaux pour grouper symétriquement des utilisateurs et des objets dans un but de filtrage collaboratif. Pour ces modèles nous avons dérivé et mis en œuvre des algorithmes d'inférence approchés de type EM, avec éventuellement des approximations variationnelles. Nous les avons appliqués au plus grand problème de filtrage collaboratif publiquement disponible, nos algorithmes réalisant une inférence dans un réseau de plus de 300 millions de variables aléatoires.

En raison de la localité des optimisations et de la taille du jeu de données, nos résultats ne sont pas aussi bons que ceux des meilleures équipes mondiales, mais ils sont meilleurs que ceux issus d'approches simples et sont certainement améliorables.

4.7.2 Perspectives

Notons que les meilleures équipes sont composées de plusieurs chercheurs travaillant à plein temps depuis un an sur ce projet, exploitant et combinant une multitude de méthodes différentes. Toutes les équipes ne publient pas leur approches, mais il semble que la combinaisons de plusieurs méthodes donne de meilleurs résultats qu'une méthode quelconque isolée : le meilleur modèle est une mixture de modèles. L'approche SVD régularisée, avec des réglages soigneux permet déjà d'obtenir une RMSE de 0.8914 (SimonFunk, septembre 2007). Une amélioration dans la méthode de régularisation et l'ajout de post-traitements par régression d'arrêtes et processus gaussiens a permis d'atteindre une RMSE de 0.8789 (Pat07). Il semble de plus que l'utilisation de l'information de date permette des améliorations importantes. L'approche ayant jusqu'à maintenant donné les meilleurs résultats est une combinaison multi-échelle entre une SVD régularisée à grande échelle et une approche fondée mémoire déterminant des voisinages locaux entre utilisateurs (BKV07). Dans cette approche, la fonction de similarité n'est pas arbitrairement posée, mais est le résultat d'une optimisation globale et plusieurs techniques de régularisation atténuent le sur-apprentissage lors de la SVD.

En dépit de nos résultats, nous pensons que la modélisation de clustering bayésien pourrait donner de bien meilleures performances. D'abord, avec une plus grande puissance de calcul, il aurait été possible d'augmenter le nombre de clusters, ce qui a toujours amélioré nos scores. De plus, la sensibilité du modèle B à l'initialisation nous fait penser que de très nombreux minima locaux sont présents et qu'une initialisation par un autre algorithme pourrait aider à mieux démarrer. Il serait aussi possible d'améliorer le modèle, par exemple en ajoutant les informations de dates et modélisant l'évolution des utilisateurs au cours du temps. Un autre point qui n'a pas été modélisé, mais qui nous paraît important est la distribution non uniforme des données $\{u_i, m_i\}$. En effet, tous les utilisateurs n'ont pas les mêmes chances de noter tous les films, il est fort probable qu'une personne voit plus souvent des films qui lui plaisent que d'autres. Dans une approche générative, ce biais devrait être pris en compte pour mieux représenter les données.

Un dernier axe d'amélioration possible est le choix du nombre de clusters. Comme dans tout modèle de mélange, il existe un K optimal, dépendant des données et éventuellement différent pour les films et pour les utilisateurs. Un trop petit K limite le pouvoir d'expression du modèle, alors qu'un trop grand K empêche le partage d'informations entre utilisateurs et conduit à du sur-apprentissage. Le temps de calcul et la place mémoire est une autre contrainte, technique, sur le choix de K. Pour cette raison, nous n'avons pas pu atteindre dans nos expériences de K trop grand. Dans une approche bayésienne complète, comme le K optimal dépend des données, il devrait aussi être inféré, ou mieux marginalisé. C'est l'idée des modèles non paramétriques comme les processus de Dirichlet (Fer73 ; BGJT04), ayant conduit à des algorithmes de mélanges avec une infinité de composantes, comme les mixtures infinies de gaussiennes (Ras00). Cependant, dériver une méthode d'inférence efficace pour le problème Netflix semble délicat, d'autant plus que ces modèles infinis s'appuient majoritairement sur des approches stochastiques.

La conclusion principale de ce travail est que, bien qu'il soit très satisfaisant d'établir un

bon modèle génératif du problème à étudier, si les inférences ne sont pas réalisables, ou si les approximations ne sont pas assez précises (optima locaux), alors il peut être raisonnable de revenir à des méthodes *ad-hoc* comme la SVD.

Troisième partie

Apprentissage de modèles

Chapitre 5

Sélection de variables

Dans la partie précédente, nous avons appliqué la méthode bayésienne à deux problèmes d'apprentissage pour lesquels le modèle était bien défini. Dans cette partie nous nous intéressons à l'apprentissage du modèle lui-même.

La majeure partie de la littérature concernant l'apprentissage de modèle se consacre à l'apprentissage des structures de dépendances lorsque les variables sont connues. C'est un problème difficile en raison de la taille super-exponentielle de l'espace des structures. Pour résoudre ce problème, la méthode bayésienne nous pousserait à calculer les vraisemblances marginales de chaque structure, et de trouver la structure de score maximal. Comme ceci n'est pas possible dès qu'il y a plus que quelques variables, différentes méthodes ont été proposées (LF05). Par exemple il est possible de parcourir l'espace des structures à l'aide d'algorithmes génétiques, de MCMC ou de modification locale de structures. Il est aussi possible de restreindre l'espace de recherche à des structures plus simple comme les arbres couvrants.

Dans cette partie nous ne nous intéresserons pas directement au problème d'apprentissage de structure, mais plutôt à l'apprentissage de variables. Nous étudions d'abord un problème de sélection de variables, puis dans le chapitre suivant nous nous intéressons à un problème de découverte d'une nouvelle variable et de sa cardinalité.

5.1 Introduction

Pour établir un modèle probabiliste bayésien, nous devons choisir un ensemble de variables pertinentes. En plus des variables cachées et des variables représentant des paramètres, nous introduisons en général une variable par donnée observée. Ainsi le nombre de variables du modèle probabiliste peut devenir très grand et rendre les inférences délicates. Nous avons vu dans la partie précédente qu'il était possible de traiter de grands modèles grâce à des méthodes d'approximation adaptées.

Cependant nous aimerions parfois réduire ce nombre, pour ne garder qu'un sous-ensemble de variables suffisantes à la réalisation d'une certaine tâche. Cette approche, la sélection de variables (*feature selection*), est souvent utilisée comme un prétraitement,

une réduction de dimensionnalité, avant les phases d'apprentissage proprement dites.

Dans ce chapitre, nous présentons et comparons différentes méthodes automatiques de sélection de variables, à partir de données.

5.1.1 Sélection : Intérêts et inconvénients

Sélectionner un sous-ensemble de variables avant de réaliser un apprentissage présente trois intérêts. Premièrement, cela permet une réduction des coûts en temps de calcul et en place mémoire de la phase d'apprentissage, rendant certaines méthodes à nouveau applicables. Deuxièmement, il est courant de constater que l'apprentissage est plus performant avec des données prétraitées (LG99). Finalement, une élimination des variables non pertinentes nous donne une vision plus claire des données, un modèle plus parlant, permettant ainsi d'améliorer notre compréhension des phénomènes sous-jacents.

L'inconvénient majeur de tels prétraitements est un risque de perte d'information. En effet, quelle que soit la méthode utilisée, la quantité d'information présente dans les données prétraitées ne peut être qu'inférieure ou égale à celle des données brutes, sauf si la méthode apporte une information extérieure. Le prétraitement permet éventuellement une meilleure présentation d'une partie de l'information, mais il ne peut pas en ajouter de façon absolue.

Imaginons par exemple que les variables sélectionnées soient ensuite utilisées pour faire des prédictions. Alors le taux de réussite d'un prédicteur *optimal* avec un sous-ensemble des variables ne peut être qu'inférieur ou égal à celui d'un prédicteur *optimal* utilisant les données brutes. La notion d'optimalité est importante : la plupart du temps, les prédicteurs utilisés ne le sont pas et il se peut que leurs performances soient meilleures avec des données prétraitées. Cette hypothèse de non croissance des performance a été expérimentalement confirmée par R. Neal[1] qui a remporté le concours de *feature selection* de NIPS-03 en gardant toutes les variables (GGBHD04).

La sélection de variables n'est donc justifiable que de deux façons. Soit l'algorithme prédicteur est sous optimal et pour des raisons extérieures nous ne pouvons pas l'améliorer. Soit l'utilisation de nombreuses variables à un coût explicite que nous voulons minimiser, tout en maximisant les performances prédictives.

5.1.2 Sélection : Application

Dans ce chapitre, notre prédicteur sera un classificateur bayésien naïf faisant de fortes hypothèses sur la structure de dépendance probabiliste. Il ne sera pas optimal pour nos données, qui ne respectent pas tout à fait ces hypothèses. De plus, comme nous travaillerons avec des variables discrétisées et des tables de probabilités, nous aurons intérêt à ne garder qu'un sous-ensemble des variables pour minimiser les coûts calculatoires.

Nous appliquerons la sélection de variables à un contexte de robotique autonome. Après avoir validé et comparé différents algorithmes sur des données simulées, nous les utiliserons

[1] Le prédicteur de Neal est un réseaux de neurones bayésiens avec des *a priori* fondés sur des arbres de diffusion de Dirichlet (Nea01) et une inférence stochastique.

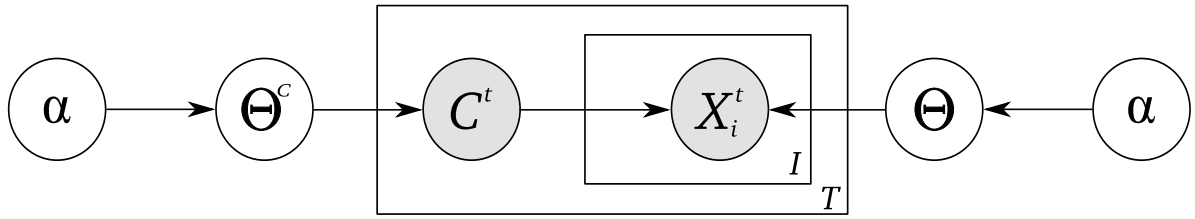


FIG. 5.1 – Modélisation bayésienne du classificateur naïf : les I variables X_i sont indépendantes, conditionnellement à la classe C. Les tables de probabilités conditionnelles sont représentées par Θ, les *a priori* sur C par Θ^C. Les T jeux de données sont i.i.d. : ils partagent les mêmes Θ. Les *a priori* sur les Θ et Θ^C sont contrôlés par le même hyper-paramètre α.

pour doter notre robot d'une capacité d'apprentissage "auto-supervisé". Notre objectif est l'automatisation de la découverte de nouveaux comportements par la recherche de régularités statistiques. Le robot devra par lui-même trouver des corrélations dans son domaine sensori-moteur lors d'une première exposition à son environnement, puis les exploiter pour réaliser une tâche similaire. Le robot devra apprendre lesquels de ses capteurs sont utiles pour localiser un objet.

5.2 Le problème de la sélection de variables

5.2.1 Sélection pour un apprentissage supervisé

Nous considérons le problème de la sélection de variables dans un contexte robotique. Le robot possède un grand nombre J de capteurs $\mathbf{X} = \{X_1, \ldots, X_J\}$ indicés par la lettre i. Chaque capteur peut retourner L valeurs possibles $X_i \in \{x_{i,1}, \ldots, x_{i,L}\}$ indicées par la lettre l. À chaque instant t, une lecture $\mathbf{x}^t$ de ces capteurs est réalisée et associée à une classe c^t. Ces T lectures sont supposées être i.i.d.. Il y a K classes possibles $C^t \in \{c_1, \ldots, c_K\}$ indicées par la lettre k. La tâche du robot est de réaliser un apprentissage supervisé de ces classes à partir de T données $\Delta^T = \{(C^1 = c^1_{k^1}, \mathbf{X}^1 = \mathbf{x}^1_{\mathbf{l}^1}), \ldots, (C^T = c^T_{k^T}, \mathbf{X}^T = \mathbf{x}^T_{\mathbf{l}^T})\}$ afin de pouvoir prédire la classe d'une nouvelle donnée $\mathbf{X}^{T+1}$.

L'algorithme d'apprentissage et de prédiction est un classificateur bayésien naïf. Pour faciliter son travail, le prétraitement consistera à sélectionner un sous-ensemble de $I \ll J$ capteurs. Avant de présenter la sélection de variables, nous décrivons le classificateur bayésien naïf.

5.2.2 Apprentissage et prédiction

Classificateur naïf : Modèle

Le modèle bayésien naïf (LR94 ; DP97), appelé modèle capteur en robotique (Leb99), suppose que les variables $\mathbf{X}^t$ soient indépendantes, conditionnellement au phénomène mesuré, ici la classe C. Il y a bien une dépendance entre X_i et X_j, mais cette dépendance passe entièrement par la variable C. Ce modèle constitue l'une des approches bayésiennes les plus répandues, car il est à la base d'un grand nombre de filtres anti-*spam* (SDHH98). Nous présentons ce modèle et dérivons un traitement bayésien complet des phases d'apprentissage et de prédiction.

En introduisant une variable $\mathbf{\Theta} := \{\Theta, \Theta^C\}$ représentant les paramètres de toutes les tables de probabilités, la distribution jointe (fig. 5.1) est :

$$p(\{C^t\}, \{\mathbf{X}^t\}, \mathbf{\Theta}) = p(\mathbf{\Theta}) \prod_t \left[p(\{C^t\} \,|\mathbf{\Theta}) \prod_i p(X_i^t \,|\breve{a} C^t, \mathbf{\Theta}) \right]. \tag{5.1}$$

La variable $\mathbf{\Theta}$ peut être vue comme une table normalisée Θ^C à une dimension (k) pour la paramétrisation de l'*a priori* sur les classes et une table Θ à trois dimensions (i, k, l) pour les distributions conditionnelles. Pour chaque X_i, pour chaque classe c_k, cette dernière donne les probabilités $\theta_{i,k,l} := p(X_i = x_{i,l} \,|\, C = c_k, \Theta)$. Ces nombres sont normalisés selon l'axe des l.

Nous supposons que l'*a priori* $p(\mathbf{\Theta})$ est factorisé et pour chaque $\theta_{i,k}$ nous choisissons un *a priori* de Dirichlet[2] symétrique d'hyper-paramètre α. Nous choisissons le même *a priori* pour Θ^C. Cet *a priori* signifie que nous avons α données virtuelles pour chaque réalisation possible des X_i (resp. C) : nous ne privilégions aucune réalisation particulière $x_{i,l}$ de X_i. Une grande valeur de α exprime une grande confiance dans l'uniformité de la distribution sur X_i (resp. C).

La distribution jointe est alors :

$$p(\{C^t\}, \{\mathbf{X}^t\}, \mathbf{\Theta}) = \left[\prod_{i,k,l} \theta_{i,k,l}^{\alpha} \right] \cdot \left[\prod_k \theta_k^{C\,\alpha} \right] \cdot \prod_t \left[\theta_{k^t}^C \prod_i \theta_{i,k^t,l^t} \right] \tag{5.2}$$

$$= \left[\prod_{i,k,l} \theta_{i,k,l}^{\alpha} \right] \cdot \left[\prod_k \theta_k^{C\,\alpha} \right] \cdot \left[\prod_{i,k,l} \theta_{i,k,l}^{\sum_t \delta(k=k^t)\,\delta(l=l^t)} \right] \cdot \left[\prod_k \theta_k^{C \sum_t \delta(k=k^t)} \right] \tag{5.3}$$

$$= \prod_{i,k,l} \theta_{i,k,l}^{\alpha + \sum_t \delta(k=k^t)\,\delta(l=l^t)} \cdot \prod_k \theta_k^{C\,\alpha + \sum_t \delta(k=k^t)}. \tag{5.4}$$

[2] La Dirichlet est à la multinomiale ce que la Bêta est à la Bernoulli, cf. sec. 1.4.3.

Classificateur naïf : Apprentissage

L'apprentissage consiste en la construction de l'*a posteriori* sur $\mathbf{\Theta}$:

$$p(\mathbf{\Theta} \mid \Delta^T) = \frac{p(\Delta^T \mid \mathbf{\Theta})\, p(\mathbf{\Theta})}{p(\Delta^T)} \tag{5.5}$$

$$\propto p(\Delta^T, \mathbf{\Theta}) \tag{5.6}$$

$$= \prod_{i,k,l} \theta_{i,k,l}^{n_{i,k,l}^T} \cdot \prod_k \theta_k^{C\, n_k^T} \tag{5.7}$$

$$= \left(\prod_i p(\Theta_i \mid \Delta^T) \right) \cdot p(\Theta^C \mid \Delta^T), \tag{5.8}$$

en définissant les exposants

$$n_{i,k,l}^T := \alpha + \sum_t \delta(k = k^t)\, \delta(l = l^t) \tag{5.9}$$

$$n_k^T := \alpha + \sum_t \delta(k = k^t). \tag{5.10}$$

Nous notons que les variables de $\mathbf{\Theta}$ sont *a posteriori* indépendantes. Construire l'*a posteriori* revient donc à compter les données réelles pour chaque cases de $\mathbf{\Theta}$, en y ajoutant le nombre de données virtuelles α. La loi *a posteriori* sur $\mathbf{\Theta}$ peut être identifiée à une distribution de Dirichlet de paramètres $n_{i,k,l}^T$ et n_k^T.

Classificateur naïf : Prédiction

Pour prédire la classe c_k^{T+1} d'une nouvelle donnée $\mathbf{x}^{T+1}$, nous marginalisons sur $\mathbf{\Theta}$ et maximisons la probabilité conditionnelle :

$$k^{\text{pred}} = \text{ArgMax}_k\ p(C^{T+1} = c_k \mid \mathbf{X}^{T+1} = \mathbf{x}_{\mathbf{l}^{T+1}}, \Delta^T) \tag{5.11}$$

$$= \text{ArgMax}_k\ \int_{\mathbf{\Theta}} p(c_k \mid \mathbf{x}_{\mathbf{l}^{T+1}}, \mathbf{\Theta})\ p(\mathbf{\Theta}|\Delta^T)\ d\mathbf{\Theta} \tag{5.12}$$

$$= \text{ArgMax}_k\ \int_{\mathbf{\Theta}} \left[\prod_i p(x_{i,l^{T+1}} \mid c_k, \mathbf{\Theta}) \right] p(c_k \mid \mathbf{\Theta})\ p(\mathbf{\Theta}|\Delta^T)\ d\mathbf{\Theta} \tag{5.13}$$

$$= \text{ArgMax}_k\ \prod_i \int_{\Theta_i} \theta_{i,k,l^{T+1}}\ p(\Theta_i|\Delta^T)\ d\Theta_i \cdot \int_{\Theta^C} \theta_{k^{T+1}}^C\ p(\Theta^C|\Delta^T)\ d\Theta^C \tag{5.14}$$

$$= \text{ArgMax}_k\ \left[\prod_i E_{p(\Theta|\Delta^T)} \left[\theta_{i,k,l^{T+1}} \right] \right] \cdot E_{p(\Theta^C|\Delta^T)} \left[\theta_k^C \right] \tag{5.15}$$

$$= \text{ArgMax}_k\ \left[\prod_i \frac{1 + n_{i,k,l^{T+1}}^T}{L + \sum_l n_{i,k,l}^T} \right] \cdot \frac{1 + n_k^T}{K + \sum_k n_k^T} \tag{5.16}$$

$$= \text{ArgMax}_k\ \left[\prod_i \frac{1 + \alpha + \sum_t \delta(k = k^t)\, \delta(l = l^t)}{L + \alpha\, L + T} \right] \cdot \frac{1 + \alpha + \sum_t \delta(k = k^t)}{K + \alpha\, K + T}. \tag{5.17}$$

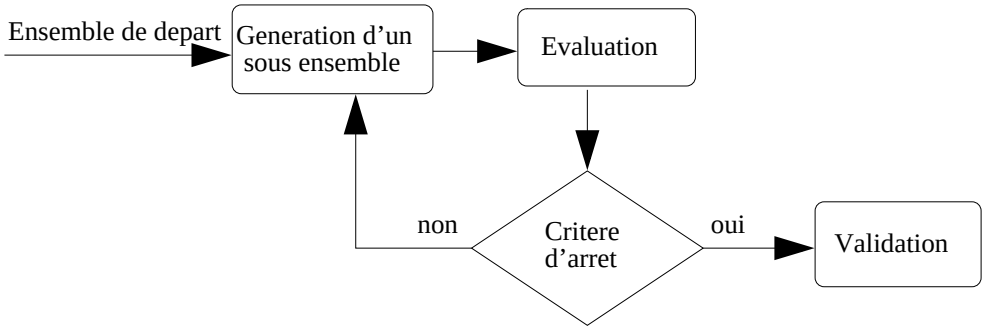


FIG. 5.2 – Procédure générale de sélection d'un sous-ensemble de variables.

L'identité 5.14 est obtenue en intervertissant les intégrales et les produits, alors que 5.16 nécessite I intégrations sur des $(L-1)$-simplexes et une sur un $(K-1)$-simplexe. Ces intégrations correspondent au calcul de l'espérance de lois de Dirichlet.

Pour l'hyper-paramètre α, nous choisissons la valeur 0, spécifiant ainsi une incertitude *a priori* totale sur la forme des tables de probabilités. Notons que ce choix $\alpha = 0$ ne correspond pas à supposer un *a priori* uniforme sur X_i, mais un *a priori* uniforme sur "$p(X_i)$".

Alors, dans ce cas particulier, la méthode de prédiction correspond à la loi de succession de Laplace ("$P(\text{succès au prochain essai} \,|\, \text{T succès passés}) \propto T+1$"). Notons que la valeur 1 est due à la marginalisation sur $\boldsymbol{\Theta}$, mais qu'il aurait été admissible de prendre d'autres valeurs (comme 5 ou 10^{-5}) en choisissant $\alpha \neq 0, (\alpha > -1)$. L'interprétation en terme d'hyper-paramètre aurait alors permis de s'assurer de la pertinence de ce choix en regard des connaissance *a priori* disponibles.

Classificateur naïf : Mise en œuvre

Pour réaliser l'apprentissage, la formule 5.17 ne nécessite que le comptage des données correspondantes à chacune des cases de $\boldsymbol{\Theta}$. La place mémoire utilisée est donc $K \times L \times I + K$ et le temps de construction est $O(T + K \times L \times I + K)$, en raison des renormalisations requises. Ces coûts n'évoluent donc que linéairement avec le nombre de capteurs I, ce qui peut néanmoins être prohibitif. Une fois les comptages effectués. Une prédiction correspond au calcul de 5.17, soit à K produits de $I+1$ termes.

5.3 État de l'art

Pour sélectionner des variables pertinentes d'une ensemble de J variables, une première idée est d'en générer tous les sous-ensembles et de ne garder que celui maximisant les performances du prédicteur et minimisant le nombre de variables I conservées. Cependant, le nombre de sous-ensemble croît exponentiellement avec J. Pour éviter ce coût, les algorithmes de sélection sont gloutons (*greedy algorithm*) : ils alternent une phase de génération de sous-ensembles candidats avec une phase d'évaluation, afin d'optimiser localement leur

qualité. La figure 5.2 présente les différentes étapes d'une sélection de variables. Nous décrivons chacune de ces étapes.

5.3.1 Génération de sous-ensembles

La génération de sous-ensembles est une procédure de recherche dans l'espace des sous-ensembles de cardinal 2^J. Plusieurs méthodes de recherche sont utilisables.

Parcours exhaustif. Tous les sous-ensembles sont générés. Le coût est prohibitif dès que $J > 10$ bien qu'un tel parcours garantisse l'optimalité de la solution. Même lorsque la fonction d'évaluation est monotone en I, le coût demeure $O(2^J)$ (NF77).

Parcours heuristique. Une heuristique est utilisée pour guider la recherche. Le coût devient $O(J^2)$. Dans cette catégorie, nous pouvons commencer par un sous-ensemble vide auquel nous ajoutons des variables une à une (*forward addition*). Il est aussi possible de commencer avec l'ensemble de toutes les variables, puis d'éliminer les moins intéressantes (*backward elimination*). Bien que des résultats expérimentaux montrent que les performances de ces deux méthodes soient souvent équivalentes (JKP94), la *forward addition* a l'avantage de fournir des plus petits candidats à la phase d'évaluation.

Parcours non déterministe. Les sous-ensembles sont générés par un processus stochastique, par exemple par des metaheuristiques de type évolutionnistes (RKFM02 ; YH98).

5.3.2 Évaluation

Il existe deux grandes classes de méthodes d'évaluation donnant deux classes d'algorithmes de sélection de variables.

Les *wrappers*

Les *wrappers* (JKP94) évaluent les sous-ensembles en mesurant les performances du prédicteur, par exemple par validation croisée. Le prédicteur est utilisé comme une boite noire. L'avantage principal des *wrappers* est qu'ils sont adaptés à l'algorithme de classification, ils prennent automatiquement en compte ses inévitables biais. C'est aussi un inconvénient dans la mesure où un changement de prédicteur rend caduque le sous-ensemble trouvé. De plus, comme chaque évaluation nécessite au moins un cycle apprentissage-prédiction, les *wrappers* sont coûteux en temps de calculs.

