École Doctorale n° 72 : Sciences Pour l'Ingénieur

THÈSE

pour obtenir le grade de docteur délivré par

l'Université des Sciences et Technologies de Lille : Lille 1
Spécialité doctorale "Informatique"

présentée et soutenue publiquement par

Hadrien Glaude

le 8 Juillet 2016

Méthodes des moments pour l'inférence de systèmes séquentiels linéaires rationnels

Directeur de thèse : **Olivier Pietquin**

Jury

Marc Tommasi,	Professeur à l'Université Lille 3	Président
François Denis,	Professeur à l'Université Aix-Marseille	Rapporteur
Joëlle Pineau,	Professeur à l'Université McGill	Rapporteur
Odalric-Ambrym Maillard,	Chargé de recherche à l'INRIA	Examinateur
Cyrille Enderli,	Ingénieur Thalès	Examinateur
Olivier Pietquin,	Professeur à l'Université Lille 1	Directeur

Centre de Recherche en Informatique, Signal et Automatique de Lille (CRIStAL),
UMR 9189, équipe SequeL, 59650, Villeneuve d'Ascq, France

Remerciements

Tout d'abord, je souhaite remercier Olivier Pietquin, mon directeur de thèse qui au fil de l'aventure est devenu un ami. Ses conseils avisés, sa disponibilité et son expérience ont été d'une grande aide dans l'aboutissement de ce travail. J'aimerais également lui dire à quel point j'ai apprécié ses retours toujours pertinents sur mon travail même dans des délais très serrés afin de soumettre à temps. Enfin, j'ai été extrêmement sensible à la confiance qu'il m'a accordée et à son soutien inconditionnel.

Je remercie mes rapporteurs François Denis et Joëlle Pineau pour l'intérêt qu'ils ont porté à mon manuscrit et leurs remarques pertinentes. Je remercie Odalric-Ambrym Maillard en tant qu'examinateur mais aussi pour ces discussions aussi passionnantes que techniques. Enfin, je remercie Marc Tommasi d'avoir accepté de présider le jury.

Je remercie ma femme, Elisa, pour sa patience face aux déplacements fréquents et au rythme parfois chaotique. Je suis reconnaissant envers mes parents de m'avoir poussé et soutenu tout au long de mes études.

Enfin, le plus important, je remercie les membres du laboratoire pour tous les moments partagés au quotidien. Bilal, Timothé, Denis, Julien, Merwan, Alexandre, nous avons passé de bons moments à Metz et à Lille, enfin surtout à Lille. Je remercie aussi les doctorants et stagiaires du laboratoire d'intelligence artificielle de Thales, où tout a commencé avec Costin et Pierre-Yves puis Romain et Ludo. Enfin, je remercie Philippe Preux, directeur de l'équipe SequeL à Lille, de m'avoir accueilli. Ces années de thèse, longues et mouvementées, m'ont amené à rencontrer des gens passionnants en nombre trop important pour tous les citer.

Résumé

L'apprentissage de modèles stochastiques générant des séquences a de nombreuses applications en traitement de la parole, du langage ou bien encore en bio-informatique. Les Automates à Multiplicité (MA) sont des modèles graphiques à variables latentes qui englobent une grande variété de systèmes linéaires pouvant en particulier représenter des langues stochastiques, des processus stochastiques ainsi que des processus contrôlés. Les algorithmes traditionnels d'apprentissage comme celui de Baum-Welch sont itératifs, lent et peuvent converger vers des optima locaux. Une alternative récente consiste à utiliser la méthode des moments (MoM) pour concevoir des algorithmes rapides et consistent avec des garanties pseudo-PAC.

Cependant, les algorithmes basés sur la MoM ont deux inconvénients principaux. Tout d'abord, les garanties PAC ne sont valides que si la dimension du modèle appris correspond à la dimension du modèle cible. Deuxièmement, bien que les algorithmes basés sur la MoM apprennent une fonction proche de la distribution cible, la plupart ne contraignent pas celle-ci à être une distribution. Ainsi, un modèle appris à partir d'un nombre fini d'exemples peut renvoyer des valeurs négatives et qui ne somment pas à un.

Ainsi, cette thèse s'adresse à ces deux problèmes. D'abord, nous proposons un élargissement des garanties théoriques pour les modèles compressés, ainsi qu'un algorithme spectral régularisé qui adapte la taille du modèle aux données. Puis, une application en guerre électronique est aussi proposée pour le séquencement des écoutes du récepteur superhétérodyne. D'autre part, nous dérivons de nouveaux algorithmes d'apprentissage ne souffrant pas du problème des probabilités négatives et dont certains bénéficient de garanties PAC.

Mots-clefs : Apprentissage automatique ; Moments, Méthode des (statistique) ; Renseignement électronique ; Modèles stochastiques d'apprentissage ; Analyse séquentielle ; Processus stochastiques ; Inférence grammaticale ; Automates pondérés

Abstract

Learning stochastic models generating sequences has many applications in natural language processing, speech recognitions or bioinformatics. Multiplicity Automata (MA) are graphical latent variable models that encompass a wide variety of linear systems. In particular, they can model stochastic languages, stochastic processes and controlled processes. Traditional learning algorithms such as the one of Baum-Welch are iterative, slow and may converge to local optima. A recent alternative is to use the Method of Moments (MoM) to design consistent and fast algorithms with pseudo-PAC guarantees.

However, MoM-based algorithms have two main disadvantages. First, the PAC guarantees hold only if the size of the learned model corresponds to the size of the target model. Second, although these algorithms learn a function close to the target distribution, most do not ensure it will be a distribution. Thus, a model learned from a finite number of examples may return negative values or values that do not sum to one.

This thesis addresses both problems. First, we extend the theoretical guarantees for compressed models, and propose a regularized spectral algorithm that adjusts the size of the model to the data. Then, an application in electronic warfare is proposed to sequence of the dwells of a super-heterodyne receiver. Finally, we design new learning algorithms based on the MoM that do not suffer the problem of negative probabilities. We show for one of them pseudo-PAC guarantees.

Keywords : Machine learning ; Method of Moments ; Electronic Support ; Stochastic models ; Stochastic languages ; Stochastic process ; Grammatical inference ; Multiplicity automata

Table des matières

Liste des figures

Liste des tableaux

Introduction

Dans un grand nombre de domaines, comme le traitement du langage naturel, la robotique, la bio-informatique, les données sont présentées comme une succession de symboles. En reconnaissance de la parole, ces symboles correspondent aux informations fréquentielles extraites pendant des courtes tranches de temps successives. En traitement du langage naturel, les symboles sont plus naturellement formés par les mots qui s'enchaînent pour former des phrases selon un ensemble de règles complexes. En bio-informatique, on cherche à aligner des séquences biologiques telles que l'ADN pour retrouver des mutations. En sécurité informatique, la plupart des systèmes enregistre un historique des événements que l'on peut analyser afin de détecter des anomalies telles que des dysfonctionnements ou des intrusions. En guerre électronique, on cherche à décoder l'activité des radars adverses interceptés sous forme d'impulsions électromagnétiques. Enfin en robotique, les séquences d'actions prises par le robot modifient l'environnement perçu sous forme de séquences d'observations. Dès lors, on recherche quelle séquence d'actions prendre pour accomplir une tâche.

Dans les exemples donnés, la première étape avant de pouvoir raisonner est la construction d'un modèle, constitué d'un ensemble de règles permettant la génération de ces séquences. Lorsque l'aléatoire est inhérent au problème ou qu'il est introduit dans la modélisation à des fins de robustesse, on utilise des règles stochastiques décrivant le comportement moyen des séquences. Une approche possible est d'apprendre ces modèles à partir d'exemples de séquences. Une fois le modèle génératif appris, on peut l'utiliser à de nombreuses fins comme la classification, la prédiction ou le décodage. Dans cette thèse, nous envisageons comme application la commande d'un récepteur d'ondes électromagnétiques sélectifs en fréquence. Dans chaque bande de fréquence, des radars émettent plus ou moins périodiquement. En modélisant par un processus contrôlé leur interception par le récepteur, on peut choisir la prochaine bande de fréquence à écouter. Finalement, ce travail se situe à l'intersection de deux grandes problématiques : l'inférence de systèmes séquentiels et l'apprentissage statistique décrit ci-dessous.

Inférence de systèmes séquentiels

L'apprentissage automatique est un domaine de l'informatique qui s'intéresse à l'inférence de modèles au travers d'un algorithme travaillant à partir d'exemples. Dans cette thèse, on s'intéresse à l'apprentissage de distributions sur des séquences. De telles distributions peuvent être très complexes. Dans le cas général, il faudrait préciser une probabilité pour chaque séquence de symboles. Ainsi, pour limiter la complexité de la tâche, on supposera que ces distributions sont réalisées par des modèles particuliers tels que les systèmes séquentiels linéaires qui ont un nombre fini de paramètres.

Les systèmes séquentiels linéaires restent néanmoins très généraux. On peut les

utiliser pour modéliser un langage stochastique (une distribution sur tous les mots de tailles finies) ou pour décrire un processus stochastique (des distributions sur tous les mots de tailles fixes). Dans ce cas, ils englobent les modèles de Markov cachés et les modèles à opérateurs observables. En considérant pour les symboles des paires action-observation, les systèmes séquentiels linéaires peuvent réaliser des processus contrôlés tels que des processus décisionnels Markoviens partiellement observables ou encore des représentations à états prédictifs. Enfin, ils sont capables de modéliser des transducteurs fonctionnant sur des paires de symboles entrée-sortie. En fait, ces systèmes séquentiels linéaires peuvent être tous décrit comme des cas particuliers d'un type de modèle graphique appelé automates à multiplicité. Dans cette thèse, on intéressera en particulier à ceux qui décrivent des langages stochastiques.

Le formalisme des automates à multiplicité en fait un cadre attrayant pour l'étude des systèmes séquentiels linéaires en général. Nous verrons que ces automates se décomposent en plusieurs sous-classes de modèles permettant des représentations plus ou moins compactes et plus ou moins expressives. De même, certaines de ces classes, telles que les automates probabilistes résiduels, permettent à des algorithmes d'inférence d'obtenir de meilleures garanties tout en étant efficaces en temps de calcul.

Ces dernières décennies, la communauté de l'inférence grammaticale et celle de l'apprentissage de processus stochastiques ou contrôlés ont évolué sans assez échanger leurs avancées. En inférence grammaticale, de nombreux progrès théoriques ont vu le jour permettant la définition de nouveaux algorithmes dans des modèles d'apprentissage de plus en plus exigeants. Quant à la communauté liée aux processus stochastiques ou contrôlés, elle s'est concentrée sur l'amélioration des modèles génératifs et leurs utilisations. Par exemple, en apprentissage par renforcement, les algorithmes de planification dans les environnements partiellement observables ont bénéficié d'énormes progrès permettant de traiter des problèmes d'ordres de grandeur au moins deux fois supérieurs. En traitements du langage, les systèmes et les modèles se sont complexifiés afin d'augmenter leur performance.

Ces deux domaines ont en commun de se baser sur des modèles génératifs à variables latentes, tels que les processus décisionnels Markoviens partiellement observables, les modèles de Markov cachés ou bien encore les grammaires stochastiques non contextuelles. Cependant, l'apprentissage de ces modèles à variables latentes est traditionnellement réalisé par des algorithmes itératifs maximisant la vraisemblance dont les garanties sont faibles. Par exemple, du côté des méthodes fréquentistes l'algorithme Espérance-Maximisation et les descentes de gradients ne garantissent que de converger vers un maximum local de la vraisemblance. Du côté des méthodes bayésiennes, des méthodes variationnelles ou bien des méthodes de Monte-Carlo par chaînes de Markov telles que l'échantillonnage de Gibbs ont été proposées. Bien que, sous certaines conditions, elles puissent converger vers un jeu de paramètres optimal *a posteriori*, la qualité du modèle inféré dépend de l'*a priori* sur les paramètres et du type de distribution utilisé. Ainsi peu de progrès ont été réalisés sur les méthodes d'inférence afin d'obtenir des garanties statistiques. De plus, ces méthodes sont assez coûteuses en temps de calcul quand le nombre de variables latentes et d'exemples augmentent. Notons tout de même que les méthodes de Monte-Carlo par chaînes de Markov ont pu bénéficier des avancées en calcul parallèle et sont maintenant beaucoup plus rapides.

Cependant, ces dernières années, les méthodes spectrales (plus généralement la méthode des moments), proposées indépendamment dans les deux communautés, ont vu le jour du côté des méthodes fréquentistes. Les algorithmes issus de la méthode des

moments se décomposent en trois grandes étapes. La première consiste à estimer des probabilités jointe du passé, du présent et du futur puis à les organiser en matrices ou tenseurs. Ensuite, ces algorithmes recherchent un espace linéaire de faible dimension dans lequel ces matrices peuvent être approximativement représentées. De cette façon, on peut représenter efficacement des statistiques suffisantes du passé pour prédire le futur. En considérant les statistiques suffisantes à deux pas de temps successifs, on peut retrouver les paramètres qui permettent une mise à jour bayésienne de ces statistiques. Ces paramètres sont ceux du modèle appris. Sous l'hypothèse de linéarité, ces méthodes permettent d'inférer des modèles au travers d'opérations algébriques peu coûteuses. Ainsi, elles sont très rapides car les probabilités jointes peuvent être estimées en un temps proportionnel au nombre d'exemples. Les étapes suivantes prennent un temps polynôme par rapport à la dimension du modèle. De plus, ces algorithmes bénéficient de fortes garanties théoriques issues de la théorie de la concentration de la mesure et de la perturbation des solutions de systèmes d'équations linéaires. Dans la communauté des processus stochastiques et contrôlés, ces méthodes sont justifiées par l'existence de variables cachées donnant lieu à des statistiques suffisantes de faibles dimensions. L'identification d'un espace de faible dimension réalise ainsi un « goulot spectral » entre le passé et le futur. Dans la communauté de l'inférence grammaticale, ces algorithmes ont une justification différente. Plutôt que des statistiques suffisantes, on considère des distributions résiduelles s'apparentant à des distributions conditionnelles. Dès lors, des résultats théoriques montrent que pour la classe de modèle que l'on considère ces distributions résiduelles appartiennent à un sous-espace vectoriel de faible dimension.

Ainsi, les méthodes spectrales ont contribué à établir un pont « algorithmique » entre les deux communautés alors que les échanges précédents concernaient principalement la théorie de l'apprentissage. Bien que le point de vue du « goulot spectral » ait suscité de nombreuses avancées, telles que la méthode des tenseurs et l'utilisation de noyaux, qui ne sont pas toujours comprises dans le formalisme de l'inférence grammaticale, celui-ci n'a pas été encore totalement exploité. En effet, ce formalisme a permis d'établir des résultats géométriques qui, nous le verrons dans cette thèse, permettent de dériver des algorithmes plus performants. De plus, ces résultats géométriques ont permis de décrire une hiérarchie complexe de modèles dont la compacité et l'expressivité ont pu être analysées en profondeur par la communauté de l'inférence grammaticale.

Le Chapitre 1 définit le formalisme de l'inférence grammaticale, présente les automates à multiplicités et décrit la hiérarchie entre les sous-classes de langages stochastiques. Plusieurs façons de caractériser les langages stochastiques sont décrites et on rappelle les principaux résultats permettant de passer d'une représentation à une autre. Enfin, ce chapitre détaille l'équivalence avec les processus stochastiques et contrôlés.

Apprentissage statistique

L'apprentissage statistique est l'étude statistique des propriétés de ces algorithmes. Il permet de garantir que sous certaines conditions un algorithme apprendra correctement un modèle de la distribution cible. Mais que signifie apprendre correctement ?

Dans un premier temps, on fait face à un problème de modélisation, différent de celui de l'apprentissage. Admettons qu'il existe un oracle que l'on peut interroger pour connaître la probabilité de n'importe quelle séquence. De plus cet oracle est capable de nous dire si un modèle réalise la distribution ou sinon renvoie un contre exemple. Est-ce qu'un algorithme ayant accès à un tel oracle est capable d'inférer un modèle

accepté par l'oracle ?

Ensuite, on pourra se demander quel sera le comportement asymptotique d'un algorithme qui apprend à partir d'une infinité d'exemples. Finit-il par converger vers un modèle réalisant la distribution cible ?

Puis, on peut étudier le comportement d'un algorithme travaillant à partir d'un ensemble fini d'exemples. Il faut considérer que nos exemples sont tirés aléatoirement selon une distribution d'entraînement qui sera souvent la distribution cible mais qui peut aussi être différente. Comme la quantité d'information est réduite aux exemples d'entraînement, la sortie d'un algorithme dépendra du tirage de ces exemples et donc sa performance aussi. Par exemple, si les exemples ne représentent pas bien la distribution cible on peut s'attendre à ce qu'un algorithme se trompe fortement. On peut alors s'intéresser au comportement moyen. C'est-à-dire que l'on cherchera, en fonction du nombre d'exemples, à savoir à quelle distance en moyenne la distribution estimée par un algorithme se trouve de la distribution cible. Cependant cela ne nous donne que peu d'information sur comment la performance d'un algorithme peut varier en fonction des exemples.

Finalement, on préférera plutôt borner la distance maximale entre la distribution apprise et la distribution cible pour les cas les plus favorables. Fort heureusement, comme les cas défavorables sont les moins probables, on pourra obtenir une borne en probabilité qui représente la part des cas favorables.

Ces différentes garanties sur la qualité de l'apprentissage sont particulièrement pertinentes quand on contraint le temps d'exécution de l'algorithme. On appelle un modèle d'apprentissage, un ensemble de garanties associées à des contraintes sur le temps d'exécution. En fonction de l'application, nous verrons que certains modèles d'apprentissage sont plus adaptés que d'autres.

Ainsi, le Chapitre 2 est dédié à l'apprentissage. D'abord, les différents modèles d'apprentissage sont présentés et critiqués, afin de proposer un modèle d'apprentissage, dérivé du modèle Probablement Approximativement Correct (PAC), adapté à notre problème. Puis, l'énumération de résultats négatifs et positifs dans ce modèle d'apprentissage, nous guidera vers l'identification d'une classe de modèles à la fois riche et apprenable. Enfin, sont passés en revue les algorithmes itératifs, puis ceux issus de l'inférence grammaticale et enfin ceux basés sur la méthode des moments. On y montre que les algorithmes issus de la méthode des moments bénéficient de meilleures garanties et d'un temps d'exécution plus faible.

Contributions

Cependant, plusieurs problèmes subsistent avec la méthode des moments. Tout d'abord, les bornes actuelles sur l'erreur supposent que le modèle appris est au moins aussi grand que celui ayant généré les exemples d'apprentissages. Lorsque, la dimension du modèle appris est inférieure à la vraie, nous qualifions le modèle de compressé. La première partie du Chapitre 3 propose d'élargir l'analyse de l'erreur d'apprentissage aux modèles compressés. En particulier, nous nous intéresserons à l'algorithme Spectral. Comme un modèle compressé ne peut pas représenter la dynamique du système dans sa totalité, nous verrons qu'un biais, étudié dans l'état de l'art, subsiste. Cependant, aucune étude ne s'intéresse au comportement statistique de l'algorithme Spectral. Nous proposons ainsi une analyse non asymptotique de l'erreur pour l'apprentissage de modèles compressés. En somme, nous obtenons une borne sur l'erreur représentant un compromis biais-variance. Celle-ci permet de conclure que la dimension du modèle doit

être choisie en fonction du nombre d'exemples. Lorsque ceux-ci sont en faible nombre, il peut être plus avantageux d'apprendre un modèle compressé et donc moins complexe afin d'éviter le sur-apprentissage. Lorsque les exemples se font plus nombreux, nous pouvons revenir à l'apprentissage d'un modèle d'une taille suffisante afin d'éliminer le biais. Enfin, comme les bornes sur l'erreur dépendent de faibles valeurs singulières, il se peut que la représentation soit mal conditionnée et l'algorithme d'apprentissage instable. Pour pallier les problèmes de stabilité et de choix de la dimension, nous proposons un algorithme Spectral régularisé qui adapte la dimension du modèle en fonction des données. Les contributions de ce chapitre ont donné lieu à une communication [Glaude et collab., 2014].

Dans le Chapitre 4, une application de l'algorithme proposé au Chapitre 3 au contrôle d'un récepteur en guerre électronique est présenté. L'objectif est de définir une stratégie qui balaye le spectre fréquentiel pour intercepter des illuminations radars afin de surveiller l'activité électromagnétique adverse. Nous proposons de modéliser l'interception des illuminations d'un radar par un processus contrôlé. Les contributions de ce chapitre ont donné lieu à une communication [Glaude et collab., 2015a].

La méthode des moments a un deuxième inconvénient. Elle a été dérivée pour apprendre des automates à multiplicité quelconques, parmi lesquels figurent ceux réalisant des distributions. Ainsi bien que l'apprentissage soit pseudo-PAC, il est dit impropre. En effet, avant de converger vers un automate réalisant un langage stochastique, la méthode des moments renvoie des automates qui n'ont aucune garantie de réaliser un langage stochastique. En général, les valeurs retournées par l'automate appris ne somment pas à un, peuvent être négatives et ne sont en aucun cas des probabilités. Cette caractéristique est souvent référence dans la littérature comme le problème des probabilités négatives. Plus formellement, on dit dans la terminologie PAC que l'ensemble d'hypothèses est plus large que l'ensemble des concepts. Cet inconvénient peut être très problématique pour l'application qui utilise le modèle appris. En effet, certaines applications nécessitent de travailler avec des probabilités. C'est le cas dès lors que l'on calcule des espérances. D'autre part, pour la planification, si le noyau de transition est décrit par une mesure non-bornée, l'algorithme d'itération de la valeur peut diverger. Enfin, ce problème empêche d'utiliser la solution trouvée pour initialiser un autre algorithme comme Espérance-Maximisation. Comme il a été montré qu'il n'existe pas d'algorithme propre avec des garanties pseudo-PAC pour apprendre des automates à multiplicité réalisant des langages stochastiques, nous sommes contraint de réduire l'ensemble des concepts. Les Chapitres 5 à 7 apportent des solutions de complexités croissantes pour retrouver des probabilités en réduisant l'ensemble d'hypothèses et l'ensemble des concepts.

Dans le Chapitre 5, nous proposons un algorithme, NNSpectral, dans la veine de l'algorithme Spectral. NNSpectral a l'avantage de contraindre dans sa conception les paramètres du modèle à être positifs. La première conséquence est la positivité des valeurs retournées. Deuxièmement, les erreurs introduites par les changements de signes des paramètres de faibles valeurs sont éradiqués. Malheureusement, en son sein cet algorithme doit identifier une enveloppe convexe. Or, dans sa généralité, ce problème est mal posé et NP-dur. NNSpectral a donc recours à des heuristiques qui condamnent toute possibilité d'analyse de la convergence. Néanmoins, NNSpectral conserve une complexité linéaire avec le nombre d'exemples. Expérimentalement, NNSpectral semble moins enclin à converger vers de mauvais minimaux locaux et obtient de très bonnes performances. Cependant, le modèle appris réalise une fonction positive qui doit être normalisée pour obtenir une distribution. Pour des raisons évidentes, cette

normalisation ne peut être effectué que localement, c'est-à-dire pour un nombre fini d'événements. Les contributions de ce chapitre ont donné lieu à deux communications [Glaude et collab., 2015b,c].

Le Chapitre 6 propose une variante de l'algorithme NNSpectral, qui effectue une normalisation globale du modèle. Celui-ci aussi se base sur l'identification d'une enveloppe convexe. Cependant, au lieu de travailler avec des probabilités jointes, celui-ci recalcule des distributions conditionnelles. De cette façon, l'algorithme CH-PNFA, renvoie un modèle globalement normalisé. Il retourne donc directement des probabilités. Comme NNSpectral, l'apprentissage n'est pas consistant à cause de l'utilisation d'heuristiques pour l'identification de l'enveloppe convexe. De plus, l'utilisation de distributions conditionnelles crée des instabilités et les performances sont légèrement moins bonnes que NNSpectral. Cependant, à la différence de NNSpectral, la solution trouvée par CH-PNFA permet d'initialiser un algorithme itératif comme Espérance-Maximisation. Ainsi, il devient possible de combiner les avantages de NNSpectral et de l'algorithme Espérance-Maximisation qui a la garantie d'améliorer à chaque itération la vraisemblance du modèle.

Le Chapitre 7 poursuit la quête d'un algorithme pseudo-PAC pour l'inférence de distributions à variables latentes. Pour ce faire, nous identifions une classe d'automates d'expressivité intermédiaire pour laquelle l'identification d'une enveloppe convexe peut être réalisée exactement de façon robuste et en temps polynomial. Nous proposons donc un dernier algorithme, CH-PRFA, qui possède des garanties pseudo-PAC. Bien qu'au sens strict l'apprentissage soit impropre, le modèle appris est contraint de réaliser une distribution. Nous évitons ainsi les inconvénients de l'algorithme Spectral. De même, que CH-PNFA, ce dernier algorithme repose l'estimation de distributions conditionnelles. L'analyse non-asymptotique, ainsi que les expériences, montrent que cette propriété crée une certaine instabilité. L'étude théorique permet de mieux cibler la source de cette instabilité et d'y apporter différentes solutions. Comme dans le chapitre précédent, nous explorons la combinaison de CH-PRFA avec l'algorithme Espérance-Maximisation. Enfin, CH-PRFA, en plus d'être très rapide, obtient de très bonnes performances. Les contributions de ce chapitre ont donné lieu à deux communications [Glaude et collab., 2015d; Glaude et Pietquin, 2016].

Plan

La première partie regroupe les définitions, les propriétés et les résultats de la littérature qui seront utilisés dans les parties suivantes. Le Chapitre 1 présente la théorie des séries rationnelles, leur représentation par des automates à multiplicité et leur lien avec les séries reconnaissables. Nous nous concentre ensuite sur les séries rationnelles qui représentent des distributions, les langages stochastiques. Puis, nous établissons une hiérarchie entre différentes classes de langages stochastiques. Pour chacune de ces classes, nous donnons une caractérisation géométrique et nous décrivons le type d'automates qui réalise les langages stochastiques de cette classe. Enfin, nous dressons un tableau d'équivalence avec les automates réalisant des processus stochastiques et contrôlés.

Le Chapitre 2 commence par passer en revue les différents modèles d'apprentissage étudiés dans la littérature de l'inférence grammaticale. À la fin de cette partie, à partir des résultats existants à la fois positifs et négatifs, nous définissons un modèle assez restrictif pour être intéressant et assez permissif pour obtenir des résultats positifs. Dans une seconde partie, nous revoyons les algorithmes d'apprentissage existants. Ceux-ci

sont classés en trois grandes familles. Enfin, nous concluons notre état de l'art et l'on dégage deux grandes problématiques attaquée dans cette thèse : l'apprentissage de modèles compressés et le problème des probabilités négatives. Ces deux problématiques forment les deux parties suivantes.

La deuxième partie s'adresse à l'apprentissage de modèles compressés avec une application dans le domaine de la guerre électronique, où le modèle, ainsi que sa taille sont appris en ligne. Le Chapitre 3 présente d'abord l'état de l'art des analyses non-asymptotiques de l'erreur d'apprentissage des algorithmes basés sur la méthode des moments. Nous y discutons les différentes techniques utilisées ainsi que les études sur le biais des modèles compressés. La second moitié du Chapitre 3 présente les contributions de l'auteur. Nous proposons une analyse non-asymptotique de l'apprentissage de modèles compressés qui, une fois combiné aux bornes précédentes, met en exergue le compromis biais-variance. Puis, nous présentons un algorithme Spectral régularisé qui fait la part entre le biais et la variance en choisissant la dimension du modèle appris en fonction des données. Une étude empirique sur des problèmes artificiels montre le gain apporté par la régularisation.

Puis, le Chapitre 4 présente une application de l'algorithme du Chapitre 3 pour le séquencement de récepteurs superhétérodynes. Une description technique du problème est donnée, incluant une présentation de la chaine de traitement en guerre électronique et la séparation du séquencement en veille et analyse. Puis, un état de l'art des stratégies de veille et d'analyse est présenté. Ensuite, nous formulons le problème du séquencement comme un problème de prédiction des prochains passages de lobes. La modélisation de ce problème par des processus contrôlés permet l'application de l'algorithme présenté au Chapitre 3. Au cours de l'expérimentation, nous abordons les problématiques d'apprentissage en ligne (des modèles et de leurs tailles), d'utilisation de caractéristiques, et de compromis entre exploration et exploitation. Enfin, des expériences sur des données simulées montrent l'apport d'une méthode d'apprentissage en ligne par rapport aux méthodes conventionnelles.

Nous passons ensuite la troisième et dernière partie guidée par la recherche d'algorithme d'apprentissage sans le problème des probabilités négatives et pourvus de garanties pseudo-PAC. Le Chapitre 5 présente l'algorithme NNSpectral basé sur la recherche de sous semi-modules. L'algorithme fait appel à des méthodes de décomposition en matrices non-négatives, ainsi qu'à des algorithmes de moindres carrés non-négatifs. Un bref état de l'art de ces méthodes est proposé. Nous décrivons ensuite les trois ensembles de données qui serviront à l'évaluation des algorithmes des Chapitres 5 à 7. Plusieurs critères d'évaluation sont présentés et le protocole expérimental est défini. Puis, les trois autres algorithmes issus de la méthodes des moments, ainsi que l'algorithme de Baum-Welch, qui sont utilisés en comparaison, sont détaillés. Ensuite, l'implémentation des différents algorithmes est discutées. En particulier, nous expliquons comment utiliser des séries auxiliaires, normaliser la variance, choisir la base et la dimension des modèles. Nous expliquons aussi comment utiliser les algorithmes présentés pour apprendre des processus stochastiques au lieu de langages stochastiques. Enfin, après présentation des résultats, nous comparons l'algorithme NNSpectral avec des approches de l'état de l'art pour l'apprentissage de chaînes de Markov cachées.

Le Chapitre 6 est plus bref. Il présente une variante de NNSpectral, appelée CH-PNFA, qui répond au problème de normalisation globale. Nous revoyons dans ce chapitre comment les moindres carrés non-négatifs sous contraintes peuvent s'exprimer comme un problème de programmation quadratique avec des contraintes linéaires.

Puis, nous étudions les propriétés d'existence et d'unicité de la solution. Ensuite, nous discutons de l'utilisation de CH-PNFA pour l'initialisation d'algorithmes itératifs. Enfin, les performances de CH-PNFA sont données, en particulier en tant que procédure d'initialisation de l'algorithme de Baum-Welch.

Le Chapitre 7 est dédié à l'algorithme CH-PRFA. Nous y décrivons d'abord les matrices séparables au centre de la caractérisation d'un langage stochastique par sa matrice de Hankel. Cette caractérisation adaptée à la sous classe de langages stochastiques que l'on considère permet de dériver l'algorithme CH-PRFA. Ensuite, les algorithmes de décomposition de matrices séparables sont classés en trois grandes catégories. Parmi les algorithmes récursifs, un en particulier, SPA est sélectionné pour l'analyse non-asymptotique de CH-PRFA. Cette analyse montre certaines faiblesses de CH-PRFA issues des algorithmes actuels de décomposition de matrices séparables. Plusieurs solutions sont proposées pour palier au problème dont une variante de SPA. Ensuite, les expériences confirment l'intuition donnée par l'analyse non-asymptotique en comparant différents algorithmes de décomposition de matrices séparables sur plusieurs tailles de base. Enfin, les performances de l'algorithme CH-PRFA, ainsi que son utilisation pour initialiser de l'algorithme de Baum-Welch, sont évaluées.

Un chapitre de conclusion clôt la thèse en comparant les algorithmes proposés dans les Chapitres 5 à 7. Puis, un plan pour des recherches futures est présenté.

Publications de l'auteur

— Glaude, H., O. Pietquin C. Enderli. 2014, «Subspace identification for predictive state representation by nuclear norm minimization», *2014 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning, ADPRL 2014, Orlando, FL, USA, December 9-12, 2014*, 1–8

— Glaude, H., C. Enderli, J. Grandin O. Pietquin. 2015a, «Learning of scanning strategies for electronic support using predictive state representations», *25th IEEE International Workshop on Machine Learning for Signal Processing, MLSP 2015, Boston, MA, USA, September 17-20, 2015*, 1–6

— Glaude, H. H. Boucard. 2015, «Method and system for determining a reception configuration and a duration of a time interval», WO Patent App. PCT/EP2014/069,804

— Glaude, H., C. Enderli O. Pietquin. 2015d, «Spectral learning with proper probabilities for finite state automaton», *Automatic Speech Recognition and Understanding Workshop (ASRU 2015)*, IEEE

— Glaude, H., C. Enderli O. Pietquin. 2015c, «Non-negative spectral learning for linear sequential systems», *Neural Information Processing - 22nd International Conference, ICONIP 2015, Istanbul, Turkey, November 9-12, 2015, Proceedings, Part II*, 143–151

— Glaude, H., C. Enderli O. Pietquin. 2015b, «Apprentissage spectral non négatif de systèmes séquentiels linéaires», *Actes de CAP*

— Glaude, H. O. Pietquin. 2016, «Pac learning of probabilistic automaton based on the method of moments», *Proceedings of the 33rd International Conference on Machine Learning, ICML 2016, New-York, NY, USA, 19-24 June 2016*, 820–829

Première partie

Définitions et état de l'art

Chapitre 1

Langages stochastiques et automates

Sommaire

Notations

Dans ce manuscrit, les vecteurs seront indiqués par des caractères gras et les matrices par des lettres majuscules. La matrice identité de taille d est noté I_d. La fonction indicatrice d'ensemble est notée $\mathbb{1}_E(F)$ et vaut 1 si $F \subset E$, 0 sinon. Cette fonction est étendue aux singletons, $\mathbb{1}_e(f)$ vaut 1 si $e = f$ et 0 sinon. Pour un ensemble E, on note 2^E l'ensemble des parties de E. Dans ce chapitre, les notations et la plupart des résultats sont repris de Denis et Esposito [2008]. La plupart des preuves sont omises mais certains résultats importants sont redémontrés.

Introduction

Dans ce chapitre, on définit les objets mathématiques et leurs propriétés, essentielles pour les chapitres suivants. Après quelques rappels en algèbre linéaire, on introduira les séries formelles rationnelles qui sont des fonctions qui associent à toute séquence de symboles un scalaire. Nous donnerons deux autres définitions équivalentes. L'une permet de représenter une série par un modèle graphique appelé automate, l'autre donne une représentation linéaire qui permet l'évaluation de la série par un calcul matriciel. Ensuite, on définira la matrice de Hankel, qui caractérise entièrement une série. Cette matrice a des propriétés intéressantes sur son rang et utile pour l'apprentissage. Ainsi, les résultats permettant de passer de la matrice de Hankel à la représentation linéaire sont fondamentaux pour dériver des algorithmes d'apprentissage basés sur la méthode des moments.

Ensuite, on se concentre sur les langages stochastiques qui sont des séries formelles rationnelles particulières, représentant des distributions sur les séquences de longueurs finies. La représentation d'un langage stochastique par une famille finie de langages stochastiques permettra d'obtenir une hiérarchie entre des sous-ensembles de langages stochastiques rationnels. Ces sous-ensembles sont définis naturellement par des propriétés géométriques. Nous verrons aussi que ces classes de langages stochastiques correspondent à des automates satisfaisants des contraintes particulières.