Les *filters*

Les filtres se basent sur des mesures de dépendance entre variables pour évaluer les sous-ensembles. Ils sont plus rapides et permettent de mieux comprendre les relations entre variables. Mais, comme ils ne prennent pas en compte les biais du prédicteur, les performances finales sont en général plus faibles.

Pour évaluer un sous-ensemble, une méthode naïve consiste à sommer les scores de chaque variable considérée indépendamment des autres (*feature ranking*). Le score d'une variable dépend d'une estimation de sa dépendance avec la classe. Cependant cette approche n'élimine pas les variables redondantes et il est possible que des variables peu dépendantes de la classe deviennent utiles lorsqu'on les considère dans le contexte des autres variables (GE03).

Il est donc préférable d'évaluer un sous-ensemble dans sa globalité (KS96) comme lors de l'apprentissage de structure de réseaux bayésiens (HGC94). Comme cet apprentissage de structure est un problème difficile, des compromis entre le *feature ranking* et le *subset ranking* ont été proposés. Hall (Hal98) propose l'utilisation d'une formule heuristique pour mesurer l'intérêt d'un sous-ensemble. L'idée est de sélectionner des variables fortement dépendantes de la classe (éliminer les X_i non pertinentes), mais peu dépendantes entre elles (minimiser la redondance). La formule heuristique proposée par Ghiselli (Ghi64) est :

$$r_{C,S} = \frac{I\,\overline{r_{C,i}}}{\sqrt{I + I(I-1)\overline{r_{i,j}}}} \tag{5.18}$$

avec $r_{C,S}$ le score du sous-ensemble, $\overline{r_{C,i}}$ la moyenne des dépendances entre C et chaque X_i et $\overline{r_{i,j}}$ la moyenne des k^2 dépendances entre variables. C'est une approximation car seules les dépendances entre couples de variables sont prises en compte.

La dépendance de deux variables peut être estimée de différentes façons. Hall (Hal00), en utilisant le coefficient de corrélation empirique, se limite aux dépendances monotones. D'autres auteurs se basent sur d'autres mesures de dépendances, comme des tests statistiques (χ^2, (LS95)). Une mesure plus générale de dépendance entre deux variables est l'information mutuelle :

$$I(X_i, X_j) = D_{\mathrm{KL}}(\, p(X_i, X_j) \,||\, p(X_i)\, p(X_j)\,), \tag{5.19}$$

qui est nulle si et seulement si les variables sont indépendantes. Cette information peut être estimée empiriquement par des sommes (Wol94), ou par une approche fondée sur un *a posteriori* bayésien (MT01).

5.3.3 Critère d'arrêt

Différents critères d'arrêt sont possibles : temps de calcul, nombre de variables sélectionnées, évaluation heuristique de la qualité du sous-ensemble produit ou évolution de la mesure d'évaluation.

5.3.4 Validation

Une fois le "meilleur" sous-ensemble trouvé, nous pouvons évaluer sa qualité en calculant le taux de réussite du prédicteur sur un jeu de données différent de celui ayant servi à la sélection. Cette méthode nous permettra de comparer différents algorithmes de sélection.

5.4 Sélection : Algorithmes

Nous présentons dans cette partie quatre algorithmes de sélection de variables que nous comparons sur des données robotiques. Le dernier algorithme est original alors les autres sont des recombinaisons de méthodes inspirées de l'état de l'art. Les algorithmes sont appelés : *WrappGA*, *FiltGA*, *FiltFA* et *FiltCIF*. Nous utiliserons aussi deux algorithmes de sélection triviaux comme référence.

5.4.1 WrappGA

Génération de sous-ensemble Algorithme génétique.

Évaluation *wrapper*.

Critère d'arrêt Nombre de générations ($\propto$ temps de calcul).

Détails Le parcours de l'espace des sous-ensembles est réalisé par un algorithme génétique (Hol75). Les algorithmes génétiques sont des metaheuristiques cherchant le minimum global d'une fonction de coût. Ils sont issus d'une analogie avec la biologie : une population d'individus candidats évolue par reproduction (*crossover*) et par mutation sous une pression darwinienne. Ils sont connus pour donner des solutions approximatives à des problèmes difficiles en des temps de calculs raisonnables (Koz92). Dans notre cas, un individu représente un sous-ensemble et la fonction d'évaluation retourne le taux de reconnaissance du prédicteur pour ce candidat. Un individu S est codé par une chaîne de J bits. Le $j^{\text{ième}}$ bits vaut un si S contient X_j, zéro sinon. La sélection est réalisée par tournoi (Spa99) et les *crossovers* sont uniformes. L'ajustement des paramètres de l'algorithme génétique, toujours délicat, a été réalisé par essais et erreurs sur différents jeux de données simulées. Ces paramètres sont le taux de mutation, le taux de *crossover*, la taille de la population et le nombre de générations.

Complexité Pour chaque individu de chaque génération, nous effectuons F apprentissages sur un fichier de T données, avec F le nombre de pliages de la validation croisée (*F-fold cross validation*). Si G est le nombre de générations et M le nombre d'individus, le coût en temps de calcul est celui de $G \times M \times F$ apprentissages et $T \times F$ prédictions soit : $O(G \times M \times F \times (K \times L \times I + T \times K \times I)$.

5.4.2 FiltGA

Génération de sous-ensemble Algorithme génétique.

Évaluation Filtre avec formule de Ghiselli et information mutuelle empirique normalisée.

Critère d'arrêt Nombre de générations.

Détails Pour évaluer la dépendance entre deux variables, nous calculons le coefficient d'incertitude symétrique empirique défini par :

$$\text{CIS} := 2\frac{I(X_i, X_j)}{H(X_i)\, H(X_j)}. \tag{5.20}$$

En normalisant par le produit des entropies, le CIS corrige le biais qu'a l'information mutuelle en faveur des variables à beaucoup de valeurs possibles (L).

Complexité $O(G \times M \times I^2 + J^2 \times L^2)$ après optimisation et principalement en raison de l'estimation des J^2 informations mutuelles. Cependant il n'est pas nécessaire de calculer toutes ces informations, mais seulement celles apparaissant lors de l'évolution et de les réutiliser ensuite.

5.4.3 FiltFA

Génération de sous-ensemble *Forward addition.*

Évaluation Filtre avec formule de Ghiselli et information mutuelle empirique normalisée.

Critère d'arrêt La *forward addition* entraînant une croissance stricte de la taille des sous-ensembles, l'algorithme est arrêté quand la fonction de coût ne peut plus croître.

Détails La seule différence entre *FiltFA* et *FiltGA* est le parcours de l'espace des sous-ensembles. *FiltFA* trouvera certainement un minimum local, alors que *FiltGA* a pour ambition de trouver le maximum global. Lors de la *forward addition*, nous ajoutons à chaque pas la meilleur variable, celle donnant le sous-ensemble maximisant l'heuristique de Ghiselli.

Complexité $O(J^2\breve{a} \times I^2 + J^2 \times L^2)$ dans le pire cas, mais beaucoup moins en pratique car l'addition s'arrête rapidement.

5.4.4 FiltCIF

Génération de sous-ensemble *Backward elimination.*

Évaluation Filtre basé sur l'hypothèse de modèle capteur du prédicteur bayésien naïf.

Critère d'arrêt Nombre de variables conservées prédéfini ($J/2$ dans nos tests).

Détails Nous commençons avec un sous-ensemble complet à J variables et, à chaque étape de l'élimination, nous déterminons la variable la plus inutile. Pour cela, nous utilisons le modèle utilisé par le prédicteur : les X_i sont indépendantes entre elles, conditionnellement à la classe C. Nous considérons que la variable à éliminer X_{i^*} est celle minimisant la divergence entre la jointe p sur les variables restantes et une jointe modifiée q dans laquelle X_{i^*} est indépendante de C. Nous minimisons pour cela la KL symétrisée :

$$D(\, p \,||\, q \,) := D_{\mathrm{KL}}(\, p \,||\, q \,) + D_{\mathrm{KL}}(\, q \,||\, p \,) \tag{5.21}$$

$$p(\{X_i\}, C) = p(C) \prod_i p(X_i|C) \tag{5.22}$$

$$q(\{X_i\}, C) = p(C)\, p(X_{i^*}) \prod_{i \neq i^*} p(X_i|C). \tag{5.23}$$

Cette divergence devient :

$$D(\,p\,||\,q\,) = \sum_{\{X_i\},C} p(C) \prod_{i \neq i^*} p(X_i|C) \left[\log \left(\frac{p(X_{i^*})}{p(X_{i^*}|C)} \right) - (p(X_{i^*}) - p(X_{i^*}|C)) \right]$$
$$= \sum_{C} p(C) \sum_{X_{i^*}} \log \left(\frac{p(X_{i^*})}{p(X_{i^*}|C)} \right) - (p(X_{i^*}) - p(X_{i^*}|C))\,.$$

Complexité $O(J \times I \times K \times L)$ dans le pire cas.

5.4.5 Exhaustif

Génération de sous-ensemble Parcours exhaustif des 2^J sous-ensembles.

Évaluation *Wrapper.*

Détails Cet algorithme de référence essaie tous les sous-ensembles possibles et retourne celui donnant les meilleurs performances prédictives. Son coût prohibitif restreint son utilisation à de petits nombres de variables.

5.4.6 Toutes

Génération de sous-ensemble Cet algorithme de référence retourne toutes les variables.

5.5 Sélection : Résultats

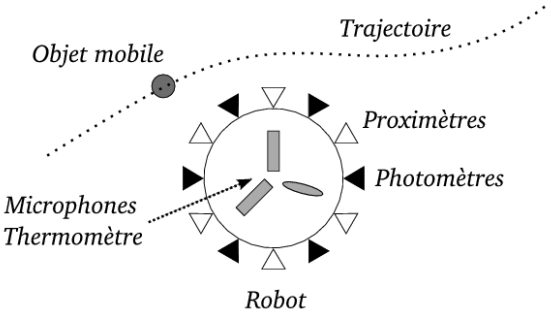
5.5.1 Contexte robotique

Notre objectif est l'automatisation de la découverte de nouveaux comportements par la recherche de régularités statistiques dans le domaine sensori-moteur. L'approche statistique peut être vue comme une fondation de l'apprentissage et est un modèle prometteur de la cognition (Bar01) et du développement mental (WMP+01).

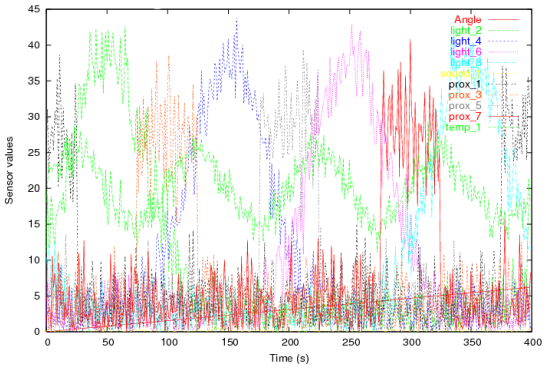
Dans notre application, nous considérons une tâche de localisation d'un objet. Le robot, muni de nombreux capteurs, est immobile et un objet se déplace dans son champ de perception. Dans nos expériences, l'objet est une balle rouge se déplaçant dans le plan horizontal. Différents capteurs du robot peuvent en détecter la position à chaque instant. Cette position, discrétisée, sera la variable de classe C. En particulier, le robot est muni d'une caméra et d'un programme de vision fondé sur la couleur qui n'est capable de détecter que des objets rouges.

Ainsi dans une première phase, la balle rouge se déplace et le robot enregistre sa position C grâce à ses facultés de vision. Dans le même temps, il enregistre les sorties de tous ses autres capteurs. Cette phase d'acquisition donne le jeu de données Δ^T, avec une variable X_i pour chacun des J capteurs et les classes (angles) correspondantes C^t.

A partir de ces données, le robot apprend lesquels de ses capteurs apportent de l'information sur C par sélection de variables. Par exemple, son capteur de température est non



(a) Représentation du monde simulé.



(b) Relevé lors du déplacement circulaire d'une source de lumière autour du robot.

FIG. 5.3 – Données simulées.

pertinent pour prédire la position de la balle, alors que ses proximètres le sont certainement. D'une certaine façon, le robot apprend à utiliser convenablement ses capteurs.

Dans un deuxième temps, au lieu de la balle rouge, c'est un cube vert qui est déplacé devant le robot. Comme le système de vision n'est plus efficace, le robot devra utiliser les capteurs sélectionnés, ainsi que les tables de probabilités apprises pour prédire la position du cube. Nous disons alors que le robot aura exploité son expérience passée pour réaliser une tâche différente. Il aura transmis ses connaissances d'un domaine sensorimoteur à un autre, en corrélant ses sensations visuelles avec ses perceptions proximétriques.

5.5.2 Données simulées

Avant de présenter l'expérience réelle, nous décrivons la comparaison des quatre algorithmes sur des données simulées. Le simulateur définit un robot immobile dans un environnement 2D. Des objets lumineux ou sonores se déplacent et sont perçus par différents capteurs. Entre autres, le robot dispose de capteurs directionnels de lumière dont la réponse décroît linéairement avec la distance de l'objet. Il possède aussi des capteurs de proximétrie directionnelle à réponse binaire et de plusieurs microphones qui nécessitent d'être associés entre eux pour estimer la position de la source. Nous avons ajoutés un capteur de "température" qui évolue indépendamment des objets ; un capteur de charge de batterie décroissant avec le temps et des capteurs parasites totalement bruités. Tous les capteurs sont artificiellement bruités. La figure 5.3 présente ce simulateur et un exemple de relevés au cours du temps, lors du passage d'un objet.

Expériences et Résultats

Nous décrivons successivement huit expériences de complexité croissante et leurs résultats pour étudier les méthodes de sélection. Nos critères de comparaison sont :

– le nombre de variables retenues,

- les performances prédictives sur un jeu de données de test non utilisées pour l'apprentissage,
- le temps de calcul sur un ordinateur standard,
- la pertinence des variables retenues (par exemple les micros pour une source sonore),
- le nombre de paramètres à fixer de l'algorithme de sélection.

Les résultats bruts sont présentés dans la table 5.5.2. Trois critères numériques sont reportés, le nombre de variables I, le taux de bonne prédiction Tx et le temps de calcul t en secondes. Interprétons ces résultats.

Expérience E1

Expérience Capteurs : deux copies de l'angle C. Source lumineuse.
But : tester l'élimination de variables trivialement redondantes, car un seule variable est suffisante.

Résultats Les algorithmes ont bien éliminé la variable redondante, sauf *WrappGA* et *FiltGA* qui n'ont pas de préférence pour les petits I.

Expérience E2

Expérience Capteurs : un copie de C et 3 bruits. Source lumineuse.
But : tester l'élimination de variables indépendantes.

Résultats Même conclusion.

Expérience E3

Expérience Capteurs : 8 proximètres répartis autour du robot et 3 bruits. Source lumineuse. But : déterminer si les algorithmes trouvent que les proximètres sont porteurs de plus d'information sur C que le bruit.

Résultats Les algorithmes ont tous éliminé les bruits, sauf *WrappGA*. *FiltCIF* n'a pas une bonne performance prédictive, car il lui est imposé de ne garder que 5 capteurs, ce qui est insuffisant dans ce cas.

Expérience E4

Expérience Capteurs : 8 proximètres et leur copie. Source lumineuse.
But : étudier l'élimination de redondances lorsque qu'un seul capteur n'est pas suffisant pour prédire C.

Résultats Nous notons d'abord que les performances sont meilleures avec un sous-ensemble de variables : les variables redondantes peuvent dégrader l'efficacité du classificateur naïf. Ce fait peut paraître surprenant mais il s'explique par deux raisons. D'abord, comme la validation se fait sur un jeu de données différent de la sélection, il est possible que trop optimiser le choix des variables sur le premier jeu de données ne soit pas la meilleure chose à faire (*overfitting*). Ensuite, en raison de la discrétisation des variables ($L = 10$), le modèle capteur n'est pas parfaitement adapté aux données et

le classificateur n'est pas optimal. Concernant les différents algorithmes, nous constatons que *WrappGA* a trouvé un sous-ensemble de 12 variables très performantes : les 8 proximètres et 4 de leur clone. Ainsi, utiliser deux fois certaines variables, c'est-à-dire élever au carré leur contribution dans l'équation de prédiction, peut améliorer les performances. Par exemple, considérons un ensemble de 3 variables avec une redondance $\{X, Y, X\}$. La prédiction calculera $[p(X|C)]^2\, p(Y|C)$ qui est une distribution différente de $p(X|C)\, p(Y|C)$. En inspectant les ensembles sélectionnés dans cette expérience, nous constatons en effet que ces 4 variables additionnelles correspondent à la zone angulaire où nous avons la majorité des données.

Expérience E5

Expérience Capteurs : 8 proximètres et 8 photomètres placés aux mêmes endroits sur le robot. Source lumineuse. But : étudier l'élimination de redondances plus fines. Il y a ici deux fois l'information sous deux formes différentes, car C est prédictible par proximétrie et par photométrie.

Résultats Dans cette expérience, le meilleur sous-ensemble trouvé par *WrappGA* et *Exhaustif* comprend tous les proximètres et la majorité de photomètres ; à l'exception de ceux placés le plus loin de la source. Encore une fois, les meilleures performances ne sont pas atteintes avec le jeu complet de capteurs. Les trois autres algorithmes, n'ont retenu que 8 capteurs, des proximètres et des photomètres, de façon à couvrir le champ angulaire le plus représenté dans les données d'apprentissage. Leur performance finale est inférieure, mais leur temps de calcul est plus raisonnable.

Expérience E6

Expérience Capteurs : 8 proximètres, 8 photomètres, 2 micros, 3 bruits, 1 témoin de la charge de la batterie et 1 capteur de température. Source lumineuse.
But : étudier les algorithmes dans un contexte plus réaliste, avec 23 capteurs.

Résultats Cette expérience avec $J = 23$ capteurs, pour laquelle la recherche exhaustive n'est déjà plus réalisable, nous pousse aux mêmes conclusions. Le meilleur sous-ensemble, trouvé par *WrappGA* et *FiltCIF*, comprend une combinaison de proximètres et de photomètres. Les performances sont meilleures qu'avec tous les capteurs et meilleures qu'avec un sous-ensemble plus petit (*FiltGA* et *FiltFA*). Le temps de calcul de *FiltCIF* est le plus faible.

Expérience E7

Expérience Capteurs : deux micros formant un angle de $2\pi/3$ et 3 bruits. Source sonore. But : étudier la sélection lorsqu'une combinaison de capteurs est systématiquement nécessaire.

Résultats Pour cette expérience, seuls les deux micros donnent de l'information sur C et nous constatons que les algorithmes ont bien détecté cette dépendance. Notons que *WrappGA* a conservé des variables inutiles, mais, comme elles ne font pas baisser la

fitness, il n'avait pas de raison de les supprimer. En revanche, *FiltGA* n'a pas trouvé les deux micros.

Expérience E8

Expérience Capteurs : deux micros et 17 autres capteurs de toute sorte. Source : sonore. But : étudier si la sélection saura retrouver les micros au milieu des 19 capteurs. De plus, nous changeons la trajectoire de la source pour simuler celle de notre expérimentation robotique : l'objet fait plusieurs allers-retours devant le robot.

Résultats E8 est similaire à E7, sauf que de nombreux autres capteurs rendent la recherche plus difficile et nous constatons que *WrappGA* a conservé 6 capteurs, dont 4 proximètres corrélés avec C mais redondants. *FiltGA* a échoué, mais *FiltFA* s'est encore bien comporté. *FiltCIF* a gardé de nombreux proximètres car I est une de ses paramètres et est fixé à $J/2$.

Ainsi nous constatons que le *wrapper* avec une recherche génétique trouve de bons sous-ensembles. Cependant, il a tendance à garder un peu trop de variables et surtout, il est très lent. Le filtre à algorithme génétiques (*FiltGA*) est un peu plus rapide et mais ne trouve que rarement les bonnes variables. *FiltFA* présente un bon compromis entre temps de calcul, performances et nombre de variables conservées. *FiltCIF* est aussi intéressant, mais son problème est que I est fixé à l'avance. Nous avons essayé d'imposer différentes valeurs de I pour *FiltCIF* et dès que cette valeur est supérieure aux nombres de capteurs suffisants, les performances de *FiltCIF* sont meilleures que celles de *WrappGA* et *FiltFA*.

De façon générale, nous devons noter qu'un sous-ensemble donne très souvent de meilleures performances que le jeu complet de capteurs, en raison de la non optimalité du prédicteur. De plus, même lorsque la recherche exhaustive est possible, il n'est pas garanti que le meilleur sous-ensemble sur le jeu d'apprentissage soit aussi le meilleur pour le jeu de données de test. Dans certains cas, nous constatons une forme de sur-apprentissage. Pour essayer de lutter contre ce problème, il serait intéressant de marginaliser sur tous les sous-ensembles, mais cette intégration n'est pas réalisable en pratique.

Le tableau ci-dessous résume les points forts et points faibles des quatre algorithmes.

5.5.3 Données réelles

Pour réaliser l'expérience de changement de modalité, nous utilisons le robot BIBA présenté par la figure 5.4(a). Ce robot d'une taille moyenne (1/0.5/0.5 m) est équipé de nombreux capteurs :

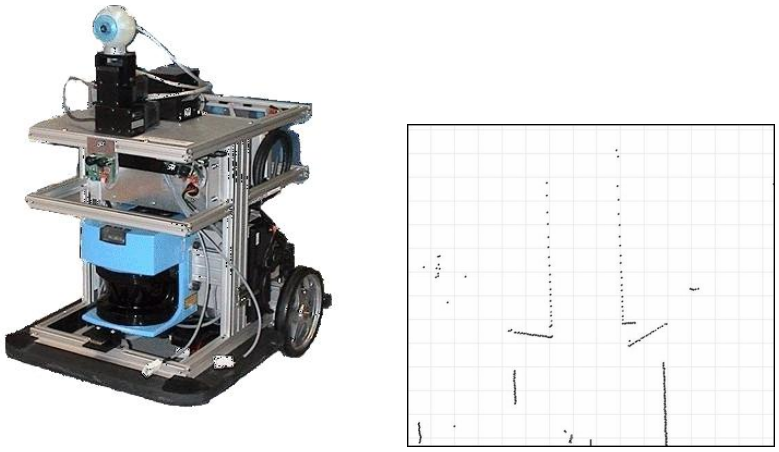
- une caméra pouvant tourner sur deux axes,
- un télémètre laser de type SICK (LMS 200 fig.5.4(b)). Ce capteur délivre la distance des plus proches objets sur un demi-plan horizontal. Il produit une mesure tous les demi degrés. Nous considérons chacun des 360 rayons laser comme une capteur indépendant,
- 3 odomètres,
- 4 détecteurs de contact (*bumpers*),

Exp.		*WrappGA*	*FiltGA*	*FiltFA*	*Exhaustif*	*FiltCIF*	*Toutes*
E1	I	2	1	2	1	1	2
	Tx	1	1	1	1	1	1
	t	2.7	0.31	0.0028	0.27	0.0029	0.0014
E2	I	3	1	1	1	2	4
	Tx	1	1	1	1	1	1
	t	3.9	0.18	0.0035	2.05	0.0031	0.0024
E3	I	11	8	8	8	5	11
	Tx	0.79	0.79	0.79	0.79	0.62	0.79
	t	13	0.42	0.14	664	0.0089	0.006
E4	I	12	15	16	13	8	16
	Tx	0.79	0.78	0.77	0,74	0.71	0.77
	t	13	0.49	0.72	15 000	0.018	0.01
E5	I	14	8	8	14	8	16
	Tx	0.93	0.88	0.9	0.93	0.9	0.92
	t	14	0.33	0.34	17 000	0.015	0.0088
E6	I	11	9	9		11	23
	Tx	0.93	0.86	0.9		0.93	0.89
	t	15	0.39	0.55		0.024	0.012
E7	I	4	2	2		2	5
	Tx	0.94	0.68	0.94		0.94	0.94
	t	4.8	0.18	0.0077		0.0036	0.003
E8	I	6	2	2		9	19
	Tx	0.81	0.37	0.83		0.78	0.74
	t	7.5	0.25	0.028		0.018	0.01

TAB. 5.1 – Comparaison d'algorithmes de sélection avec données simulées. Les lignes I représentent le nombre de variables conservées, Tx le taux de reconnaissance de la validation croisée et t le temps de calcul de l'algorithme de sélection.

	WrappGA	*FiltGA*	*FiltFA*	*FiltCIF*
Temps de calcul	- - -	- -	+ +	+ + +
Performances prédictives	+ +	- - -	+ +	+ +
Nombre de capteurs conservés	-	+	+ +	paramètre
Paramètres à fixer	- -	- -	+ +	- - -

TAB. 5.2 – Points forts et faiblesses des différents algorithmes.



(a) (b)

FIG. 5.4 – Le robot BIBA et une représentation des données de son télémètre laser lorsqu'il fait face à un couloir.

- 8 proximètres à ultrasons,
- 15 proximètres à rayons infrarouge,
- une horloge,
- et une jauge de batterie.

Nous réalisons quatre expériences mettant en œuvre la sélection de variables.

Expérience E1 Le robot enregistre les données de ses capteurs lorsqu'une balle rouge est déplacée dans son champ de vision. Son programme de vision asservit l'orientation de la caméra dans la direction de la balle. Cette position est la variable C discrétisée en 6 valeurs. Les autres capteurs sont discrétisés sur 20 valeurs. La sélection de variable est opérée avec ces données. Puis, dans un deuxième temps, la caméra est masquée et la balle rouge est de nouveau déplacée devant le robot. Le prédicteur naïf estime C grâce aux capteurs sélectionnés et asservit la direction de la caméra sur cette prédiction. Alors le robot continue de fixer la balle, bien que la caméra ne soit plus opérationnelle.

Expérience E2 Cette expérience est similaire à E1, sauf que dans la deuxième phase, ce n'est plus la balle rouge, mais un cube de bois vert qui se déplace.

Expérience E3 E3 est identique à E1, sauf que nous n'utilisons pas les capteurs lasers.

Expérience E4 Similaire à E2 sans les lasers.

Les résultats numériques sont présentés dans le tableau 5.5.3 et nous en tirons les conclusions suivantes.

Expérience E1 Nous constatons tout d'abord que la sélection de variables améliore grandement le taux de bonne prédiction. *FiltFA* a trouvé que 8 rayons lasers suffisaient pour prédire C. En effet le capteur laser est très performant et peu bruité. De plus,

Expérience		*WrappGA*	*FiltGA*	*FiltFA*	*FiltCIF*	*Toutes*
E1	Nombre de capteurs	196	118	8	8	392
	Taux de prédiction	0.143	0.143	0.85	0.55	0,143
	Temps de calcul	110.5	8.61	3.98	0.72	0,07
E2	Nombre de capteurs	183	112	8	8	392
	Taux de prédiction	0.15	0.143	0.88	0.43	0,143
	Temps de calcul	100.4	8.94	3,92	0,726	0,07
E3	Nombre de capteurs	14	4	7	8	32
	Taux de prédiction	0.66	0.2	0.55	0.51	0,55
	Temps de calcul	2.73	0.15	0.01	0.006	0,002
E4	Nombre de capteurs	20	5	7	8	32
	Taux de prédiction	0.40	0.23	0.36	0.38	0,30
	Temps de calcul	2.97	0.16	0.01	0,005	0,002

TAB. 5.3 – Résultats des expériences robotiques.

une discrétisation de C sur 6 valeurs facilite le problème. *FiltCIF* qui a conservé 6 lasers et 2 capteurs à ultrasons, donne des performances moins élevées. Nous constatons aussi que les algorithmes génétiques n'ont pas trouvé de bon optimum, bien qu'ayant nécessité un long temps de calcul. Nous pensons que cela est du à la fonction de *fitness* qui présente un maximum global très piqué : il est difficile de le trouver par mutation et reproduction avec le codage binaire utilisé. De plus, l'espace de recherche avec 392 capteurs est très vaste.

Expérience E2 Avec le cube vert les résultats sont similaires : il est aisé de le détecter avec les 8 lasers, mais c'est un peu plus dur avec les ultrasons.

Expérience E3 et E4 Comme la présence du laser SICK rend la détection assez facile, nous avons reproduit les expériences sans ce capteur. Alors nous n'avons plus que 32 variables et *WrappGA* trouve un bien meilleur sous-ensemble. De façon générale, les performances sont moins bonnes qu'avec le SICK, mais le robot arrive à retrouver le cube dans 40% des cas et lorsqu'il se trompe, ce n'est souvent que d'un domaine angulaire. Les capteurs sélectionnés sont principalement les proximètres à ultrasons, bien que *WrappGA* conserve toujours des capteurs inutiles.

Finalement, nous constatons que la sélection de variables permet une amélioration des prédictions dans un contexte de fusion simple de capteurs. *FiltFA* est une bon candidat lorsque de nombreuses variables sont présentes, car le mécanisme de *forward addition* permet rapidement de ne garder que des variables intéressantes et évite ainsi des calculs coûteux.

5.6 Conclusion

5.6.1 Contributions

Dans ce chapitre, nous avons proposé et comparé plusieurs algorithmes de sélection de variables dans le but d'améliorer les performances d'un prédicteur. Nous avons de plus redérivé une version bayésienne du classificateur bayésien naïf.

Nous avons constaté que lorsque le modèle probabiliste du classificateur est éloigné de la réalité, la sélection de variables permettait de nettement améliorer les performances prédictives finales. Concernant les différents algorithmes, nous avons constaté que les filtres donnaient de meilleurs résultats expérimentaux que les *wrappers*, en terme de temps de calcul, mais aussi en regard des performance prédictives. Nous expliquons ce phénomène par un bon choix d'heuristiques de filtre (formule de Ghiselli et information mutuelle empirique) et aussi par le fait que la *forward addition* permette une exploration parcimonieuse de l'espace des sous-ensembles.

Nous avons utilisé la sélection de variables dans un contexte de robotique autonome. Le robot est capable de se passer d'un de ses capteurs après avoir découvert des régularités statistiques dans son domaine sensori-moteur. Nous avons testé cette idée dans le cadre d'une expérience réelle.

5.6.2 Perspectives

Il serait intéressant de poursuivre l'étude de l'algorithme génétique afin d'optimiser ses paramètres pour le rendre plus rapide et plus sélectif. Il serait possible de modifier sa fonction d'adaptation en ajoutant une pénalité dépendant du nombre de variables retenues afin de le forcer à éliminer les variables inutiles. Nous pourrions aussi complexifier la formule de Ghiselli afin de détecter les dépendances conditionnelles d'ordre supérieure à deux. La dernière voie à explorer est celle d'une modification du critère d'arrêt de l'algorithme *FiltCIF*. Avec un critère automatisé, *FiltCIF* deviendrait un excellent candidat pour une application embarquée.

Dans l'esprit de la robotique autonome, nous n'avons réalisé qu'un premier pas vers la découverte entièrement automatique de nouveaux comportements. En effet l'enchaînement des phases de recherche de régularité statistique et des phases d'exploitation est, dans notre travail, décidé par le programmeur. Il serait intéressant de réfléchir à intégrer ces phases au sein d'une architecture de plus haut niveau. Ce programme pourrait décider des moments pendant lesquels le robot peut (ou doit) se consacrer à de l'apprentissage et des moments où il doit se consacrer à sa tâche principale.

Chapitre 6

Création de variables

6.1 Introduction

Dans le chapitre précédent, nous avons étudié différentes façons de sélectionner des variables pertinentes à partir d'un ensemble de variables connu à l'avance : l'ensemble des capteurs d'un robot. Cette sélection nous a permis de définir une partie du modèle probabiliste utilisé ensuite par le robot.

Dans ce chapitre nous nous intéressons à la création automatique d'une variable cachée permettant une meilleure modélisation de l'environnement. Pour cela, nous proposons une approche incrémentale dans un cadre temporel. Le robot reçoit en permanence de l'information sur son environnement à travers ses capteurs $\mathbf{X}$ et son but est de structurer ces informations en y recherchant une régularité sous-jacente. Le robot découvre alors une explication à une série de données temporelles en créant une variable cachée simplifiant l'expression des dépendances entre les variables $\mathbf{X}$.

Nous présentons d'abord les hypothèses cadrant ce travail, puis le modèle probabiliste associé. Nous décrivons ensuite l'algorithme d'apprentissage incrémental utilisé et reportons des résultats sur des données synthétiques.

6.2 Le problème de la création de variables

6.2.1 Cadre

Nous considérons dans ce chapitre un agent évoluant dans un environnement mal connu et incertain, comme par exemple un robot mobile autonome. Pour interagir d'une façon efficace avec cet environnement et pour accomplir ses tâches, le robot doit prendre en compte l'incomplétude et l'incertitude de sa représentation. Bessière *et al.* (LBDM03) ont montré que dans ce contexte, l'utilisation de probabilités subjectives comme représentation de la connaissance présentait de nombreux intérêts. L'inférence bayésienne est bien adaptée à cette problématique (DBM05 ; TBF05).

Dans ce cadre, c'est habituellement le roboticien qui définit un modèle probabiliste de

l'environnement, des capacités sensorimotrices du robot et de sa tâche. Pour définir ce modèle, le premier travail du roboticien consiste à choisir des variables de la densité de probabilité jointe.

Cet ensemble de variables devra d'abord contenir les variables sensorimotrices : une variable pour chaque capteur utile et une variable pour chaque commande motrice. Les variables capteurs seront observées et le résultat de l'inférence, c'est-à-dire du raisonnement du robot, sera un *a posteriori* sur les variables motrices. Le robot utilisera cet *a posteriori*, en combinaison avec une fonction de coût, pour prendre des décisions et exécuter certaines actions, comme par exemple se déplacer à une certaine vitesse dans une certaine direction. En plus de ces variables sensorimotrices, le roboticien ajoute souvent des variables non observées permettant de modéliser l'environnement. Un exemple de telles variables peut être une grille d'occupation de l'espace comme dans le filtre d'occupation bayésien (B.O.F. (CPL+06 ; TMC+07)).

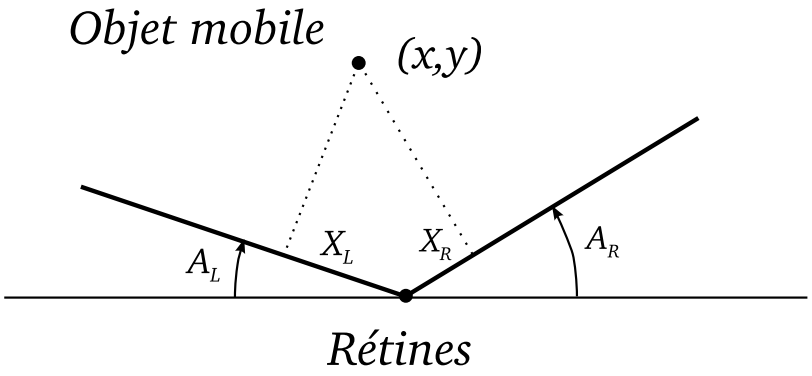
Ainsi le roboticien transcrit une partie de ses *a priori*, de sa propre connaissance du monde en un ensemble de variables avec leur structure de dépendance. Dans un cadre d'apprentissage plus général, il serait intéressant que ce soit le robot et non le roboticien, qui trouve de lui même ces variables cachées avec leur structure. Ce chapitre propose un premier pas dans cette direction.

Pour cela, nous nous plaçons dans le cadre suivant. Un robot est présent dans un environnement dynamique et enregistre les valeurs mesurées par ses capteurs $\{\mathbf{X}^t\}_{t\in[1,T]}$ à chaque instant t. Nous supposons que ces capteurs mesurent certaines caractéristiques d'un phénomène Φ ayant lieu dans l'environnement. Supposer l'existence d'un phénomène ϕ revient à dire que le monde n'est pas complètement aléatoire et incohérent : il présente certaines régularités. En d'autres termes, les mesures $\mathbf{x}^t = \{x_i^t\}_{i\in[1,I]}$ ne sont pas indépendantes : elles sont liées par l'existence même de Φ. Nous supposons de plus que, contrairement à l'exemple de la pièce où Φ était immuable dans le temps (le biais θ ne changeait pas), désormais le phénomène Φ a sa dynamique propre. Ainsi il y a une relation entre Φ au temps t et Φ au temps $t+1$. Cette relation implique une dépendance entre $\mathbf{X}^t$ et $\mathbf{X}^{t+1}$.

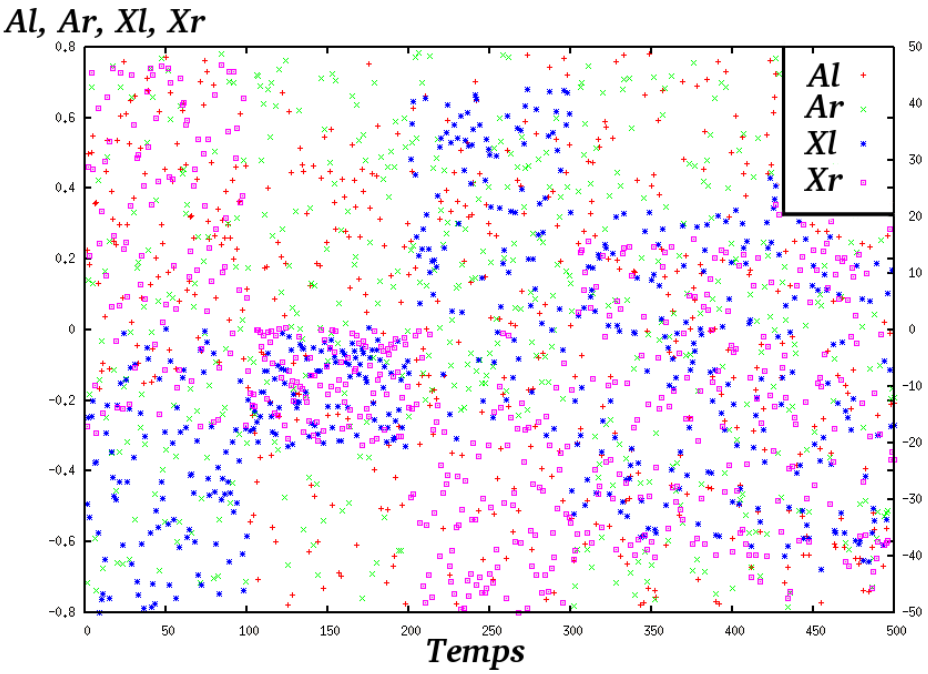
Nous ajoutons une hypothèse importante : la fréquence de variation de Φ est plus faible que celle des lectures des capteurs. Ainsi pour chaque état successif de Φ, nous avons plusieurs relevés de capteurs consécutifs. Nous pensons que cette hypothèse de *stabilité temporelle du phénomène* est nécessaire pour que le robot puisse découvrir des régularités dans les données récoltées. Il pourra alors se construire un modèle pertinent de Φ et de ses propres capacités sensorimotrices.

6.2.2 Problème étudié

Pour concrétiser notre approche, nous considérons un problème simple de vision synthétique. Notre robot n'est constitué que par deux rétines, gauche et droite (L at R) et est plongé dans un environnement 2D. La position de chaque rétine est commandée par un angle (A_L et A_R). Le phénomène Φ que nous étudions est la position d'un objet ponctuel dans le champ visuel du robot. L'objet se projette sur les rétines et les coordonnées de



(a) Représentation du monde simulé.



(b) Les 4 variables en fonction de t, avec une couleur pour chacune des variables.

FIG. 6.1 – La figure 6.1(a) présente un robot constitué de deux rétines dont les positions angulaires sont notées A_L et A_R. L'image d'un objet mobile se projette orthogonalement en X_L et X_R sur les rétines. La figure 6.1(b) montre les données brutes au cours du temps, pour cinq positions différentes. Tous les 100 pas de temps, la vraie position est modifiée et nous constatons bien que l'allure des densités des quatre variables change.

ses deux projections sont X_L et X_R. Les projections sont orthogonales. Nous avons ainsi 4 variables sensorimotrices, lues à chaque instant $\mathbf{X}^t = \{A_L^t, A_R^t, X_L^t, X_R^t\}$. La figure 6.1(a) représente le robot avec ses variables.

La position de l'objet $(x, y)^t$ est inconnue du robot. En fait, le robot ne sait même pas qu'il existe un objet, il ne connaît pas la sémantique des variables qu'il observe. Comme nous l'avons précisé, le déplacement de l'objet est lent devant la vitesse d'acquisition des capteurs. Nous supposons que l'objet peut se trouver dans N positions différentes et qu'il reste suffisamment longtemps dans chacune de ces positions. Lorsque l'objet est dans l'une de ces positions, le robot réalise plusieurs lectures de ses variables, pour différents angles de ses rétines. Par une analogie biologique, nous pouvons dire que le robot effectue des saccades oculaires, il regarde dans différentes directions et enregistre les images reçues. Les données brutes sont présentées par la figure 6.1(b) : à chaque instant t, le robot positionne aléatoirement ses rétines (A_L^t, A_R^t) et lit les projections correspondantes (X_L^t, X_R^t). Ces quatre valeurs sont représentées sur la figure 6.1(b) par quatre points de couleurs différentes. Tous les cent pas de temps, l'objet change de position et nous constatons que les données se présentent différemment. Nous voulons que le robot retrouve la sémantique de la figure 6.1(a) à partir uniquement des données brutes de la figure 6.1(b) et de l'hypothèse de variation lente.

Pour une position donnée de l'objet, il y aura une relation directe entre A_L^t et X_L^t d'une part et entre A_R^t et X_R^t d'autre part. Pour une autre position, il y aura toujours des relations entre ces couples, mais elles seront différentes. L'existence de telles relations est devinable sur la figure 6.1(b), il y a un changement de régime pour chaque nouvelle position.

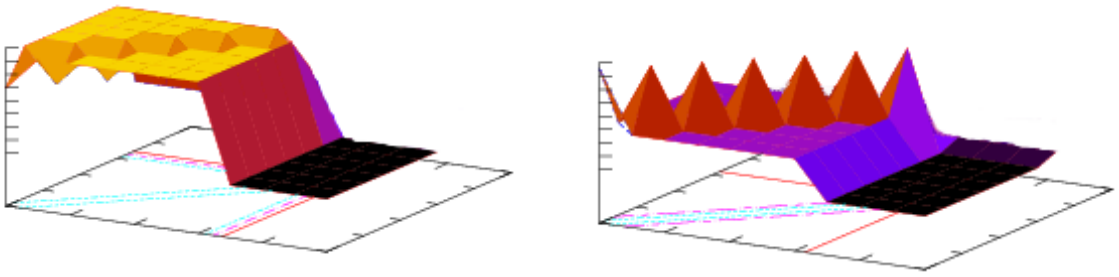
Nous voyons ainsi qu'il existe en réalité une structure de dépendance entre les quatre variables. Sachant la position, A_L^t et X_L^t sont liées et sont indépendantes du couple (A_R^t, X_R^t). C'est ce genre de dépendance que nous voulons découvrir. Pour cela il faudra que le robot crée une variable supplémentaire représentant la position de l'objet, car sans elle, les quatre variables ne présentent pas de structure de dépendance simple. Le but est donc de créer une variable C^t représentant la position de l'objet et conjointement, de découvrir la bonne structure de dépendance conditionnellement à C^t. Les différentes valeurs de C^t devront être à l'image des différentes positions de l'objet, qui ne sont pas connues à l'avance.

Nous présentons maintenant le cadre probabiliste dans lequel cet apprentissage sera réalisé.

6.3 Création : Modèle

6.3.1 Structure temporelle

Les variables de notre modèle sont donc les variables sensorimotrices $\mathbf{X}$ accompagnées d'une variable discrète C dont la cardinalité n'est pas connue à l'avance. De plus, toutes ces variables sont dupliquées pour chaque instant t. Pour modéliser les dépendances temporelles entre variables, nous nous restreignons à un cadre de Markov 1 : les variables ne dépendent



(a) Inertie $i = 0.1$. (b) Inertie $i = 0.3$.

FIG. 6.2 – La matrice des probabilités de transition $p(C^{t+1} \mid C^t)$ est supposée uniforme avec une diagonale dominante (à droite) pour représenter la lenteur de la variation du phénomène. cette matrice est spécifiée par un seul paramètre : $i = p(C^{t+1} = c^t \mid C^t = c^t)$

du passé qu'à travers leurs valeurs à l'instant précédent :

$$p(\mathbf{X}^t, C^t \mid \mathbf{X}^0, C^0, \ldots, \mathbf{X}^{t-1}, C^{t-1}) = p(\mathbf{X}^t, C^t \mid \mathbf{X}^{t-1}, C^{t-1}). \tag{6.1}$$

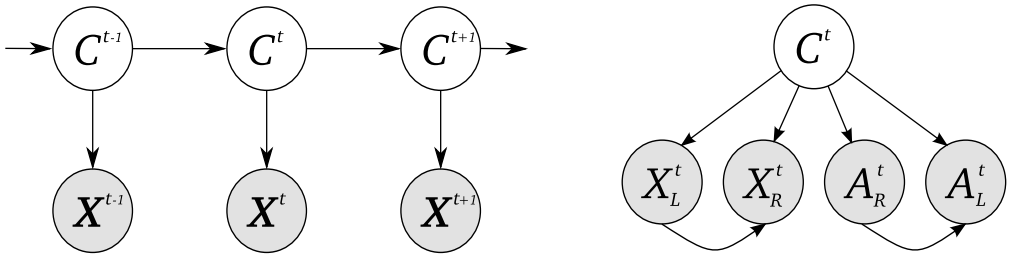
De plus, à chaque instant t, nous considérons que C^t transporte toute l'information sur le phénomène Φ, en particulier sur sa dynamique. En d'autres termes, les variables sensorimotrices n'ont pas de dynamique propre, par exemple il n'y a pas d'inertie dans le mouvement des rétines. Alors la dépendance de $\mathbf{X}^t$ au passé ne passe que par la variable C^t et nous avons une structure similaire à celle des modèles de Markov cachés comme présenté par la figure 6.3(a).

Comme les variables C^t sont discrètes, le modèle de transition $p(C^{t+1} \mid C^t)$ est une table de probabilité. Pour transcrire notre hypothèse de variation lente du phénomène, nous pouvons préciser la forme de cette table : ce sera une matrice carrée à diagonale dominante paramétrée par l'inertie i : $p(C^{t+1} = c^{t+1} \mid C^t = c^t) = i$ si $c^{t+1} = c^t$ et est uniforme sinon. En choisissant une grande valeur de i, nous modélisons une grande inertie dans le phénomène, une grande lenteur dans la changement de position de l'objet (fig. 6.2).

Ainsi la dynamique du phénomène est modélisée de la façon suivante : si i est assez grand, il est très probable que le phénomène ne change pas entre t et $t+1$. Par contre, s'il change, alors il peut aller dans n'importe quelle autre de ses configurations, avec la même probabilité.

6.3.2 Structures locale

Pour ce qui est de la structure de dépendance à chaque instant t, nous considérons un modèle "bayésien naïf augmenté". Il présente une structure d'arbre comme le modèle capteur, mais augmenté de relations possibles entre les différents $\mathbf{X}^t$. La figure 6.3(b) présente un exemple d'une telle structure. Dans cet exemple nous avons exprimé deux dépendances conditionnelles : au contraire du modèle capteur ou tous les $\mathbf{X}^t$ sont indépendants conditionnellement à C^t, ici les variables du couple (A_L^t, A_R^t) sont dépendantes et celles



(a) Dépendance temporelle. (b) Exemple de dépendance locale.

FIG. 6.3 – La structure de dépendance temporelle est similaire à celle d'un HMM d'ordre 1. À chaque instant t, les dépendances possibles entre les variables $\mathbf{X}^t$ sont de type naïf augmenté.

de (X_L^t, X_R^t) le sont aussi. Dans le cas général de I variables, notre algorithme travaillera dans cet espace de structures "naïf augmenté", mais en restreignant le nombre maximal de variables reliées à M ($M = 3$ dans notre application).

Cette restriction permet de considérer des modèles plus simples, avec moins de paramètres. En effet, si nous ne restreignons pas M, le modèle le plus flexible serait celui qui est complètement connecté. Il nécessiterait un grand nombre de paramètres, donc un grand nombre de données et il ne permettrait pas de régulariser notre apprentissage (*over-fitting*). De plus, ce modèle ne rendrait pas explicites les relations d'indépendances présentes dans les vraies données. Notre but est de trouver un modèle simple mais adapté aux données.

Cette restriction du nombre de variables peut aussi être mise en parallèle avec une idée récemment employée pour construire des algorithmes d'inférence déterministe approchés. Nous avons décrit et utilisé dans la partie précédente un EP et un *Mean-Field* EM. Dans les deux cas nous approximions une distribution $p(\mathbf{X})$ par une distribution $p(\mathbf{X}) = \prod_i q_i(X_i)$ complètement factorisée. Cette idée de factorisation totale facilite les inférences, mais conduit nécessairement à une sous optimalité : q est différent de p. Pour améliorer la qualité de l'approximation, il a été proposé de ne pas factoriser complètement q, mais de grouper localement des variables. Par exemple, l'algorithme *generalized belief propagation* (YFW00) étend BP en considérant des petits clusters de variables dans le graphe de dépendances de p. La même idée à été appliquée à EP pour construire des *structured region graphs* (WMT05) permettant un compromis entre difficulté de l'inférence et précision de l'approximation. Nous retrouvons ici cette idée de chercher une distribution faiblement connectée.