Puis, on expliquera que la classe la plus générale de langages stochastiques ne peut pas être décrite de façon récursivement énumérable. Cette propriété induit l'absence d'algorithme d'inférence consistant contraint à retourner des distributions. Ainsi, on s'intéressera à des automates décrits par un ensemble fini de contraintes. Ces classes d'automates sont moins expressives et représentent les langages de façon moins compacte mais permettent d'obtenir des algorithmes d'inférence propres. Malheureusement, on verra au chapitre suivant que l'inférence de tels langages stochastiques restent néanmoins difficile, ce qui proscrit l'existence d'algorithmes efficaces. Nous présenterons ainsi une dernière classe de langages stochastiques, moins riche, mais pour laquelle on donnera au Chapitre 7 un algorithme d'inférence consistant, efficace et contraint à retourner des distributions.

Enfin, on présentera les processus stochastiques et les processus contrôlés dont l'analyse est très similaire à celle des langages stochastiques. Ainsi une hiérarchie similaire de sous-ensembles et d'automates sera établie par analogie. En particulier, on verra que de nombreux modèles graphiques populaires entrent dans cette hiérarchie. Deux diagrammes récapitulatifs concluent ce chapitre.

1.1 Préliminaires

1.1.1 Algèbre linéaire

Dans cette section, on rappelle les définitions de trois structures algébriques peu communes : les monoïdes, les semi-anneaux et les semi-modules qui sont une généralisation respectivement des groupes, des anneaux et des espaces vectoriels.

Définition 1 (Monoïde)**.**
Soit E un ensemble, un monoïde $(E, *, e)$ est une structure algébrique telle que,

(i) E *est stable pour la loi* $*$ *(*$\forall x, y \in E,\ x * y \in E$*);*

(ii) la loi $*$ *est associative dans* E *(*$\forall x, y, z \in E,\ (x * y) * z = x * (y * z)$*);*

(iii) il existe un élément neutre e *pour la loi* $*$ *(*$\exists e \in E,\ \forall x \in E,\ x * e = e * x = x$*).*

Un monoïde est dit commutatif si $\forall x, y \in E,\ x * y = y * x$*.*

Définition 2 (Semi-anneau)**.**
Soit E un ensemble, un semi-anneau $(E, +, \times, 0, 1)$ est une structure algébrique telle que,

(i) $(E, +, 0)$ *est un monoïde commutatif;*

(ii) $(E, \times, 1)$ *est un monoïde;*

(iii) $\times$ *est distributif par rapport à* $+$ *(*$\forall x, y, z \in E,\ x \times (y + z) = x \times y + x \times z$ *et* $(x + y) \times z = x \times z + y \times z$*);*

(iv) 0 *est absorbant pour la loi* $\times$ *(*$\forall x \in E,\ x \times 0 = 0 \times x = 0$*).*

Un semi-anneau est dit commutatif si $(E, \times, 1)$ *est un monoïde commutatif. Un semi-anneau est dit unitaire si* 1 *est un élément neutre pour la loi* $\times$ *(*$\forall x \in E,\ 1 \times x = x \times 1 = 1$*).*

Dans la suite on ne considérera que des semi-anneaux unitaires commutatifs.

Exemple. $\mathbb{R}$ *et* $\mathbb{R}^+$ *sont des semi-anneaux. De plus,* $\mathbb{R}$ *est aussi un corps commutatif.*

Soit K un anneau, on note pour la suite $K^{m\times n}$ l'ensemble des matrices de taille $m \times n$ à coefficients dans K. On remarque que $K^{m\times m}$ est un semi-anneau.

Définition 3 (Semi-module)**.**
Soit $(K, +_K, \times_K, 0_K, 1_K)$ un semi-anneau, $(V, +_V, 0_V)$ un monoïde commutatif et une loi externe $\cdot$ de $K \times V$ dans V vérifiant pour tout éléments $a, b \in K$ et $x, y \in V$,

(i) $(a \times_K b) \cdot x = a \cdot (b \cdot x)$,

(ii) $(a +_K b) \cdot x = (a \cdot x) +_M (b \cdot x)$,

(iii) $a \cdot (x +_M y) = (a \cdot x) +_M (a \cdot y)$,

(iv) $1_K \cdot x = x$

alors $(V, +_V, \cdot)$ est un K-semi-module.

Soit V un K-semi-module et S un sous-ensemble de V, on note $[[S]]$ le plus petit sous semi-module de V contenant S. On appelle $[[S]]$ le sous semi-module généré par S. On peut montrer que $[[S]] = \left\{\sum_{i=1}^{d} k_i s_i | d \in \mathbb{N},\ k_i \in K,\ s_i \in S\right\}$. Les structures algébriques définies ci-dessus qualifient les ensembles qui font l'objet des sections suivantes.

1.2 Séries

Dans cette section, on définit d'abord de trois façons différentes un type de fonction qui associe un scalaire à toute séquence finie de symboles. On appelle ces fonctions des séries. La première définition, par les séries rationnelles, étend celle des polynômes. La seconde définition, donnée par les séries reconnaissables, associe une représentation linéaire. La troisième définition utilise les automates à multiplicité. Puis, on présentera un Théorème fondamental pour les séries rationnelles qui permettra de montrer que les trois définitions sont équivalentes.

1.2.1 Séries formelles rationnelles

Cette section définit formellement des objets que l'on manipulera tout au long de la thèse. On appelle un *alphabet* un ensemble de symboles. Dans la suite, on notera généralement l'alphabet Σ. Un *mot* est une suite de symboles. La *longueur* d'un mot u, notée $|u|$, est le nombre de symboles qui composent u. On note Σ^l, $\Sigma^{\leq l}$, $\Sigma^\star$ les ensembles respectivement des mots de longueur l, de longueurs inférieures ou égales à l, des mots de longueurs finies. La *concaténation* de deux mots u et v est notée uv. On note uV, la concaténation d'un mot u et d'ensemble de mots V tel que $uV = \{uv | v \in V\}$. De plus, on note $u^{-1}V = \{v | uv \in V\}$ les suffixes des mots V commençant par u. Le mot vide, c'est-à-dire de longueur nulle, est noté ε.

Définition 4 (Série formelle)**.**
Soit Σ un alphabet fini et K un semi-anneau, une série formelle r est une fonction de $\Sigma^\star$ dans K. On appelle les $r(u)$ ($u \in \Sigma^\star$) les coefficients de la série. Celle-ci peut s'écrire comme la somme formelle $r = \sum_{u\in\Sigma^\star} r(u)u$.

Le *support* d'une série est l'ensemble des mots associés à des coefficients non nuls, formellement $supp(r) = \{u | r(u) \neq 0\}$. Un *polynôme* est une série dont le support est fini.

On note $K\langle\langle\Sigma\rangle\rangle$ l'ensemble des séries formelles et $K\langle\Sigma\rangle$ l'ensemble des polynômes. On peut définir la somme de deux séries r, l en sommant leurs coefficients respectifs $r + l = \sum_{u\in\Sigma^\star}(r(u) + l(u))u$. Le produit de deux séries r et l est défini par $rl = \sum_{w\in\Sigma^\star}\left(\sum_{uv=w} r(u)l(v)\right)w$. La série 0 est celle dont tous les coefficients sont nuls. La série 1 est la série dont tous les coefficients sont nuls sauf celui de ε qui vaut 1. Ces opérations munissent $K\langle\langle\Sigma\rangle\rangle$ d'une structure de semi-anneau dont $K\langle\Sigma\rangle$ est un sous semi-anneau.

Une série formelle r est *quasi-régulière* si $r(\varepsilon) = 0$. Si une série r est quasi-régulière alors pour chaque mot $u \in \Sigma^\star$, il n'existe qu'un nombre fini d'entiers i tel que $r^i(u) \neq 0$. Ainsi, si r est quasi-régulière, la série r^+ définie par $r^+(u) = \sum_{i\geq 1} r^i(u)$ existe bien.

Définition 5 (Ensemble des séries formelles rationnelles)**.**
L'ensemble $K^{rat}\langle\langle\Sigma\rangle\rangle$ des séries formelles rationnelles est le plus petit sous semi-anneau de $K\langle\langle\Sigma\rangle\rangle$ contenant $K\langle\Sigma\rangle$ tel que pour toute série quasi-régulière $r \in K^{rat}\langle\langle\Sigma\rangle\rangle$, $r^+ \in K^{rat}\langle\langle\Sigma\rangle\rangle$.

On qualifiera dorénavant de *séries rationnelles* les éléments de $K^{rat}\langle\langle\Sigma\rangle\rangle$. Cet objet pour l'instant très abstrait possède plusieurs représentations équivalentes qui sont détaillées dans la prochaine section.

1.2.2 Séries formelles reconnaissables

Cette section explicite la représentation linéaire en termes de matrices et de vecteurs de certaines séries, dites reconnaissables.

Définition 6 (Ensemble des séries formelles reconnaissables)**.**
Une série formelle r est reconnaissable s'il existe un morphisme multiplicatif $A : \Sigma^\star \to K^{d\times d}, d \in \mathbb{N}^+$ (c'est-à-dire vérifiant $\forall u, v \in \Sigma^\star$, $A(uv) = A(u)A(v)$) et deux vecteurs $\boldsymbol{\alpha}_0, \boldsymbol{\alpha}_\infty \in K^d$ tel que $\forall u \in \Sigma^\star$, $r(u) = \boldsymbol{\alpha}_0^\top A(u)\boldsymbol{\alpha}_\infty$. On appelle le triplet $(\boldsymbol{\alpha}_0, A, \boldsymbol{\alpha}_\infty)$ la représentation linéaire de r. On appelle d la dimension de r. On note $K^{rec}\langle\langle\Sigma\rangle\rangle$ l'ensemble des séries formelles reconnaissables.

Dans la suite, pour tout symbole σ de Σ, on note A_σ la matrice $A(\sigma)$. Par abus de notation, on désigne aussi par A, la matrice créée par concaténation horizontale des $\{A_\sigma\}_{\sigma\in\Sigma}$. On note aussi A_Σ la matrice tel que $A_\Sigma = \sum_{\sigma\in\Sigma} A_\sigma$. De plus, pour tout mot $u = \sigma_1 \dots \sigma_l$, on note $A_u = A(u) = A_{\sigma_1} \dots A_{\sigma_l}$. On a de plus que $A_\varepsilon = I_d$.

Définition 7 (Représentation linéaire minimale)**.**
Une représentation linéaire d'une série est dite minimale s'il n'existe pas de représentation linéaire de la même série de dimension strictement inférieure.

Définition 8 (Représentations linéaires similaires)**.**
Deux représentations linéaires de dimension d, $(\boldsymbol{\alpha}_0, A, \boldsymbol{\alpha}_\infty)$ et $(\boldsymbol{\beta}_0, B, \boldsymbol{\beta}_\infty)$ sont similaires s'il existe une matrice $M \in K^{d\times d}$ inversible telle que $\boldsymbol{\alpha}_0 = \boldsymbol{\beta}_0 M$, $\boldsymbol{\alpha}_\infty = M^{-1}\boldsymbol{\beta}_\infty$ et pour tout mot $u \in \Sigma^\star$, $A_u = M^{-1}B_u M$. On vérifie directement que des représentations linéaires similaires définissent la même série.

1.2.3 Séries formelles réalisées par un automate

Dans cette section, on définit d'abord deux types de machines à états, appelées aussi automates finis. Lorsque l'on associe des poids aux transitions, ces automates définissent des séries.

Définition 9 (Automate fini non déterministe, ou *Non-deterministic Finite Automata* (NFA))**.**
Un automate fini non déterministe, ou Non-deterministic Finite Automata *(NFA) est un quintuplet $(\Sigma, Q, Q_I, Q_F, \delta)$ où Σ est un alphabet, Q un ensemble fini d'états contenant un ensemble d'états initiaux noté Q_I et un ensemble d'états finaux noté Q_F. δ est une fonction, dite de transition, de $Q\times\Sigma$ à valeur dans 2^Q vérifiant $\delta(q, \varepsilon) = \{q\}$, $\delta(q, u\sigma) = \cup_{q'\in\delta(q,u)}\delta(q', u)$ pour tout $q \in Q$, $\sigma \in \Sigma$ et $u \in \Sigma^\star$.*

On note aussi δ la fonction de transition étendue à $2^Q \times \Sigma^\star$ à valeur dans 2^Q et définie par $\delta(R, u) = \cup_{q\in R}\delta(q, u)$ pour tout $R \subset Q$ et $u \in \Sigma^\star$. Un mot est *reconnu* par un automate si en partant d'un état initial, il existe un chemin, défini par la fonction de transition et les symboles qui composent le mot, qui termine dans un état final.

Définition 10 (Automate fini déterministe, ou *Deterministic Finite Automata* (DFA))**.**
Un automate fini déterministe, ou Deterministic Finite Automata *(DFA) est un NFA avec un unique état initial ($Q_I = \{q_0\}$) et tel que $\forall q \in Q$, $\forall u \in \Sigma^\star$, $|\delta(q, u)| \leq 1$.*

Dans les DFA, à chaque mot n'est associé qu'au plus un seul chemin.

En associant des poids aux transitions, on peut définir un nouveau type d'automate.

Définition 11 (Automate à multiplicité dans K ou *K-Multiplicity Automata* (K-MA))**.**

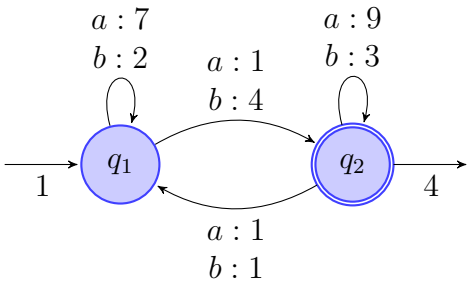
*Un automate à multiplicité dans K ou K-*Multiplicity Automata *(K-MA) est un quintuplet $(\Sigma, Q, \iota, \varphi, \tau)$ où Σ est un alphabet, Q un ensemble fini d'états, $\iota : Q \to K$ la fonction initiale, $\varphi : Q \times \Sigma \times Q \to K$ la fonction de transition et $\tau : Q \to K$ la fonction finale.*

Un exemple d'automate à multiplicité est représenté sur la Figure 1.1, ainsi que sa représentation linéaire.

De même que pour les NFA, on définit la fonction de transition étendue $\varphi : Q \times \Sigma^\star \times Q \to K$ vérifiant $\varphi(q, u\sigma, q') = \sum_{s \in Q} \varphi(q, u, s)\varphi(s, \sigma, q')$ et $\varphi(q, \varepsilon, q') = \mathbb{1}_q(q')$. De plus, on définit aussi φ sur des ensembles d'états et des ensembles de mots, soit $R, S \subset Q$ et $U \subset \Sigma^\star$ alors

$$\varphi(R, U, S) = \sum_{r \in R} \sum_{s \in S} \sum_{u \in U} \varphi(r, u, s).$$

Cela revient à sommer sur tous les chemins possibles allant d'un état de R à un état de S et émettant une séquence de symboles de U, le produit des poids de transition le long des chemins.



$$\boldsymbol{\alpha}_0 = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \qquad A_a = \begin{pmatrix} 7 & 1 \\ 1 & 9 \end{pmatrix} \qquad A_b = \begin{pmatrix} 2 & 4 \\ 1 & 3 \end{pmatrix} \qquad \boldsymbol{\alpha}_\infty = \begin{pmatrix} 0 \\ 4 \end{pmatrix}$$

FIGURE 1.1 – Un automate multiplicité à deux états (q_1 et q_2) sur l'alphabet $\Sigma = \{a, b\}$ et sa représentation linéaire.

Définition 12 (Support d'un K-MA)**.**
Soit un K-MA $M = (\Sigma, Q, \iota, \varphi, \tau)$, on note $Q_I = \{q \in Q | \iota(q) \neq 0\}$ l'ensemble des états initiaux, $Q_F = \{q \in Q | \tau(q) \neq 0\}$ l'ensemble des états finaux et $\delta(q, u) = \{q' \in Q | \varphi(q, u, q') \neq 0\}$ la fonction de transition, alors le NFA $(\Sigma, Q, Q_I, Q_F, \delta)$ est le support de M

Ce nouveau type d'automates permet de calculer des fonctions de $\Sigma^\star$ à valeur dans K.

Définition 13 (Séries réalisées par un K-MA)**.**
Pour tout K-MA $M = (\Sigma, Q, \iota, \varphi, \tau)$, on définit la série formelle r_M réalisée par M telle que pour tout $u \in \Sigma^\star$, $r_M(u) = \sum_{q,q' \in Q} \iota(q)\varphi(q, u, q')\tau(q')$. À chaque état $q \in Q$, on associe la série $r_{M,q}$ telle que pour tout $u \in \Sigma^\star$, $r_{M,q}(u) = \sum_{q' \in Q} \varphi(q, u, q')\tau(q')$. Enfin, on définit la série $\overline{r}_M$ telle que pour tout $u \in \Sigma^\star$, $\overline{r}_M(u) = \sum_{q,q' \in Q} \iota(q)\varphi(q, u, q')$.

Exemple (Série réalisée par l'automate sur la Figure 1.1)**.**

$$r(a) = 1 \times 1 \times 4 = 4$$
$$r(ab) = 1 \times 7 \times 4 \times 4 + 1 \times 1 \times 3 \times 4 = 124$$
$$r(u) = f(\sigma_1\sigma_2...\sigma_l) = \boldsymbol{\alpha}_0^\top A_{\sigma_1} A_{\sigma_2} \dots A_{\sigma_l} \boldsymbol{\alpha}_\infty$$

On dit que deux K-MA, M et M' sont *équivalents* si ils définissent la même série, c'est-à-dire si $r_M = r_{M'}$. Pour un K-MA, il arrive que l'on puisse définir un autre K-MA équivalent avec un nombre réduit d'états. On appelle ce nouveau K-MA une *K-réduction.*

Proposition 1 (K-réduction d'un K-MA [Denis et Esposito, 2008])**.**
Soit K-MA $M = (\Sigma, Q, \iota, \varphi, \tau)$ et $q \in Q$ un état de M, s'il existe pour tout $q' \in Q' = Q\backslash\{q\}$ des coefficient $a_{q'} \in K$ tel que $r_{M,q} = \sum_{q' \in Q} a_{q'} r_{M,q'}$, alors $M' = (\Sigma, Q', \iota', \varphi', \tau')$ est équivalent à M où, pour tout $r, s \in Q'$ et $\sigma \in \Sigma$, on a

$$\varphi'(r, \sigma, s) = \varphi(r, \sigma, s) + a_s \varphi(r, \sigma, q),$$
$$\iota'(r) = \iota(r) + a_r \varphi(q),$$
$$\tau'(s) = \tau(s).$$

De plus, $\forall q' \in Q'$, $r_{M,q'} = r_{M',q'}$.

Un K-MA qui ne possède pas de K-réduction est dit *K-réduit.* On remarque que certains automates peuvent posséder des états inutiles, par exemple, des états inaccessibles. Plus précisément, un état $q \in Q$ est *accessible* si à partir d'un état initial $q_0 \in Q_I$, il existe un mot $u \in \Sigma^\star$ arrivant dans q, c'est-à-dire tel que $\varphi(q_0, u, q) \neq 0$. De même un état $q \in Q$ est *co-accessible* si à partir de q il existe un mot $u \in \Sigma^\star$ arrivant dans un état final $q_f \in Q_F$, c'est-à-dire tel que $\varphi(q, u, q_f) \neq 0$.

Définition 14 (Automate émondé)**.**
Un K-MA est émondé si tous ses états sont accessibles et co-accessibles.

Dans la suite, on ne considère plus que des automates émondés car tout K-MA possède un automate émondé équivalent et calculable facilement.

1.2.4 Caractérisation par les sous semi-modules stables

Cette section est dédiée à un Théorème qui permet de représenter les séries formelles rationnelles par les sous semi-modules des fonctions de $\Sigma^\star$ dans K noté $K^{\Sigma^\star}$. En fait, $K^{\Sigma^\star}$ et $K\langle\langle\Sigma\rangle\rangle$ sont isomorphes mais on notera $K^{\Sigma^\star}$ pour désigner le semi-module et $K\langle\langle\Sigma\rangle\rangle$ pour désigner le semi-anneau. Cette caractérisation permet de montrer de nombreux résultats de cette thèse et est donc essentielle.

Pour tout mot $u \in \Sigma^\star$, on définit l'opérateur $\dot{u}$ qui à toute fonction $f \in K^{\Sigma^\star}$ associe la fonction $\dot{u}f$ définie par $\dot{u}f(v) = f(uv)$. On remarque que ces opérateurs sont linéaires.

Définition 15 (Sous ensemble stable)**.**
Un sous-ensemble F de $K^{\Sigma^\star}$ est stable si $\forall u \in \Sigma^\star$, $\forall f \in F$, $\dot{u}f \in F$.

Théorème 1 ([Fliess, 1974; Jacob, 1975])**.**
Soit K un semi-anneau commutatif et r une série formelle de $K\langle\langle\Sigma\rangle\rangle$, alors les trois affirmations suivantes sont équivalentes,

(i) r est rationnelle ($r \in K^{rat}\langle\langle\Sigma\rangle\rangle$),

(ii) $[[\{\dot{u}r|u \in \Sigma^\star\}]]$ *est contenu dans un sous semi-module stable de* $K^{\Sigma^\star}$ *généré par une famille finie,*

(iii) r *appartient à un sous semi-module stable généré par une famille finie.*

Proposition 2.
Soit r *une série rationnelle de* $K^{rat}\langle\langle\Sigma\rangle\rangle$*, si* K *est un corps commutatif alors le sous espace vectoriel* $[[\{\dot{u}r|u \in \Sigma^\star\}]]$ *est généré par une famille finie.*

Démonstration. Comme r est une série rationnelle, $[[\{\dot{u}r|u \in \Sigma^\star\}]]$ est contenu dans un sous espace vectoriel généré par une famille finie. D'après les propriétés des espaces vectoriels, il est donc généré par une famille finie. □

La proposition ci-dessus est en général fausse si K n'est pas un corps. Cette propriété est fondamentale pour la suite.

Proposition 3 (Longueur suffisante des mots générant le sous-espace vectoriel stable)**.**
Soit K *un corps commutatif,* r *une série rationnelle de* $K^{rat}\langle\langle\Sigma\rangle\rangle$ *et* E *le plus petit sous espace vectoriel stable contenant* $[[\{\dot{u}r|u \in \Sigma^\star\}]]$*, alors* E *est de dimension* d *finie et* $E = \left[\left[\left\{\dot{u}r|u \in \Sigma^{<d}\right\}\right]\right] = [[\{\dot{u}r|u \in \Sigma^\star\}]]$.

Démonstration. D'abord E est de dimension finie d car généré par une famille finie d'après le Théorème 1. On propose l'Algorithme 1 pour construire une base de $[[\{\dot{u}r|u \in \Sigma^\star\}]]$. Premièrement, comme les vecteurs de B sont linéairement indépendants par construction, B contient au plus d vecteurs et donc l'algorithme termine. De plus, on montre par récurrence qu'à chaque début d'itération de la boucle principale, si B contient i vecteurs alors $C \subset \left\{\dot{u}r|u \in \Sigma^{\leq i}\right\}$. Donc, lorsque l'algorithme termine, $B \subset \left\{\dot{u}r|u \in \Sigma^{<d}\right\}$. Il reste à montrer que B est bien une base de E. Lorsque l'algorithme termine, on a que pour tout $\sigma \in \Sigma$ $\{\dot{\sigma}f|f \in B\} \subset [[B]]$. Par linéarité de l'opérateur $\dot{\sigma}$, on a $\dot{\sigma}[[B]] \subset [[B]]$, donc $[[B]]$ est stable. Or $r \in [[B]]$, et donc par stabilité $\{\dot{u}r|u \in \Sigma^\star\} \subset [[B]]$. Comme $[[B]]$ est un espace vectoriel $[[\{\dot{u}r|u \in \Sigma^\star\}]] \subset [[B]]$ et B est bien une base de E. □

Algorithme 1 Construit une base de $[[\{\dot{u}r|u \in \Sigma^\star\}]]$ quand K est un corps commutatif.

Entrées Une série $r \in K^{rat}\langle\langle\Sigma\rangle\rangle$ où K est un corps commutatif
Sortie Une base B de $[[\{\dot{u}r|u \in \Sigma^\star\}]]$

$B \leftarrow \emptyset$
$C \leftarrow \{r\}$
tant que $C \neq \emptyset$ **faire**
    Soit f un élément de C
    $C \leftarrow C \backslash \{f\}$
    si $f \notin [[B]]$ **alors**
        $B \leftarrow B \cup \{f\}$
        $C \leftarrow C \cup \{\dot{\sigma}f|\sigma \in \Sigma\}$
    fin si
fin tant que

1.2.5 Équivalence des définitions

Trois types de séries ont été définies précédemment : les séries rationnelles qui sont contenues dans des sous semi-modules stables générés par des familles finies, les séries reconnaissables par leur représentation linéaire et les séries réalisées par les automates à multiplicité. On montre ci-dessous que ces trois représentations définissent en fait le même ensemble de séries, que l'on désignera, par la suite, par l'ensemble des séries rationnelles. Les propositions qui suivent montrent que l'on peut passer d'une représentation à une autre et établissent ensemble l'équivalence des trois définitions.

Proposition 4 (Passage des séries rationnelles aux séries reconnaissables)**.**
Soit $r \in K^{rat}\langle\langle\Sigma\rangle\rangle$ une série rationnelle et $S = \{r_1, \dots, r_d\}$ une famille finie de séries de $K\langle\langle\Sigma\rangle\rangle$ générant un sous semi-module stable de $K\langle\langle\Sigma\rangle\rangle$ contenant r, on note $a_i \in K$ et $a^\sigma_{i,j} \in K$ les coefficients pour tout $\sigma \in \Sigma$ et entiers $i, j \in [1, d]$ tels que,

$$r = \sum_{i=1}^{d} a_i r_i \qquad \text{et} \qquad \dot\sigma r_i = \sum_{j=1}^{d} a^\sigma_{i,j} r_j.$$

Soit $(\boldsymbol{\alpha}_0, A, \boldsymbol{\alpha}_\infty)$ une représentation linéaire définie par $\boldsymbol{\alpha}_0[i] = a_i$, $A_\sigma[i,j] = a^\sigma_{i,j}$ et $\boldsymbol{\alpha}_\infty[i] = r_i(\varepsilon)$, alors $(\boldsymbol{\alpha}_0, A, \boldsymbol{\alpha}_\infty)$ est une représentation linéaire de r.

Démonstration. Avant tout, on remarque que les coefficients a_i et $a^\sigma_{i,j}$ existent car $[[\{r_1, \dots, r_d\}]]$ est stable et contient r. Ensuite, on montre que $\forall u \in \Sigma^\star, \begin{pmatrix} r_1(u) & \dots & r_d(u) \end{pmatrix}^\top = A_u \boldsymbol{\alpha}_\infty$, par induction sur la longueur de u. Pour $u = \varepsilon$, c'est vrai par définition de $\boldsymbol{\alpha}_\infty$. Supposons que la relation est vraie pour $u \in \Sigma^{\leq n}$. Soit $\sigma \in \Sigma$, on pose $v = \sigma u$ et on a,

$$\begin{aligned}
A_v \boldsymbol{\alpha}_\infty &= A_\sigma A_u \boldsymbol{\alpha}_\infty \\
&= A_\sigma \begin{pmatrix} r_1(u) & \dots & r_d(u) \end{pmatrix}^\top && \text{(par hypothèse d'induction)} \\
&= \begin{pmatrix} \sum_{j=1}^d a^\sigma_{1,j} r_j(u) & \dots & \sum_{j=1}^d a^\sigma_{d,j} r_j(u) \end{pmatrix}^\top && \text{(par définition de } A\text{)} \\
&= \begin{pmatrix} \dot\sigma r_1(u) & \dots & \dot\sigma r_d(u) \end{pmatrix}^\top && \text{(par définition de } a^\sigma_{i,j}\text{)} \\
&= \begin{pmatrix} r_1(v) & \dots & r_d(v) \end{pmatrix}^\top.
\end{aligned}$$

Maintenant, pour tout $u \in \Sigma^\star$, par définition de $\boldsymbol{\alpha}_\infty$ et de a_i, on a

$$\boldsymbol{\alpha}_\infty^\top A_u \boldsymbol{\alpha}_\infty = \sum_{i=1}^{d} a_i r_i(u) = r(u).$$

□

La transformation explicite entre les deux représentations de la Proposition 4 sera au cœur des algorithmes présentés dans cette thèse.

Proposition 5 (Passage des représentations linéaires aux automates)**.**
Soit $(\boldsymbol{\alpha}_0, A, \boldsymbol{\alpha}_\infty)$ une représentation linéaire de dimension d de la série r définie sur K, et $M = (\Sigma, Q, \iota, \varphi, \tau)$ le K-MA tel que $Q = \{1, \dots, d\}$, $\iota(i) = \boldsymbol{\alpha}_0[i]$, $\varphi(i, \sigma, j) = A_\sigma[i,j]$ et $\tau[i] = \boldsymbol{\alpha}_\infty[i]$, alors $r = r_M$.

Démonstration. Pour tout mot $u \in \Sigma^\star$, on a,

$$r(u) = \boldsymbol{\alpha}_0^\top A_u \boldsymbol{\alpha}_\infty = \sum_{i,j=1}^{d} \boldsymbol{\alpha}_0[i] A_u[i,j] \boldsymbol{\alpha}_\infty[j] = \sum_{i,j=1}^{d} \iota(i)\varphi(i,u,j)\tau(j) = r_M(u).$$

□

Proposition 6 (Passage des automates aux séries rationnelles)**.**
Soit $M = (\Sigma, Q, \iota, \varphi, \tau)$ un K-MA alors $[[\{r_{M,q}|q \in Q\}]]$ est un sous semi-module stable de $K\langle\langle\Sigma\rangle\rangle$ contenant r_M et r_M est une série rationnelle.

Démonstration. Premièrement, $r_M \in [[\{r_{M,q}|q \in Q\}]]$ car $r_M = \sum_{q\in Q} \iota(q) r_{M,q}$. Ensuite, $[[\{r_{M,q}|q \in Q\}]]$ est stable car pour tout $\sigma \in \Sigma$, pour tout mot $u \in \Sigma^\star$ et pour tout état $q \in Q$,

$$\dot{\sigma} r_{M,q}(u) = r_{M,q}(\sigma u) = \sum_{q' \in Q} \varphi(q, \sigma, q') r_{M,q'}(u).$$

Enfin, $\{r_{M,q}|q \in Q\}$ est une famille finie car Q est fini. □

Théorème 2 (Équivalence des trois représentations)**.**
Les ensembles des séries rationnelles, des séries reconnaissables et des séries réalisées par les K-MA coïncident.

Démonstration. Le Théorème 2 se déduit des Propositions 4 à 6. □

Dorénavant, on confondra représentation linéaire et automates.

On termine par un dernier résultat, lorsque K est un corps commutatif, utilisant l'équivalence des représentations.

Théorème 3 (Dimension et similarité des représentations linéaires minimales définies sur les corps commutatifs)**.**
Soit K un corps, toutes les représentations linéaires minimales d'une série rationnelle r sont similaires et de dimension égale à celle de l'espace vectoriel généré par $\{\dot{u}r|u \in \Sigma\}$.

Démonstration. Soit d la dimension de $[[\{\dot{u}r|u \in \Sigma\}]]$, d'après la Proposition 4, il existe une représentation linéaire de r de dimension d. Supposons qu'il existe une représentation linéaire de r de dimension $k < d$, alors à l'aide des Propositions 5 et 6, on pourrait construire une famille génératrice de $[[\{\dot{u}r|u \in \Sigma\}]]$ avec k vecteurs or $[[\{\dot{u}r|u \in \Sigma\}]]$ est de dimension $d > k$. Ainsi toutes les représentations linéaires minimales de r sont de dimension d. Elles sont de plus similaires. En effet, soit deux représentations linéaires minimales, on y associe respectivement les automates M et M' dont les ensembles d'états respectifs sont notés Q et Q', par la Proposition 5. Posons P la matrice inversible de passage entre les bases $\{r_{M,q}|q \in Q\}$ et $\{r_{M',q'}|q' \in Q'\}$ de $[[\{\dot{u}r|u \in \Sigma\}]]$, on vérifie alors que P permet de passer d'une représentation linéaire à l'autre. Elles sont donc similaires. □

1.3 Matrice de Hankel

Dans cette section, on introduit les matrices de Hankel. Une matrice de Hankel, est simplement une matrice bi-infinie contenant toutes les valeurs d'une série formelle rationnelle organisées selon une base de préfixes et de suffixes. Ainsi, une matrice de Hankel définit entièrement une série. Nous allons montrer des propriétés intéressantes sur le rang de ces matrices. On distinguera le cas où K est un corps pour montrer une propriété à la base des algorithmes d'apprentissage issus de la méthodes des moments. Enfin, on montrera que même un sous-bloc fini bien choisi de la matrice de Hankel est suffisant pour définir complètement une série.

Les algorithmes spectraux, présentés dans le Chapitre 2, utilisent les matrices de Hankel pour représenter une série formelle rationnelle. Deux variantes existent [Bailly, 2011a]. L'une utilise une matrice bi-infinie, l'autre des sous blocs finis de cette matrice

[Balle, 2013]. Cette caractérisation permet de passer d'une décomposition en facteurs de faible dimension de la matrice de Hankel à une série rationnelle quand K est un corps. Les notions présentées ici sont fondamentales pour le Chapitre 3. Si K est un semi-anneau, la caractérisation est plus complexe car une décomposition de la matrice de Hankel sur le semi-anneau ne permet pas de retrouver à coup sûr une série formelle rationnelle. Dans la Section 2.2.3.3, on présentera une caractérisation plus complète sur les semi-anneaux. Dans les sections suivantes, K désigne un corps.

1.3.1 Matrice de Hankel infinie

Définition 16 (Matrice de Hankel)**.**
Pour toute série r, on définit la matrice bi-infinie de Hankel, noté $H \in K^{\Sigma^\star \times \Sigma^\star}$, telle que $\forall u, v \in \Sigma^\star$, $H[u, v] = r(uv)$.

$$H = \begin{array}{c} \\ \varepsilon \\ a \\ b \\ aa \\ \vdots \end{array} \begin{array}{c} \begin{array}{ccccc} \varepsilon & a & b & aa & \dots \end{array} \\ \begin{pmatrix} r(\varepsilon) & r(a) & r(b) & r(aa) & \dots \\ r(a) & r(aa) & r(ab) & r(aaa) & \dots \\ r(b) & r(ba) & r(bb) & r(baa) & \dots \\ r(aa) & r(aaa) & r(aab) & r(aaaa) & \dots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix} \end{array}$$

On élargit la définition de rang aux matrices définies sur les corps K.

Définition 17 (Rang)**.**
Pour toute matrice $H \in K^{m \times n}$, on note $\mathrm{rang}_K(H)$ *le rang de H défini comme étant la plus petite valeur k telle qu'il existe $P \in K^{d \times k}$ et $S \in K^{k \times m}$ vérifiant $H = PS$. Quand $K = \mathbb{R}$ peut être déduit du contexte, on note simplement* $\mathrm{rang}(H)$.

Proposition 7 (Rang de la matrice de Hankel d'une série formelle rationnelle)**.**
Soit $r \in K^\Sigma$ une série formelle et $H[u, v] = r(uv)$ sa matrice de Hankel, si r est réalisée par un K-MA de dimension d alors $\mathrm{rang}_K(H) \leq d$.