Nous considérons ainsi les densités jointes factorisées de la façon suivante, le modèle

d'évolution dynamique n'étant pas représenté :

$$p(\mathbf{X}^t, C^t) = \prod_k p(\mathbf{Y}_k^t \mid C^t)\, p(C^t) \tag{6.2}$$

$$\forall k \quad \mathbf{Y}_k^t \subset \mathbf{X}^t \tag{6.3}$$

$$\forall k \quad \mathrm{Card}\left(\mathbf{Y}_{\mathrm{k}}^{\mathrm{t}}\right) \leq M \tag{6.4}$$

$$\bigcup_k \mathbf{Y}_k^t = \mathbf{X}^t \tag{6.5}$$

$$\forall (k, k') \quad \mathbf{Y}_k^t \cap \mathbf{Y}_{k'}^t = \emptyset. \tag{6.6}$$

Ainsi, les $\{\mathbf{Y}_k^t\}$ forment une partition de l'ensemble des variables de $\mathbf{X}^t$, sous la contrainte que chacun de ces sous ensembles soit de cardinalité inférieure à M. Par exemple, pour 4 variables et $M = 3$, il y a 14 structures possibles :

$$\begin{aligned}
p(\mathbf{X}^t, C^t) &= p(C^t)\, p(A_L^t \mid C^t)\, p(Z_L^t \mid C^t)\, p(A_R^t \mid C^t)\, p(Z_R^t \mid C^t)\\
p(\mathbf{X}^t, C^t) &= p(C^t)\, p(A_L^t \mid C^t)\, p(Z_L^t, Z_R^t \mid C^t)\, p(A_R^t \mid C^t)\\
p(\mathbf{X}^t, C^t) &= p(C^t)\, p(A_L^t \mid C^t)\, p(Z_L^t \mid C^t)\, p(A_R^t, Z_R^t \mid C^t)\\
p(\mathbf{X}^t, C^t) &= p(C^t)\, p(A_L^t \mid C^t)\, p(Z_L^t, A_R^t \mid C^t)\, p(Z_R^t \mid C^t)\\
p(\mathbf{X}^t, C^t) &= p(C^t)\, p(A_L^t, Z_R^t \mid C^t)\, p(Z_L^t \mid C^t)\, p(A_R^t \mid C^t)\\
p(\mathbf{X}^t, C^t) &= p(C^t)\, p(A_L^t, A_R^t \mid C^t)\, p(Z_L^t \mid C^t)\, p(Z_R^t \mid C^t)\\
p(\mathbf{X}^t, C^t) &= p(C^t)\, p(A_L^t, Z_L^t \mid C^t)\, p(A_R^t \mid C^t)\, p(Z_R^t \mid C^t)\\
p(\mathbf{X}^t, C^t) &= p(C^t)\, p(A_L^t, Z_L^t, Z_R^t \mid C^t)\, p(A_R^t \mid C^t)\\
p(\mathbf{X}^t, C^t) &= p(C^t)\, p(A_L^t, Z_L^t \mid C^t)\, p(A_R^t, Z_R^t \mid C^t)\\
p(\mathbf{X}^t, C^t) &= p(C^t)\, p(A_L^t, Z_R^t \mid C^t)\, p(Z_L^t, A_R^t \mid C^t)\\
p(\mathbf{X}^t, C^t) &= p(C^t)\, p(A_L^t, A_R^t \mid C^t)\, p(Z_L^t, Z_R^t \mid C^t)\\
p(\mathbf{X}^t, C^t) &= p(C^t)\, p(A_L^t \mid C^t)\, p(Z_L^t, A_R^t, Z_R^t \mid C^t)
\end{aligned}$$

$$\begin{aligned}
p(\mathbf{X}^t, C^t) &= p(C^t)\, p(A_L^t, A_R^t, Z_R^t \mid C^t)\, p(Z_L^t \mid C^t)\\
p(\mathbf{X}^t, C^t) &= p(C^t)\, p(A_L^t, Z_L^t, A_R^t \mid C^t)\, p(Z_R^t \mid C^t).
\end{aligned}$$

Ces décompositions peuvent être trouvées en générant toutes les partitions de $\mathbf{X}^t$ par l'algorithme des *restricted growing strings* (Knu05), puis en filtrant celles ne vérifiant pas la contrainte $\forall k \;\; \mathrm{Card}\left(\mathbf{Y}_{\mathrm{k}}^{\mathrm{t}}\right) \leq M$.

La dernière hypothèse à préciser est celle de *stationnarité* : les distributions de probabilités ne dépendent pas du temps t. Cela signifie d'abord que la décomposition de $p(\mathbf{X}^t, C^t)$ est la même pour tout t. D'autre part les densités conditionnelles $p(\mathbf{Y}_k^t \mid C^t)$ sont elles aussi invariantes dans le temps. De même que dans le chapitre précédent, nous pouvons les paramétrer par une même variable $\boldsymbol{\Theta}$ telle que $\forall t \;\; p(\mathbf{Y}_k^t \mid C^t, \boldsymbol{\Theta}) = \theta_{C,\mathbf{Y}_k}$. Cette hypothèse nous permettra d'utiliser les données de différents instants pour estimer $\boldsymbol{\Theta}$.

6.4 Création : Apprentissage

Nous décrivons maintenant notre algorithme d'apprentissage. Cet algorithme prend en entrée un jeu de données $\Delta^T := \{\mathbf{X}^0, \dots, \mathbf{X}^T\}$ comme celui de la figure 6.1(b). Ce jeu de données est construit de la manière suivante : l'objet est placé dans N positions successives. Pour chacune de ces positions, O données sont générées (typiquement $N = 5$ et $O = 100$). Pour générer une donnée $\mathbf{X}^t$ à l'instant t, les angles des rétines sont tirées aléatoirement et les projections correspondantes sont calculées. Les valeurs de C^t ne sont donc jamais observées.

L'algorithme apprend alors trois quantités différentes. Pour chaque structure de dépendance locale (définie par la partition $\{\mathbf{Y}_k\}$), il apprend les tables de probabilités $p(\mathbf{Y}_k^t \,|\, C^t)$ pour les différentes valeurs de C^t. En parallèle il doit justement apprendre quelle sont ces différentes valeurs de C^t. L'idée est de débuter avec une variable C^t n'ayant qu'une valeur possible c_0, puis de créer des nouvelles valeurs à chaque fois que le besoin s'en fait sentir. Le troisième niveau d'apprentissage est celui de la structure de dépendance : l'algorithme réalise les deux premiers niveaux d'apprentissage pour les différentes structures, puis compare leurs résultats pour trouver la meilleure d'entre elles.

6.4.1 Apprentissage des tables

Comme toutes les variables sont discrétisées en R valeurs, les probabilités conditionnelles $p(\mathbf{Y}_k^t \,|\, C^t) = \theta_{\mathbf{Y}_k, C}$ sont représentées par des tables. Pour chaque valeur de k, notons I_k le nombre de variables du vecteur $\mathbf{Y}_k$. Alors pour chaque valeur de C^t, nous devons apprendre les paramètres $\mathbf{\Theta}$ de I_k R-multinomiales, soit $(I_k)^R$ paramètres. Comme dans le chapitre précédent, nous considérons un *a priori* de Dirichlet symétrique d'hyper-paramètre α. Mais cette fois, notre algorithme n'a pas pour but immédiat de faire de la prédiction et nous ne marginalisons donc pas sur $\mathbf{\Theta}$. En revanche nous estimons $\mathbf{\Theta}$ par son MAP, en choisissant $\alpha = 1$. Alors le théorème 8 nous dit que l'estimation de maximum *a posteriori* de $\mathbf{\Theta}$ a aussi la forme d'une loi de succession de Laplace.

Théorème 8 (Estimateur MAP des paramètres d'une multinomiale)**.** *Soit X une variable discrète à I valeurs et $\Delta = \{x_1, \dots, x_N\}$ un jeu de données de N réalisations de X. Si nous notons $\mathbf{\Theta} = \{\theta_i := p(X = i)\}$ et si nous considérons un a priori de Dirichlet symétrique de paramètre α sur $\mathbf{\Theta}$, alors l'estimateur de maximum a posteriori des θ_i est :*

$$\theta_i^{MAP} = \frac{\alpha + \sum_n \delta(x_n = i)}{\alpha\, I + N}. \tag{6.7}$$

Ainsi pour estimer les probabilités θ_i, il suffit de compter le nombre de réalisations dans les données de x_i, de lui ajouter α et de renormaliser. Pour $\alpha = 1$, nous retrouvons la loi de succession de Laplace.

Démonstration. Nous voulons maximiser :

$$p(\mathbf{\Theta} \mid \Delta) = p(\Delta \mid \mathbf{\Theta})\, P(\mathbf{\Theta}) \tag{6.8}$$

$$\propto \left[\prod_n \theta_{i_n}\right] \cdot \left[\prod_i \theta_i^{\alpha}\right] \tag{6.9}$$

$$\propto \left[\prod_i \theta_i^{\sum_n \delta(i_n=i)}\right] \cdot \left[\prod_i \theta_i^{\alpha}\right] \tag{6.10}$$

$$\propto \prod_i \theta_i^{\alpha+\sum_n \delta(i_n=i)}. \tag{6.11}$$

Il faut aussi vérifier la contrainte de normalisation $\sum_i \theta_i = 1$. En maximisant le logarithme de cette expression et en introduisant un multiplicateur de Lagrange, nous obtenons pour tout i :

$$\theta_i \propto \alpha + \sum_n \delta(i_n = i). \tag{6.12}$$

Le résultat final vient alors en réintroduisant la contrainte de normalisation. □

Ainsi l'apprentissage des tables conditionnelles se fait par la construction d'histogrammes de Laplace. Cette méthode est incrémentale : au fur et à mesure de la réception de nouvelles données, nous pouvons mettre à jour l'estimation courante des tables en incrémentant au bon endroit et en renormalisant. Il faut cependant prendre garde à mémoriser le nombre de données reçues afin de renormaliser convenablement.

6.4.2 Apprentissage de la cardinalité de C^t

Principe

Nous venons de décrire la méthode employée pour estimer les tables de probabilités conditionnelles. Nous présentons maintenant une méthode originale pour découvrir la cardinalité de C^t. Notre but est que cette variable créée soit à l'image des positions (non observées) de l'objet. L'idée est de faire apparaître une nouvelle valeur de C^t à chaque fois qu'une nouvelle position est perçue. En effet, à chaque fois que l'objet change de place, disons tous les $O = 100$ pas de temps pour fixer les idées, la relation probabiliste entre les $\mathbf{X}^t$ change aussi. Notre but est donc de détecter un tel changement de régime, en repérant que les nouvelles données ne sont plus cohérentes avec ce que nous avons vu par le passé.

Algorithme

Pour cela, notre méthode "en ligne" commence à $t = 0$ par une variable C^t à une seule valeur $C^t = c_0$. Cette valeur aura pour notre algorithme une sémantique particulière : elle ne correspond pas à une position, mais elle signifie que les nouvelles données sont incohérentes avec le passé et qu'il faut donc créer une nouvelle valeur pour C^t. À chaque fois qu'une nouvelle donnée arrive, nous calculons l'*a posteriori* sur C^t. Nous en cherchons le

maximum. Si c'est c_0, alors une nous créons une nouvelle classe c_{new}. Sinon, nous incrémentons les tables de probabilités conditionnelles correspondantes. Cette procédure est définie par l'algorithme 8.

Algorithme 8 : Algorithme d'apprentissage de la variable C.

Entrées : Données $\Delta^T := \{\mathbf{X}^0, \ldots, \mathbf{X}^T\}_t$ et une partition $\{\mathbf{Y}_k\}$ de $\mathbf{X}$.
Sorties : Une variable discrète C^T avec sa cardinalité et des tables de probabilités conditionnelles.

Initialiser la variable C^0 à la valeur particulière c_0;
Définir la matrice de transition $p(C^{t+1} \mid C^t)$ avec le paramètre d'inertie i;
Initialiser les tables conditionnelles : **pour** *tout* k **faire** $p(\mathbf{Y}_k^0 \mid C^0 = c_0) \leftarrow 1$;
pour *tout* $t \in [1, T]$ **faire**
    Lire la nouvelle donnée : $\forall k \;\; \mathbf{Y}_k^t = \mathbf{y}_k^t$;
    Calculer l'*a posteriori* $p(C^t \mid \{\mathbf{X}^{0,\ldots,t}\})$;
    Trouver le c^{t^*} maximisant cet *a posteriori* ;
    si $c^{t^*} \neq c_0$ **alors**
        pour *tout* k **faire**
            Incrémenter la case $p(\mathbf{Y}_k^t = \mathbf{y}_k \mid C^t = c^{t^*})$;
        fin
    sinon
        Créer une nouvelle valeur possible c_{new} pour C^t;
        pour *tout* k **faire**
            $p(\mathbf{Y}_k^t \mid C^t = c_{\text{new}}) \leftarrow 1$;
            Incrémenter la case $p(\mathbf{Y}_k^t = \mathbf{y}_k \mid C^t = c_{\text{new}})$
        fin
        Augmenter la taille de la matrice de transition $p(C^{t+1} \mid C^t)$
    fin
fin

Une fois le jeu de données entièrement parcouru, nous avons en sortie une certain nombre de valeurs possibles pour C^T, reflétant les différentes positions de l'objet, ainsi que des tables $p(\mathbf{Y}_k^T \mid C^T)$. Ces tables peuvent alors servir à classer une nouvelle observation en calculant $p(C^{T+1} \mid \mathbf{Y}_k^{T+1})$.

Calcul de l'*a posteriori* sur C^t

Nous détaillons la phase de calcul de l'*a posteriori* sur C^t de l'algorithme 8. Notre modèle est un filtre bayésien dont la densité jointe sur toutes les variables est :

$$p(\{\mathbf{X}^{0,\ldots,T}, C^{0,\ldots,T}\}) = p(\mathbf{X}^0, C^0) \prod_{t=1}^{T} p(\mathbf{X}^t \mid C^t)\, p(C^t \mid C^{t-1}). \tag{6.13}$$

Selon le vocabulaire consacré aux filtres, nous cherchons la *prédiction* sur C^t. C'est le produit de sa *vraisemblance* à la vue des nouvelles données par l'*estimation* basée sur les

anciennes données :

$$\begin{aligned}
\text{prédiction} &:= p(C^t \mid \{\mathbf{X}^{0,\dots,t}\}) && (6.14)\\
&= p(C^t \mid \{\mathbf{X}^{0,\dots,t-1}\}, \mathbf{X}^t) && (6.15)\\
&\propto p(\mathbf{X}^t \mid C^t, \{\mathbf{X}^{0,\dots,t-1}\})\, p(C^t \mid \{\mathbf{X}^{0,\dots,t-1}\}) && (6.16)\\
&\propto p(\mathbf{X}^t \mid C^t)\, p(C^t \mid \{\mathbf{X}^{0,\dots,t-1}\}) && (6.17)\\
&\propto \text{vraisemblance} \times \text{estimation}. && (6.18)
\end{aligned}$$

L'*estimation* se calcule en marginalisant sur C^{t-1} et en utilisant les propriétés de factorisation de la jointe :

$$\begin{aligned}
\text{estimation} &:= p(C^t \mid \{\mathbf{X}^{0,\dots,t-1}\}) && (6.19)\\
&= \sum_{C^{t-1}} p(C^t, C^{t-1} \mid \{\mathbf{X}^{0,\dots,t-1}\}) && (6.20)\\
&= \sum_{C^{t-1}} p(C^{t-1} \mid \{\mathbf{X}^{0,\dots,t-1}\})\, p(C^t \mid C^{t-1}, \{\mathbf{X}^{0,\dots,t-1}\}) && (6.21)\\
&= \sum_{C^{t-1}} p(C^{t-1} \mid \{\mathbf{X}^{0,\dots,t-1}\})\, p(C^t \mid C^{t--1}) && (6.22)\\
&= \sum_{C^{t-1}} \text{ancienne prédiction} \times \text{transition}. && (6.23)
\end{aligned}$$

Nous voyons ainsi apparaître la récursivité des filtres bayésiens : le calcul de la nouvelle *prédiction* $p(C^t \mid \{\mathbf{X}^{0,\dots,t}\})$ utilise l'ancienne *prédiction* $p(C^{t-1} \mid \{\mathbf{X}^{0,\dots,t-1}\})$. En pratique il suffit de ne garder en mémoire qu'une seule prédiction sur une variable C. Après chaque nouvelle donnée elle sera mise à jour pour représenter la nouvelle *prédiction* en fonction de la *vraisemblance* et de la matrice de transition.

6.4.3 Comparaison de structures

Nous avons décrit comment l'algorithme apprenait le nombre de valeurs de C^t et les tables conditionnelles pour une structure donnée $\mathcal{M}_S$, c'est-à-dire pour une partition $\{\mathbf{Y}_k\}$ de $\mathbf{X}$ donnée. Nous pouvons réaliser cet apprentissage pour différentes structures afin de comparer les résultats pour différents modèles. Pour cela nous utilisons un critère entropique : parmi les différents modèles possibles, nous cherchons celui qui est le plus informatif, celui qui contient le plus d'information.

A la fin de chacun des apprentissages, nous obtenons des tables de probabilités représentant $p(\{\mathbf{Y}_k^T\}, C^T \mid \mathcal{M}_S)$. Imaginons que pour une structure $\mathcal{M}_1$, cette distribution de probabilités soit très plate, très proche de l'uniforme et que pour un modèle $\mathcal{M}_2$, elle en soit bien différente. Alors nous pouvons dire que le modèle $\mathcal{M}_1$ est une structure qui n'a pas mis en avant les particularité de la distribution des données, elle n'a pas permis de détecter les motifs sous-jacents dans les données, car, pour toutes les différentes valeurs de C^T, les distributions $p(\{\mathbf{Y}_k^T\} \mid C^T, \mathcal{M}_1)$ sont similairement plates. En revanche, pour un modèle $\mathcal{M}_2$ de petite entropie, nous allons avoir des $p(\{\mathbf{Y}_k^T\} \mid C^T, \mathcal{M}_2)$ piquées et informatives qui

permettrons une meilleur classification de nouvelles données. Nous justifions ainsi le choix de l'utilisation de l'entropie comme heuristique de comparaison de modèles.

L'entropie à minimiser est $H\left[\mathcal{M}_S\right] := H\left[p(\{\mathbf{Y}_k^T\}, C^T \mid \mathcal{M}_S)\right]$. Pour simplifier les notations, oublions l'exposant T, notons $p_k := p(\mathbf{Y}_k^T \mid C^T, \mathcal{M}_S)$ et $p_c := P(C)$. Alors

$$H\left[\mathcal{M}_S\right] := H\left[\, p(\{\mathbf{Y}_k\}, C \mid \mathcal{M}_S)\,\right] \tag{6.24}$$

$$= -\sum_C \sum_{\{\mathbf{Y}_k\}} \left(p_c \prod_k p_k\right) \log \left(p_c \prod_k p_k\right) \tag{6.25}$$

$$= -\sum_C \sum_{\{\mathbf{Y}_k\}} \left[p_c \log p_c \prod_k p_k + \left(p_c \prod_k p_k\right) \sum_k \log p_k\right] \tag{6.26}$$

$$= -\sum_C \left[p_c \log p_c \sum_{\{\mathbf{Y}_k\}} \prod_k p_k\right] - \sum_C \left[p_c \sum_{\{\mathbf{Y}_k\}} \left[\left(\prod_k p_k\right) \sum_k \log p_k\right]\right] \tag{6.27}$$

$$= H\left[\, p_c \,\right] - \sum_C p_c \sum_{\{\mathbf{Y}_k, k\neq i\}} \sum_{\mathbf{Y}_i} p_i \left(\prod_{k\neq i} p_k\right) \left[\log p_i + \sum_{k\neq i} \log p_k\right] \tag{6.28}$$

$$= H\left[\, p_c \,\right] - \sum_C p_c \sum_{\{\mathbf{Y}_k, k\neq i\}} \left(\prod_{k\neq i} p_k\right) \sum_{\mathbf{Y}_i} \left[p_i \log p_i + p_i \sum_{k\neq i} \log p_k\right] \tag{6.29}$$

$$= H\left[\, p_c \,\right] - \sum_C p_c \sum_{\{\mathbf{Y}_k, k\neq i\}} \left(\prod_{k\neq i} p_k\right) \left(-H\left[\, p_i \,\right] + \sum_{k\neq i} \log p_k\right) \tag{6.30}$$

$$= H\left[\, p_c \,\right] - \sum_C p_c \left(-H\left[\, p_i \,\right] + \sum_{\{\mathbf{Y}_k, k\neq i\}} \left(\prod_{k\neq i} p_k\right) \sum_{k\neq i} \log p_k\right) \tag{6.31}$$

$$= H\left[\, p_c \,\right] + \sum_C p_c \sum_k H\left[\, p_k \,\right] \tag{6.32}$$

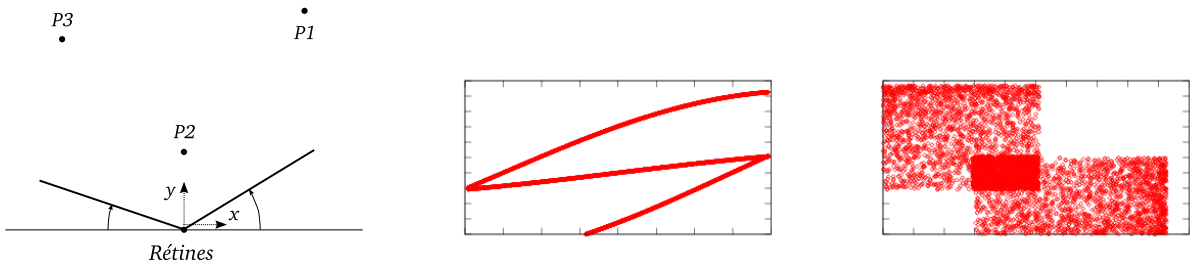
$$= H\left[\, p_c \,\right] + \sum_k \left[\sum_C p_c H\left[\, p_k \,\right]\right] \tag{6.33}$$

$$= H\left[\, p(C) \,\right] + \sum_k \left[\sum_C p(C) H\left[\, p(\mathbf{Y}_k^T \mid C^T, \mathcal{M}_S)\,\right]\right] \tag{6.34}$$

Dans l'équation 6.34 nous particularisons le terme i pour faire apparaître son entropie. La procédure est identique pour tous les k. Ainsi l'entropie se décompose sur les différents termes. Si nous voulons utiliser $\mathcal{M}_S$ pour faire de la prédiction et que nous avons un *a priori* uniforme sur la possible valeur de C^T, alors l'entropie devient simplement la somme $\sum_k \left[\sum_C H\left[\, p(\mathbf{Y}_k^T \mid C^T, \mathcal{M}_S)\,\right]\right]$.

6.5 Création : Résultats

Nous présentons les résultats de notre algorithme pour deux jeux de données synthétiques différents, générés pour trois et cinq positions réelles. Pour chacune de ces expé-



(a) Positions dans l'espace (x, y). (b) Projection sur (A_L, X_L). (c) Projection sur (X_L, X_R).

FIG. 6.4 – **Expérience 1** : données. Les 3 positions réelles sont représentées sur la figure 6.4(a). La figure 6.4(b) exhibe le jeu de 6.10^4 données $\{A_L^t, A_R^t, X_L^t, X_R^t\}$ projeté sur le sous-espace (A_L, X_L) (nous ne représentons que deux des variables et visualisons leur densité jointe) et la figure 6.4(c) sur le sous-espace (X_L, X_R). Les répartitions des données dans ces 2 sous-espaces sont différentes, celle de 6.4(b) ayant une entropie plus faible. C'est cette différence d'entropie qui nous permettra de comparer les structures entre elles.

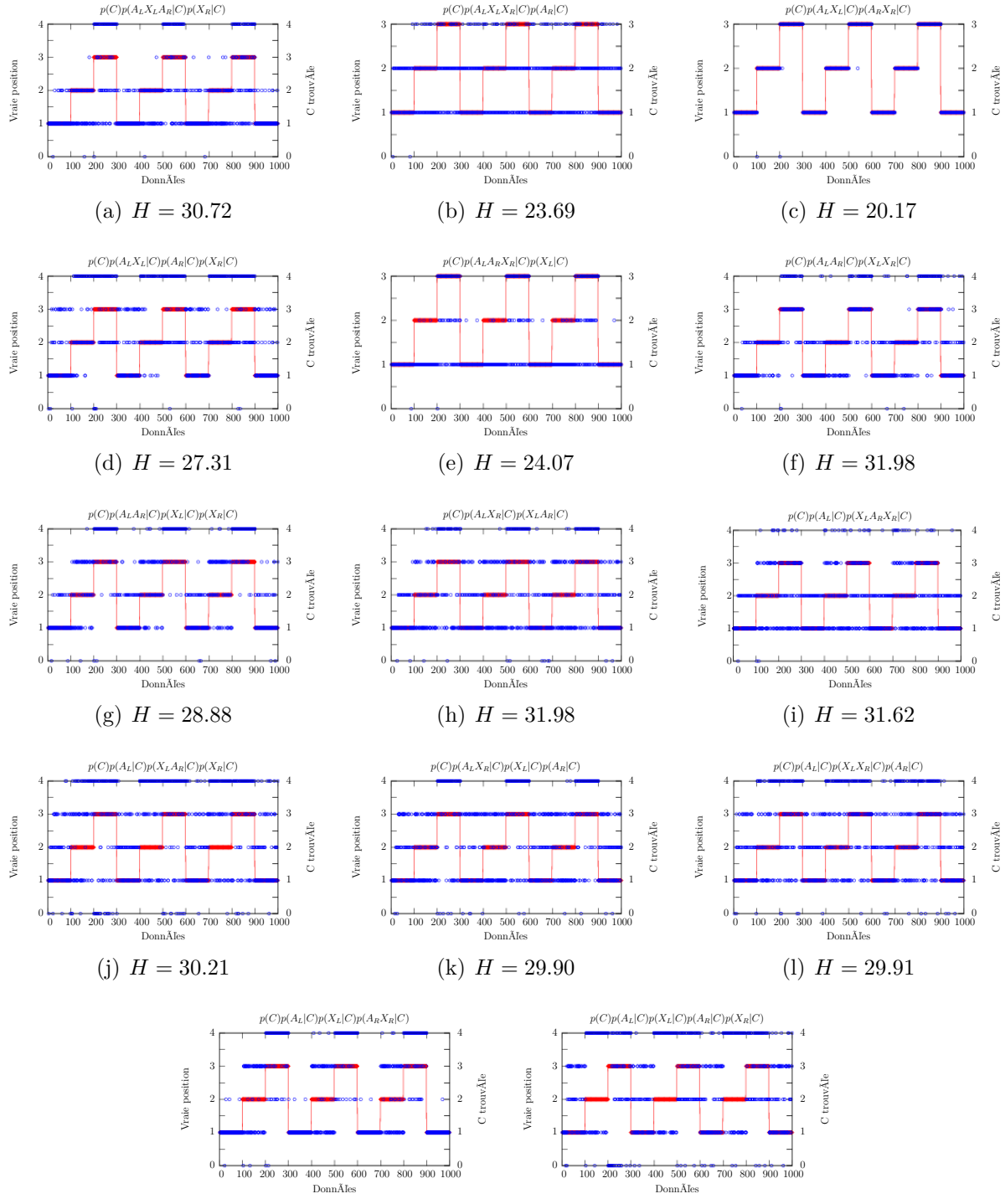
riences, nous présentons les données ainsi que le déroulement de l'apprentissage, le nombre de valeurs de C crées et l'entropie pour les différentes structures. Nous détaillons aussi une utilisation des résultats de l'apprentissage pour la réalisation d'une tâche de classification.

6.5.1 Expérience 1 : 3 positions

Nous appliquons notre algorithme au problème des deux rétines pour lequel $\mathbf{X}^t = \{A_L^t, A_R^t, X_L^t, X_R^t\}$. Le jeu de données Δ^T est généré à partir de $N = 3$ positions différentes, $O = 100$ pas de temps par position (fig. 6.4). De plus, la séquence des 3 positions est répétée $L = 20$ fois, afin de vérifier que notre algorithme reconnaît des positions anciennement vues. Les variables de $\mathbf{X}^t$ sont discrétisées en $R = 10$ cas. Ainsi le jeu contient $T = N \times O \times L = 6.10^3$ relevés. Notons n_C^t le nombre de valeurs de C^t lors du déroulement de l'algorithme.

La figure 6.5 présente le déroulement de l'algorithme pour les 1 000 premières données et pour chacune des 14 structures. Nous représentons en ordonnées deux valeurs à chaque t : en rouge la vraie position (inconnue de l'algorithme) et en bleu la valeur c^{t*} maximisant l'*a posteriori* courant $p(C^t \mid \{\mathbf{X}^{0,\dots,t}\})$. Nous constatons que parmi les structures, seule celle correspondant à la "vraie" structure, $p(C)p(A_L X_L|C)p(A_R X_R|C)$ (fig. 6.5(c)), présente une bonne adéquation entre les points bleus et les points rouges. Pour cette structure, dès la première donnée, une nouvelle valeur c_1 pour C a été créé. Pendant environ 100 données les tables $p(A_L X_L|C = c_1)$ et $p(A_R X_R|C = c_1)$ sont apprises et prennent des formes présentées par la figure 6.6(a). Vers la centième donnée, la vraisemblance des données issues de la nouvelle position devient faible dans ce modèle et la valeur c_0 correspondant à une uniforme devient plus probable. L'algorithme crée alors une nouvelle valeur c_2.

Pour comprendre pourquoi cette création a été décidée vers la centième donnée, il faut se souvenir que les tables apprises sont des lois de successions de Laplace. Considérons le cas unidimensionnel : nous apprenons une table de probabilité de Laplace pour une variable Z.



(a) $H = 30.72$ (b) $H = 23.69$ (c) $H = 20.17$

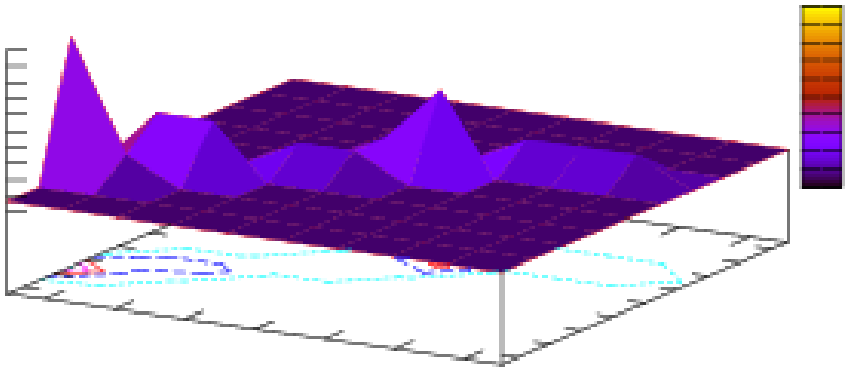
(d) $H = 27.31$ (e) $H = 24.07$ (f) $H = 31.98$

(g) $H = 28.88$ (h) $H = 31.98$ (i) $H = 31.62$

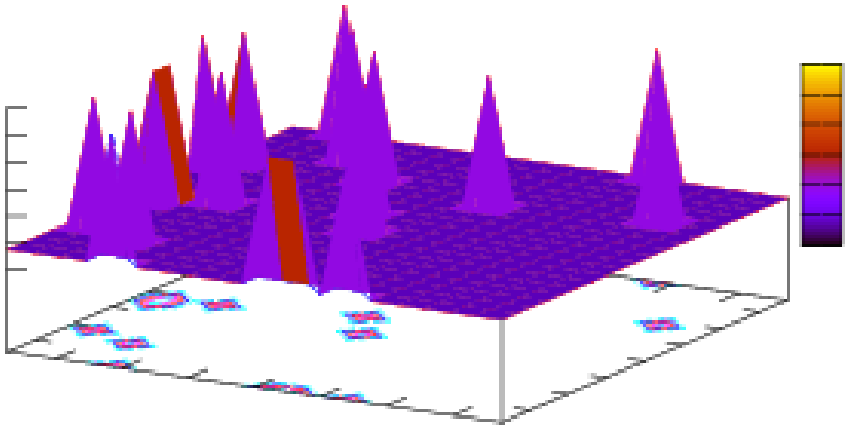
(j) $H = 30.21$ (k) $H = 29.90$ (l) $H = 29.91$

(m) $H = 27.28$ (n) $H = 29.14$

FIG. 6.5 – **Expérience 1** : déroulement de l'apprentissage avec 3 vraies positions. Pour chaque structure, un graphe représente en rouge la vraie position qui a généré la donnée. Après la lecture de chaque donnée, l'algorithme cherche la valeur de c^{t*} maximisant l'*a posteriori* $p(C^t \mid \{\mathbf{X}^{0,\dots,t}\})$. Cette valeur est reportée en bleu. Pour chaque structure, nous donnons aussi la valeur de l'entropie H de la jointe finale.



(a) Décomposition (c), table $p(A_L X_L | C = c_1)$



(b) Décomposition (f), table $p(A_L A_R | C = c_1)$

FIG. 6.6 – **Expérience 1** : allure de deux tables de probabilités apprises après 100 données. Nous retrouvons bien la répartition des données des figures 6.4(b) et 6.4(c) et nous visualisons la différence d'entropie de ces deux structures.

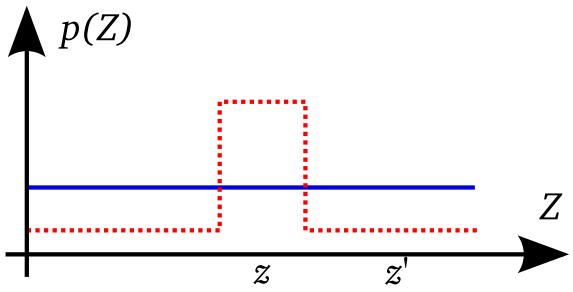


FIG. 6.7 – Illustration de la prise de décision de créer une nouvelle valeur pour C. La vraisemblance d'une loi de succession de Laplace (en rouge) apprise avec des valeurs identiques $\{z_n\}$ est plus faible que celle de l'uniforme (en bleu) pour une donnée $z' \neq z$.

Imaginons que les données $\{z_1 \dots z_N\}$ soient toute similaires, qu'elle correspondent toutes à la même case z de la discrétisation de Z (Z est discrétisée en R cases). La table apprise prend ainsi une grande valeur pour $Z = z$ et des valeurs plus faibles pour les autres éventualités. Alors la vraisemblance de cette table, pour une nouvelle donnée différente de z sera plus petite que celle d'un modèle uniforme, comme représenté par la figure 6.5.1. C'est ce principe qui s'applique pendant le déroulement de notre algorithme : la $101^{\text{ième}}$ donnée donnera une plus grande probabilité à c_0 qu'à c_1 et donc une nouvelle valeur c_2 est créée.

Ce phénomène est cependant modulé par le modèle de transition, qui donne une dynamique inertielle à l'algorithme. L'*a posteriori* sur C ne dépend pas que de la vraisemblance des tables mais dépend aussi de la matrice de transition. Ainsi, l'algorithme peut tolérer, pendant un certain temps, des données moins vraisemblables avant de décider de créer une nouvelle valeur pour C.

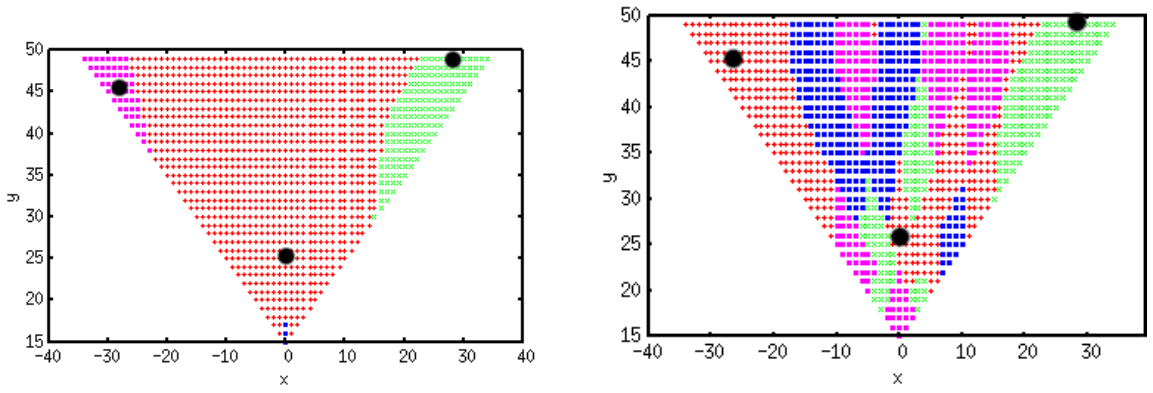
Revenons à la figure 6.5. En comparant les valeurs de C estimées et les vraies positions, nous voyons que la bonne structure (fig. 6.5(c)) donne de bons résultats, alors que les autres se trompent souvent. Cependant, comme l'algorithme ne connaît pas les vraies positions, nous devons utiliser un autre critère pour comparer les structures. Ce critère est l'entropie de la distribution finale que nous reportons en légende de chaque sous-figure. En classant les différentes structures par entropie croissante nous obtenons le tableau 6.8.

Nous trouvons ainsi que le modèle le plus informatif (de faible entropie) est $p(C)$ $p(A_L X_L|C)$ $p(A_R X_R|C)$, celui correspondant effectivement au vrai processus de génération de données. Et nous constatons aussi que le modèle le plus plat, le plus entropique est sans surprise $p(C)$ $p(A_L A_R|C)$ $p(X_L X_R|C)$. Ces découvertes sont permises par une exploration exhaustive de l'espace des modèles.

Comparons l'utilité de ces deux modèles extrêmes dans la cadre d'une classification spatiale. Une fois avoir appris les tables conditionnelles et la variable C, nous les utilisons pour assigner une classe à chaque point (x, y) de l'espace réel. L'idée est de trouver la valeur de C maximisant une probabilité de la forme $p(C \,|\, \Delta^T, (x, y))$ pour tout (x, y). Cependant, comme les variables (x, y) ne sont pas présentes dans les modèles, nous considérons que le

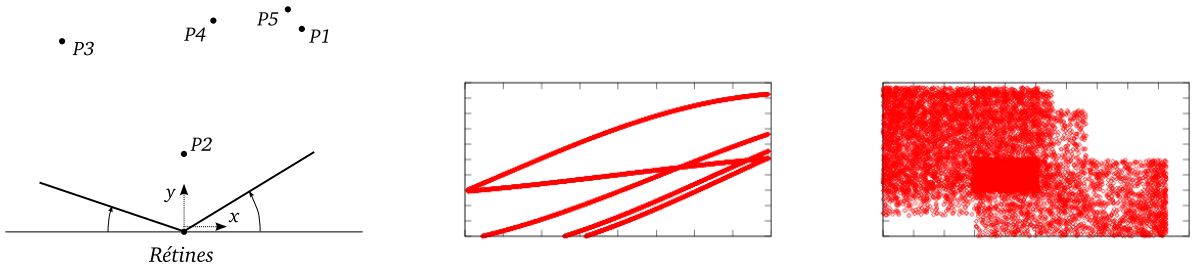
Structure	Décomposition	Entropie H
(c)	$p(C)\,p(A_LX_L\|C)\,p(A_RX_R\|C)$	20.17
(b)	$p(C)\,p(A_LX_LX_R\|C)\,p(A_R\|C)$	23.69
(e)	$p(C)\,p(A_LA_RX_R\|C)\,p(X_L\|C)$	24.07
(m)	$p(C)\,p(A_L\|C)\,p(X_L\|C)\,p(A_RX_R\|C)$	27.28
(d)	$p(C)\,p(A_LX_L\|C)\,p(A_R\|C)\,p(X_R\|C)$	27.31
(g)	$p(C)\,p(A_LA_R\|C)\,p(X_L\|C)\,p(X_R\|C)$	28.88
(n)	$p(C)\,p(A_L\|C)\,p(X_L\|C)\,p(A_R\|C)\,p(X_R\|C)$	29.14
(k)	$p(C)\,p(A_LX_R\|C)\,p(X_L\|C)\,p(A_R\|C)$	29.90
(l)	$p(C)\,p(A_L\|C)\,p(X_LX_R\|C)\,p(A_R\|C)$	29.91
(j)	$p(C)\,p(A_L\|C)\,p(X_LA_R\|C)\,p(X_R\|C)$	30.21
(a)	$p(C)\,p(A_LX_LA_R\|C)\,p(X_R\|C)$	30.72
(i)	$p(C)\,p(A_L\|C)\,p(X_LA_RX_R\|C)$	31.62
(h)	$p(C)\,p(A_LX_R\|C)\,p(X_LA_R\|C)$	31.98
(f)	$p(C)\,p(A_LA_R\|C)\,p(X_LX_R\|C)$	31.98

FIG. 6.8 – **Expérience 1** : entropie finale pour les différentes structures.



(a) $p(C)\,p(A_LX_L|C)\,p(A_RX_R|C)$ (b) $p(C)\,p(A_LA_R|C)\,p(X_LX_R|C)$

FIG. 6.9 – **Expérience 1** : classification. Valeurs de C maximisant $p(C|\Delta^T, A_L = a_L, A_R = a_R, Z_L = Z_R = 0)$ pour les deux structures extrêmes. Différentes couleurs correspondent à différentes valeurs de C. Nous constatons l'apparition de la classification de l'espace (x, y) en plusieurs zones. Cette classification est de meilleure qualité pour la bonne structure 6.9(a). Les vraies positions sont représentées par des points noirs.



(a) Positions dans l'espace (x, y). (b) Projection sur (A_L, X_L). (c) Projection sur (X_L, X_R).

FIG. 6.10 – **Expérience 2** : données. Les 5 positions réelles sont représentées dans l'espace (x, y), ainsi que les projections des données dans les espaces (A_L, X_L) et (X_L, X_R) .

robot a positionné ses yeux de façon à centrer les images sur ses rétines. Il a donc choisi, pour un objet présent en (x, y), $A_L = a_L$ et $A_R = a_R$ tels que $Z_L = Z_R = 0$. Nous pouvons alors maximiser $p(C \mid \Delta^T, A_L = a_L, A_R = a_R, Z_L = Z_R = 0)$ pour tout (x, y).

La figure 6.9 présente la classification qui en résulte, pour la bonne décomposition et la plus "mauvaise" (au sens entropique). Nous constatons tout d'abord que, dans les deux cas, les points associés à une même valeur de C sont voisins et que les classes trouvées sont assez étendues. De plus, pour le bon modèle (fig. 6.9(a)), les 3 classes correspondent approximativement aux vraies positions (fig. 6.4(a)). Ce n'est pas le cas pour la mauvaise décomposition (fig. 6.9(b)) qui ne reproduit pas bien la topologie initiale.

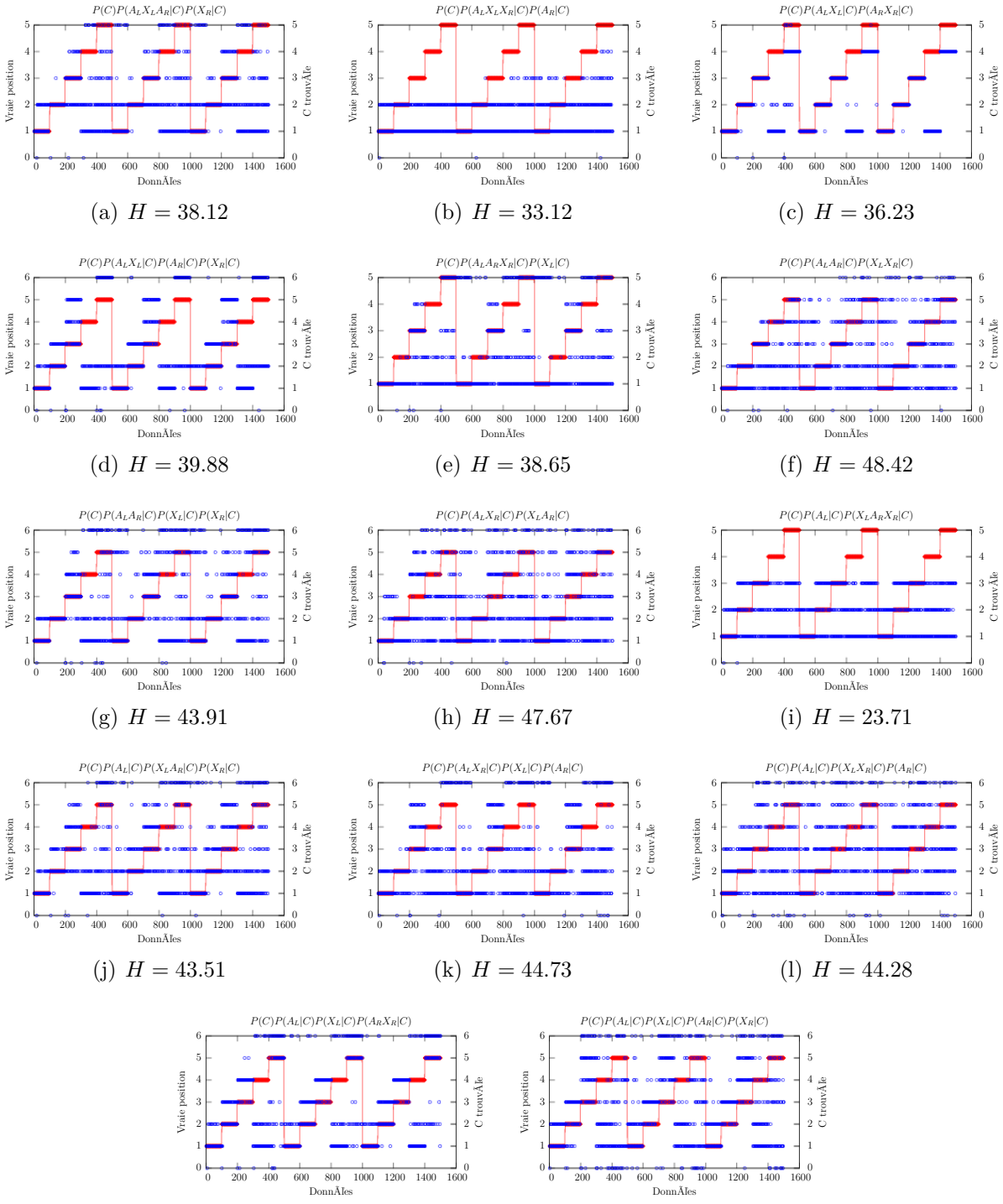
Ainsi, pour la bonne décomposition, les valeurs de la variable C créée et les distributions conditionnelles ont permis de construire une représentation de l'espace (x, y). En effet chaque valeur de C correspond en fait à une sous zone de cet espace. En combinant les informations provenant de ses deux rétines, le robot s'est construit une représentation du phénomène sous-jacent de façon incrémentale et non supervisée.

6.5.2 Expérience 2 : 5 positions

Nous reproduisons la même expérience, avec $N = 5$ positions. La figure 6.10 présente les données et la figure 6.11 le déroulement de l'algorithme pour les différentes structures.

Nous pouvons définir trois types de comportements selon la structure. Pour la plupart d'entre elles, le nombre n_c^T de valeurs créées pour C est trop grand et les c^{t*} trouvés n'ont pas de lien avec les vraies positions. À l'opposé, les structures correspondantes au figures 6.11(b) et 6.11(i) n'ont pas créé assez de valeurs et mélangent les positions. La "bonne" décomposition (fig. 6.11(c)) donne de meilleurs résultats, bien qu'elle ait regroupé les positions P5 et P1 sous une même valeur de C. En effet ces deux positions sont proches dans l'espace (x, y) (fig. 6.10(a)) et les projections des données qui leur sont associées sont presque identiques (fig. 6.10(b)) en raison de la discrétisation ($R = 10$). Il est donc difficile pour un quelconque algorithme d'apprendre à les différencier.

Le tableau 6.12 donne les entropies correspondantes, ainsi que le nombre de valeurs de C créées. Nous lisons que la bonne décomposition (c) ne se classe que troisième, car les



(a) $H = 38.12$ (b) $H = 33.12$ (c) $H = 36.23$

(d) $H = 39.88$ (e) $H = 38.65$ (f) $H = 48.42$

(g) $H = 43.91$ (h) $H = 47.67$ (i) $H = 23.71$

(j) $H = 43.51$ (k) $H = 44.73$ (l) $H = 44.28$

(m) $H = 40.41$ (n) $H = 43.71$

FIG. 6.11 – **Expérience 2** : Déroulement de l'apprentissage avec 5 vraies positions.