Démonstration. Soit r une série formelle rationnelle et $(\boldsymbol{\alpha}_0, A, \boldsymbol{\alpha}_\infty)$ une représentation linéaire de dimension d du K-MA réalisant r, alors les deux matrices $P \in K^{\Sigma^\star \times d}$ et $S \in K^{d \times \Sigma^\star}$ définies par

$$P = ((\boldsymbol{\alpha}_0^\top A_u)^\top)_{u \in \Sigma^\star}^\top \text{ et } S = (A_v \boldsymbol{\alpha}_\infty)_{v \in \Sigma^\star},$$

vérifient $H = PS$ car $H[u, v] = r(uv) = \boldsymbol{\alpha}_0 A_u A_v \boldsymbol{\alpha}_\infty$. Par les dimensions de S et P, on a $\mathrm{rang}(H) \leq d$. □

Soit r une série formelle et $H = PS$ sa matrice de Hankel telle que $P \in K^{\Sigma^\star \times d}$ et $S \in K^{d \times \Sigma^\star}$, alors les lignes $H[u, :]$, qui correspondent aux séries $\dot{u}r$, sont contenues dans le sous espace vectoriel généré par les lignes de S. Si le sous espace vectoriel est stable alors la Proposition 4 permet de montrer que r est rationnelle et même d'en calculer une représentation linéaire de dimension d. Or, comme K est un corps, $[[\{\dot{u}r | u \in \Sigma^\star\}]]$ est stable comme le montre la proposition suivante. On donne de plus les équations matricielles de passage d'une factorisation de H à la représentation linéaire.

Proposition 8 (Passage de la matrice de Hankel à la représentation linéaire pour les corps)**.**
Soit $H \in \mathbb{R}^{\Sigma^\star \times \Sigma^\star}$ une matrice de Hankel de rang d, on note, pour tout $\sigma \in \Sigma$, T_σ la matrice telle que $H[u\sigma, v] = T_\sigma H = H T_\sigma^\top = H[u, \sigma v]$. Soit $PS = H$ une factorisation

de rang d, alors il existe une série r rationnelle telle que $H[u,v] = r(uv)$ qui est réalisée par un $\mathbb{R}$-MA dont la représentation linéaire $(\boldsymbol{\alpha}_0, A, \boldsymbol{\alpha}_\infty)$ de dimension d vérifie

(i) $A_\sigma = ST_\sigma^\top S^\dagger = P^\dagger T_\sigma P$,
(ii) $\boldsymbol{\alpha}_0^\top = \mathbf{1}_\varepsilon^\top H S^\dagger = \mathbf{1}_\varepsilon^\top P$,
(iii) $\boldsymbol{\alpha}_\infty = S\mathbf{1}_\varepsilon = P^\dagger H \mathbf{1}_\varepsilon$.

Démonstration. Soit H une matrice de Hankel de rang d et $PS = H$ une factorisation associée, alors le sous espace vectoriel généré par les lignes de S, noté $\left[\left[S^\top\right]\right]$, contient le sous espace vectoriel généré par les lignes de H, noté $\left[\left[H^\top\right]\right]$. On remarque que $\left[\left[H^\top\right]\right] = [[\{\dot{u}r | u \in \Sigma^\star\}]] \subset \left[\left[S^\top\right]\right]$. Soit T_σ la représentation matricielle de l'opérateur linéaire $\dot{\sigma}$, telle que

$$T_\sigma[u,v] = \mathbb{1}_{u\sigma}(v)$$

Ainsi pour toute série formelle r, on a que

$$T_\sigma(r(u))_{u\in\Sigma^\star} = (\dot{\sigma}r(u))_{u\in\Sigma^\star}.$$

Comme P est de rang plein, on a que $(P^\top P)^{-1}$ existe et $(P^\top P)^{-1}P^\top P = I$ et on pose

$$ST_\sigma^\top = (P^\top P)^{-1}P^\top P S T_\sigma^\top = (P^\top P)^{-1}P^\top H T_\sigma^\top.$$

Par définition de H, on a que

$$T_\sigma H = H T_\sigma^\top.$$

Donc $ST_\sigma^\top = (P^\top P)^{-1}P^\top T_\sigma H = (P^\top P)^{-1}P^\top T_\sigma P S$ pour tout $\sigma \in \Sigma$ et $\left[\left[S^\top\right]\right]$ est stable. Ainsi les lignes de S définissent une famille finie génératrice qui peut être utilisée dans la Proposition 4, où l'on a

$$\mathbf{1}_\varepsilon^\top H = \boldsymbol{\alpha}_0 S,$$
$$ST_\sigma^\top = A_\sigma S.$$

De $ST_\sigma^\top = P^\dagger T_\sigma P S$, on identifie $A_\sigma = P^\dagger T_\sigma P$. De $\mathbf{1}_\varepsilon^\top H = \boldsymbol{\alpha}_0 S$, on trouve $\boldsymbol{\alpha}_0 = \mathbf{1}_\varepsilon^\top H S^\dagger = \mathbf{1}_\varepsilon P$ car S est de rang plein. Enfin, d'après la Proposition 4, on a simplement que $\boldsymbol{\alpha}_\infty = S\mathbf{1}_\varepsilon = P^\dagger H \mathbf{1}_\varepsilon$. Enfin, on a,

$$ST_\sigma^\top S^\dagger = P^\dagger P S T_\sigma^\top S^\dagger = P^\dagger T_\sigma H S^\dagger = P^\dagger T_\sigma P.$$

□

Les deux propositions précédentes permettent d'établir une caractérisation des série formelle rationnelles sur $\mathbb{R}$ par les matrices de Hankel de rang fini.

Théorème 4 (Caractérisation des séries formelles rationnelles sur un corps par les matrices de Hankel de rang fini)**.**
Soit une série formelle $r \in K^\Sigma$, où K est un corps, et $H[u,v] = r(uv)$ sa matrice de Hankel, alors r est réalisée par un K-MA de dimension d si et seulement si $\mathrm{rang}_K(H) \leq d$.

Démonstration. Conséquence directe des Propositions 7 et 8. □

Les deux propositions précédentes sont valables pour des matrices de Hankel bi-infinies. Lors de l'apprentissage une matrice bi-infinie peut être approchée par une matrice ayant un nombre fini de valeurs non nulles rendant ainsi les calculs possibles. Cependant, dans de nombreuses applications utilisant un large ensemble d'exemples avec des mots très différents, le nombre de valeurs non nulles devient trop grand et on préférera exécuter les calculs sur un sous-ensemble de ses valeurs. On définit alors la matrice de Hankel associée à une base finie.

1.3.2 Matrices de Hankel finies

Définition 18 (Matrices de Hankel finies)**.**
Soit deux ensembles finis de mots $\mathcal{P} \subset \Sigma^\star$ et $\mathcal{S} \subset \Sigma^\star$, on note $\mathcal{B} = (\mathcal{P}, \mathcal{S})$ une base formée de préfixes et de suffixes. On note $H_\mathcal{B}$ le sous bloc de la matrice de Hankel H sur la base $\mathcal{B}$. De plus, on note $H^\sigma_\mathcal{B}$ le sous bloc de H sur la base $(\mathcal{P}\sigma, \mathcal{S})$, $\mathbf{h}_\mathcal{S}$ le sous bloc de H sur la base $(\{\varepsilon\}, \mathcal{S})$, $\mathbf{h}_\mathcal{P}$ le sous bloc de H sur la base $(\mathcal{P}, \{\varepsilon\})$.

La Proposition 8 utilise la matrice T_σ définie sur la base $(\Sigma^\star, \Sigma^\star)$ et donc la preuve n'est plus valide pour le cas des matrices finies. Il convient alors de l'adapter aux bases finies comme le propose Balle [2013]. Une condition nécessaire est que la base utilisée soit complète.

Définition 19 (Base complète)**.**
Soit une matrice de Hankel H, une base $\mathcal{B}$ telle que $\operatorname{rang}_K(H_\mathcal{B}) = \operatorname{rang}_K(H)$ est complète.

Proposition 9 (Passage des matrices de Hankel finies à la représentation linéaire pour les corps [Balle, 2013, Lemme 5.2.1])**.**
Soient $H_\mathcal{B}$ $\{H^\sigma_\mathcal{B}\}_{\sigma\in\Sigma}$ des matrices de Hankel de $\mathbb{R}^{\mathcal{P}\times\mathcal{S}}$, d le rang $H_\mathcal{B}$ et $PS = H_\mathcal{B}$ une factorisation de rang d, si $\mathcal{B}$ est complète alors il existe une unique série r rationnelle qui est réalisée par un $\mathbb{R}$-MA dont représentation linéaire $(\boldsymbol{\alpha}_0, A, \boldsymbol{\alpha}_\infty)$ de dimension d vérifie

(i) $A_\sigma = P^\dagger H^\sigma_\mathcal{B} S^\dagger = P^\dagger H^\sigma_\mathcal{B} (P^\dagger H_\mathcal{B})^\dagger$,

(ii) $\boldsymbol{\alpha}_0^\top = \mathbf{h}_\mathcal{S}^\top S^\dagger = \mathbf{h}_\mathcal{S}^\top (P^\dagger H_\mathcal{B})^\dagger$,

(iii) $\boldsymbol{\alpha}_\infty = P^\dagger \mathbf{h}_\mathcal{P}$.

Ainsi la caractérisation donnée par le Théorème 4 s'étend aux matrices de Hankel finies.

1.4 Langages stochastiques

Dans cette section, on définit un sous-ensemble d'intérêt des séries formelles rationnelles, les langages stochastiques. Ceux-ci sont des distributions sur les mots de longueur finie. Nous verrons une propriété très intéressante pour les langages stochastiques rationnels. Selon qu'un langage stochastique est rationnel sur $\mathbb{R}$ ou sur $\mathbb{R}^+$, il pourra être représenté ou non par un automate décrit par un nombre fini de contraintes rendant possible ou non l'apprentissage.

1.4.1 Définitions

Définition 20 (Langage stochastique)**.**
L'ensemble des langages stochastiques, noté $S(\Sigma)$, est constitué des séries formelles de $K\langle\langle\Sigma\rangle\rangle$ telles que $\forall u \in \Sigma^\star r(u) \geq 0$ et $\sum_{u\in\Sigma^\star} r(u) = 1$.

Définition 21 (Langage stochastique rationnel)**.**
L'ensemble des langages stochastiques rationnels à coefficients dans K est noté $S^{rat}_K(\Sigma) = S(\Sigma) \cap K^{rat}\langle\langle\Sigma\rangle\rangle$.

Dans la suite du manuscrit, on suppose que $K \in \{\mathbb{R}, \mathbb{R}^+\}$.

1.4.2 Caractérisation par l'enveloppe convexe stable

Pour les langages stochastiques rationnels, on a une caractérisation plus précise par l'enveloppe convexe de familles finies génératrices de sous semi-modules stables.

Définition 22 (Enveloppe convexe)**.**
Soit $S \in S(\Sigma)$ *un ensemble de langages stochastiques, on note* $\mathrm{conv}(S) = \left\{\sum_{s\in S} a_s s | \sum_{s\in S} a_s = 1; \forall s \in S,\, a_s \in \mathbb{R}^+\right\}$*, l'enveloppe convexe de* S*.*

Théorème 5 ([Denis et Esposito, 2008])**.**
Une série formelle p *est un langage stochastique rationnel (*$p \in S_K^{rat}(\Sigma)$*) si et seulement si une famille finie* S *de langages stochastiques (*$S \subset S(\Sigma)$*) génère un sous semi-module stable dont l'enveloppe convexe contient* p *(*$p \in \mathrm{conv}(S)$*).*

Pour un langage stochastique p, pour tout ensemble de mots $U \subset \Sigma^\star$ on note $p(U) = \sum_{u\in U} p(u)$. Cette somme est bien définie, même pour des ensembles U infinis, d'après les propriétés de la Définition 20.

1.4.3 Caractérisation par les automates probabilistes

Dans cette section, on montre que les langages stochastiques rationnels sur un semi-anneau positif ($\mathbb{R}^+$) sont réalisés par une classe particulière d'automates. Le cas général des langages stochastiques rationnels sur $\mathbb{R}$ est délicat. En fait, il n'existe pas de représentation adaptée à ces langages. On détaillera plus tard ce cas dans la Section 1.4.4. On définit pour l'instant la classe d'automates suivante.

Définition 23 (Automate fini probabiliste non déterministe, ou *Probabilistic Non-deterministic Finite Automata* (PNFA))**.**
Un PNFA est un $\mathbb{R}^+$*-MA* $(\Sigma, Q, \iota, \varphi, \tau)$ *tel que* $\sum_{q\in Q} \iota(q) = 1$ *et* $\forall q \in Q$, $\tau(q) + \varphi(q, \Sigma, Q) = 1$*.*

La Définition 23 implique que les fonctions ι, φ, τ prennent valeurs dans $[0, 1]$ et peuvent donc s'interpréter comme des probabilités respectivement d'entrées, de transitions entre états et de sorties. Sous la forme matricielle de la représentation linéaire équivalente, $(\boldsymbol{\alpha}_0, A, \boldsymbol{\alpha}_\infty)$, ces contraintes s'écrivent,

$$\boldsymbol{\alpha}_0^\top \mathbf{1} = 1 \qquad \text{et} \qquad \boldsymbol{\alpha}_\infty + \sum_{\sigma\in\Sigma} A_\sigma \mathbf{1} = \mathbf{1}. \tag{1.1}$$

Les PNFA permettent de caractériser les langages stochastiques rationnels sur $\mathbb{R}^+$, comme le montrent les deux propositions suivantes établies par Denis et Esposito [2008].

Proposition 10 (Les PNFA réalisent des langages)**.**
Soit un PNFA $M = (\Sigma, Q, \iota, \varphi, \tau)$*, pour tout état* $q \in Q$*,* $p_{M,q}$ *définit un langage stochastique et donc* p_M *aussi.*

Démonstration. Par récurrence sur k, on montre que pour tout état $q \in Q$, on a

$$\sum_{u\in\Sigma^{\leq k}} p_{M,q}(u) + \varphi(q, \Sigma^{k+1}, Q) = 1. \tag{1.2}$$

Pour $k = 0$, on a par définition,

$$p_{M,q}(\varepsilon) + \varphi(q, \Sigma, Q) = \tau(q) + \varphi(q, \Sigma, Q) = 1.$$

Supposons la relation vrai pour k, on a alors,

$$\begin{aligned}
&\sum_{u\in\Sigma^{\leq k+1}} p_{M,q}(u) + \varphi(q,\Sigma^{k+2},Q) \\
&\qquad = \sum_{u\in\Sigma^{\leq k}} p_{M,q}(u) + \sum_{q'\in Q} \varphi(q,\Sigma^{k+1},q')\tau(q') + \sum_{q'\in Q} \varphi(q,\Sigma^{k+1},q')\varphi(q',\Sigma,Q) \\
&\qquad = \sum_{u\in\Sigma^{\leq k}} p_{M,q}(u) + \sum_{q'\in Q} \varphi(q,\Sigma^{k+1},q') \qquad \text{par hypothèse de récurrence} \\
&\qquad = 1 \qquad \text{par définition.}
\end{aligned}$$

Comme M est émondé, il existe un mot $u \in \Sigma^{l-1}$, tel que $p_{M,q}(u) > 0$. Ainsi on pose,

$$a = \varphi(q,\Sigma^l,Q) + \sum_{v\in\Sigma^{\leq l-1}} p_{M,q}(v) - p_{M,q}(u) = 1 - p_{M,q}(u).$$

Par définition, on a $a < 1$. De plus, la positivité de φ implique que $\varphi(q,\Sigma^l,Q) \leq a$. Ensuite, par récurrence sur k, on montre que

$$\varphi(q,\Sigma^{lk},Q) \leq a^k.$$

Enfin, soit $l = |Q|$, on a que pour tout entier k et pour tout état $q \in Q$,

$$1 \geq \sum_{u\in\Sigma^{<lk}} p_{M,q}(u) = 1 - \varphi(q,\Sigma^{lk},Q) \geq 1 - a^k.$$

Cela permet de conclure, que la somme converge bien vers 1,

$$\sum_{u\in\Sigma^\star} p_{M,q}(u) = 1$$

Finalement,

$$\sum_{u\in\Sigma^\star} p_M(u) = \sum_{u\in\Sigma^\star}\sum_{q\in Q} \iota(q)p_{M,q}(u) = 1.$$

□

Proposition 11 (Passage d'une enveloppe convexe stable à un PNFA)**.**
Soit $p \in \mathbb{R}^+\langle\langle\Sigma\rangle\rangle$ et $S = \{p_1,\ldots,p_d\}$ une famille finie de langages stochastiques de $S(\Sigma)$ générant un sous semi-module stable et telle que $p \in \mathrm{conv}(S)$, on note $a_i \in \mathbb{R}^+$ et $a^\sigma_{i,j} \in \mathbb{R}^+$ les coefficients tels que, pour tout $\sigma \in \Sigma$ et entiers $i,j \in [1,d]$,

$$p = \sum_{i=1}^d a_i p_i \qquad \text{et} \qquad \dot\sigma p_i = \sum_{j=1}^d a^\sigma_{i,j} p_j.$$

Soit $(\boldsymbol{\alpha}_0, A, \boldsymbol{\alpha}_\infty)$ la représentation linéaire définie par $\boldsymbol{\alpha}_0[i] = a_i$, $A_\sigma[i,j] = a^\sigma_{i,j}$ et $\boldsymbol{\alpha}_\infty[i] = p_i(\varepsilon)$, alors $(\boldsymbol{\alpha}_0, A, \boldsymbol{\alpha}_\infty)$ définit un PNFA réalisant p.

Démonstration. D'abord, d'après le Théorème 1, p est rationnel sur $\mathbb{R}^+$. Comme les p_i vérifient $p_i(\Sigma^\star) = 1$ et que $p \in \mathrm{conv}(S)$, on a que $p(\Sigma^\star) = 1$. Puis, d'après la Proposition 4, $(\boldsymbol{\alpha}_0, A, \boldsymbol{\alpha}_\infty)$ réalise p. Il reste montrer que $(\boldsymbol{\alpha}_0, A, \boldsymbol{\alpha}_\infty)$ définit un PNFA. En effet, on a $p = \sum_{i=1}^d a_i p_i$, en particulier $p(\Sigma^\star) = \sum_{i=1}^d a_i p_i(\Sigma^\star)$. Or

$p(\Sigma^\star) = p_i(\Sigma^\star) = 1$ pour tout $i \in [1, d]$. Donc $\sum_{i=1}^{d} a_i = 1$. Enfin, on a pour tout $i \in [1, d]$ que

$$\dot{\sigma} p_i(\Sigma^\star) = \sum_{j=1}^{d} a_{i,j}^{\sigma} p_j(\Sigma^\star),$$

$$\sum_{\sigma \in \Sigma} p_i(\sigma\Sigma^\star) = \sum_{\sigma \in \Sigma} \sum_{j=1}^{d} a_{i,j}^{\sigma},$$

$$p_i(\Sigma\Sigma^\star) = \sum_{\sigma \in \Sigma} \sum_{j=1}^{d} a_{i,j}^{\sigma}.$$

Comme p_i est un langage stochastique, $p_i(\Sigma^\star) = p_i(\varepsilon) + p_i(\Sigma\Sigma^\star)$. On obtient alors que $\boldsymbol{\alpha}_\infty + A_\Sigma \mathbf{1} = \mathbf{1}$. Donc $(\boldsymbol{\alpha}_0, A, \boldsymbol{\alpha}_\infty)$ définit un PNFA. □

Théorème 6 (Caractérisation de $S^{rat}_{\mathbb{R}^+}(\Sigma)$ par les PNFA)**.**
Une série p appartient à $S^{rat}_{\mathbb{R}^+}(\Sigma)$ si et seulement si elle est réalisée par un PNFA.

Démonstration. Conséquence directe des Propositions 6, 10 et 11. □

De plus, soit $p \in S^{rat}_{\mathbb{R}^+}(\Sigma)$, on note que le plus petit sous semi-module stable contenant p est de dimension égale à celle d'un PNFA minimal réalisant p.

1.4.4 Cas des langages stochastiques rationnels sur un corps

Nous avons associé aux langages stochastiques rationnels un type d'automates à multiplicité dont la représentation permet un apprentissage consistant (mais pas efficace) comme on le verra dans les chapitres suivants. Mais qu'en est-il de $S^{rat}_{\mathbb{R}}(\Sigma)$? En fait, il n'existe pas de « bonne » classe d'automates à multiplicité associés à cet ensemble. On pourrait définir des automates à multiplicité stochastiques par des $\mathbb{R}$-MA auxquels on aurait imposé la positivité de la série et la convergence de celle-ci vers 1. Bien que cette représentation corresponde à $S^{rat}_{\mathbb{R}}(\Sigma)$, elle n'est pas adaptée pour l'apprentissage [Denis et Esposito, 2008; Esposito, 2004].

Premièrement, les $\mathbb{R}$-MA ne sont pas une représentation robuste pour les langages stochastiques rationnels. En effet, Esposito [2004] montre qu'à partir d'un $\mathbb{R}$-MA définissant un langage stochastique, un changement aussi petit que l'on veut dans les paramètres définit un autre $\mathbb{R}$-MA qui réalise une série négative ou bien une série non bornée.

Deuxièmement, il est indécidable [Denis et Esposito, 2004b] si un $\mathbb{R}$-MA génère un langage stochastique. Plus généralement, il est indécidable si un $\mathbb{R}$-MA génère une série positive. De plus, Esposito [2004] montre qu'il n'existe pas de sous-ensemble de $\mathbb{Q}$-MA récursivement énumérable décrivant $S^{rat}_{\mathbb{Q}}(\Sigma)$. Cela permet à l'auteur de conclure, qu'il n'existe pas d'algorithme d'inférence propre pour les $\mathbb{Q}$-MA identifiant à la limite avec probabilité 1 les langages de $S^{rat}_{\mathbb{Q}}(\Sigma)$ (voir les différents modèles d'apprentissage du Chapitre 2).

Ce résultat permet aussi d'établir le théorème suivant qui montre que $S^{rat}_{\mathbb{R}}(\Sigma)$ est un ensemble plus riche que les autres.

Théorème 7 ([Denis et Esposito, 2004a])**.**

$$S^{rat}_{\mathbb{R}^+}(\Sigma) \subsetneq S^{rat}_{\mathbb{R}}(\Sigma)$$

Dans la Section 1.6, on montre que les $\mathbb{R}$-MA offrent aussi des représentations bien plus compactes. Malgré ces avantages, ils ne constituent pas une représentation adaptée à l'apprentissage.

1.5 Langages stochastiques résiduels

Dans les sections précédentes, on a établi que les langages stochastiques rationnels sur $\mathbb{R}$ ne formaient pas une classe proprement apprenable par un algorithme consistant. Ainsi, on pourrait se concentrer uniquement sur les langages stochastiques rationnels sur $\mathbb{R}^+$. Malheureusement, on verra au chapitre suivant que ces langages stochastiques ne sont pas apprenables efficacement. Ainsi, on propose une classe de langages stochastiques moins riche qui elle sera efficacement apprenable par un algorithme consistant. Cette classe de langages stochastiques fait l'objet du Chapitre 7.

1.5.1 Définitions

Dans cette section, on définit les langages rationnels résiduels qui sont des distributions conditionnelles sur les mots de longueur finie commençant par un certain préfixe.

Définition 24 (Résidus)**.**
Soit $p \in S_K^{rat}(\Sigma)$, on définit l'ensemble des résidus $\mathrm{res}(p)$ *tel que* $\mathrm{res}(p) = \{u \in \Sigma^\star | p(u\Sigma^\star) \neq 0\}$.

Soit $p \in S_K^{rat}(\Sigma)$ et $u \in \mathrm{res}(p)$, on note $u^{-1}p$ la série formelle définie par,

$$\forall v \in \Sigma^\star,\ u^{-1}p(u) = \frac{p(uv)}{p(u\Sigma^\star)}.$$

Définition 25 (Ensemble des langages stochastiques résiduels)**.**
L'ensemble des langages stochastiques résiduels d'un langage stochastique p est noté $\mathrm{Res}(p) = \{u^{-1}p | u \in \mathrm{res}(p)\}$.

Proposition 12 ([Denis et Esposito, 2008])**.**
Soit $p \in S_K^{rat}(\Sigma)$, on a $[[\mathrm{Res}(p)]] = [[\{\grave{u}p | u \in \Sigma^\star\}]]$ et $\forall u \in \Sigma^\star$, $u^{-1}p \in S_K^{rat}(\Sigma)$.

Dorénavant, on appelle semi-module résiduel de p, le sous semi-module $[[\mathrm{Res}(p)]]$.

1.5.2 Caractérisation par le sous semi-module résiduel

On se spécialise maintenant au cas où $K \in \{\mathbb{R}, \mathbb{R}^+\}$. On a vu que certains résultats ne sont valables que si K est un corps commutatif (Proposition 2 et théorème 3). En particulier, la Proposition 12 implique que un langage stochastique p est rationnel si son semi-module résiduel est généré par une famille finie de langages stochastiques. Cependant, la réciproque est fausse lorsque K n'est pas un corps. En effet, il se peut que $[[\mathrm{Res}(p)]]$ soit contenu dans un sous semi-module stable généré par une famille finie sans que celui-ci, bien que stable, ne soit généré par une famille finie. Cela amène à considérer deux sous-ensembles de $S_{\mathbb{R}^+}^{rat}(\Sigma)$.

Définition 26 (Langage stochastique dont le semi-module résiduel est généré par une famille finie)**.**
On note $S_K^{[[Res]]}(\Sigma)$, l'ensemble des Langages stochastiques de $S_K^{rat}(\Sigma)$ dont le semi-module résiduel est généré par une famille finie.

Définition 27 (Langage stochastique possèdant un nombre fini de langages stochastiques résiduels)**.**
On note $S_K^{Res}(\Sigma)$, l'ensemble des langages stochastiques de $S_K^{rat}(\Sigma)$ qui possède un nombre fini de langages stochastiques résiduels.

Les Définitions 26 et 27 impliquent le résultat évident suivant,

Proposition 13.

$$S_K^{Res}(\Sigma) \subset S_K^{[[Res]]}(\Sigma).$$

Les deux propositions suivantes montrent qu'il est inutile de considérer les ensembles $S_{\mathbb{R}}^{Res}(\Sigma)$ et $S_{\mathbb{R}}^{[[Res]]}(\Sigma)$.

Proposition 14 ([Denis et Esposito, 2008])**.**

$$S_{\mathbb{R}^+}^{Res}(\Sigma) = S_{\mathbb{R}}^{Res}(\Sigma)$$

Proposition 15.

$$S_{\mathbb{R}}^{[[Res]]}(\Sigma) = S_{\mathbb{R}}^{rat}(\Sigma)$$

Démonstration. Corollaire de la Proposition 2. □

Nous donnons maintenant un résultat d'unicité.

Proposition 16 (Unicité de l'enveloppe convexe minimale stable, [Denis et Esposito, 2008])**.**
Soit $p \in S_{\mathbb{R}^+}^{[[Res]]}(\Sigma)$, il existe un unique sous-ensemble $R \subset \mathrm{Res}(p)$ minimal qui génère un sous semi-module stable tel que $p \in \mathrm{conv}(R)$.

La Proposition 16 permet d'étendre le résultat de Denis et collab. [2002] à $S_{\mathbb{R}^+}^{[[Res]]}(\Sigma)$ sur la longueur minimale des résidus dont le langage stochatique résiduel appartient au générateur minimal R de l'enveloppe convexe stable.

Proposition 17 ([Denis et Esposito, 2008; Denis et collab., 2002])**.**
Pour tout polynôme q, il existe $p \in S_{\mathbb{R}^+}^{[[Res]]}(\Sigma)$, tel que $|u| > q(|Q|)$ pour tout résidu u dont le $u^{-1}p \in R$, où R est l'unique sous-ensemble défini Proposition 16.

1.5.3 Caractérisation par les automates probabilistes résiduels

Dans la section précédente, on a défini deux ensembles de langages stochastiques d'intérêts $S_{\mathbb{R}^+}^{Res}(\Sigma)$ et $S_{\mathbb{R}^+}^{[[Res]]}(\Sigma)$. Dans cette section, on définit deux classes d'automates avant de montrer qu'elles caractérisent ces ensembles de langages. Les résultats suivants peuvent être retrouvés dans [Denis et Esposito, 2008].

Définition 28 (Automate fini probabiliste résiduel, ou *Probabilistic Residuel Finite Automata* (PRFA))**.**
Un PRFA est un PNFA $(\Sigma, Q, \iota, \varphi, \tau)$ tel que pour tout état $q \in Q$, il existe un mot $u \in \Sigma^\star$ tel que la série associée à cet état soit égale à la série résiduelle définie par u ($p_{M,q} = u^{-1}p_M$).

Définition 29 (Automate fini probabiliste déterministe, ou *Probabilistic Deterministic Finite Automata* (PDFA))**.**
Un PDFA est un PNFA dont le support est un DFA. Autrement dit, il ne possède qu'un seul état initial et les transitions entre états sont déterministes.

Proposition 18 (Semi-module résiduel de PRFA)**.**
Soit un PRFA $M = (\Sigma, Q, \iota, \varphi, \tau)$, alors $[[Res(p_M)]]$ est généré par une famille finie de langage stochastique de $\mathrm{Res}(p_M)$.

Démonstration. D'après la Définition 28, $\{p_{M,q} | q \in Q\} \subset \mathrm{Res}(p_M)$ donc $[[\{p_{M,q} | q \in Q\}]] \subset [[\mathrm{Res}(p_M)]]$. Montrons que $\mathrm{Res}(p_M) \subset [[\{p_{M,q} | q \in Q\}]]$. En effet, on a pour tout $v \in \Sigma^\star$,

$$\begin{aligned} \dot{u}p_M(v) &= \sum_{q,q' \in Q} \iota(q)\varphi(q, uv, q')\tau(q') \\ &= \sum_{q \in Q} \iota(q) \sum_{s \in Q} \varphi(q, u, s) \sum_{q' \in Q} \varphi(s, v, q')\tau(q') \\ &= \sum_{q,s \in Q} \iota(q)\varphi(q, u, s)p_{M,s}(v). \end{aligned}$$

Ainsi, $\dot{u}p_M \in [[\{p_{M,q} | q \in Q\}]]$ et donc $[[\{\dot{u}p_M | u \in \mathrm{Res}(p_M)\}]] \subset [[\{p_{M,q} | q \in Q\}]]$. Comme $[[\{\dot{u}p_M | u \in \mathrm{Res}(p_M)\}]]$ et $[[\{u^{-1}p_M | u \in \mathrm{Res}(p_M)\}]]$ coïncident, alors $[[\{u^{-1}p_M | u \in \mathrm{Res}(p_M)\}]] \subset [[\{p_{M,q} | q \in Q\}]]$. □

Proposition 19 (Passage d'un semi-module résiduel à un PRFA)**.**
Soit $p \in S^{rat}_{\mathbb{R}^+}(\Sigma)$ et $R = \{p_1, \ldots, p_d\}$ une famille finie telle que $R \subset \mathrm{Res}(p)$ et $[[\mathrm{Res}(p)]] \subset \mathrm{conv}(R)$, on note $a_i \in \mathbb{R}^+$ et $a^\sigma_{i,j} \in \mathbb{R}^+$ les coefficients tels que, pour tout $\sigma \in \Sigma$ et entiers $i, j \in [1, d]$,

$$p = \sum_{i=1}^{d} a_i p_i \qquad \text{et} \qquad \dot{\sigma}p_i = \sum_{j=1}^{d} a^\sigma_{i,j} p_j.$$

Soit $(\boldsymbol{\alpha}_0, A, \boldsymbol{\alpha}_\infty)$ la représentation linéaire définie par $\boldsymbol{\alpha}_0[i] = a_i$, $A_\sigma[i, j] = a^\sigma_{i,j}$ et $\boldsymbol{\alpha}_\infty[i] = p_i(\varepsilon)$, alors $(\boldsymbol{\alpha}_0, A, \boldsymbol{\alpha}_\infty)$ définit un PRFA réalisant p.

Démonstration. Comme $R \subset \mathrm{Res}(p)$ et $[[\mathrm{Res}(p)]] \subset \mathrm{conv}(R)$, alors $[[\mathrm{Res}(p)]] = [[R]]$. Ainsi, le semi-module résiduel de p est généré par une famille finie. Donc d'après la Définition 26, $p \in S^{[[Res]]}_{\mathbb{R}^+}(\Sigma)$. D'après la Proposition 11, $(\boldsymbol{\alpha}_0, A, \boldsymbol{\alpha}_\infty)$ réalise p et définit un PNFA. Dans la preuve de la Proposition 4, on a montré que pour tout $u \in \Sigma^\star$,

$$\begin{pmatrix} p_1(u) & \ldots & p_d(u) \end{pmatrix}^\top = A_u \boldsymbol{\alpha}_\infty.$$

Or $A_u \boldsymbol{\alpha}_\infty = (p_{M,q}(u))_{q \in Q}$. On obtient ainsi que les $p_{M,q}$ définissent des langages résiduels. Ainsi, $(\boldsymbol{\alpha}_0, A, \boldsymbol{\alpha}_\infty)$ définit un PRFA. □

Théorème 8 (Caractérisation de $S^{[[Res]]}_{\mathbb{R}^+}(\Sigma)$ par les PRFA)**.**
Une série p appartient à $S^{[[Res]]}_{\mathbb{R}^+}(\Sigma)$ si et seulement si elle est réalisée par un PRFA.

Démonstration. Conséquence directe des Propositions 18 et 19 et du Théorème 6. □

De plus, soit $p \in S^{[[Res]]}_{\mathbb{R}^+}(\Sigma)$ et R l'ensemble minimal unique de résiduels défini dans la Proposition 16, alors le cardinal de R est égal à la dimension d'un PRFA minimal réalisant p.

On ne donnera pas ici le détail de la caractérisation pour $S^{Res}_{\mathbb{R}^+}(\Sigma)$, car les PDFA ne font pas l'objet de cette thèse. On mentionne quand même le résultat suivant [Denis et Esposito, 2008].

Théorème 9 (Caractérisation de $S^{Res}_{\mathbb{R}^+}(\Sigma)$ par les PDFA)**.**
Une série p appartient à $S^{Res}_{\mathbb{R}^+}(\Sigma)$ si et seulement si elle est réalisée par un PDFA.

1.6 Richesse des différentes classes d'automates stochastiques

Dans cette section, on mentionne des résultats sur la richesse des classes de langages stochastiques définies précédemment.

1.6.1 Expressivité

On rappelle une série de résultats sur l'expressivité des automates définis dans la section précédente.

Les Définitions 23, 28 et 29 et les Proposition 13 et théorèmes 6, 8 et 9 impliquent l'inclusion $S^{Res}_{\mathbb{R}^+}(\Sigma) \subset S^{[[Res]]}_{\mathbb{R}^+}(\Sigma) \subset S^{rat}_{\mathbb{R}^+}(\Sigma)$. Le théorème suivant établit l'inclusion stricte au travers d'exemples que l'on peut retrouver dans [Esposito, 2004].

Théorème 10 (Hiérarchie des classes de langages [Denis et Esposito, 2008])**.**

$$S^{Res}_{\mathbb{R}^+}(\Sigma) \subsetneq S^{[[Res]]}_{\mathbb{R}^+}(\Sigma) \subsetneq S^{rat}_{\mathbb{R}^+}(\Sigma)$$

1.6.2 Compacité

Après avoir établi l'inclusion stricte entre les ensembles de langages stochastiques, on peut se demander si une représentation plus riche permet de décrire un langage stochastique de façon plus compacte. C'est le cas, comme le montre les résultats suivants.