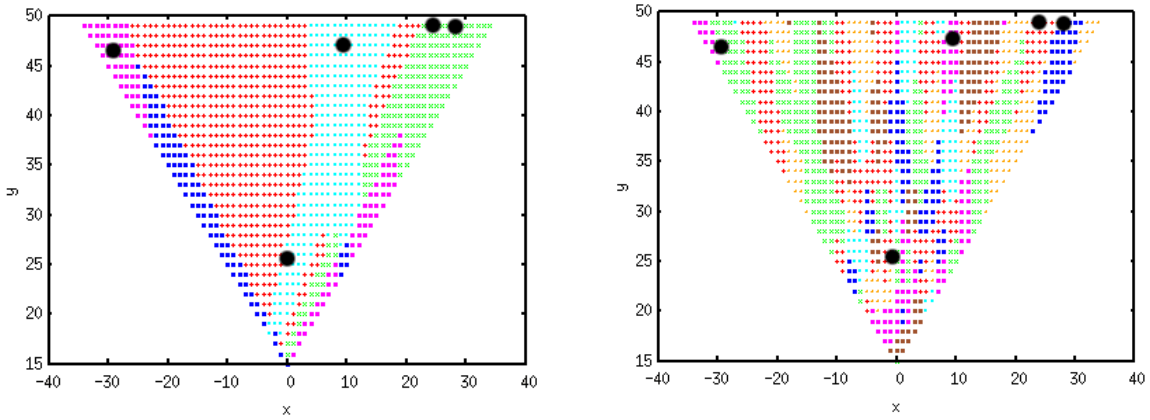
Structure	Décomposition	Entropie H	n_C^T	H/n_C^T
(i)	$p(C)\,p(A_L\|C)\,p(X_L A_R X_R\|C)$	23.71	3	7.9
(b)	$p(C)\,p(A_L X_L X_R\|C)\,p(A_R\|C)$	33.12	4	8.2
(c)	$p(C)\,p(A_L X_L\|C)\,p(A_R X_R\|C)$	36.23	5	7.2
(a)	$p(C)\,p(A_L X_L A_R\|C)\,p(X_R\|C)$	38.12	5	7.6
(e)	$p(C)\,p(A_L A_R X_R\|C)\,p(X_L\|C)$	38.65	5	7.7
(d)	$p(C)\,p(A_L X_L\|C)\,p(A_R\|C)\,p(X_R\|C)$	39.88	6	7.6
(n)	$p(C)\,p(A_L\|C)\,p(X_L\|C)\,p(A_R\|C)\,p(X_R\|C)$	43.79	6	7.3
(g)	$p(C)\,p(A_L A_R\|C)\,p(X_L\|C)\,p(X_R\|C)$	43.91	6	7.3
(j)	$p(C)\,p(A_L\|C)\,p(X_L A_R\|C)\,p(X_R\|C)$	44.1.81	6	7.3
(l)	$p(C)\,p(A_L\|C)\,p(X_L X_R\|C)\,p(A_R\|C)$	44.28	6	7.3
(k)	$p(C)\,p(A_L X_R\|C)\,p(X_L\|C)\,p(A_R\|C)$	44.73	6	7.4
(m)	$p(C)\,p(A_L\|C)\,p(X_L\|C)\,p(A_R X_R\|C)$	45.61	6	7.7
(h)	$p(C)\,p(A_L X_R\|C)\,p(X_L A_R\|C)$	47.67	6	7.9
(f)	$p(C)\,p(A_L A_R\|C)\,p(X_L X_R\|C)$	48.42	6	8.0

FIG. 6.12 – **Expérience 2** : entropie finale pour les différentes structures et nombre de valeurs pour C créées.

deux structures ayant conduit à des n_C^T trop petits ont une plus faible entropie. En effet, le calcul 6.24 montre que l'entropie se décompose comme une somme sur les différentes valeurs de C et il se trouve que H croit approximativement linéairement avec n_C^T. Nous en concluons que l'entropie globale n'est pas toujours une bonne heuristique pour comparer des structures portant sur des variables C différentes. En revanche, en divisant H par n_C^T comme dans la dernière colonne du tableau, nous retrouvons que la bonne décomposition (c) est proportionnellement la plus informative. Elle peut donc être sélectionnée par cette mesure d'information.

Pour cette expérience à cinq positions, nous présentons aussi la projection dans l'espace (x, y) pour les deux décompositions extrêmes (c) et (f) (fig. 6.13). D'abord nous constatons que la mauvaise décomposition donne une classification (fig. 6.13(b)) sans lien avec les vraies positions et très peu continue. En revanche, la classification de la figure 6.13(a) est plus pertinente. Nous retrouvons qu'il existe une position (P3) en haut à gauche (rose) et plusieurs positions différentes en haut à droite. Les positions P1 et P5 sont confondues et représentée par la même valeur de C (vert). Néanmoins cette classification n'est pas parfaite, car il semble que P2 ait été groupée avec P4 et une partie de la zone rose qui devrait correspondre à P3 se trouve en bas à droite.

Ainsi cette classification n'est pas exactement à l'image des vraies positions. Il s'agit plus d'une représentation interne de l'espace (x, y) que s'est construit l'algorithme à partir des données. Dans ces données l'information qu'il existait des vraies positions n'était pas explicitement présente, mais la figure 6.13(a) montre qu'il est tout de même possible de retrouver l'existence d'une structure spatiale sous-jacente.



(a) $p(C)\,p(A_L X_L|C)\,p(A_R X_R|C)$ (b) $p(C)\,p(A_L A_R|C)\,p(X_L X_R|C)$

FIG. 6.13 – **Expérience 2** : classification. Valeurs de C maximisant $p(C \mid \Delta^T, A_L = a_L, A_R = a_R, Z_L = Z_R = 0)$ pour les deux structures extrêmes (c) à gauche et (f) à droite. Les vraies positions sont représentées par des points noirs.

6.6 Conclusion

6.6.1 Contributions

Dans ce chapitre, nous nous sommes intéressés à un problème de découverte de la cardinalité d'une variable afin d'apprendre une partie du modèle probabiliste. Dans un cadre robotique, les variables sensorimotrices sont *a priori* toutes dépendantes entre elles, mais en supposant que le robot fait face à un environnement régulier, il est possible de simplifier cette structure de dépendance en introduisant une nouvelle variable.

Nous avons donc proposé un algorithme qui, sous l'hypothèse de l'existence d'un phénomène lent, découvre la cardinalité d'une telle variable de façon incrémentale et non supervisée. Lorsque les données deviennent trop peu vraisemblables dans le modèle courant, il décide automatiquement d'augmenter la cardinalité de la variable créée afin de considérer un nouveau régime.

Pour comparer différentes structures de dépendance, nous avons utilisé une heuristique basée sur l'entropie : une décomposition est meilleure si elle donne lieu à une jointe plus informative.

Nous avons appliqué cet algorithme à des jeux de données synthétiques, avec des résultats mitigés. Lorsque le problème est simple, avec peu de vraies positions, nous avons bien retrouvé que la décomposition correspondante au phénomène réel de génération de données est la moins entropique. De plus, cette décomposition et les tables de probabilités apprises semblent bien représenter la structure spatiale sous-jacente du problème.

Cependant les résultats sont moins bons avec plus de positions et l'algorithme possède plusieurs points faibles. D'abord le choix de la résolution, du pas de discrétisation des variables a une influence sur les résultats, en en particulier sur le calcul de l'entropie. En

effet, si la discrétisation est très fine, chaque donnée est presque seule dans sa case et l'entropie de la bonne décomposition est la même que celle de la mauvaise. Ce problème est illustré par la figure 6.6 : si la discrétisation était plus grande, les deux distributions auraient la même entropie. En effet le calcul d'une entropie ne tient pas compte de la notion de voisinage : que les pics soient alignés ou disposés aléatoirement dans l'espace, l'entropie est identique. Ainsi notre algorithme a fonctionné car la résolution avait une valeur permettant un lissage des distributions.

Plus fondamentalement, utiliser l'entropie est intéressant dans la mesure où cela permet de choisir la décomposition la plus informative, la moins plate. Cependant, en terme de sélection de modèle, la méthode bayésienne nous demanderait plutôt de maximiser la vraisemblance marginale.

De plus, nous comparons les entropies de distributions portant sur des ensembles de variables différents. En effet, pour deux décompositions différentes, deux variables C sont apprises avec des n_C^T parfois différents. Dans ce cas, l'heuristique peut échouer et il vaut mieux utiliser le ratio H/n_C^T, mais nous n'avons pas trouvé de justification théorique pour cela.

6.6.2 Perspectives

Notre méthode peut être améliorée de plusieurs façons. D'abord, pour pallier le problème de la sensibilité à la résolution, nous pouvons introduire une meilleur estimation des probabilités conditionnelles que les histogrammes de Laplace. Par exemple, nous avons tenté d'introduire une notion de voisinage en utilisant une estimation par noyaux gaussiens : chaque donnée est associée à une petite gaussienne de variance σ et la représentation de la densité est une mixture (estimateur de Parzen). Cependant deux problèmes se sont posés : techniquement les calculs deviennent rapidement trop longs et d'autre part le choix de σ est tout aussi délicat que celui de R. L'approche bayésienne nous recommanderait donc de marginaliser ce paramètre, ce qui n'est pas réalisable en pratique.

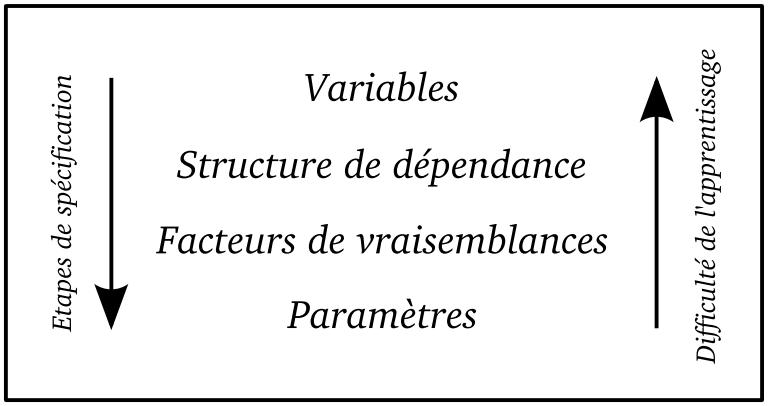
Un autre axe de développement est le parcours de l'espace des structures. Dans les exemples présentés, nous avons testé exhaustivement toutes les structures qui nous intéressaient, mais ceci devient prohibitif avec plus de variables. Il faudrait alors parcourir parcimonieusement l'espace des structures, par exemple par une méthode gloutonne de modification locale de structure, ou à l'aide d'algorithmes génétiques. Cela nous mennerai à étudier le problème de découverte de modèle dans le cadre de l'apprentissage de structure de réseaux bayésiens (LF05).

En raison des limitations citées ci-dessus, de notre incertitude quant à la pertinence de l'heuristique entropique, de l'apparition de résultats moins convainquants avec des jeux de données plus conséquents et d'un manque de temps, nous avons renoncé à étudier plus en profondeur cette méthode pour davantage nous concentrer sur d'autres parties de nos travaux de thèse.

Conclusion

Contributions

Dans ce document, nous avons montré que la méthode bayésienne était une solution pertinente pour résoudre des problèmes d'apprentissage dans un contexte d'incertitude. Nous nous sommes intéressés à quatre problèmes liés par ce formalisme probabiliste. Deux de ces problèmes concernent un apprentissage de paramètres avec un modèle probabiliste connu alors que pour les deux autres notre but est la découverte d'une partie du modèle.



Comme la figure 6.6.2 le rappelle, différents types d'apprentissages présentent différents degrés de difficultés. Cependant nous avons vu que même des questions d'estimation paramétrique simples peuvent être délicats, en particulier quand de grandes quantités de données doivent être traitées ou quand les inférences ne peuvent pas être menées analytiquement.

Il est en fait assez rare de pouvoir suivre complètement et rigoureusement la méthode bayésienne dès que les modèles ne sont plus triviaux. Ainsi nous avons dû proposer des approximations et des heuristiques pour obtenir nos résultats. Cette remarque est une conclusion générale de notre travail : lorsque les problèmes sont complexes, des solutions approchées proposant un compromis équilibré entre précision et rapidité doivent être employées. Si nous ne trouvons pas de telles solutions, il peut être pertinent de revoir le modèle ou de revenir à des méthodes *ad hoc* moins élégantes, mais plus performantes.

Contributions de la partie 1

Dans cette partie, nous avons couvert en détail les fondations philosophiques et la méthodologie de l'apprentissage bayésien. Nous avons aussi établi un état de l'art des différentes spécifications d'*a priori* et des méthodes d'inférences disponibles.

Contributions de la partie 2

Nous avons y proposé deux algorithmes d'apprentissage de paramètres à partir de grandes bases de données.

Pour le classement des joueurs d'échecs, nos contributions portent principalement sur la définition d'un modèle temporel permettant la propagation d'information du futur vers le passé. En effet le développement de TrueChess a été initié lors d'une collaboration de trois mois avec Ralf Herbrich, Thore Graepel et Tom Minka de *Microsoft Research Cambridge*. TrueChess est une extension du système TrueSkill (HG06) introduit par ces chercheurs. La nouveauté majeure de TrueChess est le lien temporel entre les parties qui permet d'utiliser les résultats de matchs futurs pour améliorer l'estimation des cotations passées. De plus nous avons dû inventer des ordonnanceurs efficaces car l'ajout de ces liens temporels fait apparaître des boucles qui n'étaient pas présentes dans TrueSkill. Ces liens temporels ont permis d'estimer des cotations historiques des joueurs d'échecs. Ce travail fait l'objet d'une publication à NIPS-2007 (DHMG07) et d'un dépôt de brevet aux États-Unis.

Concernant le problème du filtrage collaboratif, notre contribution est la proposition de nouveaux modèles génératifs de double *clustering*, associés avec leurs algorithmes d'inférences. Nous soulignons aussi l'introduction d'une méthode de *clustering* hiérarchique pour accélérer les calculs. Les modèles introduits sont porteurs de sens et permettent de faire des prédictions. Cependant nos performances prédictives sont moins bonnes que celles des leaders actuels de la compétition en raison de la difficulté des inférences.

Contributions de la partie 3

Dans cette partie nous avons d'abord étudié le problème de la sélection de variables. Dans ce cadre, nous avons comparé différents algorithmes de sélection. L'application à une tâche robotique a été l'objet d'une publication à la conférence internationale ICRA 2005 (DBS05).

Dans le dernier chapitre nous avons proposé un algorithme original de découverte de variable et de comparaison de structures sous l'hypothèse d'une régularité de l'environnement d'un robot. Bien que souffrant de quelques faiblesses, cette méthode a donné des résultats satisfaisant sur de petits jeux de données.

Perspectives

Concernant les perspectives particulières à chacun des quatre algorithmes, nous renvoyons le lecteur aux conclusions de leur chapitre respectif (sections 3.7.2, 4.7.2, 5.6.2 et

6.6.2).

Bayésien

Inférence D'une façon générale, nous pensons que l'approche subjective présente de nombreuses qualités pour résoudre les problèmes d'analyse de données et d'apprentissage. Nous avons déjà souligné ses limitations, surtout en termes de coût calculatoire. Cependant nous pensons qu'avec le développement actif de nouvelles méthodes d'inférence (en particulier les méthodes déterministes approchées) de plus en plus de problèmes deviennent traitables.

Modèles Concernant les récentes avancées en terme de modélisation, des approches non paramétriques comme les processus gaussiens (RW05) et les processus de Dirichlet (BGJT04) sont prometteuses. Elles permettent d'introduire des modèles de taille infinie (comme une mixture d'une infinité de gaussiennes (Ras00)) au prix d'un temps de calcul fini. Nous pourrions utiliser de tels modèles, par exemple les HMM infinis (BGR02), pour mieux formaliser notre méthode de découverte de variables. D'autre part, un processus gaussien serait pertinent pour améliorer le modèle d'évolution dynamique de TrueChess, et permettrait une plus grande flexibilité dans la spécification de nos *a priori*.

Généralisation Le sur-apprentissage est un défaut parfois délicat à éviter. Pour cela de nombreuses méthodes font explicitement appel au principe philosophique du rasoir d'Occam. Nous avons vu que la règle de marginalisation conduisait à un rasoir d'Occam automatique : la vraisemblance marginale d'un modèle décroît automatiquement avec sa complexité. Ainsi il n'est nul besoin d'ajouter une règle de parcimonie supplémentaire au cadre bayésien. Cependant, cette même règle de marginalisation conduit aussi au principe d'Epicure quand elle prescrit de moyenner sur tous les modèles au lieu de n'en sélectionner qu'un. En effet, la méthode rigoureuse ne recommande jamais d'établir des estimations ponctuelles (le choix d'un seul modèle). Ainsi le rasoir d'Occam automatique n'est qu'un effet de bord de notre volonté d'approximation. Choisir un seul modèle implique donc des conséquences en termes de parcimonie. Il serait certainement intéressant d'approfondir ces questions et leurs conséquences pratiques.

Apprentissage automatique

Dans ce document nous avons choisi de nous restreindre à l'étude des modèles bayésiens génératifs. Mais une grande partie des travaux en apprentissage automatique portent sur les approches discriminatives (comme les SVMs), qui donnent très bons résultats expérimentaux. Nous aimerions étudier de plus près les liens entre ces deux types de modèles (BT04 ; LBM06).

Bibliographie

[ABR64] M. A. Aizerman, E. A. Braverman, and L. Rozonoer. Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control*, 25:821–837, 1964.

[Bae03] John Baez. Bayesian probability theory and quantum mechanics. http://math.ucr.edu/home/baez/bayes.html, September 2003.

[Bar01] H Barlow. The exploitation of regularities in the environment by the brain. *Behavioral and Brain Sciences*, 24(3):602–607, 2001.

[Bax82] R. J. Baxter. Exactly solved models in statistical physics. *Academic press*, 1982.

[Bay63] T. Bayes. An essay towards solving a problem in the doctrine of chances. *Philosophical Transactions of the Royal Society of London*, 53, 1763.

[BB89] J. Berger and J. Bernardo. Estimating a product of means: Bayesian analysis with reference priors. *Journal of the American Statistical Association*, 89:200–2007, 1989.

[BDL+98a] Pierre Bessière, Eric Dedieu, Olivier Lebeltel, Emmanuel Mazer, and Kamel Mekhnacha. Interprétation ou description (i) : Proposition pour une théorie probabiliste des systèmes cognitifs sensori-moteurs. *Intellectica*, 26-27:257–311, 1998.

[BDL+98b] Pierre Bessière, Eric Dedieu, Olivier Lebeltel, Emmanuel Mazer, and Kamel Mekhnacha. Interprétation ou description (ii) : Fondements mathématiques de l'approche f+d. *Intellectica*, 26-27:313–336, 1998.

[Bea03] M.J. Beal. *Variational Algorithms for Approximate Bayesian Inference*. PhD thesis, Gatsby Computational Neuroscience Unit, University College London., 2003.

[Ber13] Jakob Bernoulli. *Ars Conjectandi: Usum & Applicationem Praecedentis Doctrinae in Civilibus, Moralibus & Oeconomicis*. 1713.

[Ber07] J. L. Bertrand. *Calcul des probabilités*. Gauthier-Villars,Paris, 1907.

[Ber97] José-Miguel Bernardo. Noninformative priors do not exist: A discussion. *Journal of Statistical Planning and Inference*, 65:159–189, 1997.

[BG96] C. Berrou and A. Glavieux. Near optimum error correcting coding and decoding: Turbo-codes. *IEEE Transactions on Communications*, pages 1261–1271, 1996.

[BGJT04] David M. Blei, Thomas L. Griffiths, Michael I. Jordan, and Joshua B. Tenenbaum. Hierarchical topic models and the nested chinese restaurant process. In *Advances in Neural Information Processing Systems*, 2004.

[BGR02] M. J. Beal, Z. Ghahramani, and C. E. Rasmussen. The infinite hidden Markov model. In *Advances in Neural Information Processing Systems 14*. MIT Press, 2002.

[BGV92] Bernhard E. Boser, Isabelle M. Guyon, and Vladimir N. Vapnik. A training algorithm for optimal margin classifiers. In *COLT '92: Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152, New York, NY, USA, 1992. ACM Press.

[BHK98] John Breese, David Heckerman, and Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the 14th Annual Conference on Uncertainty in Artificial Intelligence (UAI-98)*, pages 43–52, San Francisco, CA, 1998. Morgan Kaufmann.

[Bis95] Christopher M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, November 1995.

[Bis06] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[BKV07] Robert Bell, Yehuda Koren, and Chris Volinsky. Modeling relationships at multiple scales to improve accuracy of large recommender systems. In *KDD '07: Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 95–104, New York, NY, USA, 2007. ACM Press.

[Bor09] E. Borel. *Élements de la théorie des probabilités*. Hermann et Fils, Paris., 1909.

[BT04] G. Bouchard and Bill Triggs. The tradeoff between generative and discriminative classifiers. In *IASC International Symposium on Computational Statistics (COMPSTAT)*, pages 721–728, Prague, August 2004.

[BW88] J. O. Berger and R.L. Wolpert. The likelihood principle. *Institute of Mathematical Statistics*, 6, 1988. Lecture Notes-Monograph Series, Hayward, CA.

[CAH06] Y.L. Chan, C.N.K Anderson, and E.A. Hadly. Bayesian estimation of the timing and severity of a population bottleneck from ancient dna. *PLoS Genetics*, 2, 2006.

[Cam06] Greg Campbell. Guidance for the use of bayesian statistics in medical device clinical trials - drafts. http://www.fda.gov/cdrh/osb/guidance/1601.html, May 2006.

[CFS06] C. M. Caves, C. A. Fuchs, and R. Schack. Subjective probability and quantum certainty. *ArXiv Quantum Physics e-prints*, August 2006.

[CH97] David Maxwell Chickering and David Heckerman. Efficient approximations for the marginal likelihood of bayesian networks with hidden variables. *Machine Learning*, 29(2-3):181–212, 1997.

[Cha87] David Chandler. *Introduction to Modern Statistical Mechanics*. Oxford University Press, USA, September 1987.

[CKHO98] N. D. Chatterjee, R. Krüger, G. Haller, and W. Olbricht. The Bayesian approach to an internally consistent thermodynamic database: theory, database, and generation of phase diagrams. *Computer*, 133:149–168, 1998.

[CMB07] Laurent Candillier, Frank Meyer, and Marc BoullÃĭ. Comparing state-of-the-art collaborative filtering systems. In Petra Perner, editor, *5th International Conference on Machine Learning and Data Mining in Pattern Recognition (MLDM'2007)*, volume LNAI 4571 of *LNCS*, pages 548–562, Leipzig, Germany, july 2007. Springer Verlag.

[Coo90] Gregory F. Cooper. The computational complexity of probabilistic inference using bayesian belief networks. *Artificial Intelligence*, 42(2-3):393–405, March 1990.

[Cou95] R. D. Cousins. Why isn't every physicist a Bayesian? *American Journal of Physics*, 63:398–410, May 1995.

[Cox61] Richard Threlkeld Cox. *The Algebra of Probable Inference*. The Johns Hopkins Press, Baltimore, 1961.

[CP04] Ariel Caticha and Roland Preuss. Maximum entropy and bayesian data analysis: Entropic priors. *Physical Review E*, 70:046127, 2004.

[CPL+06] Christophe Coué, Cédric Pradalier, Christian Laugier, Thierry Fraichard, and Pierre Bessière. Bayesian occupancy filtering for multitarget tracking: an automotive application. *Int. Journal of Robotics Research*, 25(1):19–30, January 2006.

[Dan07] Pierre Dangauthier. Distribution de maximum d'entropie de moyenne contrainte. Research report 6279, INRIA, August 2007.

[Dav88] H. A. David. *The method of paired comparisons.* Oxford University Press, New York, 1988.

[DBM05] Julien Diard, Pierre Bessière, and Emmanuel Mazer. Merging probabilistic models of navigation: the bayesian map. In *Proc. of the IEEE-RSJ Int. Conf. on Intelligent Robots and Systems*, pages 668–673, 2005.

[DBS05] Pierre Dangauthier, P. Bessière, and A. Spalanzani. Auto-supervised learning in the bayesian programming framework. In *Proc. of the IEEE Int. Conf. on Robotics and Automation*, Barcelona (ES), April 2005.

[DDF+99] Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407, January 1999.

[DDFG01] Arnaud Doucet, Nando De Freitas, and Neil Gordon. *Sequential Monte Carlo methods in practice.* Springer, 2001.

[Dem06] L. Demortier. Bayesian Reference Analysis. In L. Lyons and M. Karagöz Ünel, editors, *Statistical Problems in Particle Physics, Astrophysics and Cosmology*, pages 11–+, 2006.

[Den05] Sophie Deneve. Bayesian inference in spiking neurons. In Lawrence K. Saul, Yair Weiss, and Léon Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 353–360. MIT Press, Cambridge, MA, 2005.

[DHMG07] Pierre Dangauthier, Ralf Herbrich, Tom Minka, and Thore Graepel. Trueskill through time: Revisiting the history of chess. In *Advances in Neural Information Processing Systems 21.* MIT press, December 2007.

[Die00] Thomas G. Dietterich. Ensemble methods in machine learning. *Lecture Notes in Computer Science*, 1857:1–15, 2000.

[DLR77] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977.

[Dos03] V. Dose. Bayesian inference in physics: case studies. *Rep. Prog. Phys.*, 2003.

[DP97] Pedro Domingos and Michael J. Pazzani. On the optimality of the simple bayesian classifier under zero-one loss. *Machine Learning*, 29(2-3):103–130, 1997.

[DR84] P. J. Davis and P. Rabinowitz. *Methods of numerical integration*. Academic press, 1984.

[Edw06a] Rod Edwards. Edo historical chess ratings. http://www.edochess.ca/, 2006.

[Edw06b] Rod Edwards. Fide chess rating inflation. http://members.shaw.ca/redwards1/, 2006.

[Elo86] Arpad E. Elo. *The Rating of Chessplayers, Past and Present*. Arco Publishing, Inc., New York, 2nd edition, 1986.

[EMK06] G. Elidan, I. McGraw, and D. Koller. Residual belief propagation: Informed scheduling for asynchronous message passing. In *Proceedings of the Twenty-second Conference on Uncertainty in AI*, 2006.

[Fer73] Thomas S. Ferguson. A bayesian analysis of some nonparametric problems. *The Annals of Statistics*, 1(2):209–230, 1973.

[FID05] Fédération Internationale des Echecs FIDE. Fide handbook. http://www.fide.com/official/handbook.asp, 2005.

[Fie05] Stephen A. Fienberg. When did bayesian inference become "bayesian"? *Bayesian Analysis*, 1(1):1–40, July 2005.

[Fin37] Bruno de Finetti. La prÃlvision: Ses lois logiques, ses sources subjectives. In *Annales de l'Institut Henri PoincarÃl 7*, pages 1–68. Paris, 1937. Translated into English by Henry E. Kyburg Jr., Foresight: Its Logical Laws, its Subjective Sources. In Henry E. Kyburg Jr. and Howard E. Smokler (1964, Eds.), Studies in Subjective Probability, 53-118, Wiley, New York.

[Fin74] Bruno de Finetti. *Theory of Probability, Vol. I*. J. Wiley, New York, 1974.

[Fis25] R. A. Fisher. *Statistical Methods, Experimental Design, and Scientific Inference: A Re-issue of Statistical Methods for Research Workers, The Design of Experiments, and Statistical Methods and Scientific Inference*. Oxford University Press, USA, 1925.

[FS95] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *European Conference on Computational Learning Theory*, pages 23–37, 1995.

[Gal63] R. G. Gallager. *Low Density Parity-Check Codes*. MIT Press, 1963.

[GCSR03] Andrew Gelman, John B. Carlin, Hal S. Stern, and Donald B. Rubin. *Bayesian Data Analysis*. Chapman & Hall, second edition edition, 2003.

[GE03] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.

[GG84] S. Geman and D. Geman. Stochastic relaxation, gibbs distribution and the bayesian restoration of images. *IEEE Transaction in Pattern Analysis and Machine Intelligence*, 6:721–741, 1984.

[GGBHD04] Isabelle Guyon, Steve R. Gunn, Asa Ben-Hur, and Gideon Dror. Result analysis of the nips 2003 feature selection challenge. In *NIPS*, 2004.

[Ghi64] Edwin E. Ghiselli. *Theory of Psychological Measurement*. McGraw-Hill Book Company, 1964.

[GJ99] Mark E. Glickman and Albyn C. Jones. Rating the chess rating system. *Chance*, 12:21–28, 1999.

[Gli99] Mark E. Glickman. Parameter estimation in large dynamic paired comparison experiments. *Applied Statistics*, 48:377–394, 1999.

[GONT92] D. Goldberg, B. Oki, D. Nichols, and D.B. Terry. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35:61–70, 1992.

[Got12] Hermann von Gottschall. *Adolf Anderssen, der Altmeister deutscher Schachspielkunst. Sein Leben und Schaffen*. Leipzig, Veit & Comp, 1912.

[Gre95] P. Green. Reversible jump markov chain monte carlo computation and bayesian model determination. *Biometrika*, 82:711–732, 1995.

[GS05] Pau Gargallo and Peter Sturm. Bayesian 3d modeling from images using multiple depth maps. In *IEEE Conference on Computer Vision and Pattern Recognition*, San Diego, California, 2005.

[GW05] Jeff Gill and Davis Lee D. Walker. Elicited priors for bayesian model specifications in political science research. *The Journal of Politics*, 67(7), August 2005. University of Kentucky.

[Haj03] Alan Hajek. Interpretations of probability. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. The Metaphysics Research Lab, 2003.

[Hal98] Mark A. Hall. *Correlation-based Feature Selection for Machine Learning*. PhD thesis, Waikato University, Hamilton, NZ, 1998.

[Hal00] Mark A. Hall. Correlation-based feature selection for discrete and numeric class machine learning. In *Proc. 17th International Conf. on Machine Learning*, pages 359–366. Morgan Kaufmann, San Francisco, CA, 2000.

[Has70] W.K. Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57:97–109, 1970.

[HG06] Ralf Herbrich and Thore Graepel. Trueskill: A bayesian skill rating system. Technical Report MSR-TR-2006-80, Microsoft Research Ltd., June 2006.

[HGC94] David Heckerman, Dan Geiger, and David Maxwell Chickering. Learning bayesian networks: The combination of knowledge and statistical data. In *KDD Workshop*, pages 85–96, 1994.

[HGC01] Ralf Herbrich, Thore Graepel, and Colin Campbell. Bayes point machines. *J. Mach. Learn. Res.*, 1:245–279, 2001.

[Hol75] John Holland. *Adaptation in Natural and Artificial Systems.* University of Michigan Press, 1975.

[HOW+05] Tom Heskes, Manfred Opper, Wim Wiegerinck, Ole Winther, and Onno Zoeter. Approximate inference techniques with expectation constraints. *Journal of Statistical Mechanics: Theory and Experiment*, 2005.

[HP99] Thomas Hofmann and Jan Puzicha. Latent class models for collaborative filtering. In *IJCAI '99: Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, pages 688–693, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc.

[iAN00] Shun ichi Amari and Hiroshi Nagaoka. *Methods of Information Geometry*, volume 191. Translations of Mathematical Monographs, 2000.

[Jac86] Peter Jackson. *Introduction to expert systems.* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1986.

[Jak04] Aleks Jakulin. Modelling modelled. Computer and Information Science, 2004.

[Jay57] E. T. Jaynes. Information theory and statistical mechanics. *Physical Review*, 106(4):620+, May 1957.

[Jay90] E. T. Jaynes. *Complexity, Entropy and the Physics of Information*, chapter Probability in Quantum Theory, pages 38–403. Addison–Wesley, 1990.

[Jay03] E. T. Jaynes. *Probability Theory : The Logic of Science.* G. Larry Bretthorst, 2003.

[Jen96] F. Jensen. *An introduction to Bayesian Networks.* Springer, 1996.

[JJ94] Finn Verner Jensen and Frank Jensen. Optimal junction trees. In *Uncertainty in Artificial Intelligence*, pages 360–366, 1994.

[JKP94] George H. John, Ron Kohavi, and Karl Pfleger. Irrelevant features and the subset selection problem. In *International Conference on Machine Learning*, pages 121–129, 1994.

[JLO90] F.V. Jensen, S.L. Lauritzen, and K.G. Olesen. Bayesian updating in causal probabilistic networks by local computations. *Computational Statistics Quaterly*, 4:269–282, 1990.

[Kal60] Rudolph E. Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82:35–45, 1960.

[Kar01] George Karypis. Evaluation of item-based top-n recommendation algorithms. In *CIKM*, pages 247–254, 2001.

[KFL01] Kschischang, Frey, and Loeliger. Factor graphs and the sum-product algorithm. *IEEETIT: IEEE Transactions on Information Theory*, 47, 2001.

[KGV83] S. Kirkpatrick, Cd Gelatt, and Mp Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.

[Knu05] Donald E. Knuth. *The Art of Computer Programming, Volume 4, Fascicle 2: Generating All Tuples and Permutations*. Addison-Wesley Professional, 2005.

[Koi05] Carla Koike. *Bayesian Approach to Action Selection and Attention Focusing. Application in Autonomous Robot Programming*. Thèse de doctorat, Inst. Nat. Polytechnique de Grenoble, Grenoble (FR), November 2005.

[Kol33] A. N. Kolmogorov. *Grundbegriffe der Wahrscheinlichkeitsrechnung*. Springer, Berlin, 1933. Second English Edition, *Foundations of Probability* 1950, published by Chelsea, New York.

[Kor04] K.P. Kording. Bayesian integration in sensorimotor learning. *Nature*, 15(427):244–7, Jan 2004.

[Koz92] John R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection (Complex Adaptive Systems)*. The MIT Press, December 1992.

[KS80] R. Kindermann and J.L. Snell. Markov random fileds and their applications. *American Mathematical Society*, 1980.

[KS96] Daphne Koller and Mehran Sahami. Toward optimal feature selection. In *International Conference on Machine Learning*, pages 284–292, 1996.

[KS04] Jon Kleinberg and Mark Sandler. Using mixture models for collaborative filtering. In *STOC '04: Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, pages 569–578. ACM Press, 2004.

[KW96] R.E. Kass and L.A. Wasserman. The selection of prior distributions by formal rules. *Journal of the American Statistical Association*, 91:1343 – 1370, 1996. A goood review.

[Lap14] Pierre Simon de Laplace. *Essai philosophique sur les probabilités*. Courcier, 1814.

[LBDM03] Olivier Lebeltel, Pierre Bessière, Julien Diard, and Emmanuel Mazer. Bayesian robots programming. *Autonomous Robot*, 16(1):49–79, 2003.

[LBM06] Julia A. Lasserre, Christopher M. Bishop, and Thomas P. Minka. Principled hybrids of generative and discriminative models. In *CVPR '06: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 87–94, Washington, DC, USA, 2006. IEEE Computer Society.

[Leb99] Olivier Lebeltel. *Programmation BayÃĬsienne des Robots*. PhD thesis, Institut National Polytechnique de Grenoble, France, 1999.

[LF05] Philippe Leray and Olivier Francois. Bayesian network structural learning and incomplete data. In *Proceedings of the International and Interdisciplinary Conference on Adaptive Knowledge Representation and Reasoning (AKRR 2005)*, pages 33–40, Espoo, Finland, 2005.

[LG99] Philippe Leray and Patrick Gallinari. Feature selection with neural networks. *Behaviormetrika*, 26, January 1999.

[LHABL04] Ronan Le Hy, Anthony Arrigoni, Pierre Bessière, and Olivier Lebeltel. Teaching bayesian behaviours to video game characters. *Robotics and Autonomous Systems*, 47:177–185, 2004.

[Lor99] Thomas J. Loredo. Computational technology for bayesian inference. In D. M. Mehringerand R. L. Plante and D. A. Roberts, editors, *Astronomical Data Analysis Software and Systems VIII*, volume 172 of *ASP Conference Series*, page 297, 1999.

[Loy02] Jim Loy. Louis Eichborn. http://www.jimloy.com/chess/eichborn.htm, 2002.

[LR94] David D. Lewis and Marc Ringuette. A comparison of two learning algorithms for text categorization. In *Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval*, pages 81–93, Las Vegas, US, 1994.

[LS95] H. Liu and R. Setiono. Chi2: Feature selection and discretization of numeric attributes. In *Proceedings of 7th IEEE Int'l Conference on Tools with Artificial Intelligence*, 1995.

[LTBS00] David J. Lunn, Andrew Thomas, Nicky Best, and David Spiegelhalter. Winbugs - a bayesian modelling framework: Concepts, structure, and extensibility. *Statistics and Computing*, 10(4):325–337, 2000.

[MA00] R. McEliece and S. M. Aji. The generalized distributive law. *EEE Trans. Inform. Theory*, 46:325–343, 2000.

[Mac67] J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In L. M. Le Cam and J. Neyman, editors, *Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297. University of California Press, 1967.

[Mac95] D.J.C. Mackay. Ensemble learning and evidence maximization. Technical report, Cavendish Laboratory, University of Cambridge, 1995.

[Man99] Christopher D. Manning. *Foundations of Statistical Natural Language Processing*. MIT Press, 1999.

[Mar04] Benjamin Marlin. Modeling user rating profiles for collaborative filtering. In Sebastian Thrun, Lawrence Saul, and Bernhard Schölkopf, editors, *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA, 2004.

[Mat] ChessBase Matrimony. Chessbase. http://www.chessbase.com/.

[May79] Peter S. Maybeck. *Stochastic models, estimation, and control*, volume 141 of *Mathematics in Science and Engineering*. Press, Academic, 1979.

[Min01a] Thomas Minka. The ep energy function and minimization schemes. Technical report, MIT Media Lab, 2001.

[Min01b] Thomas Minka. Expectation propagation for approximate bayesian inference. In *Proceedings of the 17th Annual Conference on Uncertainty in Artificial Intelligence (UAI-01)*, pages 362–36, San Francisco, CA, 2001. Morgan Kaufmann.

[Min01c] Thomas Minka. *A family of algorithms for approximate Bayesian inference*. PhD thesis, MIT Media Lab, 2001.

[Min05] Thomas Minka. Divergence measures and message passing. Technical report, Microsoft Research, 2005.

[Mit97] Tom M. Mitchell. *Machine Learning*. McGraw-Hill, March 1997.

[MT01] Dimitris Margartitis and Sebastian Thrun. A bayesian multiresolution independence test for continuous variables. In *Uncertainty in Artificial Intelligence: Proceedings of the Seventeenth Conference (UAI-2001)*, pages 346–353, San Francisco, CA, 2001. Morgan Kaufmann Publishers.

[MWJ99] Kevin Murphy, Yair Weiss, and Michael Jordan. Loopy belief propagation for approximate inference: An empirical study. In *Proceedings of the 15th Annual Conference on Uncertainty in Artificial Intelligence (UAI-99)*, San Francisco, CA, 1999. Morgan Kaufmann.

[Nea93] R. M. Neal. Probabilistic inference using Markov chain Monte Carlo methods. Technical Report CRG-TR-93-1, University of Toronto, University of Toronto, 1993.

[Nea96] Radford M. Neal. *Bayesian Learning for Neural Networks*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1996.

[Nea01] Radford M. Neal. Defining priors for distributions using dirichlet diffusion trees. Technical report, Technical Report No. 0104, Dept. of Statistics, University of Toronto, March 2001.

[NF77] P. Narendra and K. Fukunaga. A branch and bound algorithm for feature subset selection. In *IEEE Transactions on Computers*, pages 917–922, 1977. 26(9).

[ORP00] Nuria M. Oliver, Barbara Rosario, and Alex Pentland. A bayesian computer vision system for modeling human interactions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):831–843, 2000.

[Pas65] Blaise Pascal. TraitÃĬ du triangle arithmétique. online, 1665.

[Pat07] Arkadiusz Paterek. Improving regularized singular value decomposition for collaborative filtering. *Proceedings of KDD Cup and Workshop*, 2007.

[PD06] A. Pohorille and E. Darve. A Bayesian Approach to Calculating Free Energies in Chemical and Biological Systems. In *Bayesian Inference and Maximum Entropy Methods In Science and Engineering*, volume 872 of *American Institute of Physics Conference Series*, pages 23–30, November 2006.

[PDZ03] A. Pouget, P. Dayan, and R. S. Zemel. Inference and computation with population codes. *Annu Rev Neurosci*, 26:381–410, 2003.

[Pea88] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc., 1988.

[PHK+05] Cédric Pradalier, Jorje Hermosillo, Carla Koike, Christophe Braillon, Pierre Bessière, and Christian Laugier. The cycab: a car-like robot navigating autonomously and safely among pedestrians. *Robotics and Autonomous Systems*, 50(1):51–68, 2005.

[Poi12] H. Poincaré. *Calcul des probabilités*. Gauthier Villars, Paris, 1912.

[QM03] Yuan Qi and Thomas Minka. Expectation propagation for signal detection in flat-fading channels. In *Proceedings of IEEE International Symposium on Information Theory*, 2003.

[Qui93] Ross J. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, January 1993.

[Ram31] Frank Ramsey. *Foundations of Mathematics*. Kegan Paul, London, 1931.

[Ras00] C. E. Rasmussen. The infinite gaussian mixture model. In S.A. et al. Solla, editor, *Advances in information processing systems 12*, pages 554–560. MIT Press, 2000.

[RIS+94] P. Resnick, N. Iacovou, M. Suchak, P. Bergstorm, and J. Riedl. Grouplens: An open architecture for collaborative filtering of netnews. In *Proceedings of ACM 1994 Conference on Computer Supported Cooperative Work*, pages 175–186, Chapel Hill, North Carolina, 1994. ACM.

[RKFM02] Oliver Ritthoff, Ralf Klinkenberg, Simon Fischer, and Ingo Mierswa. A hybrid approach to feature selection and generation using an evolutionary algorithm. Technical Report CI-127/02, Collaborative Research Center 531, University of Dortmund, Germany, 2002.

[Run97] Bob Runyan. World football elo ratings. http://www.eloratings.net/, 1997.

[RW05] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005.

[Sav54] Leonard J. Savage. *The Foundations of Statistics*. Dover Publications, 1954.

[Sch78] Gideon E. Schwarz. Estimating the dimension of a model. *Annals of Statistics*, 6(2):461–464, 1978.

[SDHH98] Mehran Sahami, Susan Dumais, David Heckerman, and Eric Horvitz. A bayesian approach to filtering junk e-mail. In *Learning for Text Categorization: Papers from the 1998 Workshop*, Madison, Wisconsin, 1998. AAAI Technical Report WS-98-05.

[SDI06] Gregory Shakhnarovich, Trevor Darrell, and Piotr Indyk. *Nearest-Neighbor Methods in Learning and Vision: Theory and Practice (Neural Information Processing)*. The MIT Press, 2006.

[Ski88] J. Skilling. *Maximum-Entropy and Bayesian Methods in Science and Engineering*, chapter The Axioms of Maximum Entropy. G. J. Erickson and C. R. Smith, 1988.

[SM95] Upendra Shardanand and Patti Maes. Social information filtering: Algorithms for automating "word of mouth". In *Proceedings of ACM CHI'95 Conference on Human Factors in Computing Systems*, volume 1, pages 210–217, 1995.

[SMD03] H. Snoussi and A. Mohammad-Djafari. Information geometry and prior selection. In *AIP Conf. Proc. 659: Bayesian Inference and Maximum Entropy Methods in Science and Engineering*, pages 307–327, 2003.

[Smy04] Gordon K. Smyth. Linear models and empirical bayes methods for assessing differential expression in microarray experiments. *Statistical Applications in Genetics and Molecular Biology*, 13, 2004.

[Son05] Jeff Sonas. Chessmetrics. http://db.chessmetrics.com/, 2005.

[Spa99] Anne Spalanzani. *Algorithmes évolutionnaires pour l'étude de la robustesse des systèmes de reconnaissance de la parole*. PhD thesis, Université Joseph Fourier, 1999.

[Syv98] Anne Randi Syversveen. Noninformative bayesian priors. interpretation and problems with construction and applications. Technical report, Department of Mathematical Sciences, NTNU, Trondheim, 1998.

[TBF05] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, September 2005.

[Tip00] M. Tipping. The relevance vector machine. In *Advances in Neural Information Processing Systems, San Mateo, CA*. Morgan Kaufmann, 2000.

[TMC+07] Christopher Tay, Kamel Mekhnacha, Cheng Chen, Manuel Yguel, and Christian Laugier. An efficient formulation of the bayesian occupation filter for target tracking in dynamic environments. *International Journal Of Autonomous Vehicles*, 2007.

[Tuk65] James W. Cooleyand John W. Tukey. An algorithm for the machine calculation of complex fourier series. *Mathematics of Computation*, 19:297–201, 1965.

[Uff95] Jos Uffink. The constraint rule of the maximum entropy principle. *Studies in History and Philosophy of Modern Physics*, 27(1):47–79, 1995.

[Val84] L. G. Valiant. A theory of the learnable. In *STOC '84: Proceedings of the sixteenth annual ACM symposium on Theory of computing*, pages 436–445, New York, NY, USA, 1984. ACM Press.

[Vap99] Vladimir N. Vapnik. *The Nature of Statistical Learning Theory (Information Science and Statistics)*. Springer, November 1999.

[VEW93] K. S. Vines, R.F. Evilia, and S.L. Whittenburg. Bayesian analysis investigation of chemical exchange above and below the coalescence point. *Journal of physical chemistry*, 97:4941–4944, 1993.

[Vit67] Andrew J. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, 13:260–269, 1967.

[VM05] Manolis G. Vozalis and Konstantinos G. Margaritis. Applying svd on item-based filtering. In *ISDA '05: Proceedings of the 5th International Conference on Intelligent Systems Design and Applications*, pages 464–469, Washington, DC, USA, 2005. IEEE Computer Society.

[Wal06] J. M. Walsh. Dual optimality frameworks for expectation propagation. In *IEEE Conference on Signal Processing Advances in Wireless Communications (SPAWC)*, 2006.

[WB05] J. Winn and C. Bishop. Variational message passing. *Journal of Machine Learning Research*, 6:661–694, 2005.

[WCNiA03] Si Wu, Danmei Chen, Mahesan Niranjan, and Shun ichi Amari. Sequential bayesian decoding with a population of neurons. *Neural Comput.*, 15(5):993–1012, 2003.

[Web06] Brandyn Webb. Netflix update: Try this at home. http://sifter.org/~simon/journal/20061211.html, 2006.

[Wil07] Darren J J. Wilkinson. Bayesian methods in bioinformatics and computational systems biology. *Brief Bioinform*, April 2007.

[WMG+06] Jianshu Weng, Chunyan Miao, Angela Goh, Zhiqi Shen, and Robert Gay. Trust-based agent community for collaborative recommendation. In *AAMAS '06: Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, pages 1260–1262, New York, NY, USA, 2006. ACM Press.

[WMP+01] J. Weng, J. McClelland, A. Pentland, O. Sporns, I. Stockman, M. Sur, and E. Thelen. Autonomous mental development by robots and animals. *Science*, 291(291):599–600, Jan 2001.

[WMT05] Max Welling, Thomas Minka, and Yee W. Teh. Structured region graphs: Morphing ep into gbp. In *Proceedings of the 21th Annual Conference on Uncertainty in Artificial Intelligence (UAI-05)*, pages 609–616, 2005.

[Wol94] David R. Wolf. Mutual Information as a Bayesian Measure of Independence, 1994. arXiv:comp-gas/9511002.

[WP63] B. Welch and H. Peers. On formulae for confidence points based on integrals of weighted likelihoods. *journal of Royal Statistical Society*, 25:318–329, 1963.

[YFW00] Jonathan S. Yedidia, William T. Freeman, and Yair Weiss. Generalized belief propagation. In Todd K. Leen, Thomas G. Dietterich, and Volker Tresp, editors, *NIPS*, pages 689–695. MIT Press, 2000.

[YFW03] Jonathan S. Yedidia, William T. Freeman, and Yair Weiss. *Understanding belief propagation and its generalizations*, pages 239–269. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2003.

[YH98] Jihoon Yang and Vasant Honavar. Feature subset selection using a genetic algorithm. *IEEE Intelligent Systems*, 13:44–49, 1998.

[You01] S. Youssef. Physics with exotic probability theory. *ArXiv High Energy Physics - Theory e-prints*, October 2001.

[YSX94] Alan L. Yuille, Stelios M. Smirnakis, and Lei Xu. Bayesian self-organization. In Jack D. Cowan, Gerald Tesauro, and Joshua Alspector, editors, *Advances in Neural Information Processing Systems*, volume 6, pages 1001–1008. Morgan Kaufmann Publishers, Inc., 1994.

[ZH05] Onno Zoeter and Tom Heskes. Gaussian quadrature based expectation propagation. In Z. Ghahramani and R. Cowell, editors, *AISTATS*, 2005.

[ZR95] Huaiyu Zhu and Richard Rohwer. Information geometric measurements of generalization. Technical Report NCRG4350, Dept. Comp. Sci. and Appl. Math., Aston University, August 1995.

Annexe A

Annexe

A.1 Famille exponentielle

A.1.1 Définition

Définition 11. *Un ensemble de distributions sur $\mathbb{R}^d$ est une famille exponentielle si elles s'écrivent*

$$p_{\boldsymbol{\theta}}(\mathbf{x}) = \frac{1}{Z(\boldsymbol{\theta})} \exp\left(\boldsymbol{\theta}^\top \boldsymbol{\psi}(\mathbf{x})\right), \tag{A.1}$$

où $\boldsymbol{\psi}(\boldsymbol{X})$ est un vecteur de statistiques suffisantes et $Z(\boldsymbol{\theta})$ une constante de normalisation. Le vecteur $\boldsymbol{\theta}$ est appelé vecteur des paramètres canoniques.

Une famille est donc définie par la donnée de $\boldsymbol{\psi}(\boldsymbol{X})$ et elle est paramétrée par $\boldsymbol{\theta}$. Cette définition se généralise au cas discret et de nombreuses distributions usuelles font partie de familles exponentielles (distribution gaussienne, gamma, exponentielle, bêta, Dirichlet, Bernoulli, binomiale, multinomiale, Poisson, binomiale négative, géométrique).

A.1.2 Propriétés

Entropie

Les familles exponentielles sont liées au principe de maximum d'entropie. En effet la distribution de plus grande entropie vérifiant des contraintes sur des espérances fait partie d'une famille exponentielle. Ses paramètres canoniques sont les multiplicateurs de Lagrange introduits lors de la maximisation. Le vecteur $\boldsymbol{\psi}(\mathbf{X})$ est défini par la définition des contraintes et $\boldsymbol{\theta}$ par leurs niveaux.

Conjugaison

Si la fonction de vraisemblance d'un modèle probabiliste est membre de la famille exponentielle, alors elle admet un *a priori* conjugué, lui aussi dans une famille exponentielle. Par exemple l'*a priori* conjugué sur les paramètres d'une distribution multinomiale est une

distribution de Dirichlet, celui sur la moyenne d'une gaussienne de variance fixée est une autre gaussienne.