Proposition 20 (Compacité non bornée par des $\mathbb{R}$-MA [Esposito, 2004])**.**
Pour tout entier $k \geq 3$, il existe un $\mathbb{R}$-MA à trois états réalisant un langage stochastique qui peut être réalisé par au choix un PNFA, PRFA, PDFA minimal à k états.

Ainsi, la compacité des PNFA, PRFA, PDFA par des $\mathbb{R}$-MA est non bornée.

Proposition 21 (Compacité par des PRFA [Esposito, 2004])**.**
Pour tout entier $n \geq 1$, il existe un PRFA à $2n$ états réalisant un langage stochastique qui peut être réalisé par un PDFA minimal à $\frac{|\Sigma|^n-1}{|\Sigma|-1}+1$ états.

Ainsi, la compacité des PDFA par des PRFA est en $\mathcal{O}(|\Sigma|^n)$.

1.7 Autres systèmes séquentiels linéaires

On appelle systèmes séquentiels linéaires, les $\mathbb{R}$-MA qui réalisent un type de séries formelles bien précis comme les langages stochastiques. Les systèmes séquentiels linéaires ont été formellement introduits par Thon et Jaeger [2015] pour unifier le traitement des langages stochastiques, des processus stochastiques et des processus contrôlés. Dans ce manuscrit, le cadre général utilisé est celui des K-MA plutôt que des systèmes séquentiels linéaires. Comme précisé dans [Thon et Jaeger, 2015], ces deux modèles sont équivalents quand K est un corps commutatif. Or, comme on l'a montré pour les langages stochastiques, il est théoriquement essentiel de considérer la possibilité que K soit un semi-anneau afin d'établir une hiérarchie dans les ensembles de langages stochastiques naturellement déduite de propriétés simples sur les semi-modules résiduels. Cette hiérarchie sur les ensembles se traduit en une hiérarchie sur K-MA réalisant ces ensembles. Des résultats de complexité montre que la classe la plus générale de K-MA n'est pas la plus adaptée à l'apprentissage de langages stochastiques.

D'autres types d'automate tels que les PNFA, les PRFA et les PDFA sont moins expressifs, moins compacts mais se définissent à partir d'un nombre fini de contraintes, ce qui permet leur apprentissage comme on le verra dans les chapitres suivants.

Dans cette section, on montre que le raisonnement sur les différentes classes de langages stochastiques et les automates adaptés à leur apprentissage, peut être calqué pour d'autres systèmes séquentiels linéaires largement utilisés. Cela permettra dans la suite, d'appliquer les algorithmes décrits dans cette thèse pour les langages stochastiques aux autres systèmes séquentiels linéaires définis dans cette section.

1.7.1 Processus stochastiques

1.7.1.1 Définitions

Pour les processus stochastiques, on assimile l'alphabet à l'ensemble des observations.

Définition 30 (Processus stochastique)**.**
Un processus stochastique est une série formelle p tel que, $\forall u \in \Sigma^\star$, $p(u) \in [0,1]$, $p(\varepsilon) = 1$ et $p(u) = \sum_{\sigma\in\Sigma} p(u\sigma)$. On note $SP(\Sigma)$ l'ensemble des processus stochastiques.

De la Définition 30, on en déduit que $\forall l \in \mathbb{N}$, $\sum_{u\in\Sigma^l} p(u) = 1$. Ainsi p définit une distribution sur toutes les séquences d'observations de même longueur.

1.7.1.2 Modèles graphiques

Dans la littérature, il existe de nombreux modèles graphiques pour les processus stochastiques. On rappelle la définition de trois de ces modèles.

Définition 31 (Chaîne de Markov, ou *Markov Chain* (MC))**.**
Une chaîne de Markov, ou Markov Chain *(MC) (homogène et discrète) est une suite de variables aléatoires $\mathbf{X}_0, \mathbf{X}_1, \mathbf{X}_2, \dots$ à valeurs dans un ensemble fini d'états Q qui vérifie la propriété de Markov suivante $\forall n \in \mathbb{N}$, $\forall q_0, \dots, q_{n+1} \in Q$,*

$$\begin{aligned}\mathbb{P}\left(\mathbf{X}_{n+1} = q_{n+1} | \mathbf{X}_n = q_n, \dots, \mathbf{X}_0 = q_0\right) &= \mathbb{P}\left(\mathbf{X}_{n+1} = q_{n+1} | \mathbf{X}_n = q_n\right) \\ &= \mathbb{P}\left(\mathbf{X}_1 = q_{n+1} | \mathbf{X}_0 = q_n\right).\end{aligned}$$

Si on indexe les états alors on peut stocker les probabilités de transitions dans une matrice notée T tel que $T[i,j] = \mathbb{P}\left(\mathbf{X}_1 = j | \mathbf{X}_0 = i\right)$. De plus, on note μ la distribution initiale sur X_0. On peut alors définir une MC par le triplet (Q, T, μ).

Définition 32 (Chaîne de Markov, ou *Markov Chain* (MC) d'ordre k)**.**
Une MC d'ordre $k \geq 1$ est une suite de variables aléatoires $\mathbf{X}_0, \mathbf{X}_1, \mathbf{X}_2, \dots$ à valeurs dans un ensemble fini d'états Q, telle que $\mathbf{Z}_0, \mathbf{Z}_1, \mathbf{Z}_2, \dots$ avec $\mathbf{Z}_i = (\mathbf{X}_i, \dots, \mathbf{X}_{i+k-1})$ définit une MC.

Étant donné que toute MC d'ordre fini, peut s'exprimer comme une MC d'ordre 1, on ne traite que des MC d'ordre 1.

Définition 33 (Chaîne de Markov cachée, ou *Hidden Markov Model* (HMM))**.**
Une HMM (discrète) est une suite $(\mathbf{X}_0, \mathbf{Y}_0), (\mathbf{X}_1\mathbf{Y}_1), (\mathbf{X}_2, \mathbf{Y}_2), \dots$ de couples de variables aléatoires à valeurs dans $Q \times \Sigma$, où Q est l'ensemble d'états cachés et Σ est l'ensemble d'observations. De plus, $\mathbf{X}_0, \mathbf{X}_1, \mathbf{X}_2, \dots$ est une MC de matrice de transition T et de distribution initiale μ. Enfin pour tout $n \in \mathbb{N}$, $\mathbf{Y}_n$ ne dépend que de $\mathbf{X}_n$ tel que $\mathbb{P}\left(\mathbf{Y}_n = o_n | \mathbf{X}_n = q_n\right) = \mathbb{P}\left(\mathbf{Y}_0 = o_n | \mathbf{X}_0 = q_n\right)$. Si on indexe les observations, on peut stocker les probabilités d'observations dans une matrice O tel

que $O[i,j] = \mathbb{P}(\mathbf{Y}_n = j | \mathbf{X}_n = i)$*. On peut alors définir une HMM par le quintuplet* (Q, Σ, T, O, μ)*.*

Définition 34 (Modèle à opérateur observable, ou *Observable Operator Model* (OOM))**.**
Un modèle à opérateur observable, ou Observable Operator Model *(OOM) est un* $\mathbb{R}$*-MA qui réalise un processus stochastique.*

1.7.1.3 Liens avec les automates à multiplicité

Dans cette section, on montre que les MCs, les HMMs et les OOMs sont des K-MA. Ainsi de la même façon que pour les langages stochastiques, on parlera de processus stochastiques rationnels, lorsque ceux-ci sont réalisés par un K-MA et on note $SP_K^{rat}(\Sigma) = SP(\Sigma) \cap K^{rat}\langle\langle\Sigma\rangle\rangle$ leur ensemble.

En suivant le même cheminement que pour les langages stochastiques, on peut définir un type de $\mathbb{R}^+$-MA similaire aux PNFA réalisant des processus stochastiques dont l'acronyme est inspiré de PNFA.

Définition 35 (Automate fini non déterministe réalisant des processus stochastiques, ou *Non-deterministic Finite Automata realizing Stochastic Processes* (SP-NFA))**.**
Un automate fini non déterministe réalisant des processus stochastiques, ou Non-deterministic Finite Automata realizing Stochastic Processes *(SP-NFA) est un* $\mathbb{R}^+$*-MA* $(\Sigma, Q, \iota, \varphi, \tau)$ *tel que* $\sum_{q\in Q} \iota(q) = 1$*,* $\forall q \in Q$*,* $\varphi(q, \Sigma, Q) = 1$ *et* $\forall q \in Q$*,* $\tau(q) = 1$*.*

Sous la forme matricielle de la représentation linéaire équivalente, $(\boldsymbol{\alpha}_0, A, \boldsymbol{\alpha}_\infty)$, ces contraintes s'écrivent,

$$\boldsymbol{\alpha}_0^\top \mathbf{1} = 1, \qquad \sum_{\sigma\in\Sigma} A_\sigma \mathbf{1} = \mathbf{1} \qquad \text{et} \qquad \boldsymbol{\alpha}_\infty = \mathbf{1}. \tag{1.3}$$

Proposition 22.
Soit un SP-NFA $M = (\Sigma, Q, \iota, \varphi, \tau)$*, pour tout état* $q \in Q$*,* $p_{M,q}$ *définit un processus stochastique et donc* p_M *aussi.*

Démonstration. Pour tout état $q \in Q$, on a par définition,

$$p_{M,q}(\varepsilon) = \sum_{q'\in Q} \varphi(q, \varepsilon, q')\tau(q') = \tau(q) = 1.$$

De plus, on a pour tout état $q \in Q$ et tout mot $u \in \Sigma^\star$,

$$\begin{aligned}\sum_{\sigma\in\Sigma} p_{M,q}(u\sigma) &= \sum_{q'\in Q} \varphi(q, u\Sigma, q')\tau(q') \\ &= \sum_{q'\in Q} \varphi(q, u, q')\varphi(q', \Sigma, Q) \\ &= \sum_{q'\in Q} \varphi(q, u, q') \\ &= p_{M,q}(u)\end{aligned}$$

Finalement,

$$p_M(\varepsilon) = \sum_{q\in Q} \iota(q) p_{M,q}(\varepsilon) = 1,$$
$$p_M(u) = \sum_{q\in Q} \iota(q) p_{M,q}(u) = \sum_{q\in Q}\sum_{\sigma\in\Sigma} \iota(q) p_{M,q}(u\sigma) = \sum_{\sigma\in\Sigma} p_M(u\sigma).$$

Notons que la positivité de ι, φ, τ entraine celle de $p_{M,q}$ pour tout $q \in Q$ et celle de p_M. Comme $\forall l \in \mathbb{N}$, $\sum_{u\in\Sigma^l} p_M(u) = 1$ et $\sum_{u\in\Sigma^l} p_{M,q}(u) = 1$, on a bien que $\forall u \in \Sigma^\star$, $p_M(u) \in [0,1]$ et $p_{M,q}(u) \in [0,1]$. □

On définit maintenant deux types d'automates hérités des SP-NFA et correspondant aux PDFA et aux PRFA.

Définition 36 (Automate fini déterministe réalisant des processus stochastiques, ou *Deterministic Finite Automata realizing Stochastic Processes* (SP-DFA))**.**
Un automate fini déterministe réalisant des processus stochastiques, ou Deterministic Finite Automata realizing Stochastic Processes *(SP-DFA) est un SP-NFA dont le support est un DFA.*

Définition 37 (Automate fini résiduel réalisant des processus stochastiques, ou *Residual Finite Automata realizing Stochastic Processes* (SP-RFA))**.**
Un automate fini résiduel réalisant des processus stochastiques, ou Residual Finite Automata realizing Stochastic Processes *(SP-RFA) est un SP-NFA $(\Sigma, Q, \iota, \varphi, \tau)$ tel que pour tout état $q \in Q$, il existe un mot $u \in \Sigma^\star$ tel que $p_{M,q} = u^{-1}p_M$.*

On peut maintenant relier ces modèles à ceux présentés dans la section précédente.

Proposition 23.
Toute MC peut être modélisée par un SP-DFA à $|\Sigma| + 1$ états, où Σ est l'ensemble d'états de la chaine.

Démonstration. Il suffit de construire un $\mathbb{R}^+$-MA sur l'alphabet Σ à partir d'une MC $\mathbf{X}_0, \mathbf{X}_1, \ldots$. On associe à chaque observation $\sigma \in \Sigma$ un état q_σ et on ajoute un état initial q_0 pour former l'espace d'état Q. Puis, naturellement on pose $\forall \sigma, \sigma_1, \sigma_2 \in \Sigma$,

$$\begin{aligned} \iota(q_\sigma) &= 0, \\ \varphi(q_{\sigma_1}, \sigma, q_{\sigma_2}) &= \mathbb{1}_{\sigma_2}(\sigma)\, \mathbb{P}\left(X_1 = \sigma | \mathbf{X}_0 = \sigma_1\right), \\ \tau(q) &= 1, \end{aligned}$$

et

$$\begin{aligned} \iota(q_0) &= 0, \\ \varphi(q_0, \sigma, q_{\sigma_1}) &= \mathbb{1}_{\sigma_1}(\sigma)\, \mathbb{P}\left(X_0 = \sigma\right), \\ \varphi(q_{\sigma_1}, \sigma, q_0) &= 0, \\ \tau(q_0) &= 1. \end{aligned}$$

Par définition, le $\mathbb{R}^+$-MA $M = (\Sigma, Q, \iota, \varphi, \tau)$ a bien comme support un DFA et est équivalent à la MC. De plus, on vérifie facilement que M satisfait les contraintes (1.3). □

Proposition 24.
Tout SP-DFA peut être modélisé par une MC d'ordre $|Q|$ états, où Q est l'ensemble d'états du SP-DFA.

Démonstration. Soit un SP-DFA $M = (\Sigma, Q, \iota, \varphi, \tau)$, montrons que les variables aléatoires $\mathbf{X}_0, \mathbf{X}_1, \ldots$ à valeurs dans Σ définies tel que pour tout $i \in \mathbb{N}$, $\mathbb{P}\left(\mathbf{X}_0 = \sigma_0, \mathbf{X}_1 = \sigma_1, \ldots, \mathbf{X}_i = \sigma_i\right) = p_M(\sigma_0\sigma_1\ldots\sigma_i)$ suivent une MC d'ordre $|Q|$. Pour cela, il suffit de montrer que la distribution d'un $\mathbf{X}_i$ avec $i > |Q|$ ne dépend au plus que de la valeur des $|Q|$ variables précédentes. Soit $u \in \Sigma^{|Q|}$ le mot construit à partir de la réalisation de $\mathbf{X}_{i-|Q|}, \ldots, \mathbf{X}_{i-1}$, alors il existe au moins un état noté

tel que le chemin suivi par u dans le SP-DFA y passe au moins deux fois car M a comme support un DFA et car $|u| = |Q|$. Soit q_u un de ces états, et u_s le suffixe de u associé au reste du chemin après son dernier passage par q_u, alors l'état du SP-DFA à l'instant $i-1$ est $\delta(q_u, u_s)$ et est donc parfaitement déterminé. Ainsi, on a $\mathbb{P}\left(\mathbf{X}_i = \sigma \middle| \mathbf{X}_{i-|Q|} \dots \mathbf{X}_{i-1} = u\right) = \varphi(\delta(q_u, u_s), \sigma, Q)$ et $\mathbf{X}_0, \mathbf{X}_1, \dots$ est bien une MC d'ordre $|Q|$. □

Les liens entre HMMs et SP-NFA ont été étudiés dans [Dupont et collab., 2005], où les deux résultats suivants sont montrés.

Proposition 25 ([Dupont et collab., 2005])**.**
Tout HMM peut être modélisé par un SP-NFA à $|Q|$ états, où Q est l'ensemble d'états cachés.

Proposition 26 ([Dupont et collab., 2005])**.**
Tout SP-NFA défini sur l'alphabet Σ et avec comme espace d'état Q peut être modélisé par un HMM avec au plus $\min(|Q|^2, |Q|\,|\Sigma|)$ *états.*

Par définition, les OOMs sont des $\mathbb{R}$-MA et la classe de langages qu'ils réalisent coïncident avec $SP^{rat}_{\mathbb{R}}(\Sigma)$.

Finalement, toutes les définitions et les résultats des Sections 1.4 et 1.6 peuvent être adaptés pour les processus stochastiques, la seule différence étant l'utilisation des contraintes (1.3) au lieu de (1.1). Ces contraintes ne changent pas fondamentalement les preuves des propositions et des théorèmes pour les langages stochastiques qui peuvent se transposer aux cas des processus stochastiques. Ainsi on donne sans preuves explicites les propositions suivantes où les processus résiduels et les ensembles $SP^{[[Res]]}_{\mathbb{R}^+}(\Sigma)$s et $SP^{Res}_{\mathbb{R}^+}(\Sigma)$ sont définis de la même façon que pour les langages stochastiques.

Proposition 27.
La classe de langages réalisée par les SP-NFA coïncide avec $SP^{rat}_{\mathbb{R}^+}(\Sigma)$

Proposition 28.
La classe de langages réalisée par les SP-RFA coïncide avec $SP^{Res}_{\mathbb{R}^+}(\Sigma)$.

Proposition 29.
La classe de langages réalisée par les SP-DFA coïncide avec $SP^{Res}_{\mathbb{R}^+}(\Sigma)$.

Théorème 11 (Hiérarchie des classes de processus stochastiques)**.**

$$SP^{Res}_{\mathbb{R}^+}(\Sigma) \subsetneq SP^{[[Res]]}_{\mathbb{R}^+}(\Sigma) \subsetneq SP^{rat}_{\mathbb{R}^+}(\Sigma) \subsetneq SP^{rat}_{\mathbb{R}}(\Sigma)$$

On remarque que les résultats de compacité se transposent de même. Cette similarité justifie théoriquement l'utilisation des algorithmes développés dans cette thèse aussi bien pour les processus stochastiques que pour les langages stochastiques et aussi, comme on va le voir dans la section suivante, pour les processus contrôlés.

1.7.2 Processus contrôlés

1.7.2.1 Définitions

Pour les processus contrôlés, on assimile l'alphabet à l'ensemble des paires d'action-observation. On note $\Sigma = \Sigma_A \times \Sigma_O$. Le but est de pouvoir prédire les observations sachant les actions prises.

Définition 38 (Processus contrôlé)**.**
Un processus contrôlé est une série formelle p telle que, $\forall u \in \Sigma^\star$, $p(u) \in [0,1]$,

$p(\varepsilon) = 1$ *et* $\forall a \in \Sigma_A$, $p(u) = \sum_{o \in \Sigma_O} p(u(a,o))$. *On note* $CP(\Sigma)$ *l'ensemble des processus stochastiques.*

Ainsi p définit une distribution conditionnelle par séquence d'actions sur toutes les séquences d'observations de même longueur que la séquence d'actions.

Les problèmes de décisions sont souvent formulés en termes de cumul de récompenses. On suppose qu'à chaque pas, après que l'agent ait pris une action, il observe son environnement et reçoit en même temps une récompense en fonction des objectifs atteints. Le but de l'agent est alors de maximiser le cumul des récompenses dans le temps. Pour obtenir un processus contrôlé à récompenses, il existe deux possibilités. Soit, on définit une fonction de récompense $R : \Sigma_O \to \mathbb{R}$ sur les observations, soit, on inclut les récompenses dans les observations. On choisit la deuxième option, plus générale car elle permet de prendre en compte une dynamique différente pour la génération des observations et des récompenses. On note alors $\Sigma_{\tilde{O}} = \Sigma_O \times \Sigma_R$ où Σ_O est l'ensemble fini et discret des observations, et Σ_R est l'ensemble fini et discret des récompenses. Dès lors, on pose $\Sigma = \Sigma_A \times \Sigma_{\tilde{O}}$. Enfin, pour obtenir la probabilité des récompenses, il suffit de marginaliser les observations.

1.7.2.2 Modèles graphiques

Dans la littérature, il existe de nombreux modèles graphiques pour les processus contrôlés. On rappelle la définition de quatre de ces modèles.

Définition 39 (Processus décisionnel de Markov, ou *Markov Decision Process* (MDP))**.**
Un processus décisionnel de Markov, ou Markov Decision Process *(MDP) est un quadruplet* $(\Sigma_O, \Sigma_A, T, R)$*, où l'ensemble d'observations* Σ_O *forme un ensemble fini et discret d'états,* $T : \Sigma_O \times \Sigma_A \times \Sigma_O \to [0;1]$ *est la fonction de transition du système et* $R : \Sigma_O \times \Sigma_A \times \Sigma_O \to \mathbb{R}$ *est la fonction de récompense. De plus,* $T(o,a,o')$ *donne la probabilité de passer dans l'état* o' *à partir de l'état* o *en effectuant l'action* a*. La fonction de récompense* $R(o,a,o')$ *donne la valeur obtenue si en effectuant l'action* a *dans l'état* o *le système transite en* o'*.*

Définition 40 (Processus décisionnel de Markov partiellement observable, ou *Partially Observable Markov Decision Process* (POMDP))**.**
Un processus décisionnel de Markov partiellement observable, ou Partially Observable Markov Decision Process *(POMDP) est un tuple* $(Q, \Sigma_A, T, R, \Sigma_O, O)$*, où* Q *est un ensemble fini et discret d'états,* (Q, Σ_A, T, R) *définit un MDP et* $O : Q \times \Sigma \to [0;1]$ *est la fonction d'observation qui à un état associe la probabilité d'observer un symbole.*

Définition 41 (Modèle à opérateur observable à entrée-sortie, ou *Input-Output Observable Operator Model* (IO-OOM) [Jaeger, 1998])**.**
Un modèle à opérateur observable à entrée-sortie, ou Input-Output Observable Operator Model *(IO-OOM) est un* $\mathbb{R}$*-MA qui réalise un processus contrôlé.*

Quelques définitions préalables sont nécessaires à la définition du dernier modèle. On rappelle que l'on note $\Sigma = \Sigma_A \times \Sigma_{\tilde{O}}$ l'ensemble des paires d'action-observation, où les observations contiennent les récompenses. On appelle un historique une séquence de paires de Σ dans le passé et un test une séquence de paires de Σ dans le futur. Pour un test t et un historique h, on note $P(t|h)$ la probabilité d'observer les observations de t conditionnée aux actions de t et à l'historique d'actions-observations de h. On note $P(t) = P(t|\varepsilon)$ lorsque l'historique est vide. Un ensemble de tests Q est dit « core » si pour tout historique h, $(P(q|h))_{q \in Q}$ est une statistique suffisante du passé pour prédire le futur. Pour un ensemble de « core » tests Q, on définit, pour tout test $t \in Q$,

les fonctions de projections $f_t : [0,1]^{|Q|} \to [0,1]$ telles que pour tout historique h, $f_t((P(q|h))_{q\in Q}^\top) = P(t|h)$.

Définition 42 (Représentation à état prédictif, ou *Predictive State Representation* (PSR) [Littman et collab., 2001])**.**
Un représentation à état prédictif, ou Predictive State Representation *(PSR) est un tuple* $(Q, \Sigma_A, \Sigma_{\tilde{O}}, F, F_Q, \mathbf{p}_0)$ *où* Q *est un ensemble de « core » tests,* $\mathbf{p}_0$ *est le vecteur des probabilités initiales sur les « core » tests tel que* $\mathbf{p}_0 = (\mathbb{P}(q))_{q\in Q}^\top$ *et* F *et* F_Q *sont des ensembles tels que* $F = \{f_{ao}\}_{a\in\Sigma_A, o\in\Sigma_{\tilde{O}}}$ *et* $F_Q = \{f_{aoq}\}_{a\in\Sigma_A, o\in\Sigma_{\tilde{O}}, q\in Q}$ *contiennent des fonctions de projections. Lorsque le PSR est linéaire, les fonctions de projections sont définies par des vecteurs* $\mathbf{m}_t \in \mathbb{R}^{|Q|}$ *tels que pour tout* $\mathbf{p} \in [0,1]^{|Q|}$, $f_t(\mathbf{p}) = \mathbf{m}_t^\top \mathbf{p}$.

La donnée d'un PSR permet de prédire la réalisation de n'importe quelle trajectoire en définissant récursivement,

$$\mathbf{p}_{hao} = \left(\frac{P(aoq|h)}{P(ao|h)}\right)_{q\in Q} = \left(\frac{f_{aoq}(\mathbf{p}_h)}{f_{ao}(\mathbf{p}_h)}\right)_{q\in Q} = \left(\frac{\mathbf{m}_{aoq}^\top \mathbf{p}_h}{\mathbf{m}_{ao}\mathbf{p}_h}\right)_{q\in Q},$$
$$P(ao|h) = \mathbf{m}_{ao}^\top \mathbf{p}_h.$$

En fait, bien que leur formulation diffère, PSRs et IO-OOMs sont équivalents comme l'a montré Thon et Jaeger [2015].

1.7.2.3 Liens avec les automates à multiplicité

De même que pour les processus stochastiques, on peut définir un type de $\mathbb{R}^+$-MA similaire aux PNFA réalisant des processus stochastiques.

Définition 43 (Automate fini non déterministe réalisant des processus contrôlés, ou *Non-deterministic Finite Automata realizing Control Processes* (CP-NFA))**.**
Un automate fini non déterministe réalisant des processus contrôlés, ou Non-deterministic Finite Automata realizing Control Processes *(CP-NFA) est un* $\mathbb{R}^+$*-MA* $(\Sigma, Q, \iota, \varphi, \tau)$ *tel que* $\Sigma = \Sigma_A \times \Sigma_{\tilde{O}}$, $\sum_{q\in Q} \iota(q) = 1$, $\forall q \in Q$, $\forall a \in \Sigma_A \sum_{o\in\Sigma_{\tilde{O}}} \varphi(q, o, Q) = 1$ *et* $\forall q \in Q$, $\tau(q) = 1$.

Sous la forme matricielle de la représentation linéaire équivalente, $(\boldsymbol{\alpha}_0, A, \boldsymbol{\alpha}_\infty)$, ces contraintes s'écrivent,

$$\boldsymbol{\alpha}_0^\top \mathbf{1} = 1, \qquad \forall a \in \Sigma_A, \sum_{o\in\Sigma_{\tilde{O}}} A_{ao}\mathbf{1} = \mathbf{1} \qquad \text{et} \qquad \boldsymbol{\alpha}_\infty = \mathbf{1}. \tag{1.4}$$

Proposition 30.
Soit un CP-NFA $M = (\Sigma, Q, \iota, \varphi, \tau)$*, pour tout état* $q \in Q$*,* $p_{M,q}$ *définit un processus contrôlé et donc* p_M *aussi.*

Démonstration. Pour tout état $q \in Q$, on a par définition,

$$p_{M,q}(\varepsilon) = \sum_{q'\in Q} \varphi(q, \varepsilon, q')\tau(q') = \tau(q) = 1.$$

De plus, on a pour tout état $q \in Q$, toute action a et toute séquence de paires d'action-

observation $u \in \Sigma^\star$,

$$\begin{aligned}\sum_{o\in\Sigma_{\tilde{O}}} p_{M,q}(u(a,o)) &= \sum_{q'\in Q} \varphi(q, u(a,\Sigma_{\tilde{O}}), q')\tau(q') \\ &= \sum_{q'\in Q} \varphi(q,u,q')\varphi(q',\Sigma,Q) \\ &= \sum_{q'\in Q} \varphi(q,u,q') \\ &= p_{M,q}(u)\end{aligned}$$

Finalement,

$$p_M(\varepsilon) = \sum_{q\in Q} \iota(q) p_{M,q}(\varepsilon) = 1,$$

$$p_M(u) = \sum_{q\in Q} \iota(q) p_{M,q}(u) = \sum_{q\in Q}\sum_{o\in\Sigma_{\tilde{O}}} \iota(q) p_{M,q}(u\sigma) = \sum_{o\in\Sigma_{\tilde{O}}} p_M(u(a,o)).$$

Notons que la positivité de ι, φ, τ entraine celle de $p_{M,q}$ pour tout $q \in Q$ et celle de p_M. Comme $\forall l \in \mathbb{N}$, $\sum_{u\in\Sigma^l} p_M(u) = 1$ et $\sum_{u\in\Sigma^l} p_{M,q}(u) = 1$, on a bien que $\forall u \in \Sigma^\star$, $p_M(u) \in [0,1]$ et $p_{M,q}(u) \in [0,1]$. □

Le cheminement des idées étant le même que pour les processus stochastiques, on termine par la définition des automates suivants.

Définition 44 (Automate fini déterministe réalisant des processus contrôlés, ou *Deterministic Finite Automata realizing Control Processes* (CP-DFA))**.**
Un automate fini déterministe réalisant des processus contrôlés, ou Deterministic Finite Automata realizing Control Processes *(CP-DFA) est un CP-NFA dont le support est un DFA.*

Définition 45 (Automate fini résiduel réalisant des processus contrôlés, ou *Residual Finite Automata realizing Control Processes* (CP-RFA))**.**
Un automate fini résiduel réalisant des processus contrôlés, ou Residual Finite Automata realizing Control Processes *(CP-RFA) est un CP-NFA* $(\Sigma, Q, \iota, \varphi, \tau)$ *tel que pour tout état* $q \in Q$*, il existe un mot* $u \in \Sigma^\star$ *tel que* $p_{M,q} = u^{-1}p_M$*.*

Les résultats sur l'expressivité de ces modèles se déduisent naturellement comme pour les processus stochastiques.

Proposition 31.
Les IO-OOMs sont équivalents aux PSRs et la classe de langages qu'ils réalisent coïncide avec $CP^{rat}_{\mathbb{R}}(\Sigma) = CP(\Sigma) \cap \mathbb{R}^{rat}\langle\langle\Sigma\rangle\rangle$*.*

Proposition 32.
Les CP-NFA sont équivalents aux POMDPs et la classe de langages qu'ils réalisent coïncide avec $CP^{rat}_{\mathbb{R}^+}(\Sigma) = CP(\Sigma) \cap \mathbb{R}^{+rat}\langle\langle\Sigma\rangle\rangle$*.*

Proposition 33.
La classe de langages réalisée par les CP-RFA coïncide avec $CP^{[[Res]]}_{\mathbb{R}^+}(\Sigma)$*.*

Proposition 34.
Les CP-DFA sont équivalents aux MDPs et la classe de langages qu'ils réalisent coïncide avec $CP^{Res}_{\mathbb{R}^+}(\Sigma)$*.*

Théorème 12 (Hiérarchie des classes de processus stochastiques)**.**

$$CP^{Res}_{\mathbb{R}^+}(\Sigma) \subsetneq CP^{[[Res]]}_{\mathbb{R}^+}(\Sigma) \subsetneq CP^{rat}_{\mathbb{R}^+}(\Sigma) \subsetneq CP^{rat}_{\mathbb{R}}(\Sigma)$$

1.8 Diagrammes récapitulatifs

Nous terminons ce chapitre avec deux diagrammes. Le premier, représenté Figure 1.2, montre les ponts établis entre les différents systèmes séquentiels. Dans ce diagramme, on indique par « résiduel », en fonction du contexte, un langage stochastique résiduel, un processus stochastique résiduel ou encore un processus contrôlé résiduel. Les modèles sont regroupées en fonction du type de systèmes séquentiels qu'ils représentent et de leur caractérisation par les résiduels qui détermine à la fois la richesse du modèle et les algorithmes d'apprentissage. Le deuxième diagramme, représenté Figure 1.2, récapitule les principales propositions utilisées dans la thèse pour l'élaboration des algorithmes d'apprentissage. Certaines des propositions présentes sur ce diagramme sont données dans les chapitres concernés. Ce diagramme met en avant pour différentes classes de séries formelles et de langages stochastiques les liens entre les différentes caractérisations. Comme on le verra dans le chapitre suivant, les algorithmes présentés dans ce manuscrit se basent sur la caractérisation par la matrice de Hankel.

	espace vectoriel généré par les résiduels de dimension finie	résiduels contenus dans un semi-module stable de dimension finie	semi-module généré par les résiduels de dimension finie	résiduels en nombre fini
langage stochastique	$\mathbb{R}$-MA	PNFA	PRFA	PDFA
processus stochastique	OOM $\mathbb{R}$-MA	HMM SP-NFA	SP-RFA	MC d'ordre fini SP-DFA
processus contrôlé	IO-OOM PSR $\mathbb{R}$-MA	POMDP CP-NFA	CP-RFA	MDP d'ordre fini CP-DFA

FIGURE 1.2 – Similarité entre les différents types de systèmes séquentiels linéaires.

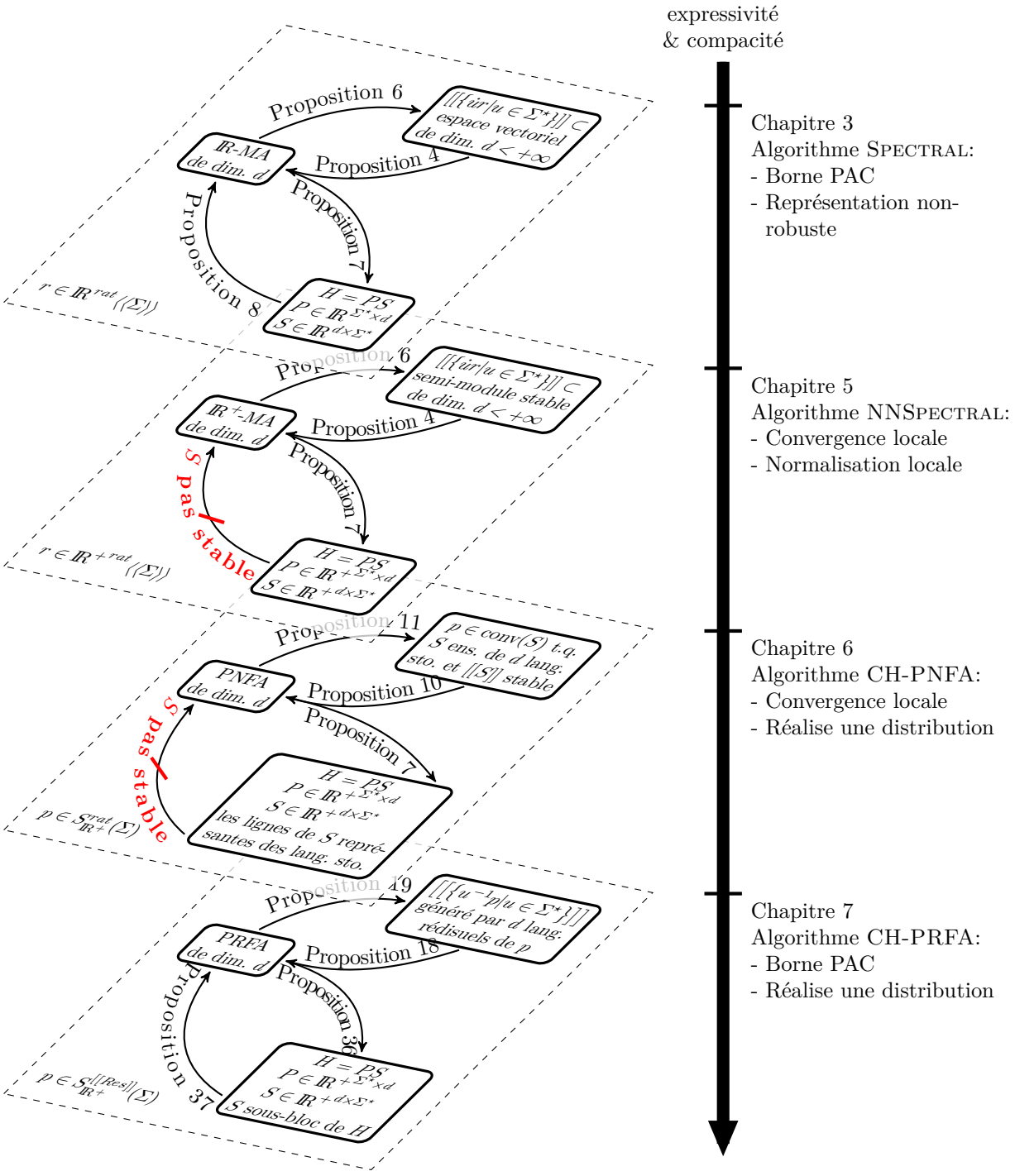


FIGURE 1.3 – Relations entre la matrice de Hankel infinie, les semi-modules et les automates pour différents types de séries formelles et algorithmes les exploitant.