Multiplication

Les distributions de familles exponentielles ont d'agréables propriétés algébriques. Par exemple, pour deux distributions de la même famille :

$$p_{\boldsymbol{\theta}_1}(\mathbf{x}) \cdot p_{\boldsymbol{\theta}_2}(\mathbf{x}) \propto p_{\boldsymbol{\theta}_1+\boldsymbol{\theta}_2}(\mathbf{x}).$$

A.2 Distribution gaussienne

A.2.1 Définition

Définition 12. *Un vecteur aléatoire* $\mathbf{X}$ *réel de dimension* d *est dit gaussien ssi sa fonction caractéristique s'écrit :*

$$\phi_{\mathbf{X}}(\boldsymbol{u}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \exp\left(i\boldsymbol{\mu}^\top \mathbf{u} - \frac{1}{2}\mathbf{u}^\top \boldsymbol{\Sigma}\right),$$

avec $\boldsymbol{\mu}$ *un vecteur de* d *réels et* $\boldsymbol{\Sigma}$ *une matrice symétrique réelle semi-définie positive.*

Le vecteur $\boldsymbol{\mu}$ est l'espérance de $\mathbf{X}$ et $\boldsymbol{\Sigma}$ sa matrice de covariance. Nous notons $\mathbf{x} \sim \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$. Souvent $\boldsymbol{\Sigma}$ est inversible et la densité de $\mathbf{X}$ s'écrit :

$$p(\mathbf{x}) = (2\pi)^{-d/2} \left(\det(\boldsymbol{\Sigma})\right)^{-1/2} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right).$$

La densité de la gaussienne centrée réduite $\mathcal{N}(\mathbf{x}; \mathbf{0}, \boldsymbol{I})$ sera notée $\mathcal{N}(\mathbf{x})$. Dans le cas où $d = 1$, la fonction de *répartition* est définie par $\Phi(t; \mu, \sigma^2) = p_{x\sim\mathcal{N}(x;\mu,\sigma^2)}(x \leq t)$. Alors $\Phi(.; 0, 1)$ sera notée $\Phi(.)$.

A.2.2 Paramétrisation canonique

La distribution gaussienne faisant partie de la famille exponentielle, elle admet une paramétrisation canonique. Nous noterons $\mathbf{x} \sim \mathcal{G}(\mathbf{x}; \boldsymbol{\eta}, \boldsymbol{\Lambda}) = [\boldsymbol{\eta}, \boldsymbol{\Lambda}]$.

$$p(\mathbf{x}) = \exp\left(\alpha + \boldsymbol{\eta}^\top \mathbf{x} - \frac{1}{2}\mathbf{x}^\top \boldsymbol{\Lambda}\mathbf{x}\right)$$

La valeur $\alpha = -\frac{1}{2}(d \log 2\pi - \log\det \boldsymbol{\Lambda} + \boldsymbol{\eta}^\top \boldsymbol{\Lambda}^{-1}\boldsymbol{\eta})$ est une constante de normalisation. Dans certains cas, la paramétrisation canonique est plus pratique que la paramétrisation par les moments. Les paramètres sont liés par :

$$\begin{cases} \boldsymbol{\eta} &= \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu} \\ \boldsymbol{\Lambda} &= \boldsymbol{\Sigma}^{-1} \end{cases} \qquad \begin{cases} \boldsymbol{\mu} &= \boldsymbol{\Lambda}^{-1}\boldsymbol{\eta} \\ \boldsymbol{\Sigma} &= \boldsymbol{\Lambda}^{-1}. \end{cases}$$

A.2.3 Propriétés

Transformations affines

Soit $\mathbf{x} \sim \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$ un vecteur gaussien d-dimensionnel, A une matrice de taille q par d et $\boldsymbol{b}$ un vecteur q-dimensionnel. Si $\mathbf{z} = A\mathbf{x} + \boldsymbol{b}$ alors

$$\mathbf{z} \sim \mathcal{N}\left(\mathbf{z}; A\boldsymbol{\mu} + \boldsymbol{b}, A\boldsymbol{\Sigma}A^\top\right).$$

Somme

Soient $\mathbf{x}_1 \sim \mathcal{N}(\mathbf{x}_1; \boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)$ et $\mathbf{x}_2 \sim \mathcal{N}(\mathbf{x}_2; \boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2)$ deux vecteurs gaussiens d-dimensionnels. Si $\mathbf{z} = \mathbf{x}_1 + \mathbf{x}_2$ alors

$$\mathbf{z} \sim \mathcal{N}(\mathbf{z}; \boldsymbol{\mu}_1 + \boldsymbol{\mu}_2, \boldsymbol{\Sigma}_1 + \boldsymbol{\Sigma}_2).$$

Produit

Soit la densité $p(\mathbf{x}) \propto p_1(\mathbf{x})p_2(\mathbf{x})$ proportionnelle au produit de deux gaussiennes. Alors

$$\mathbf{x} \sim \mathcal{G}(\mathbf{x}; \boldsymbol{\eta}_1 + \boldsymbol{\eta}_2, \boldsymbol{\Lambda}_1 + \boldsymbol{\Lambda}_2).$$

Convolution

Soit deux vecteurs $\mathbf{x}$ et $\mathbf{y}$ tels que $p(\mathbf{x}, \mathbf{y}) = p(\mathbf{y}|\mathbf{x})\, p(\mathbf{x})$ et

$$\begin{aligned} \mathbf{x} &\sim \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) \\ \mathbf{y}|\mathbf{x} &\sim \mathcal{N}(\mathbf{y}; A\mathbf{x}, \boldsymbol{\Gamma}) \end{aligned}$$

alors la marginale de $\mathbf{y}$ est la convolution de ces deux distributions

$$\mathbf{y} \sim \mathcal{N}\left(\mathbf{x}; A\boldsymbol{\mu}, \boldsymbol{\Gamma} + A\boldsymbol{\Sigma}A^\top\right).$$

On remarque que la distribution de $\mathbf{y}$ a sensiblement la même forme que celle d'une somme de deux variables, $\mathbf{y}$ étant la somme de $\mathbf{x}$ et d'un incrément suivant la distribution de $\mathbf{y}|\mathbf{x}$. L'incertitude sur $\mathbf{y}$ est la somme de celle sur $\mathbf{x}$ et de celle de l'incrément. En effet, la distribution d'une somme de deux variables est la convolution de leurs distributions.

De plus

$$\mathbf{x}|\mathbf{y} \sim \mathcal{N}\left(\mathbf{x}; \boldsymbol{\Psi}\left(A^\top\boldsymbol{\Gamma}^{-1}\mathbf{y} + \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}\right), \boldsymbol{\Psi}\right)$$

avec $\boldsymbol{\Psi} = \left(A^\top\boldsymbol{\Gamma}^{-1}A + \boldsymbol{\Sigma}^{-1}\right)^{-1}$.

Commentaires

Il existe une dualité entre les deux paramétrisations. La paramétrisation par les moments est naturelle pour la distribution de probabilités et la paramétrisation canonique est naturelle pour la fonction caractéristique. Or la fonction caractéristique est la transformée de Fourier de la distribution. De plus, la famille des gaussiennes a la propriété d'être close par transformation de Fourier. En substance, la transformée d'une gaussienne est une gaussienne de variance inverse.

Cette dualité permet de comprendre les propriétés des gaussiennes. Par exemple nous avons vu que pour obtenir le produit de deux gaussiennes, il suffit d'ajouter leurs paramètres canoniques. Comme la transformée d'un produit est un produit de convolution (et inversement), nous en déduisons que pour calculer la convolution (conditionnement) de gaussiennes, il faut faire le produit de leur transformées. Donc ajouter leurs paramètres canoniques dans l'espace de Fourier. Or ces paramètres sont les moments. Donc pour calculer une convolution, il suffit d'ajouter les variances.

Marginalisation

Soit le vecteur composé $\mathbf{x}$ tel que

$$\mathbf{x} \sim \mathcal{N}\left(\left[\begin{array}{c} \mathbf{x}_1 \\ \mathbf{x}_2 \end{array}\right]; \left[\begin{array}{c} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \end{array}\right], \left[\begin{array}{cc} \boldsymbol{\Sigma}_{11} & \boldsymbol{\Sigma}_{12} \\ \boldsymbol{\Sigma}_{12}^\top & \boldsymbol{\Sigma}_{22} \end{array}\right]\right).$$

Alors

$$\begin{aligned}
\mathbf{x}_1 &\sim \mathcal{N}\left(\mathbf{x}_1; \boldsymbol{\mu}_1, \boldsymbol{\Sigma}_{11}\right) \\
\mathbf{x}_2 &\sim \mathcal{N}\left(\mathbf{x}_2; \boldsymbol{\mu}_2, \boldsymbol{\Sigma}_{22}\right) \\
\mathbf{x}_1|\mathbf{x}_2 &\sim \mathcal{N}\left(\mathbf{x}_1; \boldsymbol{\mu}_1 + \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}(\mathbf{x}_2 - \boldsymbol{\mu}_2), \boldsymbol{\Sigma}_{11} - \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{12}^\top\right) \\
\mathbf{x}_2|\mathbf{x}_1 &\sim \mathcal{N}\left(\mathbf{x}_2; \boldsymbol{\mu}_2 + \boldsymbol{\Sigma}_{12}^\top\boldsymbol{\Sigma}_{11}^{-1}(\mathbf{x}_1 - \boldsymbol{\mu}_1), \boldsymbol{\Sigma}_{22} - \boldsymbol{\Sigma}_{12}^\top\boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{\Sigma}_{12}\right).
\end{aligned}$$

Translation et changement d'échelle

Le premier paramètre $\boldsymbol{\mu}$ est un paramètre de position, de centrage de la distribution alors que le second un paramètre d'échelle, induisant une distance dite de Mahalanobis.

$$\begin{aligned}
\mathcal{N}\left(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}\right) &= \left(\det \boldsymbol{\Sigma}\right)^{-\frac{1}{2}} \mathcal{N}\left(\boldsymbol{\Sigma}^{-\frac{1}{2}}(\mathbf{x} - \boldsymbol{\mu})\right) \\
\Phi\left(t; \mu, \sigma^2\right) &= \Phi\left(\frac{t - \mu}{\sigma}\right)
\end{aligned}$$

Cas unidimensionnel

Soit X une variable réelle. Les deux premiers moments sont $E[X] = \mu$ et $E[X^2] = \boldsymbol{\Sigma} + \boldsymbol{\mu}\boldsymbol{\mu}^\top = \sigma^2 + \mu^2$. L'écart type est σ et l'intervalle $[\mu - 2\sigma, \mu + 2\sigma]$ contient 95% de la masse de probabilité.

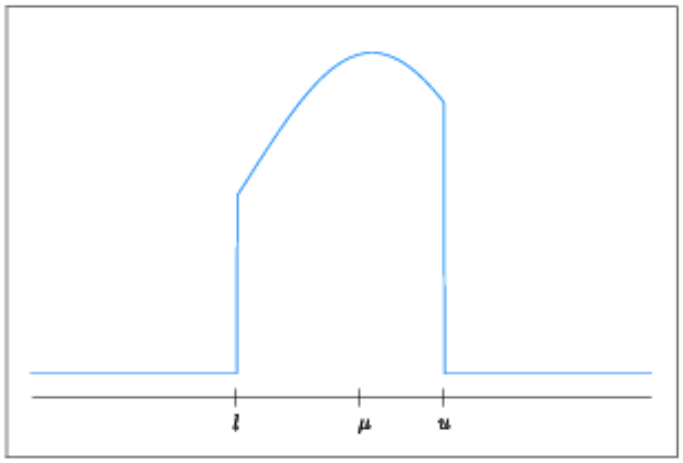


FIG. A.1 – Une gaussienne tronquée rectifiée est normalisée.

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}}\, e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Notons τ et π les paramètres canoniques unidimensionnels.

$$\begin{cases} \tau &= \mu/\sigma^2 \\ \pi &= 1/\sigma^2 \end{cases} \qquad\qquad \begin{cases} \mu &= \tau/\pi \\ \sigma^2 &= 1/\pi \end{cases}$$

Le paramètre π est appelé *précision* de la gaussienne. Lorsque cela ne sera pas source d'ambiguïté, nous noterons $\mathcal{G}(x;\tau,\pi) = [\tau,\pi]$.

La paramétrisation canonique nous permet de définir une distribution uniforme impropre sur $\mathbb{R}$ par $\mathcal{G}(x;0,0)$, distribution de précision nulle, de variance infinie et dont l'espérance n'est pas définie. Cette distribution peut représenter un état d'absence d'information sur la valeur de x.

Symétries

La densité gaussienne centrée réduite est à symétrie sphérique.

$$\begin{aligned} \mathcal{N}(-\mathbf{x}) &= \mathcal{N}(\mathbf{x}) \\ \Phi(t) &= 1 - \Phi(-t) \\ \Phi\left(t;\mu,\sigma^2\right) &= 1 - \Phi\left(-t;-\mu,\sigma^2\right) \end{aligned}$$

Gaussienne tronquée rectifiée

Définition 13. *Une gaussienne tronquée rectifiée (fig. A.1) est une gaussienne unidimensionnelle doublement tronquée et renormalisée définie par la densité*

$$\mathcal{R}\left(x;\mu,\sigma^2,l,u\right) := \mathbb{1}_{x\in[l,u]}\,\frac{\mathcal{N}\left(x;\mu,\sigma^2\right)}{\Phi\left(u;\mu,\sigma^2\right)-\Phi\left(l;\mu,\sigma^2\right)}$$
$$= \mathbb{1}_{x\in[l,u]}\,\frac{\mathcal{N}\left(\frac{x-\mu}{\sigma}\right)}{\sigma\cdot\left(\Phi\left(\frac{u-\mu}{\sigma}\right)-\Phi\left(\frac{l-\mu}{\sigma}\right)\right)}.$$

Si la gaussienne n'est tronquée qu'à gauche, nous parlons de gaussienne simplement tronquée rectifiée.

$$\mathcal{R}\left(x;\mu,\sigma^2,l\right) := \lim_{u\to+\infty}\mathcal{R}\left(x;\mu,\sigma^2,l,u\right)$$

On a de plus

$$\lim_{l\to-\infty}\mathcal{R}\left(x;\mu,\sigma^2,l\right) = \mathcal{N}\left(x;\mu,\sigma^2\right).$$

Les moyenne et variance d'une gaussienne doublement tronquée rectifiée sont :

$$\begin{aligned} E_{\mathcal{R}}[X] &= \mu+\sigma\cdot v\left(\frac{\mu}{\sigma},\frac{l}{\sigma},\frac{u}{\sigma}\right)\\ Var_{\mathcal{R}}[X] &= \sigma^2\cdot\left(1-w\left(\frac{\mu}{\sigma},\frac{l}{\sigma},\frac{u}{\sigma}\right)\right), \end{aligned} \tag{A.2}$$

où les fonctions v et w sont données par

$$v(t,l,u) := \frac{\mathcal{N}\left(l-t\right)-\mathcal{N}\left(u-t\right)}{\Phi(u-t)-\Phi(l-t)}$$
$$w(t,l,u) := v^2(t,l,u)+\frac{(u-t)\cdot\mathcal{N}\left(u-t\right)-(l-t)\cdot\mathcal{N}\left(l-t\right)}{\Phi(u-t)-\Phi(l-t)}.$$

Dans notre application TrueChess, nous utiliserons les fonctions :

$$\begin{aligned} W_{>}(t,\epsilon) &:= w(t,\epsilon,\infty)\\ W_{|.|}(t,\epsilon) &:= w(t,-\epsilon,\epsilon)\\ W_{<}(t,\epsilon) &:= w(t,-\infty,-\epsilon). \end{aligned} \tag{A.3}$$

A.3 Football

Nous avons appliqué notre méthode de classement à des résultats de matchs de football, de la ligue 1 française et des phases finales des coupes du monde FIFA. Les paramètres sont ceux maximisant la preuve et le modèle est celui à marges fixes. Toutes les dates ont été arrondies à l'année. Les données précisent le nombre de buts marqués par chaque équipe, mais nous n'avons pas pris en compte cette information et n'avons conservé que le nom du vainqueur. Un meilleur modèle, par exemple $p_1 - p_2 > n\ \epsilon$, avec n le nombre de buts de différence en cas de victoire de p_1, aurait certainement une meilleure vraisemblance, mais nous voulons simplement illustrer la généralité de TrueChess.

A.3.1 Classement de clubs français

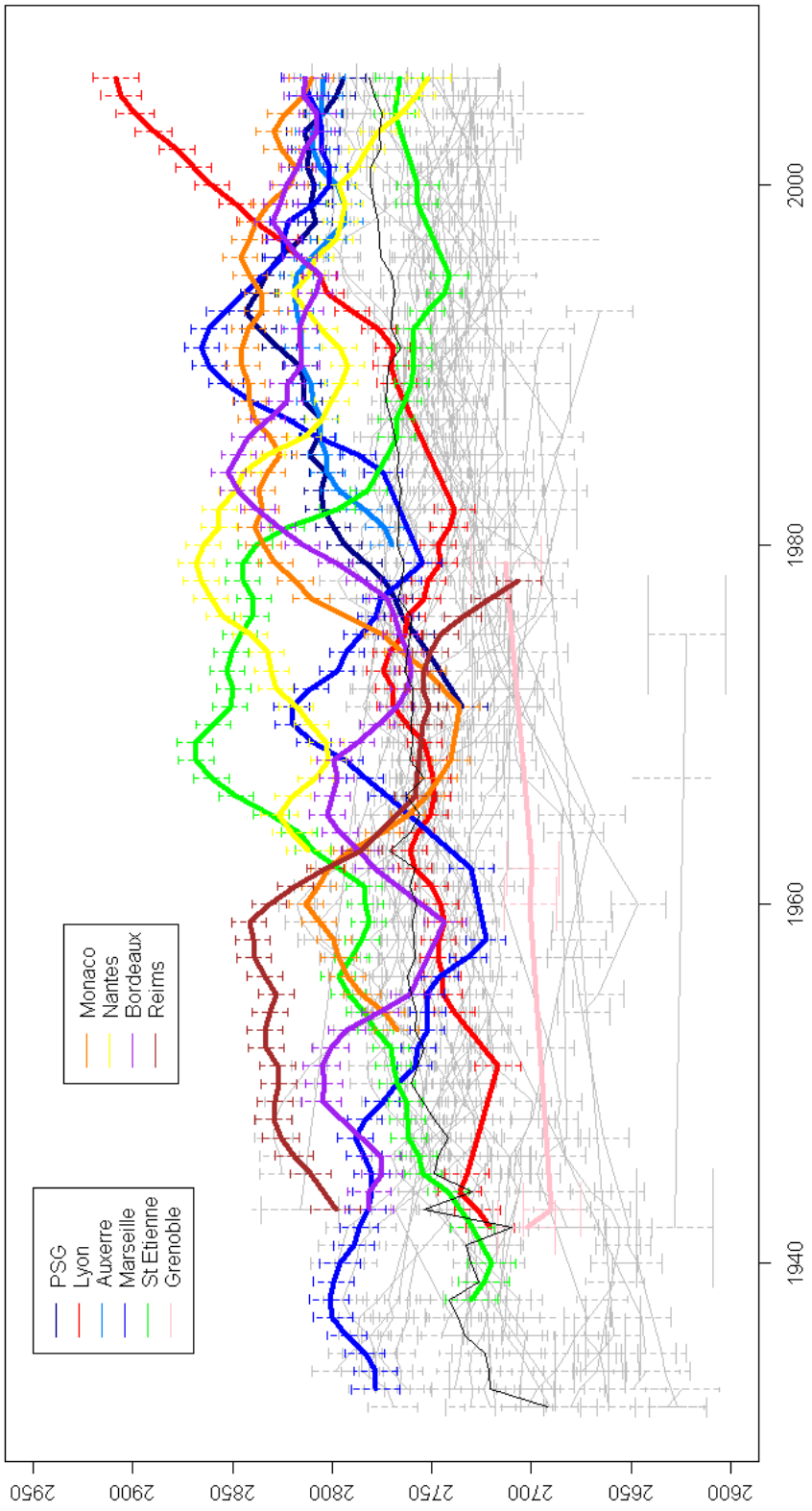


FIG. A.2 – Application de notre méthode de cotation aux équipes de football de la première division française (ligue 1) entre 1932 et 2007. Le jeu de 24 254 matchs comprend 25% de nuls et 71 équipes différentes. Notons la nette domination de Lyon ces dernières années, ainsi que les règnes de Reims, Saint-Étienne, Nantes et Marseille. Le niveau moyen (noir) augmente légèrement.

A.3.2 Classement internationnaux

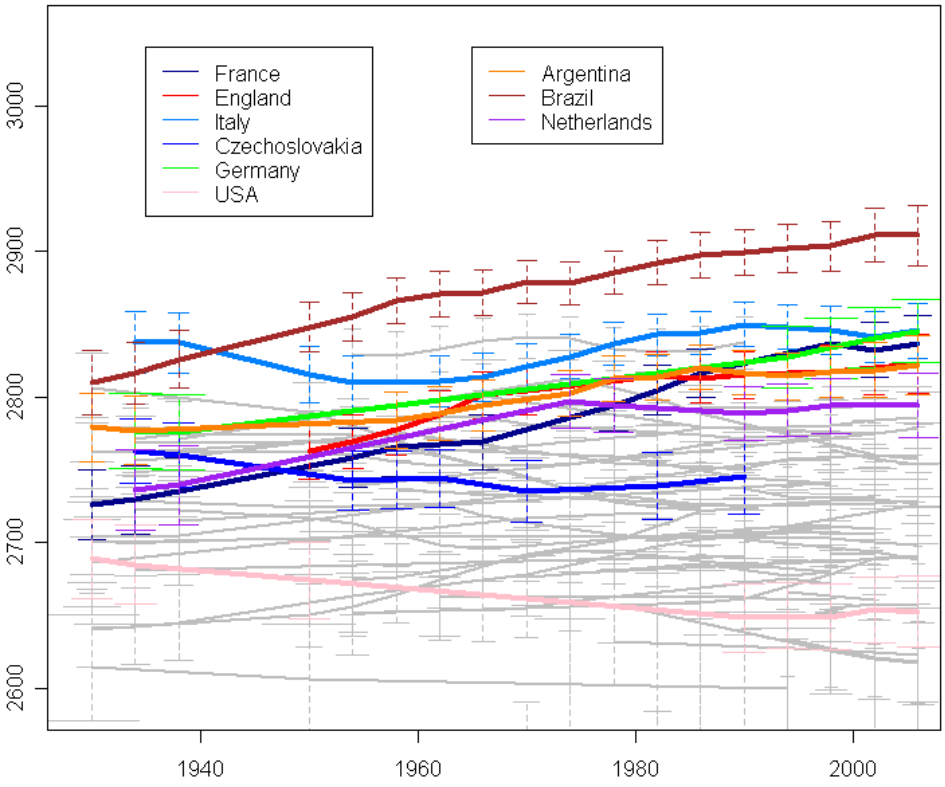


FIG. A.3 – Application de notre méthode de cotation aux équipes de football internationales. Les 708 matchs sont ceux des phases finales des coupes du monde FIFA depuis 1930. Le taux de matchs nuls est de 22.1%, ce qui est plus faible que pour le championnat national (la cause étant le principe même d'une coupe). Nous notons que les forces relatives des équipes internationales sont beaucoup plus stables dans le temps que celles des équipes de ligue 1. Le Brésil domine le football mondial. La France, L'Allemagne et l'Italie sont ses plus sérieux adversaires.

Bayesian learning : foundations, method and applications.

Abstract.

Bayesian probabilities are an efficient tool for addressing machine learning issues. However, because such problems are often difficult, trade-offs between accuracy and efficiency must be implemented. Our work presents the Bayesian learning method, its philosophical foundations and several innovative applications. Firstly, we study two data analysis problem with hidden variables. We propose a method for ranking chess players and a collaborative filtering system for movie recommendations. The second part of our work deals with model learning, with the selection and the creation of relevant variables for a robotic application.

Keywords : bayesian learning, subjective probabilities, generative models, ranking from pairwise comparisons, chess, collaborative filtering, feature selection, model selection, robotics.

Fondations, méthode et applications de l'apprentissage bayésien.

Résumé.

Le domaine de l'apprentissage automatique a pour but la création d'agents synthétiques améliorant leurs performances avec l'expérience. Pour pouvoir se perfectionner, ces agents extraient des régularités statistiques de données incertaines et mettent à jour leur modèle du monde. Les probabilités bayésiennes sont un outil rationnel pour répondre à la problématique de l'apprentissage. Cependant, comme ce problème est souvent difficile, des solutions proposant un compromis entre précision et rapidité doivent être mises en œuvre. Ce travail présente la méthode d'apprentissage bayésien, ses fondations philosophiques et plusieurs applications innovantes.

Nous nous intéressons d'abord à des questions d'apprentissage de paramètres. Dans ce cadre nous étudions deux problèmes d'analyse de données à variables cachées. Nous proposons d'abord une méthode bayésienne pour classer les joueurs d'échecs qui améliore sensiblement le système Elo. Le classement produit permet de répondre à des questions intéressantes comme celle de savoir qui fut le meilleur joueur d'échecs de tous les temps. Nous étudions aussi un système de filtrage collaboratif dont le but est de prévoir les goûts cinématographiques d'utilisateurs en fonction de leurs préférences passées.

La deuxième partie de notre travail concerne l'apprentissage de modèles. D'abord nous nous intéressons à la sélection de variables pertinentes dans le cadre d'une application robotique. D'un point de vue cognitif, cette sélection permet au robot de transférer ses connaissances d'un domaine sensorimoteur vers un autre. Finalement, nous proposons une méthode permettant de découvrir automatiquement une nouvelle variable cachée afin de mieux modéliser l'environnement d'un robot.

Mots clés : apprentissage bayésien, probabilités subjectives, modèle génératif, classement historique, échecs, filtrage collaboratif, sélection de variables, sélection de modèle, robotique.