Chapitre 2

Apprentissage d'automates

Sommaire

Ce chapitre est dédié à l'inférence de séries formelles rationnelles et commence par présenter différents modèles d'apprentissage. Parmi eux, le modèle *Probably Approximately Correct* (PAC) est le plus adapté à l'apprentissage. En effet, il définit à la fois des contraintes sur la complexité temporelle et sur le nombre d'échantillons. Cependant, dans sa formulation première, même les familles simples de langages stochastiques ne sont pas apprenables.

Bien que les autres modèles d'apprentissage présentés sont moins adaptés à l'apprentissage, ils définissent des conditions nécessaires à la satisfaction du modèle PAC. Des résultats positifs et négatifs dans ces modèles ont été établis pour l'inférence de certains sous-ensembles de langages stochastiques. Ces résultats permettent de cerner quels types de langages stochastiques sont apprenables dans un modèle PAC relâché, ou pseudo-PAC.

En effet, une fois plusieurs contraintes relâchées, le modèle PAC permet d'établir des résultats positifs pour les langages stochastiques rationnels tout en restant intéressant d'un point de vue pratique. Ainsi, l'obtention d'algorithmes pour l'inférence de familles riches de langages stochastiques satisfaisant le modèle pseudo-PAC est une des motivations principales de cette thèse. En conclusion, on verra qu'autoriser un apprentissage impropre permet d'obtenir des garanties PAC pour les langages stochastiques rationnels. Cependant, un apprentissage impropre peut être problématique pour les applications qui nécessitent de vraies distributions. Par exemple, utiliser en planification une mesure non borné comme noyau de transition peut faire diverger l'algorithme d'itération de la valeur.

En deuxième partie, on propose de revoir les algorithmes d'inférence existants en les classant en trois grandes familles : les méthodes itératives, les méthodes par fusion d'états et les algorithmes issus de la Méthode des Moments ou, *Method of Moments* (MoM). On s'intéresse en particulier à la MoM dont sont issus les algorithmes présentés dans cette thèse.

2.1 Modèles d'apprentissage

Les modèles d'apprentissage consistent en des cadres théoriques que doivent respecter des algorithmes d'inférence. Ils définissent certains bons comportements que doit avoir l'algorithme.. Obtenir des algorithmes satisfaisant ces contraintes, comparé à de bonnes performances empiriques, n'est pas une fin en soi. En effet, on observe que les bornes théoriques sur l'erreur sont souvent trop pessimistes. Néanmoins, ces modèles témoignent des capacités de généralisation des algorithmes. Il existe dans la littérature des modèles d'apprentissage plus ou moins restrictifs. Dans un modèle trop restrictif, les problèmes d'inférence compliqués risquent ne pas avoir de solution algorithmique. Au contraire, un modèle trop laxiste ne se traduira pas en algorithmes efficaces et performants dans les applications pratiques. En fonction de la richesse du type de langages stochastiques, on relâchera certaines contraintes afin d'obtenir des garanties.

2.1.1 Probablement Approximativement Correct

Dans le modèle PAC de Valiant [1984], un algorithme apprend un concept à partir d'un nombre fini d'exemples tirés aléatoirement, et doit retourner, dans la plupart des cas, une hypothèse proche du concept. On note $\mathcal{D}$ l'ensemble des distributions définissant le type de langages stochastiques que l'on souhaite apprendre. Dans notre cas, on étudie des sous-ensembles de langages stochastiques comme $S_K^{rat}(\Sigma)$, $S_{\mathbb{R}^+}^{[[Res]]}(\Sigma)$

ou encore $S^{Res}_{\mathbb{R}^+}(\Sigma)$. On note S_N, un ensemble de N mots tirés aléatoirement selon la distribution $p \in \mathcal{D}$. L'ensemble d'hypothèses, noté $\mathcal{H}$, forme la classe des modèles retournés par l'algorithme A, par exemple les PNFA, les PRFA ou encore les PDFA. La qualité de l'hypothèse $M \in \mathcal{H}$ retournée par A est mesurée par la distance ou divergence, $d(p_M, p)$, entre le langage stochastique p et la série formelle réalisée par M.

Définition 46 (Modèle d'apprentissage PAC pour les langages stochastiques)**.**
Un ensemble de distributions $\mathcal{D}$ est efficacement apprenable par A dans le modèle PAC, si pour tout $\epsilon > 0$ et tout $1 \geq \delta \geq 0$, il existe $N_0 \in \mathbb{N}$, tel que pour tout $N \geq N_0$, l'algorithme A retourne une hypothèse M calculée à partir d'un ensemble D_N d'exemples tirés selon $p \in \mathcal{D}$ vérifiant

$$\mathbb{P}\left(d(p_M, p) \leq \epsilon\right) \geq 1 - \delta,$$

sous la contrainte que A s'exécute un temps $t(\epsilon^{-1}, \delta^{-1}, |\Sigma|, |\mathcal{H}|)$ et que le nombre d'exemples soit supérieur à $N_0 = s(\epsilon^{-1}, \delta^{-1}, |\Sigma|, |\mathcal{H}|)$, où s et t sont des polynômes. Ci-dessus, $|\mathcal{H}|$ indique une mesure de complexité de la classe $\mathcal{H}$, par exemple, le nombre d'états de l'automate.

Historiquement, $\mathcal{D}$ décrit l'ensemble des distributions sur Σ^l et la définition des polynômes t et s est étendue pour dépendre de l. Les travaux plus récents étudient le cas plus difficile où $\mathcal{D}$ décrit l'ensemble des distributions sur $\Sigma^\star$.

Le choix de la distance d est primordiale pour obtenir une analyse pertinente. Par exemple, la distance ℓ_∞ définie par $\ell_\infty(p, q) = \max_{u \in \Sigma^\star} |p(u) - q(u)|$ est trop permissive. Elle permet à un algorithme apprenant par cœur en construisant l'arbre des préfixes de montrer que $S(\Sigma)$ est apprenable dans le modèle PAC [Esposito, 2004]. Lorsque la divergence Kullback-Leibler (KL) [Kullback et Leibler, 1951] est utilisée on qualifiera le modèle par KL-PAC et par extension ℓ_p-PAC pour la distances ℓ_p. La divergence KL apparait comme étant la plus restrictive car elle borne de nombreuses distances sur les distributions, tout en étant elle-même non bornée.

On termine par la définition de quelques propriétés que posséder le modèle PAC.

Définition 47 (Modèle PAC *distribution-free*)**.**
On qualifie le modèle PAC de distribution-free *lorsque l'on ne fait aucune hypothèse sur la distribution ayant généré les exemples servant à l'apprentissage. Dans ce cas, on contrôle l'erreur de l'hypothèse apprise M à la meilleure hypothèse $M_{opt} = \operatorname{argmin}_{M_{opt} \in \mathcal{H}} d(p_{M_{opt}}, p)$ au travers de la relation*

$$\mathbb{P}\left(d(p_M, p) - d(p_{M_{opt}}, p) \leq \epsilon\right) \geq 1 - \delta.$$

On note $\mathcal{P}_\mathcal{H}$ les distributions réalisées par l'ensemble d'hypothèses $\mathcal{H}$.

Définition 48 (Apprentissage propre ou impropre)**.**
Si $\mathcal{P}_\mathcal{H} = \mathcal{D}$, on qualifie l'apprentissage de propre. À l'inverse, si $\mathcal{D} \subsetneq \mathcal{P}_\mathcal{H}$ l'apprentissage est impropre. On dira alors que $\mathcal{D}$ est PAC-apprenable par $\mathcal{H}$.

On remarque que si $\mathcal{P}_\mathcal{H} \subsetneq \mathcal{D}$, alors à part dans le modèle *distribution-free*, aucun algorithme ne peut avoir de garanties PAC.

Le modèle PAC *distribution-free* a été étudié pour l'inférence de PNFA par Abe et Warmuth [1990]. Dans leur travail, ils considèrent l'inférence des paramètres de PNFA quand la structure est donnée. Sous l'hypothèse que **RP** $\neq$ **NP**, Abe et Warmuth [1990] montrent que la complexité temporelle d'un algorithme ne peut pas être polynomiale dans le modèle KL-PAC pour les PNFA à deux états, bien qu'un nombre polynomial d'exemples soit suffisant. En d'autres termes, les PNFA ne peuvent

pas approximer toutes les distributions. Plus tard, Kearns et collab. [1994] ont mis de côté l'aspect *distribution-free* pour définir un modèle pour les PDFA en supposant que $\mathcal{D} = S^{Res}_{\mathbb{R}^+}(\Sigma)$. Même dans ce modèle simplifié, ils prouvent que $S^{Res}_{\mathbb{R}^+}(\Sigma)$ n'est pas KL-PAC-apprenable par réduction à l'inférence de fonction de parité en présence de bruit uniforme. L'identification de fonctions de parité bruitées est reconnue difficile en cryptographie. En effet, seuls des algorithmes super-polynomiaux sont connus pour ce problème. Ce résultat a été étendu au modèle ℓ_1-PAC dans la thèse de Guttman [2006]. De plus, l'auteur montre que la réduction à l'inférence de fonction de parité ne s'applique pas dans les modèles ℓ_p-PAC pour $p \geq 1$. C'est pourquoi, dans la suite, on s'intéresse principalement à la divergence KL et à la distance ℓ_1. Les travaux précédemment cités considèrent uniquement les distributions sur Σ^l mais leurs résultats se généralisent directement aux distributions sur $\Sigma^\star$.

2.1.2 Minimal Adequate Teacher

Le modèle *Minimal Adequate Teacher* (MAT) a été proposé par Angluin [1987] puis adapté aux séries formelles par Beimel et collab. [2000] et Bergadano et Varricchio [1994]. Dans ce modèle, l'algorithme peut faire appel à un oracle au travers de deux types de requêtes. Lors des requêtes d'appartenance, l'oracle retourne la probabilité d'un mot $u \in \Sigma^\star$ choisi par l'algorithme. Lors des requêtes d'équivalence, l'algorithme demande si un K-MA M réalise la série formelle cible notée r. Si l'oracle répond par l'affirmative, alors $\forall u \in \Sigma^\star$, $r_M(u) = r(u)$ et l'algorithme retourne M, sinon l'oracle renvoie un contre-exemple u tel que $r_M(u) \neq r(u)$. Une classe de séries formelles rationnelles $\mathcal{R}$ est MAT-apprenable s'il existe un algorithme de complexité temporelle polynomiale retournant pour tout $r \in \mathcal{R}$, une hypothèse M tel que $r_M = r$.

Beimel et collab. [2000] ont montré que $S^{rat}_K(\Sigma)$ est MAT-apprenables par les K-MA lorsque K est un corps. Leur algorithme, d'une complexité polynomiale en la dimension du semi-module résiduel et en la longueur maximale des exemples, est basé sur l'identification d'une base comme dans l'Algorithme 1. En fait, leur algorithme est même PAC, si l'on considère que les exemples sont les paires $(u, r(u))$ pour u tiré aléatoirement. Pour les langages stochastiques, les processus stochastiques, et les processus contrôlés, ce cadre n'est pas réaliste car on suppose que les exemples sont uniquement composés de mots u tirés selon la distribution cible. Ainsi, les vraies valeurs de $p(u)$ sont inconnues et doivent être estimées à partir des exemples.

2.1.3 Identification à la limite avec probabilité 1

Le modèle d'apprentissage d'identification à la limite avec probabilité 1, correspond au modèle de Gold [1967] adapté au cas stochastique par Angluin [1988]. Dans ce modèle, on suppose qu'il existe une source infinie d'exemples tirés selon la distribution à inférer et l'on s'intéresse au comportement asymptotique de l'algorithme. Celui-ci retourne après chaque nouvel exemple une hypothèse.

Définition 49 (Identification à la limite avec probabilité 1)**.**
Une classe de distribution $\mathcal{D}$ est dite identifiable à la limite avec probabilité 1 *s'il existe un algorithme A qui pour un échantillon de n mots renvoie une hypothèse $M_n \in \mathcal{H}$, tel que pour tout $p \in \mathcal{D}$,*

$$\mathbb{P}\left(\lim_{n\to\infty} p_{M_n} = p\right) = 1.$$

Ainsi, le modèle ne contraint ni le nombre d'exemples, ni le temps d'exécution de l'algorithme. Plus exactement, tout algorithme identifiant à la limite une classe de distribution peut servir à un autre algorithme ayant les même garanties mais un temps d'exécution polynomial en fonction du nombre d'exemples [Esposito, 2004]. En pratique, un tel algorithme aurait peu d'utilité. En fait, dans ce modèle, un algorithme identifiant à la limite, peut retourner des hypothèses arbitrairement mauvaises si la quantité requise d'exemples n'est pas fournie. Ce modèle est donc trop permissif et apparait plutôt comme une condition nécessaire. Pour les PDFA, l'identification de la structure à la limite a été prouvée par Carrasco et Oncina [1999]. Un an après, la preuve a été étendue par de la Higuera et Thollard [2000] pour montrer aussi l'identification à la limite des paramètres des PDFA définie sur $\mathbb{Q}$.

Dans sa thèse, Esposito [2004] étudie des classes plus larges de langages stochastiques rationnels et propose de rajouter plusieurs contraintes au modèle pour le rendre plus réaliste. En particulier, il impose qu'à partir d'un certain rang le support des hypothèses soit constant et que la convergence des paramètres se fasse vers une cible unique. Il qualifie l'identification de *forte* si à partir d'un certain rang le support et les paramètres des hypothèses sont constants. Ainsi, Esposito [2004] montre que $S^{rat}_{\mathbb{Q}}(\Sigma)$ n'est pas identifiable à la limite avec probabilité 1 par un algorithme propre, et par conséquent $S^{rat}_{\mathbb{R}}(\Sigma)$ non plus. Ce résultat négatif, qui est une conséquence des problèmes évoqués Section 1.4.4, annihile tout espoir d'obtenir des algorithmes réalisant un apprentissage propre avec des garanties PAC même relâchées pour la classe la plus générale de langages stochastiques rationnels.

2.2 Algorithmes d'apprentissage

2.2.1 Méthodes itératives

Les méthodes itératives sont les plus utilisées, surtout en traitement de la parole, mais bénéficient de peu de garanties. Parmi ces méthodes, deux algorithmes [Ephraim et Merhav, 2002] sont particulièrement connus pour l'estimation des paramètres de PNFA : l'algorithme Baum-Welch (BW) (qui est une instance particulière de l'algorithme Espérance-Maximisation, ou *Expectation-Maximization* (EM) [Dempster et collab., 1977]) et l'algorithme de Baum-Viterbi (BV). L'algorithme de BV remplace l'étape E de l'algorithme EM, où sont calculées l'espérance conditionnelle des variables latentes, par une maximisation de la probabilité *a posteriori* à l'aide de l'algorithme de Viterbi. Alors que BW cherche à maximiser la vraisemblance, BV maximise le maximum *a posteriori* des variables latentes. Ces algorithmes et leurs variantes convergent seulement vers des optimaux locaux dans le cas général. Pour les PNFA non-ambigus (incluant les PDFA), BW et BV convergent vers le même optimum global qui est unique [Vidal et collab., 2005].

Les algorithmes BW et BV et leurs variantes sont des méthodes fréquentistes. À l'opposé, des méthodes Bayésiennes ont aussi été appliquées à l'estimation des paramètres de PNFA. Dans ce cas, l'apprentissage de modèles se traduit par l'inférence d'hyper-paramètres. Parmi ces méthodes, on trouve celles basées sur les méthodes de Monte-Carlo par chaînes de Markov, ou *Monte Carlo Markov Chains* (MCMCs) (ces méthodes sont comparés à EM dans [Rydén et collab., 2008])telles que l'échantillonnage de Gibbs mais aussi les méthodes Variationnelles de Bayes [MacKay, 1997]. Shibata et Yoshinaka [2014] ont développé de nombreuses variantes très performantes de ces méthodes en se basant sur les travaux de Teh [2006]. Une implémentation

de ces algorithmes est fournie dans la bibliothèque TREBA. De plus, [Shibata et Yoshinaka, 2014] fournit une comparaison empirique de nombreuses autres méthodes pour l'inférence de langages stochastiques.

Un dernier type de méthodes itératives Bayésiennes se propose d'inférer la structure et les paramètres. Il s'agit des méthodes Bayésiennes non-paramétriques, basées sur les processus hiérarchiques de Dirichlet et leur généralisation, les processus hiérarchiques de Pitman-Yor. D'abord proposés pour l'inférence de N-grammes [MacKay et Peto, 1995; Teh, 2006], les modèles ont été étendus à l'inférence de PDFA « infinis » [Pfau et collab., 2010] et de HMMs [Beal et collab., 2001; Blunsom et Cohn, 2011]. Ces modèles ont ensuite été étendus aux processus contrôlés par Doshi-Velez et collab. [2015] qui fournit une comparaison détaillée de ceux-ci sur la plupart des problèmes jouets rencontrés en apprentissage par renforcement dans les domaines partiellement observables. L'inférence de ces modèles est le plus souvent réalisée par des algorithmes de MCMC comme Metropolis-Hasting et l'échantillonnage de Gibbs.

Notons que pour les méthodes Bayesiennes, même si certaines d'entre elles convergent vers le maximum de vraisemblance, il est difficile d'obtenir des garanties en échantillon fini sans aucune hypothèse sur la véracité du *prior*. Si celui-ci est mauvais, le nombre nécessaire d'exemples pour obtenir un *a posterio* correct peut être arbitrairement grand. L'approche PAC-Bayésienne tente d'unir les avantages des deux mondes : fréquentiste et Bayésien. Elle permet de développer des algorithmes qui partant d'un *prior* mettent à jour la distribution *a posteriori* selon un compromis optimal entre la mise à jour Bayésienne du *posterior* et la distribution empirique. Les inégalités de transport sont au cœur de cette approche.

Il existe d'autres algorithmes itératifs inférant à la fois la structure et les paramètres. Ces algorithmes fréquentistes sont les prémisses de la MoM présentée Section 2.2.3 et ont été développées pour les PSRs. L'identification de la structure est généralement basée sur les mêmes idées que les méthodes par fusion d'états, présentées dans la section suivante. Une telle méthode est proposée pour les PSRs par McCracken et Bowling [2005]. Leur algorithme construit de façon incrémentale l'ensemble des core tests $\mathbb{Q}$ en cherchant les processus résiduels linéairement indépendants de ceux précédemment identifiés. Leur critère est basé sur le conditionnement de la matrice construite par les processus résiduels. L'identification des paramètres est basée sur une descente de gradient.

Les deux prochaines sections détaillent des alternatives fréquentistes décrivant des algorithmes avec des garanties de convergence à la limite ou dans le modèle PAC.

2.2.2 Méthodes par fusion d'états

Les méthodes à fusion d'états proviennent des travaux sur l'inférence de langages réguliers, comme les DFA [Oncina et García, 1992]. Pour les PDFA, ces méthodes procèdent en partant généralement d'un automate sous forme d'arbre réalisant la distribution empirique. Puis afin d'éviter le sur-apprentissage, les états sont successivement fusionnés selon certains critères. Généralement, des tests statistiques sont conduits pour décider si deux états possèdent des distributions empiriques semblables. Dans le cas favorable, ces états sont fusionnés. Cette méthode a ensuite inspiré d'autres algorithmes pour des classes de langages stochastiques plus riches.

2.2.2.1 Algorithmes pour les PDFA

Section 2.1.1, on a présenté le modèle PAC ainsi que plusieurs résultats négatifs pour l'apprentissage de langages stochastiques. Cette section commence par exposer des algorithmes permettant d'obtenir des garanties dans un modèle PAC moins restrictif. Ce modèle autorise les algorithmes à avoir une complexité polynomiale dépendant d'un paramètre supplémentaire mesurant la complexité de la distribution cible $p \in \mathcal{D}$. On note ce paramètre $|p|$. Ainsi, s et t sont des polynômes sur les variables $\{\epsilon^{-1}, \delta^{-1}, |\Sigma|, |\mathcal{H}|, |p|\}$. Une telle notion de complexité est apparue la première fois dans [Ron et collab., 1995].

Définition 50 (PDFA μ_∞-distinguables)**.**
Un PDFA M est μ_∞-distinguable si et seulement si pour toute paire d'états $q, q' \in Q$, il existe un mot $u \in \Sigma^\star$, tel que

$$|p_{M,q}(u) - p_{M,q'}(u)| \geq \mu_\infty.$$

Autrement dit,

$$\mu_\infty = \min_{q,q' \in Q} \ell_\infty(p_{M,q}, p_{M,q'}).$$

Ron et collab. [1995] montrent que la classe des PDFA acycliques est KL-PAC-apprenable, où $|p| = \mu_\infty^{-1}$ mesure la complexité de $\mathcal{D}$. Bien que leur analyse soit conduite en prenant des distributions sur $\Sigma^\star$, la longueur des mots est bornée par la profondeur du PDFA acyclique. Ainsi, leurs résultats se restreignent à Σ^l. Une décennie plus tard, Clark et Thollard [2004] ont généralisé ces travaux aux PDFA réalisant des distributions sur $\Sigma^\star$. En plus de μ_∞^{-1}, les polynômes t et s incluent une borne supérieure, notée L, sur la longueur espérée des mots générés par n'importe quel état. Parallèlement, Gavaldà et collab. [2006] montrent que $S^{Res}_{\mathbb{R}^+}(\Sigma)$ est ℓ_∞-PAC-apprenable avec une complexité dépendante de L. Ce résultat est moins puissant que celui de [Palmer et Goldberg, 2005], qui en étudiant l'erreur ℓ_1, proposent une complexité indépendante de L. En effet, comme Clark et Thollard [2004] le suggèrent et comme l'ont montré Palmer et Goldberg [2005], la dépendance de la borne en L est due à l'utilisation de la divergence KL.

D'autres mesures de la complexité de $\mathcal{D}$, ont été définies par Guttman et collab. [2005]. Dans leur travail, ils définissent les PDFA μ_p-distinguables en utilisant la distance ℓ_p à la place de ℓ_∞. Cela leur permet de prouver que leur algorithme est KL-PAC avec $|p| = \mu_2$. Dans sa thèse, Guttman [2006] définit de plus les PDFA ρ_p-distinguables lui permettant de généraliser les travaux de Clark et Thollard [2004]; Palmer et Goldberg [2005]. Dans [Castro et Gavaldà, 2008], les auteurs prouvent que $S^{Res}_{\mathbb{R}^+}(\Sigma)$ est KL-PAC-apprenable en utilisant comme mesure de la complexité μ'_∞, le maximum entre μ_∞ et $\min q, q' \in Q, u \in \Sigma^\star |p_{M,q}(u\Sigma^\star) - p_{M,q'}(u\Sigma^\star)|$. Par définition du maximum, μ'_∞ qui utilise les distributions sur les préfixes borne μ_∞ et est donc plus restrictive. Balle et collab. [2013] montrent que pour certains PDFA, μ_∞ est exponentiellement plus petit que μ'_∞. De plus, les bornes proposées par Castro et Gavaldà [2008] sont plus fines que les précédentes.

Alors que ces précédents travaux s'intéressent à prouver des bornes supérieures, Balle et collab. [2013] prouvent une borne inférieure pour toute une classe d'algorithmes fonctionnant en fusionnant les états qui sont statistiquement μ_∞-distinguables. Cette classe d'algorithme inclut en particulier celui de Clark et Thollard [2004] et ses variantes. Par réduction à l'inférence des fonctions de parité, Balle et collab. [2013] montrent que dans les pires des cas, μ_∞ et μ'_∞ coïncident et doivent forcément

apparaitre dans la borne inférieure sur l'erreur. Dans leur travail, ils proposent un algorithme pour les PDFA μ'_∞-distinguables, ainsi qu'une borne supérieure plus fine que les précédentes. Finalement, les auteurs proposent une généralisation de leur analyse aux PDFA μ_∞-distinguables.

Un traitement systématique du problème de la fusion d'états sous la forme de test statistique est proposé dans [Balle, 2013]. Cela permet à l'auteur de proposer de nouveaux algorithmes, utilisant des techniques d'amorçage ou *bootstrap* [Balle et collab., 2012a], qui bénéficient d'une meilleure efficacité statistique. Ce traitement systématique permet d'utiliser des tests sur des alphabets généralisés incluant une composante structurée ou continue. De plus, l'auteur s'intéresse à l'apprentissage à partir de flux de données.

Les résultats positifs précédents [Clark et Thollard, 2004; Guttman, 2006; Palmer et Goldberg, 2005; Ron et collab., 1995], passent par la construction d'algorithmes d'apprentissage. Pour obtenir des garanties PAC, ces algorithmes utilisent des heuristiques basées sur des inégalités de concentration telles que l'inégalité de Hoeffding [McDiarmid, 1989] pour la fusion d'états. Cependant, bien que disposant de garanties PAC, certains de ces algorithmes ne sont pas utilisables en pratique. Parfois, parce qu'ils utilisent trop de paramètres inconnus ou reposent sur des bornes trop lâches et donc nécessitent un trop grand nombre d'exemples avant de pouvoir retourner une hypothèse correcte. Dans la suite, on présente d'autres algorithmes, dépourvus de garanties PAC, basés sur des heuristiques différentes. Certains ont de bonnes performances empiriques et bénéficient parfois de garanties plus faibles telles que l'identification à la limite.

Pour l'apprentissage de PDFA, l'algorithme ALERGIA a été proposé par Carrasco et Oncina [1994]. Plus tard, Carrasco et Oncina [1999] ont développé l'algorithme RLIPS, une simplification de ALERGIA, qui identifie la structure du PDFA à la limite avec probabilité 1.

L'algorithme MDI a été proposé par de la Higuera et Thollard [2000] pour identifier aussi les probabilités. C'est une variante de ALERGIA qui utilise une autre heuristique pour la fusion d'états permettant d'adapter la taille du modèle. Leur algorithme identifie fortement $S^{Res}_{\mathbb{Q}^+}(\Sigma)$ à la limite avec probabilité 1.

2.2.2.2 Extension aux autres familles de langages stochastiques

Les méthodes à fusion d'états ont aussi inspiré l'algorithme DEES, du nom de ses auteurs Denis et collab. [2006], qui a été décliné pour $S^{rat}_K(\Sigma)$ ($K \in \{\mathbb{R}, \mathbb{Q}\}$) et $S^{[[Res]]}_K(\Sigma)$ ($K \in \{\mathbb{R}^+, \mathbb{Q}^+\}$). L'algorithme DEES apparait comme la dernière étape avant la MoM. En effet, il étend le principe de la fusion d'états aux langages stochastiques. Ainsi, DEES construit par itération la structure de l'automate. Au contraire, la MoM effectue l'identification de la structure de façon globale, en une seule fois. Néanmoins, ces algorithmes reposent sur les mêmes propriétés mathématiques. Détaillons le fonctionnement général de cet algorithme par similarité aux algorithmes pour les PDFA.

Les PDFA possèdent un nombre fini de langages stochastiques résiduels dont l'ensemble est noté R. Soit $p \in S^{Res}_{\mathbb{R}^+}(\Sigma)$, le semi-module résiduel $[[R]]$ est un sous semi-module généré par un ensemble fini, stable et contenant p. D'après la Proposition 4, R est donc suffisant pour retrouver la représentation linéaire de p. Ainsi, le problème se résume à l'identification de R. Bien que R soit fini, les résidus de p ne sont pas en nombre fini (sauf si le PDFA est acyclique [Ron et collab., 1995]). Ainsi, les algorithmes d'inférence de PDFA construisent d'abord un automate arborescent correspondant à

la distribution empirique, puis recherchent les états (associés aux résidus) qui semblent générer les mêmes langages stochastiques résiduels pour les fusionner. Le but est donc de garder le nombre minimal de résidus dont les langages stochastiques résiduels forment R. Ainsi, l'identification de la structure pour les PDFA s'apparente à du *clustering*. Une fois la structure identifiée, il ne reste plus qu'à estimer les probabilités [de la Higuera et Thollard, 2000].

L'algorithme DEES construit également un automate arborescent mais l'étape de fusion est différente. Pour trouver l'ensemble fini R tel que $[[R]] = [[\mathrm{Res}(p)]]$, l'algorithme DEES construit comme pour les PDFA un ensemble de résiduels de façon incrémentale. Cependant, au lieu de tester statistiquement si un nouveau résiduel est associé à un état distinct des états précédemment identifiés, DEES utilise d'autres tests. Pour $p \in S^{rat}_{\mathbb{R}}(\Sigma)$, DEES teste si le nouvel état appartient au sous-espace vectoriel généré par les autres états. Pour $p \in S^{[[Res]]}_{\mathbb{R}^+}(\Sigma)$, DEES teste si le nouvel état appartient à l'enveloppe convexe générée par les autres états. Une fois la structure identifiée, les paramètres de la représentation linéaire sont estimés.

Au niveau des garanties, dans le cas des K-MA stochastiques [Denis et collab., 2006], DEES identifie $S^{rat}_{K}(\Sigma)$ ($K \in \{\mathbb{R}, \mathbb{Q}\}$) à la limite avec probabilité 1. Pour $K = \mathbb{Q}$, l'identification est forte (constante à partir d'un certain nombre d'exemples). Malheureusement, l'apprentissage est impropre comme le prévoit la théorie (voir Section 2.1.3). Pour $K = \mathbb{R}$, l'identification à la limite de la structure est forte mais pas celle des paramètres. Or la classe des K-MA n'est pas robuste et l'automate M retourné par DEES ne réalise pas forcément un langage stochastique. Ainsi, quel que soit le nombre d'exemples, rien ne garantit d'obtenir une distribution. Néanmoins [Habrard et collab., 2006], la série retournée p_M est pseudo-stochastique à partir d'un certain nombre d'exemples (non explicité) et peut alors être convertie en langage stochastique. Bien que celui-ci puisse ne pas être rationnel, il converge à la limite vers le langage stochastique cible. Habrard et collab. [2006] montrent expérimentalement que leur algorithme est efficace et performant par rapport à ALERGIA et MDI.

Dans le cas de $S^{rat}_{\mathbb{R}^+}(\Sigma)$, Denis et Esposito [2004b] proposent un algorithme garantissant l'identification à la limite avec probabilité 1. De plus, l'identification est forte si les paramètres sont rationnels. Cependant, cet algorithme n'est ni efficace, ni utilisable en pratique. En effet, dans le cas des PNFA, on ne possède pas de procédure efficace pour trouver un ensemble fini générant un sous semi-module résiduel stable contenant le langage. L'algorithme proposé essaie alors de trouver directement les paramètres de la représentation linéaire. Cette procédure est inefficace mais permet de prouver l'identification à la limite avec probabilité 1.

Denis et Esposito [2004b] préfèrent alors s'intéresser à $S^{[[Res]]}_{\mathbb{R}^+}(\Sigma)$. Ils montrent d'abord que cette classe de langages stochastiques peut être réalisée par des PRFA particuliers, qualifiés de préfixes.

Définition 51 (PRFA préfixes [Esposito, 2004])**.**
Un PRFA préfixes est un PRA dont l'ensemble d'états Q est un ensemble U de mots tel que $U = \{u \in U | \exists v \in \Sigma^\star, uv \in U\}$ et dont les fonctions initiale, finale et de transition vérifient :

(i) $\iota(\varepsilon) = 1$,

(ii) $\forall u, v \in Q, \forall \sigma \in \Sigma, \varphi(u, \sigma, v) \neq 0 \implies v = u\sigma \vee u\sigma \notin Q$.

Contrairement aux PRFA, les PRFA préfixes ont de particulier que leur forme réduite est unique. De plus, la partie estimation des paramètres de l'algorithme DEES peut être conduite efficacement. Le problème d'identification à la limite est alors bien

posé et peut être résolu par l'algorithme DEES. Ainsi, $S_{\mathbb{R}^+}^{[[Res]]}(\Sigma)$ est identifiable à la limite avec probabilité 1. De plus l'identification est forte si les paramètres sont rationnels. Enfin, l'apprentissage est propre contrairement à DEES pour les K-MA stochastiques et se fait en temps polynomial contrairement à DEES pour les PNFA. Malheureusement, la dimension d'un PRFA préfixe peut être super polynomiale par rapport à celle du PRFA miminal équivalent, comme le suggère la Proposition 17.

Citons, un dernier travail réalisé indépendamment dans le contexte des PSRs se situant entre les méthodes par fusion d'états et la MoM. Wiewiora [2005] propose un algorithme identifiant la structure de façon itérative et calculant les paramètres du PSR par régression. Son algorithme converge vers l'optimum dans la limite d'une infinité d'échantillons avec probabilité 1.

2.2.3 Méthode des moments

Alors que les algorithmes par fusion d'états présentés précédemment sont gloutons et locaux pendant l'identification incrémentale de la structure, les algorithmes issus de la MoM réalisent une identification globale et non-incrémentale de la structure. Dans cette section, on présente de nombreux algorithmes issus de la MoM. La MoM est aussi appelée méthode spectrale, à cause de l'algorithme Spectral. Proposé pour la première fois dans [Rosencrantz et collab., 2004] pour apprendre des PSRs, cet algorithme tire son nom de l'utilisation de la Décomposition en Valeurs Singulières, ou *Singular Value Decomposition* (SVD) pour l'identification de la structure de $\mathbb{R}$-MA. À cause des nombreuses variantes ayant vu le jour autour de cet algorithme, on dénomme par la MoM tout algorithme identifiant la structure de façon globale à partir des premiers moments de la série, en particulier de la matrice de Hankel.

2.2.3.1 Décomposition de faible rang

Le Théorème 4 montre qu'une matrice de Hankel caractérise complètement une série formelle rationnelle sur un corps. Celle-ci permet alors de retrouver une représentation linéaire associée, même quand on utilise une base finie de préfixes et de suffixes. C'est le cadre envisagé par l'approche de Beimel et collab. [2000] pour apprendre des $\mathbb{R}$-MA dans le modèle MAT. Leur algorithme est basé sur l'Algorithme 1. Dans le modèle MAT l'indépendance peut être testée exactement. Supposons que les valeurs de la série soient parfaitement connues alors la représentation linéaire peut être retrouvée grâce à n'importe quelle décomposition en matrices de faible rang de la matrice de Hankel infinie, comme le prévoit la Proposition 8. Dans le cas des matrices de Hankel finies, le résultat de la Proposition 9 est analogue.

Dans les problèmes d'inférence, la première étape est d'estimer la matrice de Hankel. Pour des séries formelles représentant des langages stochastiques, les probabilités peuvent être simplement estimées par comptage. De l'estimateur $\hat{H}$ de H on peut apprendre une représentation linéaire de dimension $\text{rang}(\hat{H})$ réalisant la série $\hat{r}$ telle que $\hat{r}(u,v) = \hat{H}[u,v]$. Cependant, cette procédure est très sensible au sur-apprentissage. En effet, il est fort probable que le modèle ayant généré les données possède une représentation linéaire minimale de dimension bien inférieure à $\text{rang}(\hat{H})$. En effet, $\hat{H}$ est de rang plein avec probabilité 1. Ainsi, pour éviter le sur-apprentissage, les méthodes spectrales contraignent la taille de la représentation linéaire et donc la richesse du modèle appris. Pour ce faire, elles cherchent une approximation $\tilde{H}$ de faible rang de $\hat{H}$ à partir de laquelle une représentation linéaire de faible dimension peut être calculée. Malheureusement, en général, l'approximation de faible rang n'aura pas une structure

de Hankel. Donc, les équations linéaires permettant de retrouver la représentation matricielle ne seront pas exactes. Si bien qu'en général, la série rationnelle $\tilde{r}$ calculée à partir d'une factorisation de $\tilde{H}$ ne vérifiera pas $\tilde{r}(u,v) = \tilde{H}[u,v]$. En considérant le sous-espace vectoriel décrit par une décomposition de $\tilde{H}$, on peut décrire plus précisément la série apprise par l'algorithme. Deux cas de figure se présentent. Soit on utilise toutes les séquences pour estimer une matrice de Hankel infinie $\hat{H}$ mais contenant un nombre fini de valeurs non nulles. Soit on choisit une base complète, et on calcule les estimateurs $\hat{H}_{\mathcal{B}}, \hat{H}^{\sigma}_{\mathcal{B}}, \hat{\mathbf{h}}_{\mathcal{S}}, \hat{\mathbf{h}}_{\mathcal{P}}$ des matrices $H_{\mathcal{B}}, H^{\sigma}_{\mathcal{B}}, \mathbf{h}_{\mathcal{S}}, \mathbf{h}_{\mathcal{P}}$. Détaillons le premier cas.

Un estimateur $\hat{H}$ de H possède généralement un rang bien plus élevé que H. Dès lors plutôt que de chercher une décomposition exacte de H en matrices de faible rang P et S, on cherchera la meilleure approximation possible de $\hat{H}$ par $\hat{P}\hat{S}$ pour un rang donné d. Les lignes de $\hat{S}$ forment alors un sous-espace vectoriel de $\mathbb{R}^{\mathcal{S}}$ que l'on note $\left[\left[\hat{S}^{\top}\right]\right]$. Ce sous-espace est en quelque sorte proche de $\left[\left[S^{\top}\right]\right]$. On a vu Chapitre 1 que $\left[\left[S^{\top}\right]\right]$ est stable car $\mathbb{R}$ est un corps. De plus, $\left[\left[S^{\top}\right]\right]$ contient, par définition, $H^{\top}\mathbf{1}_{\varepsilon}$ la représentation vectorielle de la série formelle cible r. Ainsi par la Proposition 8, $\left[\left[S^{\top}\right]\right]$ caractérise la série formelle décrite par H. Pour l'inférence, $\left[\left[\hat{S}^{\top}\right]\right]$ contient bien $\tilde{r}$ définie par $\tilde{r}(u,v) = \tilde{H}[u,v]$ où $\tilde{H} = \hat{P}\hat{S}$. Malheureusement, $\left[\left[\hat{S}^{\top}\right]\right]$ n'est pas stable car il ne contient pas en général $\hat{S}T^{\top}_{\sigma}$. Une condition nécessaire et suffisante est que $\tilde{H}$ possède une structure de Hankel. On pourrait alors écrire $\hat{S}T^{\top}_{\sigma} = \hat{P}^{\dagger}\hat{P}\hat{S}T^{\top}_{\sigma} = \hat{P}^{\dagger}\tilde{H}T^{\top}_{\sigma} = \hat{P}^{\dagger}T_{\sigma}\tilde{H} = \hat{P}^{\dagger}T_{\sigma}\hat{P}\hat{S}$. Cependant, comme $\tilde{H} \approx \hat{H}$, on suppose que $\tilde{H}$ possède une structure de Hankel et on estime $\hat{A}_{\sigma} = \hat{S}T^{\top}_{\sigma} \approx A_{\sigma}\hat{S}$. Cette procédure permet d'obtenir une représentation linéaire $(\hat{\boldsymbol{\alpha}}_0, \hat{A}, \hat{\boldsymbol{\alpha}}_{\infty})$ telle que

$$\begin{aligned}\hat{A}_{\sigma} &= \hat{S}T^{\top}_{\sigma}\hat{S}^{\dagger},\\ \hat{\boldsymbol{\alpha}}_0^{\top} &= \mathbf{1}_{\varepsilon}^{\top}\hat{P},\\ \hat{\boldsymbol{\alpha}}_{\infty} &= \hat{S}\mathbf{1}_{\varepsilon}.\end{aligned}$$

En général $(\hat{\boldsymbol{\alpha}}_0, \hat{A}, \hat{\boldsymbol{\alpha}}_{\infty})$ ne réalisera pas la série r, $\hat{r}$ ou encore $\tilde{r}$ décrite respectivement par H, $\hat{H}$, ou $\tilde{H}$. En pratique, on constatera que la matrice de Hankel reconstruite à partir de $(\hat{\boldsymbol{\alpha}}_0, \hat{A}, \hat{\boldsymbol{\alpha}}_{\infty})$ donne une meilleure approximation de H que $\hat{H}$ ou $\tilde{H}$. On remarque que si l'on pouvait garantir que $\tilde{H}$ ait une structure de Hankel et que ses coefficients somment à 1, alors la série retournée par l'algorithme serait $\tilde{r}$ qui est bien un langage stochastique rationnel. L'algorithme serait ainsi propre. Ceci ne contredit pas les résultats de complexité du Chapitre 1 car algorithmiquement trouver $\tilde{H}$ est impossible. En effet, cette matrice possède potentiellement un nombre infini de valeurs non nulles, même si ce n'est pas le cas de $\hat{H}$.

Lorsque l'on travaille sur une base finie complète, les résultats sont légèrement différents. On commence, de même, par chercher la meilleure approximation possible de $\hat{H}_{\mathcal{B}}$ par $\tilde{H}_{\mathcal{B}} = \hat{P}\hat{S}$ pour un rang donné d. Comme pour la matrice infinie, on a que $\left[\left[S^{\top}\right]\right]$ contient les lignes de $P^{\dagger}H^{\sigma}_{\mathcal{B}}$ et $\mathbf{h}_{\mathcal{S}}$ la représentation vectorielle de la série formelle cible r restreinte à la base $\mathcal{S}$. Comme pour la matrice de Hankel infinie, $\left[\left[\hat{S}^{\top}\right]\right]$ ne contient pas forcément $\hat{\mathbf{h}}_{\mathcal{S}}$, ni les lignes de $\hat{P}^{\dagger}\hat{H}^{\sigma}_{\mathcal{B}}$.

Notons $\{r_i\}$ l'ensemble des colonnes de $\hat{S}^{\top}$. Dans le cas de la matrice infinie, les $\{\dot{\sigma}r_i\}$ peuvent être extraites de $\hat{S}$ grâce à l'opérateur de décalage $T^{\top}_{\sigma}$. Dans le cas des matrices finies, ce n'est pas possible. Dès lors, on propose d'estimer les $\{\dot{\sigma}r_i\}$ en utilisant $\hat{H}^{\sigma}_{\mathcal{B}}$. Si la représentation linéaire étaient connue, on pourrait identifier les $\{\dot{\sigma}r_i\}$ aux lignes de $\hat{A}_{\sigma}\hat{S}$. Or, on a que $\hat{H}^{\sigma}_{\mathcal{B}} \approx \hat{P}\hat{A}_{\sigma}\hat{S}$. On définit alors les $\{\dot{\sigma}r_i\}$ par régression

linéaire au sens des moindres carrés entre $\hat{H}^{\sigma}_{\mathcal{B}}$ et $\hat{P}$. Ainsi les $\{\dot{\sigma}r_i\}$ sont estimées par les lignes de $\hat{P}^{\dagger}\hat{H}^{\sigma}_{\mathcal{B}}$. De même, si la base $\mathcal{S}$ ne contient pas ε, il faut estimer les $\{r_i(\varepsilon)\}$. On procède de nouveau par régression linéaire entre $\hat{\mathbf{h}}_{\mathcal{P}}$ et $\hat{\boldsymbol{\alpha}}_{\infty}\hat{P}$.

On passe maintenant à l'estimation de la représentation linéaire. Contrairement au cas de la matrice infinie, il n'y a aucune raison pour que les $\{r_i\}$ et les $\{\dot{\sigma}r_i\}$ appartiennent au même sous-espace vectoriel $\left[\left[\hat{S}^{\top}\right]\right]$. On estime alors les paramètres de la représentation linéaire par régression linéaire. Pour les poids de transition, on obtient alors $\hat{P}^{\dagger}\hat{H}^{\sigma}_{\mathcal{B}}\hat{S}^{\dagger}$. Pour les poids initiaux, $\left[\left[\hat{S}^{\top}\right]\right]$ ne contient pas forcément $\hat{\mathbf{h}}_{\mathcal{S}}$. On procède de nouveau par régression linéaire. Les poids finaux sont données par $\{r_i(\varepsilon)\}$. Les quatre étapes qui constituent la procédure d'écrite sont résumées ci-dessous.

1. Choisir une base $\mathcal{B} = (\mathcal{P}, \mathcal{S})$ finie supposée complète.
2. Estimer les matrices de Hankel $(\hat{H}_{\mathcal{B}}, \hat{H}^{\sigma}_{\mathcal{B}}, \hat{\mathbf{h}}_{\mathcal{S}}, \hat{\mathbf{h}}_{\mathcal{P}})$.
3. Factoriser $\hat{H}_{\mathcal{B}} \approx \hat{P}\hat{S}$ avec $\hat{P} \in \mathbb{R}^{\mathcal{S}\times d}$ et $\hat{P} \in \mathbb{R}^{d\times\mathcal{P}}$
4. Calculer la représentation linéaire $(\hat{\boldsymbol{\alpha}}_0, \hat{A}, \hat{\boldsymbol{\alpha}}_{\infty})$ de dimension d par régression linéaire

$$\begin{aligned}\hat{A}_{\sigma} &= \hat{P}^{\dagger}\hat{H}^{\sigma}_{\mathcal{B}}\hat{S}^{\dagger},\\ \hat{\boldsymbol{\alpha}}_0^{\top} &= \hat{\mathbf{h}}_{\mathcal{S}}^{\top}\hat{S}^{\dagger},\\ \hat{\boldsymbol{\alpha}}_{\infty} &= \hat{P}^{\dagger}\hat{\mathbf{h}}_{\mathcal{P}}.\end{aligned}$$

Quelle est alors la différence avec l'algorithme travaillant avec une matrice de Hankel infinie ? L'estimation des $\{r_i\}$ et des $\{\dot{\sigma}r_i\}$ dans le cas des matrices de Hankel finies ne contraint pas celles-ci à appartenir au même sous-espace vectoriel. Les séries formelles estimées par les deux méthodes ne sont pas les mêmes, bien qu'elles convergent vers la même série r dans la limite d'une infinité d'échantillons.

L'algorithme Spectral est un cas particulier de la seconde méthode où la factorisation de $\hat{H}_{\mathcal{B}}$ est calculée par SVD. L'approximation de faible rang trouvée est donc la meilleure au sens de la norme ℓ_2. La norme ℓ_2 correspond particulièrement bien au bruit d'échantillonnage. Les échantillons étant tirés indépendamment, la distribution de la moyenne tend vers la loi Normale. Or minimiser la norme ℓ_2 correspond à chercher le maximum de vraisemblance en supposant la normalité du bruit. Rosencrantz et collab. [2004] proposent en premier une version préliminaire de l'algorithme Spectral pour les PSRs en utilisant la Matrice de la Dynamique du Système, ou *System Dynamic Matrix* (SDM) plutôt que la matrice de Hankel. Puis, Hsu et collab. [2009] proposent l'algorithme Spectral pour les HMMs de rang plein ainsi que des analyses en échantillons finis permettant d'établir que les HMMs sont PAC-apprenables. Malheureusement, l'algorithme Spectral est impropre. En même temps, Bailly et collab. [2009] proposent un algorithme basé sur l'analyse en composantes principales ainsi qu'une procédure pour estimer le rang. Leur algorithme n'est pas analysé. Ensuite, les travaux de Hsu et collab. [2009] sont adaptés à d'autres modèles. Boots et collab. [2010a] adaptent l'algorithme Spectral pour les PSRs. Pour apprendre à partir d'observations continues, les auteurs proposent d'apprendre comme série l'espérance conditionnelle des observations plutôt que leur probabilité. Une version en ligne de l'algorithme est présentée dans Boots et Gordon [2011]. Des expériences sur des données réelles démontrent l'intérêt de la MoM en robotique. Parallèlement, Siddiqi et collab. [2010] montrent que l'algorithme de Hsu et collab. [2009] peut très bien apprendre des HMMs de faible rang avec des garanties similaires. Puis une analyse non-asymptotique de Spectral pour les $\mathbb{R}$-MA réalisant des langages stochastiques

est donnée par Bailly [2011a]. Cette analyse permet en outre de montrer que $S^{rat}_{\mathbb{R}}(\Sigma)$ est PAC-apprenable, bien que l'apprentissage soit impropre. Enfin, le champ d'application de l'algorithme Spectral pour les $\mathbb{R}$-MA est élargi par Balle et collab. [2014a] avec des applications en traitement du langage naturel. De nombreux aspects pratiques sont discutés par les auteurs tels que le choix de la base ou l'apprentissage de séries auxiliaires pour augmenter l'efficacité statistique. Dans [Kulesza et collab., 2015b], les auteurs proposent une heuristique itérative pour le choix de la base en analysant les valeurs propres maximales de matrices A_σ.

D'autres chercheurs proposent des algorithmes basés sur des décompositions différentes. Les travaux sur l'apprentissage de PSRs ont été poursuivis par Hamilton et collab. [2013] mais cette fois en proposant l'utilisation de projections aléatoires pour l'estimation du sous-espace vectoriel résiduel. Des expériences en planification sur des problèmes jouets de grandes dimensions démontrent les capacités de l'expressivité des PSRs et de passage à l'échelle de la MoM. Les auteurs proposent une analyse en échantillons finis se distinguant des précédentes par l'usage de résultats pour la régression linéaire de faible rang utilisant des projections aléatoires. Dans [Jaeger et collab., 2005; Zhao et collab., 2009] les auteurs proposent deux algorithmes Efficiency Sharpening et Error Controlling basés sur des décompositions différentes pour apprendre des OOMs. Ces deux algorithmes ont été simplifiés et généralisés dans [Thon et Jaeger, 2015]. Efficiency Sharpening améliore itérativement $P^\dagger$ pour minimiser la variance de $\hat{A}_\sigma = \hat{P}^\dagger \hat{H}^\sigma_{\mathcal{B}} (\hat{P}^\dagger \hat{H}_{\mathcal{B}})^\dagger$. En supposant que la pseudo-inverse $(\hat{P}^\dagger \hat{H}_{\mathcal{B}})^\dagger$ est principalement responsable de l'erreur dans les paramètres, Efficiency Sharpening utilise une représentation linéaire déjà calculée à partir d'une première factorisation $\hat{P}_t \hat{S}_t$ afin d'estimer $\hat{P}^\dagger_{t+1}$ minimisant la variance de $\hat{P}^\dagger \hat{H}_{\mathcal{B}}$. Une nouvelle représentation linéaire est alors calculée à partir de $\hat{P}_{t+1}$. L'algorithme itère jusqu'à convergence. L'algorithme Error Controlling propose plutôt de minimiser le conditionnement du système linéaire dans la régression. En fait, Thon et Jaeger [2015] montre que la procédure incrémentale de Error Controlling est équivalente à un algorithme EM pour l'analyse en composantes principales et converge vers la SVD utilisée dans l'algorithme Spectral.

Nous terminons cette partie en citant deux travaux [Boots et collab., 2013; Song et collab., 2010] mariant représentation des distributions dans un espace de Hilbert à noyau reproduisant et l'algorithme Spectral pour l'apprentissage de HMMs et PSRs. Récemment, Hefny et collab. [2015] proposent un algorithme d'inférence pour des systèmes non linéaires. Leur présentation englobe l'algorithme Spectral et [Boots et collab., 2013; Song et collab., 2010].

2.2.3.2 Optimisation convexe

Les méthodes précédentes possèdent plusieurs défauts. D'abord, la structure de Hankel n'est pas présente dans l'approximation de faible rang. Puis, le rang de la décomposition approximant la matrice de Hankel est choisi *a priori*. Enfin, la SVD donne la meilleure approximation ℓ_2 de faible rang. Trouver une méthode pour d'autres fonctions de coût n'est pas trivial.

Dans cette section, on va montrer que chercher une approximation de faible rang par optimisation convexe permet de prendre en compte la structure, de s'adapter facilement à différentes fonctions d'erreur et aussi d'adapter le rang de la décomposition aux données.

Balle et collab. [2012b] proposent pour la première fois l'utilisation de l'optimisation

convexe pour l'inférence de $\mathbb{R}$-MA. Sa méthode permet de traiter de façon commune le problème d'approximation de faible rang (usuellement traité par décomposition) et le problème de structure de Hankel (usuellement traité par régression pour apprendre la représentation linéaire). Comme l'optimisation est faite directement sur la représentation linéaire, le problème de l'absence de structure dans l'approximation de faible rang de la matrice de Hankel est évité. Pour simplifier, on suppose que ε soit inclus dans la base de préfixes $\mathcal{P}$ et de suffixes $\mathcal{S}$. Notons $H_{\mathcal{B}}^{\Sigma}$ la concaténation verticale des matrices $\{H_{\mathcal{B}}^{\sigma}\}_{\sigma\in\Sigma}$. Soit $X \in \mathbb{R}^{\mathcal{S}\times d}$ une matrice telle que les colonnes de $H_{\mathcal{B}}X$ forment un espace vectoriel de dimension d stable et contenant r la série cible. Soit $(\boldsymbol{\alpha}_0, A, \boldsymbol{\alpha}_\infty)$ la représentation linéaire donnée par Proposition 9 alors $\boldsymbol{\alpha}_0 = \mathbf{h}_{\mathcal{S}}X$, $H_{\mathcal{B}}X\boldsymbol{\alpha}_\infty = \mathbf{h}_{\mathcal{P}}$ et $H_{\mathcal{B}}XA = H_{\mathcal{B}}^{\Sigma}X$. Pour l'inférence, les deux dernières équations ne sont pas généralement vérifiées, on propose donc de les résoudre au sens des moindres carrés. Afin d'éviter le sur-apprentissage, la dimension d est ajoutée à la fonction de coût pour obtenir

$$(\hat{d}, \hat{X}, \hat{A}_\Sigma, \hat{\boldsymbol{\alpha}}_\infty) = \underset{d, X\in\mathbb{R}^{\mathcal{S}\times d}, A, \boldsymbol{\alpha}_\infty}{\operatorname{argmin}} \left\|\hat{H}_{\mathcal{B}}XA - \hat{H}_{\mathcal{B}}^{\Sigma}X\right\|_F + \left\|\hat{H}_{\mathcal{B}}X\boldsymbol{\alpha}_\infty - \mathbf{h}_{\mathcal{P}}\right\|_2 + \lambda d,$$

où λ est un paramètre de régularisation. Ce problème est fortement non-convexe. À la place, Balle et collab. [2012b] proposent de minimiser une enveloppe convexe de ce coût en posant $X = I$ et en faisant intervenir la norme nucléaire, notée $\|\cdot\|_*$. Pour simplifier, seule une minimisation sur A est conduite,

$$\hat{A}_\Sigma = \underset{A}{\operatorname{argmin}} \left\|\hat{H}_{\mathcal{B}}A - \hat{H}_{\mathcal{B}}^{\Sigma}\right\|_F + \tau\|A\|_*,$$

où τ est un paramètre de régularisation. La représentation linéaire est alors donnée par $(\hat{\mathbf{h}}_{\mathcal{S}}, \hat{A}, \mathbf{e})$. La solution trouvée $\hat{A}_\Sigma$ sera en général de faible rang. Bien que cette méthode évite le sur-apprentissage grâce à la régularisation en norme nucléaire, la représentation linéaire trouvée est de grande taille ($\mathcal{S}\times\mathcal{S}$). L'algorithme est aussi plus lent que Spectral.

L'optimisation convexe a aussi été proposée par Balle et Mohri [2012] et Quattoni et collab. [2014] dans un contexte agnostique. Les auteurs supposent que les N exemples fournis sont des paires $\{(u_i, x_i)\}_{i=1..N}$ appartenant à $\Sigma^\star \times \mathbb{R}$ générés selon une distribution inconnue p. Le but est de trouver un $\mathbb{R}$-MA M minimisant $\mathbb{E}_{u,x\sim p}\left[l(r_M(u), x)\right]$ où l est une fonction de coût (généralement $l(y,x) = (x-y)^2$). Afin d'apprendre un $\mathbb{R}$-MA, l'optimisation est formulée sur l'ensemble des matrices de Hankel $\mathbb{H}_{\mathcal{B}}$ correspondant à la base $\mathcal{B}$. Ainsi $\mathbb{H}_{\mathcal{B}}$ décrit l'ensemble des matrices H tel que pour tout $w \in \Sigma^\star$, pour tout $(u,v) \in \mathcal{B}$, les $H[u,v]$ sont égaux. On note cette valeur $H[w]$. Pour éviter le sur-apprentissage, un terme de régularisation sur la matrice de Hankel est ajouté à la fonction de coût. Balle et Mohri [2012] proposent d'utiliser la norme q de Schatten notée ici $\|\cdot\|_q^{Schatten}$ (remarquons que $\|\cdot\|_1^{Schatten} = \|\cdot\|_*$). Le problème d'optimisation est alors

$$\tilde{H} = \min_H \frac{1}{N}\sum_{i=1}^{N} l(H[u_i], x_i) + \tau\|H\|_q^{Schatten},$$

où τ est le paramètre de régularisation. Balle et Mohri [2012] proposent d'utiliser $\tilde{H}$ pour calculer une représentation linéaire de faible dimension par l'algorithme Spectral. Bien que $\tilde{H}$ respecte la structure de Hankel, $\tilde{H}$ n'a aucune garantie de posséder une factorisation de faible rang. En pratique, à cause de la projection sur l'espace $\mathbb{H}_{\mathcal{B}}$ interne aux algorithmes d'optimisation, la solution sera souvent de rang

plein. Balle et Mohri [2012] proposent une analyse non-asymptotique de l'algorithme complet pour $q = 2$ et $l(y, x) = |y, x|$ mais pas de simulation. L'analyse est étendue au cas $q = 1$ (la norme nucléaire) dans [Balle et Mohri, 2015a] en utilisant de nouveaux résultats présentés dans [Balle et Mohri, 2015b]. Quattoni et collab. [2014] proposent un algorithme similaire pour apprendre $\tilde{H}$ en utilisant une approximation du coût *hinge* dans le cas des OOM à entrées-sorties. Deux types régularisations sont expérimentés, la norme nucléaire et la norme de Frobenius. Les auteurs n'utilisent pas ensuite l'algorithme Spectral mais se servent directement des valeurs contenues dans $\tilde{H}$ pour calculer des alignements de tri-grammes. Une application en traitement du langage naturel est proposée ayant pour but de trouver la séquence de phonèmes associée à un mot.

Enfin, on cite un dernier travail n'utilisant pas l'optimisation convexe mais néanmoins similaire. Gybels et collab. [2014] s'attaquent directement au problème d'approximation d'une matrice de Hankel par une matrice de Hankel de rang d sous la norme de Frobenius. Les auteurs montrent que l'optimisation est équivalente à une double minimisation en cascade. Bien que la minimisation interne soit convexe, le problème complet ne l'est pas. Il ne peut donc être résolu que pour de petites matrices de Hankel. Quelques expériences prouvent l'intérêt de cette approche. Notamment, les auteurs proposent d'appliquer cette méthode sur des sous-blocs de la matrice de Hankel afin de la débruiter puis d'utiliser l'algorithme Spectral. Cette procédure améliore légèrement les résultats par rapport à Spectral.

2.2.3.3 Décomposition en matrices non-négatives

Dans la section précédente, l'algorithme Spectral et ses variantes pour l'apprentissage de $\mathbb{R}$-MA se basent sur le Théorème 4 pour retrouver une série formelle rationnelle sur un corps à partir de sa matrice de Hankel. Ils identifient d'abord une base du sous-espace vectoriel contenant les $\{\dot{u}r | u \in \Sigma^\star\}$ par factorisation de faible rang de la matrice de Hankel. Puis, une version matricielle de la Proposition 4 est utilisée pour retrouver la représentation linéaire associée. Malheureusement, pour les $\mathbb{R}^+$-MA, le Théorème 4 n'est pas valable sur les semi-anneaux. En effet, envisageons une méthode reposant sur l'identification d'une famille finie de séries formelles de $\mathbb{R}^{+\Sigma^\star}$. Pour pouvoir appliquer la Proposition 4, cette famille doit générer un sous semi-module contenant les $\{\dot{u}r | u \in \Sigma^\star\}$ et stable. Dans le cas des $\mathbb{R}$-MA, on a montré que toute factorisation de H (ou $H_{\mathcal{B}}$ dans le cas fini) définissait un sous semi-module stable. Ce n'est pas le cas sur $\mathbb{R}^+$. Ainsi pour les $\mathbb{R}^+$-MA, toute factorisation $H = PS$ sur $\mathbb{R}^+$ ne permet pas forcément de retrouver un $\mathbb{R}^+$-MA. Une série r peut très bien avoir une matrice de Hankel H de rang non-négatif fini et ne pas être rationnelle sur $\mathbb{R}^+$.

Pour les langages stochastiques rationnels sur $\mathbb{R}^+$, la caractérisation par la matrice de Hankel doit être complétée. La finitude du rang n'est pas suffisante. En effet, pour qu'une série soit rationnelle sur $\mathbb{R}^+$, il faut que H possède une factorisation de rang fini PS telle que les lignes de S définissent un semi-modules stable. Formuler autrement, une condition nécessaire est que $\forall \sigma \in \Sigma$, $P^\dagger H^\sigma_{\mathcal{B}} S^\dagger \in \mathbb{R}^{+n \times n}$. Trouver une telle factorisation est compliqué. En effet, le problème NMF, ne résolvant que partiellement le problème, est déjà **NP**-dur [Vavasis, 2009]. Au Chapitre 5, nous proposons un algorithme qui se contentera de chercher des factorisations « simple » à l'aide d'heuristique et qui est donc dépourvu de garanties de convergence globale. Expérimentalement, cet algorithme produit néanmoins de très bons résultats. Enfin, pour les PNFA, il faut aussi rajouter les contraintes linéaires définies Équation (1.1).

Cette approche sera explorée au Chapitre 6. Les travaux qui suivent traitent en particulier de l'inférence de HMMs à partir de la matrice de Hankel estimée $\hat{H}_{\mathcal{B}} \in \mathbb{R}^{\Sigma\times\Sigma}$. Comme pour les algorithmes des Chapitres 5 et 6, les travaux qui suivent ne fournissent pas de garanties de convergence globale.

Finesso et collab. [2010] proposent un algorithme qui, comme SPECTRAL, commence par chercher une approximation de faible rang de $\hat{H}_{\mathcal{B}} \approx \hat{P}\hat{S}$ sous la contrainte que $\hat{P}, \hat{S} \geq 0$, $\mathbf{1}^\top \hat{P}\mathbf{1} = 1$ et $\hat{S}\mathbf{1} = \mathbf{1}$. La qualité de l'approximation est mesurée par la divergence KL entre $\hat{H}_{\mathcal{B}}$ et $\hat{P}\hat{S}$. L'algorithme de NMF utilisé est décrit dans [Lee et Seung, 1999]. Procédant à une mise à jour alternée de $\hat{P}$ et $\hat{S}$, cette heuristique ne converge que vers un minimum local. Deux régressions linéaires en divergence KL sous contraintes linéaires permettent de retrouver la représentation linéaire d'un HMM.

Cybenko et Crespi [2011] généralisent l'approche précédente en estimant les probabilités jointes de séquences d'observations plutôt que de paires. Ils proposent d'abord une estimation du rang par SVD. Puis, une décomposition de faible rang minimisant la divergence KL est trouvée par NMF. L'étape de régression est légèrement différente et est résolue par programmation linéaire pour minimiser un coût ℓ_1. À la différence de [Finesso et collab., 2010], la représentation linéaire trouvée est utilisée pour calculer une nouvelle décomposition de faible rang qui sert d'initialisation à une nouvelle itération de la NMF. Ces itérations sont répétées jusqu'à l'obtention d'un modèle satisfaisant.

Une autre possibilité [Vanluyten et collab., 2008] est de décomposer la matrice de Hankel H en trois matrices non-négatives $H = OSO^\top$. On note Y_t (resp. X_t) la variable aléatoire représentant l'observation (resp. l'état caché) au temps t du HMM cible. On montre alors que l'on peut écrire $H[\sigma, \sigma'] = \mathbb{P}(Y_t = \sigma, Y_{t+1} = \sigma')$, $O[\sigma, q] = \mathbb{P}(Y_t = \sigma | X_t = \sigma')$ et $S[q, q'] = \mathbb{P}(X_t = q, X_{t+1} = q')$. On peut ensuite retrouver la matrice de transition à partir de S et de la distribution stationnaire. Pour trouver une telle décomposition sous la contrainte que $\mathbf{1}^\top S\mathbf{1} = \mathbf{1}$ et $\mathbf{1}^\top O = \mathbf{1}^\top$, deux coûts sont envisagés : la divergence KL [Vanluyten et collab., 2008] et la distance ℓ_2 [Lakshminarayanan et Raich, 2010]. Vanluyten et collab. [2008] proposent un algorithme itératif inspiré de [Lee et Seung, 1999] mettant à jour alternativement P et S. Lakshminarayanan et Raich [2010] retiennent la distance ℓ_2 pour proposer un algorithme inspiré des moindres carrés alternés. Dans les deux cas, la convergence est seulement locale car le problème est non convexe. Une extension utilisant des noyaux est proposée dans [Lakshminarayanan et Raich, 2010] pour apprendre à partir d'observations continues. La décomposition en trois matrices de Lakshminarayanan et Raich [2010] a été ensuite exploitée par Tewari et Giering [2014] et Yang et Oja [2012]. Tewari et Giering [2014] proposent de trouver la décomposition par *Probabilistic Latent Semantic Analysis* qui est une instantiation particulière de l'algorithme EM. Quant à Yang et Oja [2012], ils proposent de minimiser la norme de Frobenius entre H et sa décomposition. Leur algorithme a été ensuite amélioré par Zhang et collab. [2014].

2.2.3.4 Modèles à opérateurs jointement factorisés

Dans cette section, on détaille les travaux s'adressant à un certain type d'automate dont les matrices de transitions se factorisent simultanément.

Définition 52 (Automates à multiplicité factorisés, ou *Factorized Multiplicity Automaton* (FMA))**.**
Un FMA est un $\mathbb{R}$*-MA ayant une représentation linéaire* $(\boldsymbol{\alpha}_0, A, \boldsymbol{\alpha}_\infty)$ *de dimension* n *telle qu'il existe une matrice* $T \in \mathbb{R}^{n\times n}$*, et, pour tout* $\sigma \in \Sigma$*, une matrice diagonale*

$O_\sigma \in \mathbb{R}^{n\times n}$ *vérifiant* $\forall \sigma \in \Sigma$, $A_\sigma = O_\sigma T$.

Les HMMs sont un cas particulier de FMA où T est la matrice de transition et O tel que $O = (\mathrm{diag}(O_\sigma))^\top_{\sigma\in\Sigma}$ est la matrice d'observation.

L'idée commune aux algorithmes présentés dans cette section est la possibilité pour les FMA de retrouver explicitement les matrices O_σ puis les matrices T, $A_\sigma = O_\sigma T$, $\boldsymbol{\alpha}_0$ et $\boldsymbol{\alpha}_\infty$. Ces méthodes supposent l'identifiabilité du modèle. Elles font les hypothèses que $|Q| \leq |\Sigma|$ et que O et T soit de rang plein. À partir de la matrice de Hankel, ces algorithmes calculent un ensemble de matrices $\{C_\sigma\}_{\sigma\in\Sigma}$ simultanément diagonalisables. En utilisant les valeurs propres des $\{C_\sigma\}_{\sigma\in\Sigma}$, la matrice O peut être retrouvée, puis en utilisant la base commune la matrice T peut être identifiée. Selon les algorithmes, afin d'obtenir des garanties, différentes conditions sont utilisées pour s'assurer que le problème de diagonalisation simultanée est bien posé.

Le premier travail s'adressant aux modèles factorisés, ou plus exactement aux HMMs, est celui de Mossel et Roch [2005]. Leur approche pour obtenir une base unique est de diagonaliser une matrice dont les valeurs propres sont bien séparées. Une technique utilisée par la plupart des algorithmes est d'effectuer une combinaison aléatoire des matrices $\{C_\sigma\}_{\sigma\in\Sigma}$. Cela permet de s'assurer que les valeurs propres sont séparées avec forte probabilité. Mossel et Roch [2005] ne traitent que des HMMs ayant un espace d'états et d'observations de même dimension. En ce sens, leur algorithme n'effectue pas l'approximation de faible rang au cœur des approches spectrales. Le travail a cependant inspiré Hsu et collab. [2009] qui proposent une procédure similaire de post-traitement après l'algorithme Spectral pour retrouver les matrices T et O d'un HMM. Une amélioration de cet algorithme, une présentation unifiée pour l'inférence de mélange de modèles, ainsi qu'une analyse en échantillons finis sont proposés dans [Anandkumar et collab., 2012b]. Une autre approche [Song et Chen, 2015], en lien avec le Chapitre 7, pour retrouver une base unique, est de chercher pour chaque état une observation caractéristique de celui-ci. Pour chacune de ces observations, qualifiées d'ancres, le premier vecteur propre de C_σ est extrait. L'ensemble de ces vecteurs propres forme la base recherchée. La méthode repose sur le fait que les valeurs propres associées à ces vecteurs propres sont en général bien séparées.

Les précédents algorithmes utilisent la diagonalisation jointe des $\{C_\sigma\}_{\sigma\in\Sigma}$ uniquement pour retrouver les paramètres O et T une fois le sous-espace vectoriel latent identifié par SVD. Un second type de méthode basé sur la décomposition jointe de Schur, permet d'extraire directement une base de faible dimension. Proposé sans analyse théorique dans [Balle et collab., 2011] pour les transducteurs, le même principe a été appliqué et analysé par Colombo et Vlassis [2015]. Expérimentalement, ces méthodes paraissent plus stables que les précédentes.

Un dernier type de méthode [Anandkumar et collab., 2014] empile les matrices $\{C_\sigma\}_{\sigma\in\Sigma}$ pour former un tenseur qui après transformation devient symétrique et orthogonal. Cette propriété permet de trouver une décomposition unique et des garanties en échantillons finis. Dans [Anandkumar et collab., 2014], les auteurs font un rapprochement entre sa méthode et celle basée sur une diagonalisation jointe, montrant que moins de conditions sont nécessaires pour obtenir l'unicité de la base. L'algorithme de Anandkumar et collab. [2014] pour les HMMs a été ensuite généralisé par Balle et collab. [2014b] pour apprendre des FMA.

Dans le cas d'inférence de HMMs, pour retrouver des probabilités à partir des méthodes précédentes, deux heuristiques ont été proposées dans la littérature. La plus simple [Colombo et Vlassis, 2015; Mossel et Roch, 2005] consiste à annuler les valeurs négatives puis à normaliser. Mossel et Roch [2005] analysent l'erreur introduite par

cette étape et montrent que les garanties pseudo-PAC sont conservées. Dans [Balle et collab., 2014b], les auteurs proposent d'appliquer une projection sur le simplexe. Malheureusement, le résultat de la projection ne produit pas de bons résultats en pratique.

Enfin, la plupart des travaux sur les modèles factorisés s'adressent aussi bien au modèle multi-vues [Anandkumar et collab., 2014]. En effet, dans le modèle multi-vues, plusieurs vues observables dépendent d'une même variable cachée. Un modèle factorisé est un modèle à trois vues : le passé, le présent et le futur. Comme les probabilités conditionnelles du passé, du présent et du futur font intervenir la matrice d'observation, de transition et la probabilité initiale de façon séparée, on peut retrouver chacun de ces paramètres par des opérations d'algèbre linéaire. Bien que le passé et le présent peuvent consister en des séquences d'observations, le présent n'est formé que d'une observation. Ainsi un désavantage majeur de cette méthode est son incapacité à apprendre des modèles dont le nombre d'états est supérieur au nombre d'observations. De plus, la matrice d'observation et de transition doivent être de rang plein.

2.2.3.5 Autres modèles

Face aux problèmes dus à l'apprentissage impropre de l'algorithme Spectral, deux travaux proposent d'apprendre des modèles réalisant uniquement des séries formelles positives.

Zhao et Jaeger [2010] proposent une descente de gradient sur la vraisemblance pour une forme particulière de OOMs, appelée Norm-OOMs. Cet algorithme ne bénéficie malheureusement pas de garanties de convergence globale.

Bailly [2011b] s'intéresse à l'apprentissage d'automates à multiplicité quadratique (QWA). Son algorithme effectue une SVD comme Spectral, mais pour apprendre la racine carrée de la série cible. Celle-ci est ensuite mise au carré de façon à retrouver une série non-négative. Par sa filiation avec Spectral, cet algorithme fournit une méthode d'inférence consistante et des bornes d'erreurs sur l'estimation des paramètres sont prouvées. Malheureusement, l'algorithme de Bailly [2011b] ne garantit pas d'obtenir une distribution, bien que la série apprise soit positive. Notons que QWA et les Norm-OOMs sont moins expressifs que les HMMs.

Enfin, dans [Bailly, 2011b], l'auteur utilise le modèle retourné par son algorithme pour initialiser une descente de gradient sur la vraisemblance. Cette procédure n'est valable que si la série apprise converge. Bien que cette procédure annule les précédentes garanties théoriques, elle semble augmenter nettement les performances du modèle appris.

2.2.3.6 Initialisation d'algorithmes itératifs

L'apprentissage impropre est un problème récurrent pour l'inférence d'automates qui empêche la plupart du temps l'initialisation d'algorithmes itératifs tels que EM par le modèle appris par la MoM. Néanmoins quelques travaux, en plus de ceux présentés dans la section précédente, ont mis au point des heuristiques permettant une telle initialisation.

Gybels et collab. [2014] proposent de partir d'un $\mathbb{R}$-MA appris par l'algorithme Spectral. En supposant que la série réalisée par le $\mathbb{R}$-MA converge, les auteurs proposent de normaliser celui-ci pour qu'il converge vers 1. Ensuite, le $\mathbb{R}$-MA normalisé est transformé en un $\mathbb{R}$-MA préfixe équivalent. Puis, le $\mathbb{R}$-MA préfixe est transformé d'itération en itération en un $\mathbb{R}$-MA préfixe équivalent mais avec plus d'états et donc

la somme des coefficients négatifs est moindre. Lorsque celle-ci est suffisamment petite, la valeur absolue des coefficients est prise et la représentation linéaire est normalisée pour satisfaire les contraintes d'un langage stochastique. Ces contraintes sont données Équation (1.1). La représentation linéaire normalisée du $\mathbb{R}$-MA préfixe est utilisée pour initialiser l'algorithme de BW. Leur procédure, bien que dépourvue de garantie, améliore généralement les résultats de l'algorithme Spectral à conditions que la somme finale des coefficients négatifs soit relativement faible.

Balle et collab. [2014b] expérimentent l'utilisation de la méthode des tenseurs et de la projection sur le simplexe, détaillée Section 2.2.3.4, pour initialiser un algorithme EM. Comme discuté précédemment, les résultats sont mitigés car le modèle projeté est souvent mauvais.

Partant de cette observation, Shaban et collab. [2015] proposent d'utiliser une méthode d'optimisation par points extérieurs pour l'inférence de HMMs. En effet, le problème d'optimisation de la représentation linéaire sous la contrainte que celle-ci définisse un HMM est non-convexe. L'algorithme EM ne converge alors que vers un minimum local accessible depuis le point d'initialisation. Le point d'initialisation ainsi que toutes les représentations linéaires intermédiaires calculées par EM satisfont les contraintes. Il s'agit donc d'une méthode de points intérieurs (à l'ensemble des contraintes). Conscients que l'étape de projection sur le simplexe peut dégrader fortement le modèle et ainsi potentiellement le sortir de la cuvette contenant l'optimum global, Shaban et collab. [2015] proposent une méthode de points extérieurs. Cette méthode ne contraint pas les représentations linéaires intermédiaires à satisfaire les contraintes. Cependant, en ajoutant les contraintes à la fonction de coût, au fil des itérations, la représentation linéaire se rapproche de l'ensemble faisable. Une fois suffisament proche la projection peut alors avoir lieu sans trop dégrader le modèle. Leurs expériences confirment leur intuition.

2.3 Conclusions sur l'état de l'art

Le Chapitre 1 décrit une hiérarchie de modèles d'expressivité et de compacité différentes. Cette-ci est rappelée sur la Figure 2.1. Dans ce chapitre, on a commencé par définir des modèles d'apprentissage dans le but d'analyser l'apprenabilité de ces différentes classes de langages stochastiques.

Ainsi, l'identification à la limite et le modèle MAT offrent peu de garanties pour les applications pratiques mais ils apparaissent comme des conditions nécessaires. À l'opposé, le modèle PAC dans sa formulation première est trop restreint pour permettre l'apprentissage de modèles tels que les PDFA, simples mais plus riches que les traditionnelles chaines de Markov et n-grammes. Pour les PDFA, le modèle PAC doit prendre en compte la complexité de la distribution à apprendre au travers de la distinguabilité des états. Dans ce cas, les méthodes à fusion d'états fournissent des algorithmes PAC apprenant proprement les PDFA.

Les modèles comme les HMMs, OOMs, POMDPs et PSRs, étudiés par la communauté des processus stochastiques et des processus contrôlés, sont plus riches et plus compacts que les PDFA. Traditionnellement, appris par des algorithmes itératifs et dépourvus de garanties, ces modèles ont bénéficié des avancées apportées par la MoM. Introduite comme une extension des méthodes par fusion d'états inférant la structure de façon globale, la MoM permet d'obtenir des garanties non-asymptotiques pour l'apprentissage de $\mathbb{R}$-MA, représentés en noir sur la Figure 2.1. Pour ces modèles plus compliqués, on verra qu'une notion similaire à la distinguabilité, comme la séparation

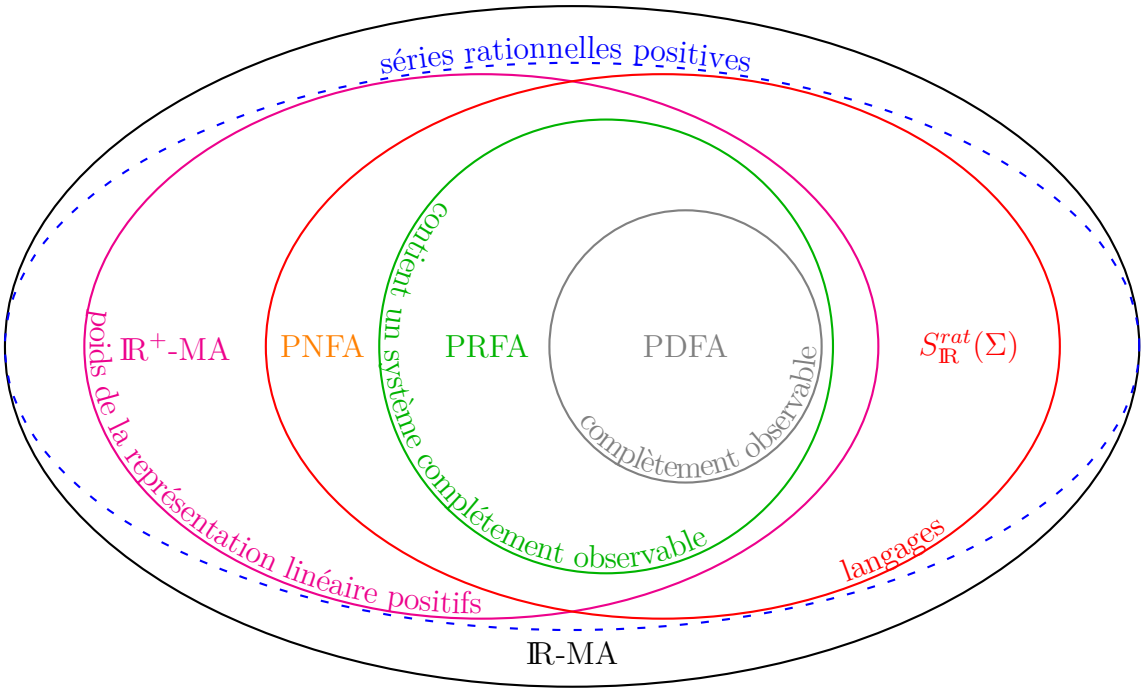


FIGURE 2.1 – Hiérarchie entres les classes d'automates. La classe des automates réalisant des langages stochastiques rationnels sur $\mathbb{R}$, en rouge, n'est pas identifiable à la limite avec probabilité 1. A l'inverse des garanties pseudo-PAC existent pour l'inférence de PDFA. L'inférence de PNFA semble dure dans le cas général. Au milieu, les PRFA semblent être de bons candidats pour obtenir des garanties pseudo-PAC tout en étant partiellement observable.

des valeurs singulières dans la matrice de Hankel, apparait souvent dans les bornes non-asymptotiques sur l'erreur. Malheureusement, les $\mathbb{R}$-MA sont bien plus simples à apprendre que les langages stochastiques rationnels. En effet, relâcher le modèle PAC n'est pas suffisant pour apprendre proprement l'ensemble des langages stochastiques rationnels qui ne sont même pas proprement identifiable à la limite avec probabilité 1. L'ensemble des automates réalisant des langages stochastiques rationnels sur $\mathbb{R}$, est représenté en rouge sur la Figure 2.1.

L'espoir se tourne alors vers les PNFA, en orange sur la Figure 2.1, qui sont proprement identifiables à la limite avec probabilité 1. Malheureusement, cette fois l'obstacle est calculatoire comme prévu par Abe et Warmuth [1990]. En d'autres termes, ce n'est pas le nombre d'exemples qui fait défaut mais le nombre nécessaire d'opérations à l'inférence. Cette problématique est confirmée par les algorithmes de décomposition en matrices non-négatives qui s'appuient tous sur des heuristiques pour résoudre une optimisation **NP**-dur.

Néanmoins, la littérature regorge d'algorithmes impropres PAC pour l'apprentissage de langages stochastiques rationnels sur $\mathbb{R}$ ou $\mathbb{R}^+$ basés sur différentes décompositions de la matrice de Hankel. On a, d'une part, les travaux issus de l'algorithme SPECTRAL cherchant une décomposition en deux matrices de faible rang. D'autre part, on trouve de nombreux algorithmes pour les modèles factorisés, basés sur des décompositions jointes de matrices ou d'un tenseur offrant une certaine symétrie. Dans les deux cas, l'apprentissage est impropre. Or, pour certaines applications, obtenir une vraie distribution est cruciale. De plus, un apprentissage propre permet d'initialiser d'autres algorithmes itératifs ayant de bonnes propriétés de convergence locale tel que EM. De nombreuses heuristiques ont vu le jour afin de transformer les $\mathbb{R}$-MA ou les FMA appris en PNFA ou en HMMs.

L'apprentissage impropre est, en particulier, dû à la perte de la structure de Hankel par les approximations de faible rang. C'est ce problème que l'optimisation convexe tente de résoudre en relâchant le problème de minimisation de rang pour une régularisation en norme nucléaire. Malheureusement, posséder une structure de Hankel et être de faible rang semblent être des objectifs antagonistes. Si bien que les algorithmes basés sur l'optimisation convexe sont propres mais apprennent des modèles de grande taille. Cependant, grâce à la régularisation, ces algorithmes évitent tout de même le sur-apprentissage. Notons que certains auteurs, en tentant de résoudre exactement le problème d'approximation de faible rang en conservant la structure de Hankel, se sont heurtés à des obstacles calculatoires liés à la non-convexité du problème.

Ainsi, les PNFA et autres modèles équivalents semblent trop riches pour être proprement PAC-apprenables. Notre recherche s'oriente alors vers les modèles d'expressivité intermédiaire entre les PDFA, en gris, et les PNFA, en orange sur la Figure 2.1. Dans la littérature, on trouve quelques travaux proposant des modèles, comme les Norm-OOMs et les QWA, qui évitent les probabilités négatives par construction mais ne sont pas entièrement satisfaisants. Les PRFA, en vert sur la Figure 2.1, par contre sont assez prometteurs. En plus d'être assez expressifs, ils sont proprement identifiables à la limite avec probabilité 1 par un algorithme efficace fonctionnant par fusion d'états. Dans le Chapitre 7, nous proposons une méthode globale issue de la MoM pour apprendre des PRFA avec des garanties PAC.

Deuxième partie

Apprentissage de modèles compressés : application en guerre électronique

Chapitre 3

Apprentissage spectral d'automates compressés

Sommaire

3.1 Introduction

Dans ce chapitre, nous nous intéressons en particulier à l'algorithme Spectral, présenté dans le chapitre précédent, pour l'apprentissage de $\mathbb{R}$-MA compressés à partir d'exemples générés par un langage stochastique rationnel. On a vu au chapitre précédent que Spectral prend en argument une variable d indiquant la dimension de la représentation linéaire minimale. Les analyses non-asymptotiques montrent que l'inférence se comporte bien quand $d \geq \text{rang}(H)$. Cependant aucune garantie n'est donnée lorsque $d < \text{rang}(H)$, c'est-à-dire lorsque l'on apprend un modèle compressé. Bien entendu apprendre un modèle compressé introduit un biais dans l'estimation de la représentation linéaire, mais cela peut permettre d'obtenir une meilleure estimation quand peu d'échantillons sont utilisés. En développant une analyse non-asymptotique de l'erreur, nous illustrons le compromis-biais variance pour les $\mathbb{R}$-MA compressés.

Cette analyse permet de mettre en lumière deux problèmes dans l'algorithme Spectral. Premièrement, dans la plupart des applications, $\text{rang}(H)$ est inconnu. Certains auteurs [Bailly, 2011b] proposent des méthodes pour l'estimer à partir de $\hat{H}$. Deuxièmement, utiliser le rang de H ou une estimation de celui-ci est souvent sous-optimal quand peu d'échantillons sont disponibles pour l'inférence. Dans certaines applications la taille du modèle peut être choisie par validation croisée. D'une part cette procédure est coûteuse en temps de calcul. D'autre part, elle est irréalisable dans certains contextes comme l'apprentissage en ligne où il est souhaitable d'adapter la taille du modèle aux données.

En fonction de la quantité de données, l'erreur d'estimation de la matrice de Hankel peut avoir plusieurs effets. L'algorithme Spectral identifie d'abord le sous-espace vectoriel contenant la dynamique du système en cherchant les dimensions associées aux plus grandes valeurs singulières de $\hat{H}$. Pour une série r, nous appelons le sous espace contenant la dynamique du système le sous espace vectoriel (stable par définition) contenant les $\{\dot{u}r | u \in \Sigma^\star\}$. La SVD tronquée a pour effet de supprimer l'erreur d'échantillonnage orthogonale à l'espace identifié et d'éviter le sur-apprentissage. Cependant, lorsque l'erreur $\left\|H - \hat{H}\right\|$ est forte, les valeurs singulières sont aussi perturbées et les dimensions associées aux plus grandes valeurs singulières de $\hat{H}$ ne correspondent plus à celle de H. Autrement dit, le bruit d'échantillonnage peut masquer le sous espace vectoriel contenant la dynamique. Ainsi choisir d trop petit augmente le risque d'ignorer certains états du système. Choisir d trop grand ne permet pas de bien généraliser car cela augmente l'expressivité de la représentation linéaire.

Dès lors, nous faisons face à un compromis. Vaut-il mieux risquer d'ignorer une partie de la dynamique ou d'être plus sensible au sur-apprentissage ? Comme nous allons le voir, la réponse dépend entièrement du modèle à estimer. En effet, toutes les dimensions (chacune représentant un état du système) ne sont pas égales. Les dimensions associées à des faibles valeurs singulières seront plus sujettes à être noyées dans l'erreur d'échantillonnage alors que celles associées aux fortes valeurs singulières seront facilement identifiables. Plus exactement, c'est le fossé entre les valeurs singulières consécutives qui caractérisera la difficulté de l'identification. Ainsi, imaginons une série caractérisée par l'existence d'un grand fossé dans le spectre de H, séparant explicitement l'espace d'état en deux parties. Une partie est facilement distinguable de l'erreur d'échantillonnage, l'autre est noyée dedans. La première possibilité est d'identifier un petit sous espace vectoriel correspondant à la partie facilement distinguable. Une partie de la dynamique sera alors ignorée, ce qui causera une certaine erreur dans la représentation linéaire estimée. La deuxième possibilité est

d'identifier un espace bien plus grand que le rang de H afin de limiter le risque que l'espace identifié soit orthogonal à celui associé aux faibles valeurs singulières. Une grande partie de l'erreur d'échantillonnage ne sera pas non plus éliminée par projection ce qui se répercutera sur la qualité de la représentation linéaire. Donc, en fonction du nombre d'échantillons, il peut être soit préférable d'apprendre un modèle compressé soit d'apprendre un modèle plus grand que la cible.

Enfin, nous remarquons que l'algorithme Spectral et sa version régularisée présentée ici sont capables d'apprendre tous types de séries formelles, en particulier les processus stochastiques et les processus contrôlés. En général, les résultats d'analyses non-asymptotiques se transposent naturellement aux processus stochastiques. La section suivante discute des légères différences dans les bornes obtenues. L'adaptation des résultats aux processus contrôlés dépend de la politique ayant généré les résultats. Nous ne nous étendrons pas plus dans ce chapitre sur ce cas.

3.2 État de l'art

Dans cette partie, M est l'automate retourné par l'algorithme Spectral. De plus, on note $(\hat{\boldsymbol{\alpha}}_0, \hat{A}, \hat{\boldsymbol{\alpha}}_\infty)$ la représentation linéaire de M. De même, on note $(\boldsymbol{\alpha}_0, A, \boldsymbol{\alpha}_\infty)$ une représentation linéaire minimale de la série rationnelle cible r. Lorsque r est un langage stochastique ou processus stochastique, on note la série p. Plusieurs mesures de l'erreur entre la série cible r et la série apprise r_M ont été étudiées. Plusieurs distances ont été proposées pour le contrôle de l'erreur. Comme discuté Chapitre 2, nous nous intéressons en particulier à la distance ℓ_1. Le plus simple est de contrôler l'erreur ponctuelle, c'est-à-dire pour tout mot u, $|r_M(u) - r(u)|$. Selon le type de série que l'on considère, on utilisera des bornes plus restrictives.

Par exemple pour les langages stochastiques, on préférera analyser l'erreur sur la distribution apprise $|p_M - p| = \ell_1(p_M, p) = \sum_{u\in\Sigma^\star} |p_M(u) - p(u)|$ en variation totale. Pour obtenir des bornes sur l'erreur en variation totale, une des difficultés principales [Bailly, 2011a; Balle, 2013] est le contrôle de la queue de distribution. En effet, décomposons

$$\sum_{u\in\Sigma^\star} |p_M(u) - p(u)| = \sum_{u\in\Sigma^{\leq l}} |p_M(u) - p(u)| + \sum_{u\in\Sigma^{>l}} |p_M(u) - p(u)|\,,$$

il existe de nombreuses façons de borner $\sum_{u\in\Sigma^{\leq l}} |p_M(u) - p(u)|$. La difficulté est alors de contrôler la queue de la distribution $\sum_{u\in\Sigma^{>l}} |p_M(u) - p(u)| \leq \sum_{u\in\Sigma^{>l}} |p_M(u)| + \sum_{u\in\Sigma^{>l}} |p(u)|$, ce qui peut se faire en bornant à la fois $\sum_{u\in\Sigma^{>l}} |p_M(u)|$ et $\sum_{u\in\Sigma^{>l}} |p(u)|$. Commençons par $\sum_{u\in\Sigma^{>l}} |p(u)|$. Pour les langages stochastiques rationnels, et plus généralement pour toute série formelle rationnelle convergente r, la suite des $(\left|r(\Sigma^l)\right|)_t$ est exponentiellement décroissante [Bailly et collab., 2009]. Plus précisément, il existe des réels $\rho \in]0,1[$ et $c \geq 0$ tels que pour tout $t \in \mathbb{N}$, $r(\Sigma^l) \leq c\rho^t$, de plus ρ et c sont calculables en temps polynomial. Ainsi $|\sum_{u\in\Sigma^{>l}} r(u)|$ est aussi exponentiellement décroissant. Si r est positive (un langage stochastique est positif par définition), alors $\sum_{u\in\Sigma^{>t}} |r(u)| = |\sum_{u\in\Sigma^{>l}} r(u)|$. Pour le contrôle de $\sum_{u\in\Sigma^{>l}} |p_M(u)|$, les choses ne sont pas si simples. En effet, l'algorithme Spectral ne renvoie pas forcément une série positive, ni convergente, ni absolument convergente. Cependant, Denis et collab. [2006, Théorème 3] ont montré que si $\hat{A}$ est suffisamment proche de A alors M est une série absolument convergente et $\sum_{u\in\Sigma^{>l}} |p_M(u)|$ décroît exponentiellement avec t. Le désavantage de cette approche est qu'elle ne fournit pas la valeur des constantes.

Pour un processus stochastique qui représente une infinité de distributions chacune sur des mots de même taille, on s'intéressera à borner, pour tout t, $\sum_{u\in\Sigma^t} |p_M(u) - p(u)|$. Il semble trop difficile de contrôler la somme sur t de ces erreurs comme le remarque Kulesza et collab. [2015a]. En effet, comme la série formelle décrite par un processus stochastique rationnel ne converge pas, il semble difficile de contrôler l'erreur. Dans [Kulesza et collab., 2015a], les auteurs proposent de pondérer la série par une suite qui décroit suffisamment vite pour la rendre convergente avant de l'analyser.

On remarque que, partant d'une borne ponctuelle $|r_M(u) - r(u)|$, on peut établir une borne sur $\sum_{u\in\Sigma^{\leq t}} |p_M(u) - p(u)|$ en sommant sur tous les mots u de longueur inférieure ou égale à t. Cependant, cela induit un nombre exponentiel en t de termes dans la somme. Or quand la série cible est réalisée par une série formelle rationnelle sur $\mathbb{R}^+$, une analyse plus fine est possible [Balle, 2013; Hsu et collab., 2012]. Cette analyse permet d'éviter le facteur exponentiel en t en supposant un nombre d'échantillons suffisamment grand et ainsi d'obtenir des garanties PAC. Bailly [2011a] propose une astuce similaire pour les séries formelles rationnelles sur $\mathbb{R}$ absolument convergente, mais son analyse ne permet pas l'obtention des coefficients dans la borne. Dans les deux cas, il faut supposer que le nombre d'échantillons est suffisamment grand par rapport à certains paramètres, inconnus, qualifiant la complexité de la série cible, tels que la valeur singulière minimale de $H_{\mathcal{B}}$. Malheureusement, cette astuce ne semble pas s'appliquer aux modèles compressés, comme expliqué en Section 3.4.5. Enfin, le passage à une borne sur la variation totale se fait en utilisant le Denis et collab. [2006, Théorème 3] comme cela est fait pour démontrer le Théorème 8 dans [Bailly, 2011a]. Pour ces raisons, on présentera les résultats sous la forme de bornes sur l'erreur ponctuelle $|r_M(u) - r(u)|$.

3.3 Borne sur l'erreur d'apprentissage des automates compressés

Dans les Chapitres 1 et 2, nous avons redémontré plusieurs résultats connus. Dans la suite de ce manuscrit, les résultats démontrés sont des contributions originales de l'auteur.

Lors de l'inférence de modèles compressés, il y a trois sources d'erreurs. Premièrement, un sous-espace sous-dimensionné n'est pas assez grand pour contenir la dynamique du système. Deuxièmement, comme la matrice Hankel est estimée, le sous-espace peut être identifié de manière incorrecte. Troisièmement, l'erreur d'estimation de la matrice de Hankel se répercute directement dans l'estimation des poids initiaux. Dans la suite, nous dénommons ces sources d'erreur, respectivement, par erreurs de modélisation, erreurs d'identification et erreurs d'estimation. Notre approche est de traiter séparément chaque source d'erreur. Pour cela, en plus de la vraie représentation linéaire $(\boldsymbol{\alpha}_0, A, \boldsymbol{\alpha}_\infty)$ de dimension d et celle apprise $(\hat{\boldsymbol{\alpha}}_0, \hat{A}, \hat{\boldsymbol{\alpha}}_\infty)$ de dimension k, nous en définissons deux autres. Nous notons $(\tilde{\boldsymbol{\alpha}}_0, \tilde{A}, \tilde{\boldsymbol{\alpha}}_\infty)$ la représentation linéaire apprise en utilisant la vraie matrice de Hankel H mais le sous-espace estimé de dimension k avec,

$$\tilde{A}_\sigma = \hat{V}^\top T_\sigma^\top \hat{V}, \qquad \tilde{\boldsymbol{\alpha}}_0^\top = \mathbf{1}_\varepsilon{}^\top H \hat{V}, \qquad \tilde{\boldsymbol{\alpha}}_\infty = \hat{V}^\top \mathbf{1}_\varepsilon.$$

La représentation linéaire inférée à l'aide de H et du vrai sous-espace, tronqué pour

être de dimension k, est notée $(\overline{\boldsymbol{\alpha}}_0, \overline{A}, \overline{\boldsymbol{\alpha}}_\infty)$ avec,

$$\overline{A}_\sigma = \overline{V}^\top T_\sigma^\top \overline{V}, \qquad \overline{\boldsymbol{\alpha}}_0^\top = \mathbf{1}_\varepsilon{}^\top H \overline{V}, \qquad \overline{\boldsymbol{\alpha}}_\infty = \overline{V}^\top \mathbf{1}_\varepsilon,$$

où $\overline{V}$ représente les k premières colonnes de V.

Nous définissons les quatre séries formelles rationnelles associées aux représentations linéaires définies précédemment par,

$$\begin{aligned} \overline{p}(u) &= \overline{\boldsymbol{\alpha}}_0^\top \overline{A}_u \overline{\boldsymbol{\alpha}}_\infty, & p_M(u) &= \hat{\boldsymbol{\alpha}}_0^\top \hat{A}_u \hat{\boldsymbol{\alpha}}_\infty, \\ \tilde{p}(u) &= \tilde{\boldsymbol{\alpha}}_0^\top \tilde{A}_u \tilde{\boldsymbol{\alpha}}_\infty, & p(u) &= \boldsymbol{\alpha}_0^\top A_u \boldsymbol{\alpha}_\infty. \end{aligned}$$

Ainsi, nous pouvons décomposer l'erreur totale en

$$|p_M(u) - p(u)| \leq |p_M(u) - \tilde{p}(u)| + |\tilde{p}(u) - \overline{p}(u)| + |\overline{p}(u) - p(u)|\,.$$

Cette décomposition fait apparaitre respectivement l'erreur d'estimation ($|p_M(u) - \tilde{p}(u)|$), l'erreur d'identification ($|\tilde{p}(u) - \overline{p}(u)|$) et l'erreur de modélisation ($|\overline{p}(u) - p(u)|$).

Notre résultat, résumé dans le Théorème 13, est une borne sur l'erreur ponctuelle $|p_M(u) - \overline{p}(u)|$ prenant en compte l'erreur d'estimation et l'erreur d'identification. Cette borne peut facilement être associée aux bornes existantes sur l'erreur de modélisation [Kulesza et collab., 2015a, 2014].

Théorème 13.
Soit p un langage stochastique rationnel, H sa matrice de Hankel associée et un ensemble de N mots tirés indépendamment selon p, on note $\hat{H}$ la matrice de Hankel empirique, σ_k (resp. $\hat{\sigma}_k$) la k^e valeur singulière de H (resp. $\hat{H}$), alors pour tout $t > 0$, pour tout mot u de longueur quelconque l, avec probabilité $1 - 2t(e^t - t - 1)^{-1}$,

$$|\overline{p}(u) - p_M(u)| \leq \left(\sqrt{\frac{2St}{N}} + \frac{2t}{3N}\right)\left(1 + \frac{l+1}{\gamma_k}\right),$$

où $\gamma_k = \max\{\sigma_k - \hat{\sigma}_{k+1}, \hat{\sigma}_k - \sigma_{k+1}\}$ est le fossé spectral et $S = \boldsymbol{\alpha}_0^\top (I - A_\Sigma)^{-2} \boldsymbol{\alpha}_\infty$ est la variance de p.

Nous remarquons que par définition γ_k est strictement positif si $\hat{\sigma}_k > \hat{\sigma}_{k+1}$ ou $\sigma_k > \sigma_{k+1}$. Dans cette borne, nous verrons que $(l+1)/\gamma_k$ correspond à l'erreur d'identification. Si $k \geq d$, nous avons alors $\gamma_k = \hat{\sigma}_k$ et la borne se réduit à

$$\left(\sqrt{\frac{2St}{N}} + \frac{2t}{3N}\right)\left(1 + \frac{l+1}{\hat{\sigma}_k}\right),$$

où le seul terme inconnu est la variance S. Celle-ci pourrait être remplacée par la variance empirique en utilisant des inégalités de Bernstein empiriques pour les sommes de matrices aléatoires. Si $d < k$, nous pouvons remplacer γ_k par une version empirique en utilisant des bornes telles que celles développées par Gittens et Tropp [2011]. Nous remarquons que le Théorème 13 propose une borne indépendante de la taille de la base comparée aux bornes existantes grâce aux résultats récents de Denis et collab. [2014].

3.4 Analyse non-asymptotique des automates compressés

Les sections qui suivent sont dédiées à la preuve du Théorème 13. Nous commençons par rappeler quelques propriétés utiles sur les normes. Nous analysons ensuite l'erreur

dans la matrice de Hankel estimée. Puis, nous propageons celle-ci pour obtenir une borne sur les perturbations dans le projecteur sur le sous-espace singulière. Ensuite, de celle-ci, nous pouvons déduire une borne sur les perturbations dans la série estimée. Nous terminons en commentant les résultats obtenus.

3.4.1 Quelques propriétés utiles

Dans cette section, nous donnons quelques propriétés de base sur les normes matricielles utiles pour la suite. La norme induite $\|.\|_p$ pour les matrices est définie par $\|M\|_p = \max_{\|\mathbf{x}\|_p=1} \|M\mathbf{x}\|_p$. En particulier, nous avons $\|M\|_2 = \sigma_{\max}(M)$ où $\sigma_{\max}(M)$ est la valeur singulière maximale de M. Toute norme induite est sous-multiplicative ($\|AB\| \leq \|A\| \, \|B\|$). Les normes invariantes par transformation unitaire sont telles que, pour toute matrice M et matrice unitaire O, nous avons $\|MO\| = \|OM\| = \|M\|$. Dans les preuves, $\|\cdot\|$ est une norme quelconque invariante par transformation unitaire. En particulier, $\|\cdot\|_2$ est invariante par transformation unitaire. De plus, pour toute matrice M et V telle que V possède des colonnes orthogonales, nous avons $\|MV\|_2 \leq \|M\|_2$.

3.4.2 Concentration des matrices de Hankel infinies

La première étape de la preuve est de borner avec forte probabilité l'erreur en norme dans la matrice de Hankel empirique. Plusieurs résultats ont déjà été établis pour des matrices de Hankel finies [Bailly, 2011a; Balle, 2013; Hsu et collab., 2012]. À la différence des précédents travaux, nous utilisons le résultat de Denis et collab. [2014] qui donne une borne indépendante de la taille de la base et donc qui s'applique aux matrices de Hankel infinies. Dans la suite, nous notons A_Σ la matrice définie par $A_\Sigma = \sum_{\sigma\in\Sigma} A_\sigma$. Comme p est un langage stochastique rationnel, $\sum_{k\geq 0} A_\Sigma^k$ converge vers $(I - A_\Sigma)^{-1}$ et $\sum_{u\in\Sigma^\star} p(u)$ converge vers $\boldsymbol{\alpha}_0^\top (I - A_\Sigma)^{-1}\boldsymbol{\alpha}_\infty = 1$. Soit S la limite de $\sum_{u,v\in\Sigma^\star} p(uv)$, alors $S = \boldsymbol{\alpha}_0^\top (I - A_\Sigma)^{-2}\boldsymbol{\alpha}_\infty$.

Théorème 14.
En utilisant les notations précédentes, nous avons avec probabilité $1 - 2t(e^t - t - 1)^{-1}$,

$$\varepsilon = \left\|H - \hat{H}\right\|_2 \leq \sqrt{\frac{2St}{N}} + \frac{2t}{3N}.$$

3.4.3 Perturbations du projecteur sur le sous-espace singulier

Dans cette section, nous établissons une propriété fondamentale pour le contrôle de l'erreur d'identification. Nous notons $\hat{W} = h\hat{V}\hat{V}^\top$, $\overline{W} = \overline{V}\overline{V}^\top$ et $W = VV^\top$ les projecteurs orthogonaux sur les sous-espaces images de respectivement $\hat{V}$, $\overline{V}$ et V. D'abord le Lemme 1 relie l'erreur $\left\|\hat{W} - \overline{W}\right\|$ aux angles canoniques entre $\mathrm{Im}(\hat{V})$ et $\mathrm{Im}(\overline{V})$. Ensuite, ces angles peuvent être contrôlés en utilisant le Théorème 15.

Théorème 15 (Théorème Sin de [Wedin, 1972])**.**
Soit une matrice X et une perturbation de celle-ci $\tilde{X} = X + E$, on note $(U_1, U_2, D_1, D_2, V_1, V_2, V_3)$ (resp. $(\tilde{U}_1, \tilde{U}_2, \tilde{D}_1, \tilde{D}_2, \tilde{V}_1, \tilde{V}_2, \tilde{V}_3)$) la SVD de X (resp. $\tilde{X}$) tels que,

$$\begin{pmatrix} U_1 \\ U_2 \end{pmatrix} X \begin{pmatrix} V_1 & V_2 & V_3 \end{pmatrix} = \begin{pmatrix} D_1 & 0 & 0 \\ 0 & D_2 & 0 \end{pmatrix},$$

$$\begin{pmatrix} \tilde{U}_1 \\ \tilde{U}_2 \end{pmatrix} \tilde{X} \begin{pmatrix} \tilde{V}_1 & \tilde{V}_2 & \tilde{V}_3 \end{pmatrix} = \begin{pmatrix} \tilde{D}_1 & 0 & 0 \\ 0 & \tilde{D}_2 & 0 \end{pmatrix}.$$

Soit α, δ deux réels positifs tels que

$$\min \sigma\left(\tilde{D}_1\right) \geq \alpha + \delta, \qquad \max \sigma\left(\tilde{D}_2\right) \leq \alpha,$$

alors,

$$\max\{\|\sin\Phi\|, \|\sin\Theta\|\} \leq \frac{\max\{\|R\|, \|S\|\}}{\delta},$$

avec $R = X\tilde{V}_1 - \tilde{U}_1\tilde{D}_1$ et $S = \tilde{X}^\top\tilde{U}_1 - \tilde{V}_1\tilde{D}_1$ et Φ (resp. Θ) la matrice diagonale des angles canoniques entre $\mathrm{Im}(U_1)$ and $\mathrm{Im}(\tilde{U}_1)$ (resp. $\mathrm{Im}(V_1)$ and $\mathrm{Im}(\tilde{V}_1)$). De plus, R et S peuvent être remplacées par E car $\max\{\|S\|, \|R\|\} \leq \|E\|$, et donc

$$\max\{\|\sin\Phi\|, \|\sin\Theta\|\} \leq \frac{\|E\|}{\delta}.$$

Lemme 1.
Soit une matrice X et une perturbation de celle-ci $\tilde{X} = X + E$. Si $\overline{V} \in \mathbb{R}^{n\times k}$ (resp. $\hat{V}$) sont les vecteurs singuliers de gauche associés aux k plus grandes valeurs singulières $\sigma_1 \leq \sigma_2 \leq ... \leq \sigma_k$ (resp. $\hat{\sigma}_1 \leq \hat{\sigma}_2 \leq ... \leq \hat{\sigma}_k$) de X (resp. $\hat{X}$) alors $\overline{W}$ (resp. $\hat{W}$) sont des projecteurs orthogonaux sur $\mathrm{Im}(\hat{V})$ (resp. $\mathrm{Im}(\overline{V})$) et,

$$\left\|\hat{W} - \overline{W}\right\|_2 \leq \frac{\|E\|_2}{\gamma_k},$$

avec $\gamma_k = \max\{\sigma_k - \hat{\sigma}_{k+1}, \hat{\sigma}_k - \sigma_{k+1}\}$.

Démonstration. En utilisant la décomposition CS, Stewart et guang Sun [1990] a montré que les valeurs singulières de $\hat{W} - \overline{W}$ sont, avec une multiplicité de 2, les angles canoniques entre $\mathrm{Im}(\hat{V})$ et $\mathrm{Im}(\overline{V})$. En utilisant $\left\|\hat{W} - \overline{W}\right\|_2 = \|\sin\Theta\|_2$ dans le Théorème 15, nous obtenons le résultat. □

Proposition 35.
En utilisant les notations précédentes, nous avons

$$\left\|\hat{W} - \overline{W}\right\|_2 \leq \frac{\varepsilon}{\gamma_k},$$

avec $\gamma_k = \max\{\sigma_k - \hat{\sigma}_{k+1}, \hat{\sigma}_k - \sigma_{k+1}\}$.

Démonstration. La preuve découle directement du Lemme 1 en utilisant $X = H$, $\hat{X} = \hat{H}$ and $E = \hat{H} - H$. □

3.4.4 Perturbations dans la série

Pour prouver le Théorème 13, nous décomposons $|\overline{p}(u) - p_M(u)|$ en deux parties correspondantes aux erreurs d'identification et d'estimation,

$$|\overline{p}(u) - p_M(u)| \leq |\overline{p}(u) - \tilde{p}(u)| + |\tilde{p}(u) - p_M(u)|.$$

Chacune des sources d'erreur est bornée séparément. Avant tout, nous démontrons quelques propriétés utiles. Soit un mot $u = \sigma_1\sigma_l$, nous avons,

$$\begin{aligned}
\overline{p}(u) &= \mathbf{1}_\varepsilon{}^\top H\overline{W}(T_{\sigma_1}^\top\overline{W})\dots(T_{\sigma_l}^\top\overline{W})\mathbf{1}_\varepsilon{}^\top,\\
p_M(u) &= \mathbf{1}_\varepsilon{}^\top\hat{H}\hat{W}(T_{\sigma_1}^\top\hat{W})\dots(T_{\sigma_l}^\top\hat{W})\mathbf{1}_\varepsilon{}^\top,\\
\tilde{p}(u) &= \mathbf{1}_\varepsilon{}^\top H\hat{W}(T_{\sigma_1}^\top\hat{W})\dots(T_{\sigma_l}^\top\hat{W})\mathbf{1}_\varepsilon{}^\top,\\
p(u) &= \mathbf{1}_\varepsilon{}^\top HVV^\top(T_{\sigma_1}^\top VV^\top)\dots(T_{\sigma_l}^\top VV^\top)\mathbf{1}_\varepsilon{}^\top.
\end{aligned}$$

Comme $\hat{W}$, $\overline{W}$ et W sont des projecteurs orthogonaux, nous avons

$$\left\|\hat{W}\right\|_2 = \left\|\overline{W}\right\|_2 = \|W\|_2 = 1. \tag{3.1}$$

Par définition de $T_\sigma^\top$, nous vérifions que $T_\sigma^\top T_\sigma = (\delta_{u=v})_{u,v\in\Sigma^\star}$ est l'identité. Donc, nous avons

$$\left\|T_\sigma^\top\right\|_2 = 1. \tag{3.2}$$

Comme p est un langage stochastique, nous avons

$$\left\|\mathbf{1}_\varepsilon{}^\top H\right\|_2 = \sqrt{\sum_{u\in\mathcal{P}} p(u)^2} \leq \sqrt{\sum_{u\in\Sigma^\star} p(u)^2} \leq \sum_{u\in\Sigma^\star} p(u) = 1. \tag{3.3}$$

Par définition, nous avons

$$\|\mathbf{1}_\varepsilon\|_2 = 1. \tag{3.4}$$

Erreur d'estimation En utilisant la sous-multiplicativité des normes induites, nous avons

$$|\tilde{p}(u) - p_M(u)| \leq \left\|\mathbf{1}_\varepsilon{}^\top H\hat{W} - \mathbf{1}_\varepsilon{}^\top \hat{H}\hat{W}\right\|_2 \times \left\|(T_{\sigma_1}^\top \hat{W})\dots(T_{\sigma_l}^\top \hat{W})\mathbf{1}_\varepsilon{}^\top\right\|_2.$$

La sous-multiplicativité et les Équations (3.1) et (3.2) donnent,

$$\left\|(T_{\sigma_1}^\top \hat{W})\dots(T_{\sigma_l}^\top \hat{W})\mathbf{1}_\varepsilon{}^\top\right\|_2 \leq 1.$$

Par le Théorème 14 et les Équations (3.1) et (3.4), nous obtenons que

$$|\tilde{p}(u) - p_M(u)| \leq \left\|\mathbf{1}_\varepsilon{}^\top \left(H - \hat{H}\right)\hat{W}\right\|_2 \leq \left\|H - \hat{H}\right\|_2 \leq \epsilon \tag{3.5}$$

Erreur d'identification En utilisant la sous-multiplicativité des normes induites et les Équations (3.3) et (3.4), nous avons

$$\begin{aligned}|\overline{p}(u) - \tilde{p}(u)| \leq& \left\|\mathbf{1}_\varepsilon{}^\top H\right\|_2 \left\|\overline{W}(T_{\sigma_1}^\top \overline{W})\dots(T_{\sigma_l}^\top \overline{W}) - \hat{W}(T_{\sigma_1}^\top \hat{W})\dots(T_{\sigma_l}^\top \hat{W})\right\|_2 \left\|\mathbf{1}_\varepsilon{}^\top\right\|_2 \\ \leq& \left\|\overline{W}(T_{\sigma_1}^\top \overline{W})\dots(T_{\sigma_l}^\top \overline{W}) - \hat{W}(T_{\sigma_1}^\top \hat{W})\dots(T_{\sigma_l}^\top \hat{W})\right\|_2.\end{aligned}$$

Nous décomposons l'erreur par inégalité triangulaire,

$$\begin{aligned}|\overline{p}(u) - \tilde{p}(u)| \leq& \left\|\hat{W} - \overline{W}\right\|_2 \left\|(T_{\sigma_1}^\top \hat{W})\dots(T_{\sigma_l}^\top \hat{W})\right\|_2 \\ &+ \left\|\overline{W}\right\|_2 \left\|(T_{\sigma_1}^\top \overline{W})\dots(T_{\sigma_l}^\top \overline{W}) - (T_{\sigma_1}^\top \hat{W})\dots(T_{\sigma_l}^\top \hat{W})\right\|_2.\end{aligned}$$

Des Équations (3.1) et (3.2), nous obtenons que

$$|\overline{p}(u) - \tilde{p}(u)| \leq \left\|(T_{\sigma_1}^\top \overline{W})\dots(T_{\sigma_l}^\top \overline{W}) - (T_{\sigma_1}^\top \hat{W})\dots(T_{\sigma_l}^\top \hat{W})\right\|_2 + \left\|\hat{W} - \overline{W}\right\|_2.$$

Le deuxième terme est borné en utilisant la Proposition 35, tandis que le premier terme nécessite un peu plus de travail. Nous définissons deux nouvelles quantités,

$$\begin{aligned}\Delta_l &= (T_{\sigma_1}^\top \overline{W})\dots(T_{\sigma_l}^\top \overline{W}) - (T_{\sigma_1}^\top \hat{W})\dots(T_{\sigma_l}^\top \hat{W}) \\ \Delta^l &= T_{\sigma_l}^\top \overline{W} - T_{\sigma_l}^\top \hat{W} = T_{\sigma_l}^\top (\overline{W} - \hat{W})\end{aligned}$$

Alors, en utilisant de nouveau les propriétés des normes induites, nous avons

$$\begin{aligned}\|\Delta_l\|_2 =& \left\|\Delta^l(T_{\sigma_1}^\top \overline{W})\dots(T_{\sigma_{l-1}}^\top \overline{W}) + (T_{\sigma_{l-1}}^\top \hat{W})\Delta_{l-1}\right\|_2 \\ \leq& \left\|\Delta^l\right\|_2 \left\|(T_{\sigma_1}^\top \overline{W})\dots(T_{\sigma_{l-1}}^\top \overline{W})\right\|_2 \left\|T_{\sigma_{l-1}}^\top \hat{W}\right\|_2 \|\Delta_{l-1}\|_2 \\ \leq& \left\|\Delta^l\right\|_2 + \|\Delta_{l-1}\|_2 \text{ par les Équations (3.1) et (3.2)} \\ \leq& \sum_{i=1}^{l} \left\|\Delta^i\right\|_2, \text{ par induction.}\end{aligned}$$

Pour borner $\|\Delta^i\|_2$, nous utilisons les Équation (3.2) et Proposition 35. Nous obtenons alors,

$$\left\|\Delta^i\right\|_2 \leq \left\|T_{\sigma_l}^\top\right\|_2 \left\|\overline{W} - \hat{W}\right\|_2 \leq \left\|\overline{W} - \hat{W}\right\|_2 \leq \frac{\epsilon}{\gamma_k}$$

Donc $\|\Delta_l\|_2 \leq \frac{l\epsilon}{\gamma_k}$ et nous obtenons que

$$|\overline{p}(u) - \tilde{p}(u)| \leq \frac{(l+1)\epsilon}{\gamma_k}. \tag{3.6}$$

Finalement, en combinant les résultats des Équations (3.5) et (3.6), nous obtenons que

$$|\overline{p}(u) - p_M(u)| \leq |\overline{p}(u) - \tilde{p}(u)| + |\tilde{p}(u) - p_M(u)| \leq \epsilon\left(1 + \frac{l+1}{\gamma_k}\right).$$

Ce qui termine la preuve du Théorème 13.

3.4.5 Discussion

Comparé aux autres analyses non-asymptotiques de l'algorithme Spectral, la borne du Théorème 13 est indépendante de la taille du nombre de préfixes et de suffixes différents présents dans l'ensemble d'apprentissage. De plus, elle ne fait pas l'hypothèse que le vrai rang est connu et utilisé ($d = k$). Enfin, notre analyse permet d'obtenir une dépendance beaucoup plus faible par rapport aux petites valeurs singulières dans le dénominateur de la borne.

Malheureusement, l'obtention d'une borne en variation totale se fait au prix d'une dépendance exponentielle en l. L'astuce proposée dans [Bailly, 2011a; Balle, 2013; Hsu et collab., 2012] ne semble pas applicable pour les modèles compressés. En effet, pour contrôler l'erreur de la série estimée, il faut que celle-ci ne diverge pas avec l. Or celle-ci est censée être proche de la série cible qui est convergente. En fait, si tous les coefficients des matrices de transition estimés sont proches des vrais, nous obtenons la convergence de la série estimée, comme le prouvent Denis et collab. [2006, Théorème 3]. Une façon de montrer que les coefficients estimés de transition se rapprochent des vrais est de les comparer dans une base commune. L'astuce permettant cette analyse plus fine est de remarquer que si le nombre d'échantillons est suffisamment grand, le sous-espace vectoriel estimé n'intersecte pas le complémentaire orthogonal du vrai avec grande probabilité. Dans les preuves, cette astuce s'illustre par l'existence d'une matrice de passage entre le sous-espace vectoriel estimé et le vrai (voir par exemple [Hsu et collab., 2012, Lemme 9] et [Bailly, 2011a, Proposition 2 et Lemme 19]). Or, par définition, dans les modèles compressés une telle matrice de passage n'existe pas puisque les espaces sont de dimensions différentes. Néanmoins, nous pourrions essayer

de trouver une série auxiliaire possédant une représentation linéaire de même dimension et absolument convergente [Bailly, 2011a] ou bien rationnelle sur $\mathbb{R}^+$ [Balle, 2013; Hsu et collab., 2012]. Cet espoir semble vain. En tout cas, nous comparons p_M à $\overline{p}$ et $\overline{p}$ n'est pas forcément absolument convergent ou rationnel sur $\mathbb{R}^+$ comme le requière l'astuce précédente. En contrepartie, aucune hypothèse sur le nombre d'échantillons n'est faite dans notre analyse.

Nous pouvons nous demander si notre décomposition de l'erreur est légitime. En effet, nous pourrions essayer de comparer directement p_M à p. Comme les dimensions ne sont pas les mêmes, supposons que l'on complète par des 0 la représentation linéaire compressée. Dans ce cas, il n'y a pas de raison que les $\left\|\overline{A}_\sigma - A_\sigma\right\| = \left\|\overline{V}^\top T_\sigma \overline{V} - V^\top T_\sigma V\right\|$ tendent vers 0 avec N. Nous ne pouvons donc pas obtenir la convergence de la série estimée à partir d'un certain rang par [Denis et collab., 2006, Théorème 3].

La convergence de la série estimée n'est pas forcément nécessaire et d'autres hypothèses permettent d'établir des résultats partiels. Dans [Kulesza et collab., 2015a], les auteurs considèrent l'erreur de modélisation entre la vraie série p (convergente) et le vrai modèle compressé $\overline{p}$ *a priori* divergent. En pondérant par une fonction décroissante en la longueur des mots, l'erreur quadratique entre ces deux séries reste contrôlable. Pour résumer, une analyse plus fine consisterait à contrôler la vitesse de divergence de la série compressée.

Détaillons une autre borne sur l'erreur de modélisation. Dans [Kulesza et collab., 2014], les auteurs supposent toujours que la matrice de Hankel H est connue. Ils établissent alors une borne en variation totale entre p et $\overline{p}$, où p est un processus stochastique réalisé par un HMM. Par conséquent, il n'y a pas d'erreurs d'estimation ou d'identification. Ils analysent d'abord les restrictions sur la matrice d'observation O et la distribution initiale π nécessaires pour garantir que les erreurs de compression ne soient pas arbitrairement grandes. Ils montrent ensuite que si $\sigma_d(O)$ est assez grande et si π est aussi la distribution stationnaire ($T\pi = \pi$), alors les erreurs de modélisation sont dominées par σ_{k+1}. Leur résultat est résumé dans Théorème 16.

Théorème 16.
Soit O, T, et π définissant un HMM de dimension $d \geq 4$ tel que $\operatorname{rang}(O) = d$, $\pi > 0$ *et $T\pi = \pi$, alors pour tout l*

$$\sum_{u \in \Sigma^l} |\overline{p}(u) - p(u)| \leq \sqrt{d}\left(\sigma_d(O)^{-1}\sqrt{d}\right)^{l+3} \sigma_{k+1}.$$

Ce Théorème montre que l'erreur de modélisation est proportionnelle à la plus grande valeur singulière associée aux dimensions omises. En approfondissant leur analyse nous pouvons montrer que l'erreur est proportionnelle à $\|\Pi_{\mathrm{Im}(H_\mathcal{B}^T \hat{U})} H_\mathcal{B} - H_\mathcal{B}\|_2$, où $\Pi_{\mathrm{Im}(H_\mathcal{B}^T \hat{U})}$ est le projecteur orthogonal sur $\mathrm{Im}(H_\mathcal{B}^T \hat{U})$. Quand $\hat{U}$ est une version tronquée de U, $\|\Pi_{\mathrm{Im}(H_\mathcal{B}^T \hat{U})} H_\mathcal{B} - H_\mathcal{B}\|_2 = \sigma_{k+1}$ et nous obtenons le Théorème 16. Cependant, quand $\hat{U}$ est calculée à partir de $H_\mathcal{B}$, $\|\Pi_{\mathrm{Im}(H_\mathcal{B}^T \hat{U})} H_\mathcal{B} - H_\mathcal{B}\|_2$ ne peut pas être bornée facilement. À cause de cela, leur analyse ne permet pas de prendre en compte l'erreur d'estimation ni d'identification. Comme notre analyse concerne l'erreur d'estimation et d'identification, elle peut être combinée avec le Théorème 16, sous les mêmes hypothèses, pour obtenir un résultat complet.

3.5 Algorithme Spectral régularisé

La borne précédente est riche d'enseignement pour l'inférence de modèles compressés. Nous commençons par réécrire γ_k en fonction du fossé spectral $\delta_k = \sigma_k - \sigma_{k+1}$.

$$\gamma_k = \delta_k + \max\{\sigma_{k+1} - \hat{\sigma}_{k+1}, \hat{\sigma}_k - \sigma_k\}$$

Prenons l'exemple d'un langage stochastique possédant un grand fossé dans son spectre. Nous posons $m = \operatorname{argmax}_i \delta_i$ et nous supposons que $m < d$. En estimant un modèle de taille m, en fonction du nombre d'échantillons, nous pouvons espérer que l'erreur d'identification soit faible par rapport à l'erreur de modélisation commise par l'inférence d'un modèle compressé. Dans ce cas, apprendre un modèle compressé est bénéfique. Partant de cette observation, nous concluons que pour l'inférence le choix du rang doit se faire en fonction des données. Bien que l'estimation d'un modèle compressé introduise un biais, la variance peut en être grandement diminuée.

3.5.1 Minimisation de la norme nucléaire

Nous proposons alors un algorithme inspiré des algorithmes d'optimisation convexe qui adapte la taille du modèle en fonction des données. L'algorithme Spectral cherche la meilleure approximation en norme de Frobenius de rang k de $\hat{H}$. Cette approximation est donnée par $\hat{U}\hat{D}\hat{V}^\top$, calculée par SVD tronquée. Nous proposons à la place de résoudre le problème suivant,

$$\min_{\tilde{H}} \operatorname{rang}(H)$$
$$\text{tel que } \left\|\tilde{H} - \hat{H}\right\|_H \leq \varepsilon,$$

pour un ε donné. Malheureusement, ce problème est **NP**-dur. À la place, nous résolvons un autre problème qui en est, en un certain sens [Cai et collab., 2010], la plus petite relaxation convexe,

$$\min_{\tilde{H}} \mu\left\|\tilde{H}\right\|_* + \frac{1}{2}\left\|\tilde{H} - \hat{H}\right\|_F^2,$$

où μ met en évidence le compromis entre le rang de $\tilde{H}$ et la qualité d'approximation de $\hat{H}$. Comme précédemment, nous notons $\|X\|_*$ la norme nucléaire de X, égale à la somme des valeurs singulières de X. Ce problème admet une solution analytique simple donnée par le Seuillage des Valeurs Singulières, ou *Singular Value Thresholding* (SVT). Soit la SVD de X,

$$X = UDV^\top, \qquad D = \operatorname{diag}(\{\sigma_i\}_{1 \leq i \leq r}),$$

alors pour $\mu \geq 0$, nous introduisons l'opérateur $\mathcal{D}_\tau$ réalisant le SVT de X, tel que

$$\mathcal{D}_\mu(X) = U\mathcal{D}_\mu(\Sigma)V^\top, \qquad \mathcal{D}_\mu(D) = \operatorname{diag}((7\sigma_i - \mu)_+),$$

où $(x)_+ = \max\{x, 0\}$. Cai et collab. [2010] ont montré que,

$$\mathcal{D}_\mu(\hat{H}) = \min_{\tilde{H}} \mu\left\|\tilde{H}\right\|_* + \frac{1}{2}\left\|\tilde{H} - \hat{H}\right\|_F^2,$$

Nous proposons alors de remplacer la SVD tronquée de rang k par un SVT de paramètre μ. Cette méthode permet d'adapter la taille du modèle appris aux données, ce qui est particulièrement utile dans les applications d'apprentissage en ligne. Plus exactement, soit $\tilde{H} = \mathcal{D}_\mu(\hat{H}) = \hat{U}\hat{D}\hat{V}^\top$, l'algorithme Spectral régularisé, estime la représentation linéaire par,

$$\hat{A}_\sigma = \hat{V}^\top T_\sigma^\top \hat{V}, \qquad \hat{\boldsymbol{\alpha}}_0^\top = \mathbf{1}_\varepsilon{}^\top \hat{H}\hat{V} = \mathbf{1}_\varepsilon{}^\top \hat{U}\hat{D}, \qquad \hat{\boldsymbol{\alpha}}_\infty = \hat{V}^\top \mathbf{1}_\varepsilon.$$

3.5.2 Algorithme pour les matrices de Hankel finies

Au Chapitre 2, nous avons expliqué la différence fondamentale entre l'algorithme Spectral travaillant avec une matrice de Hankel infinie et celui qui utilise une base. Les deux algorithmes identifient une famille finie de séries formelles $\{r_1, \ldots, r_d\}$ à partir des vecteurs singuliers à droite. Quand l'algorithme travaille à partir d'une matrice de Hankel infinie, $\{\dot{\sigma}r_1, \ldots, \dot{\sigma}r_d\}$ peut être calculé à partir de $\{r_1, \ldots, r_d\}$ grâce à l'opérateur de décalage T_σ. Quand l'algorithme travaille sur un base finie $\mathcal{B} = (\mathcal{P}, \mathcal{S})$, les séries $\{r_1, \ldots, r_d\}$ sont décrites sur $\mathcal{S}$ et donc $\{\dot{\sigma}r_1, \ldots, \dot{\sigma}r_d\}$ ne peut pas être calculé à partir de $\{r_1, \ldots, r_d\}$. Néanmoins, en utilisant $\hat{H}^\sigma_\mathcal{B}$, nous pouvons estimer les $\{\dot{\sigma}r_1, \ldots, \dot{\sigma}r_d\}$ par les lignes de $\hat{D}^{-1}\hat{U}^\top H^\sigma_\mathcal{B}$. Cependant, ces séries formelles estimées ne sont pas forcément contenues dans $[[\{r_1, \ldots, r_d\}]]$.

Bien que l'algorithme Spectral travaillant sur une base soit consistant, cette particularité semble expérimentalement rendre l'algorithme Spectral moins stable. C'est pourquoi nous proposons ici, une version légèrement différente qui contraint tout $\sigma \in \Sigma$ $[[\{\dot{\sigma}r_1, \ldots, \dot{\sigma}r_d\}]]$ et $[[\{r_1, \ldots, r_d\}]]$ à appartenir au même sous espace vectoriel.

De la même façon, l'algorithme Spectral pour les matrices de Hankel finies peut être modifié en remplaçant la SVD pour un SVT. Pour simplifier supposons que $(\varepsilon, \varepsilon) \in \mathcal{B}$. Dans ce cas, nous construisons la matrice $\hat{F}$ par empilement des $\hat{H}_\mathcal{B}$ et $\{\hat{H}^\sigma_\mathcal{B}\}_{\sigma\in\Sigma}$, telle que,

$$\hat{F} = \begin{pmatrix} \hat{H}_\mathcal{B} \\ \hat{H}^{\sigma_1}_\mathcal{B} \\ \vdots \\ \hat{H}^{\sigma_{|\Sigma|}}_\mathcal{B} \end{pmatrix}. \tag{3.7}$$

Nous calculons ensuite $\tilde{F} = D_\mu(\hat{F}) = \hat{U}\hat{D}\hat{V}^\top$ dont nous extrayons $\tilde{H}_\mathcal{B}$ et $\{\tilde{H}^\sigma_\mathcal{B}\}_{\sigma\in\Sigma}$. Cette procédure, diffère de l'algorithme Spectral qui estime la représentation linéaire en utilisant les $\hat{H}^\sigma_\mathcal{B}$. En passant par les matrices $\hat{F}$ et $\tilde{F}$, nous nous assurons que toutes les matrices $\tilde{H}_\mathcal{B}$ et $\{\tilde{H}^\sigma_\mathcal{B}\}_{\sigma\in\Sigma}$ partagent le même sous-espace singulier à droite. En d'autres termes, soit les $\{r_i\}$ la famille finie décrite par les lignes de $\hat{V}^\top$, nous nous assurons que pour tout $\sigma \in \Sigma$, les $\{\dot{\sigma}r_i\}$ appartiennent à l'espace généré par les $\{r_i\}$. Cette procédure semble apporter plus de stabilité lors de l'inférence et ne modifie pas la consistance de l'algorithme. Nous notons alors $\hat{U}_\varepsilon$, l'espace singulier à gauche de $\hat{H}_\mathcal{B}$ ($\hat{U}_\varepsilon$ correspond aux premières lignes de $\hat{U}$). La représentation linéaire est alors estimée par les équations,

$$\begin{aligned}
\hat{A}_\sigma &= \hat{D}^{-1}\hat{U}_\varepsilon^\top \tilde{H}^\sigma_\mathcal{B}\hat{V} = \hat{D}^{-1}\hat{U}_\varepsilon^\top \tilde{H}^\sigma_\mathcal{B}(D^{-1}\hat{U}_\varepsilon^\top \tilde{H}_\mathcal{B})^\dagger, \\
\hat{\boldsymbol{\alpha}}_0^\top &= {\mathbf{1}_\varepsilon}^\top \tilde{H}^\sigma_\mathcal{B}\hat{V}, \\
\hat{\boldsymbol{\alpha}}_\infty &= \hat{V}^\top \mathbf{1}_\varepsilon.
\end{aligned}$$

3.5.3 Régularisation de Tikhonov

Nous considérons dans cette section que l'on travaille avec des matrices de Hankel définies sur une base finie, mais la discussion s'applique aussi dans l'autre cas. En introduction, nous avons expliqué qu'une partie de l'erreur contenue dans $\hat{H}$ était annulée car orthogonale au sous-espace vectoriel décrit par les lignes de $\hat{V}$. L'autre partie de l'erreur se répercute lors de la régression linéaire. En effet, lors de la régression entre les $\{r_i\}$ et les $\{\dot{\sigma}r_i\}$, une partie de l'erreur est supprimée. Une astuce bien connue pour réduire l'erreur est d'ajouter une régularisation lors de la régression

entre $\hat{V}^\top = D^{-1}\hat{U}^\top \tilde{H}_\mathcal{B}$ et $\hat{D}^{-1}\hat{U}^\top \tilde{H}^\sigma_\mathcal{B}$ (dans la section précédente, nous faisons une régression entre $\hat{V}^\top = D^{-1}\hat{U}_\varepsilon^\top \tilde{H}_\mathcal{B}$ et $\hat{D}^{-1}\hat{U}_\varepsilon^\top \tilde{H}^\sigma_\mathcal{B}$). Bien que cela introduise un biais, la régularisation réduit l'amplification de l'erreur due au mauvais conditionnement de $\hat{V}^\top$. La représentation linéaire donnée par l'algorithme de la section précédente auquel nous avons ajouté une régularisation de Tikhonov est donnée par

$$\begin{aligned}
\hat{A}_\sigma &= \hat{D}^{-1}\hat{U}_\varepsilon^\top \tilde{H}^\sigma_\mathcal{B}\hat{V}' = \hat{D}^{-1}\hat{U}_\varepsilon^\top \tilde{H}^\sigma_\mathcal{B}(D^{-1}\hat{U}_\varepsilon^\top \tilde{H}_\mathcal{B})^{\lambda\dagger},\\
\hat{\boldsymbol{\alpha}}_0^\top &= \mathbf{1}_\varepsilon{}^\top \tilde{H}^\sigma_\mathcal{B}\hat{V}',\\
\hat{\boldsymbol{\alpha}}_\infty &= \hat{V}^\top \mathbf{1}_\varepsilon,
\end{aligned}$$

où $X^{\lambda\dagger} = (X^\top X + \lambda I)^{-1}X^\top$.

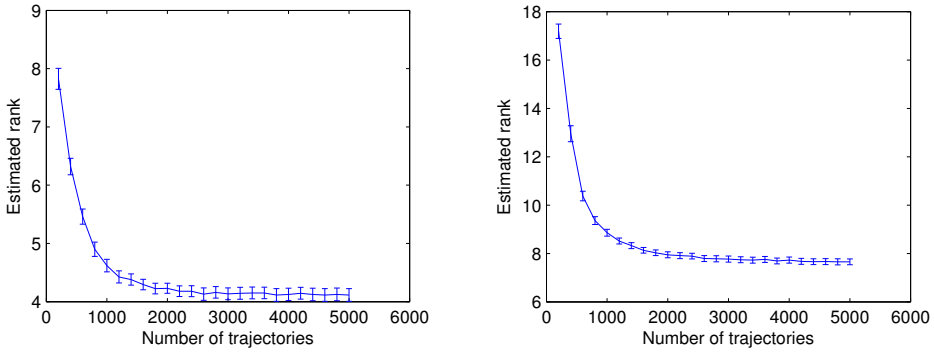
3.6 Expériences

3.6.1 Génération des données d'apprentissage

Dans nos expériences, nous nous désignons par SVD(d) l'algorithme Spectral où l'espace vectoriel identifié possède autant de dimensions que le rang de T. L'algorithme SVT se réfère à l'algorithme proposé dans ce chapitre. Pour SVT, nous avons pris $\mu = \sigma_{\max}(\hat{F})/100$. Enfin, nous avons évalué l'algorithme Spectral, mais en utilisant le rang estimé par SVT à la place du vrai rang d. Cet algorithme est désigné par SVD(l). Ces trois algorithmes sont évalués sur des HMMs générés aléatoirement. Pour générer les matrices de transition et d'observation, nous avons contraint le facteur de ramification. C'est à dire que chaque état peut transiter vers (resp. émettre) seulement un petit sous-ensemble des états (resp. observations). Les probabilités sont ensuite tirés uniformément. Pour générer une matrice de transition avec le rang r, nous générons seulement r de ses colonnes, qui sont dupliquées au hasard obtenir une matrice carrée. Deux tailles de HMMs sont sélectionnés pour évaluer les algorithmes. Les HMMs de petites tailles ont 5 états, 10 observations et la matrice de transition est de plein rang. Le facteur de ramification pour les petits HMMs est de 3 pour les transitions et les observations. Les HMMs de tailles moyennes ont 20 états, 25 observations et le rang de la matrice de transition est 10. Le facteur de ramification est 5 pour les transitions et les observations. Pour chaque taille de HMMs, 10 HMMs sont générés et pour chaque HMM, nous générons 20 ensembles de trajectoires. Chaque trajectoire est composée de 20 pas. Les résultats sont données en fonction du nombre de trajectoires d'apprentissage et du paramètre de régularisation λ.

3.6.2 Estimation du rang

La Figure 3.1 montre comment le rang estimé diminue lorsque le nombre d'exemples croît. Les barres d'erreurs indiquent un intervalle de confiance à 95% calculé à partir d'une Gaussienne. Quand peu d'exemples sont disponibles, le rang est surestimé. Toutefois, il converge rapidement vers sa valeur finale. Nous remarquons que notre algorithme tend à sous-estimer légèrement le véritable rang, même si cela ne se verra pas dans les performances. Comme nous avons utilisé la même méthode pour choisir μ pour les deux tailles de HMMs, cela montre que SVT peut estimer correctement le rang sans avoir besoin d'être adapté à chaque problème.



(a) Estimation du rang dans les petits HMMs.

(b) Estimation du rang dans les HMMs de taille moyenne.

FIGURE 3.1 – Estimation du rang en fonction de la taille de l'ensemble d'apprentissage.

3.6.3 Prédiction à un pas

La qualité du modèle appris est mesurée par l'erreur moyenne au carré entre la distribution apprise et la vraie distribution sur la prochaine observation. Le score est moyenné sur les 20 pas de chaque séquence et sur les 200 séquences de l'ensemble de test. Comme l'apprentissage est impropre, le modèle appris peut retourner des valeurs négatives et ne sommant pas à un. Nous avons choisi de seuiller à zéro puis de normaliser ces valeurs. Les Figures 3.2 et 3.3 montrent que l'erreur décroît avec la taille de l'ensemble d'apprentissage. Chaque courbe représente les performances pour une valeur différente de λ et une dernière courbe utilise pour chaque simulation et chaque algorithme la meilleure valeur de λ trouvée. Les barres d'erreurs indiquent un intervalle de confiance à 95% calculé à partir d'une Gaussienne.

Globalement, les résultats illustrent le compromis biais-variance incarné par le choix de λ. Quand l'ensemble d'apprentissage est petit, les grandes valeurs de λ produisent les meilleurs résultats. Mais quand l'ensemble d'apprentissage grossit, les grandes valeurs de λ introduisent un biais qui fait décroître les performances.

Deuxièmement, en comparant SVD(d) et SVD(l), nous pouvont voir qu'utiliser le rang estimé produit de meilleurs résultats pour la même valeur de λ quand le paramètre de régularisation est adapté à la taille de l'ensemble d'apprentissage. Quand la meilleure valeur de λ pour chaque algorithme (dernière courbe) est utilisée les résultats sont similaires. Ainsi, SVD(d) utilise une valeur de λ plus élevée que SVD(l) pour obtenir des performances similaires. Donc, adapter le rang aux données plutôt que de choisir systématiquement le vrai rang repousse le compromis biais-variance dans la régression.

Enfin, nous comparons SVT à SVD(l). Les deux algorithmes apprennent des modèles de même taille mais SVT réduit légèrement les valeurs singulières et cherche un sous-espace vectoriel contenant simultanément les $\{r_i\}$ et les $\{\dot{\sigma} r_i\}$. Globalement, SVT(l) obtient de meilleurs résultats, sauf pour la combinaison de grands ensembles d'apprentissage et d'un λ est élevé qui est expérimentalement injustifiée. En conclusion, nous retrouvons expérimentalement les avantages cités de SVT par rapport à SVD(l).

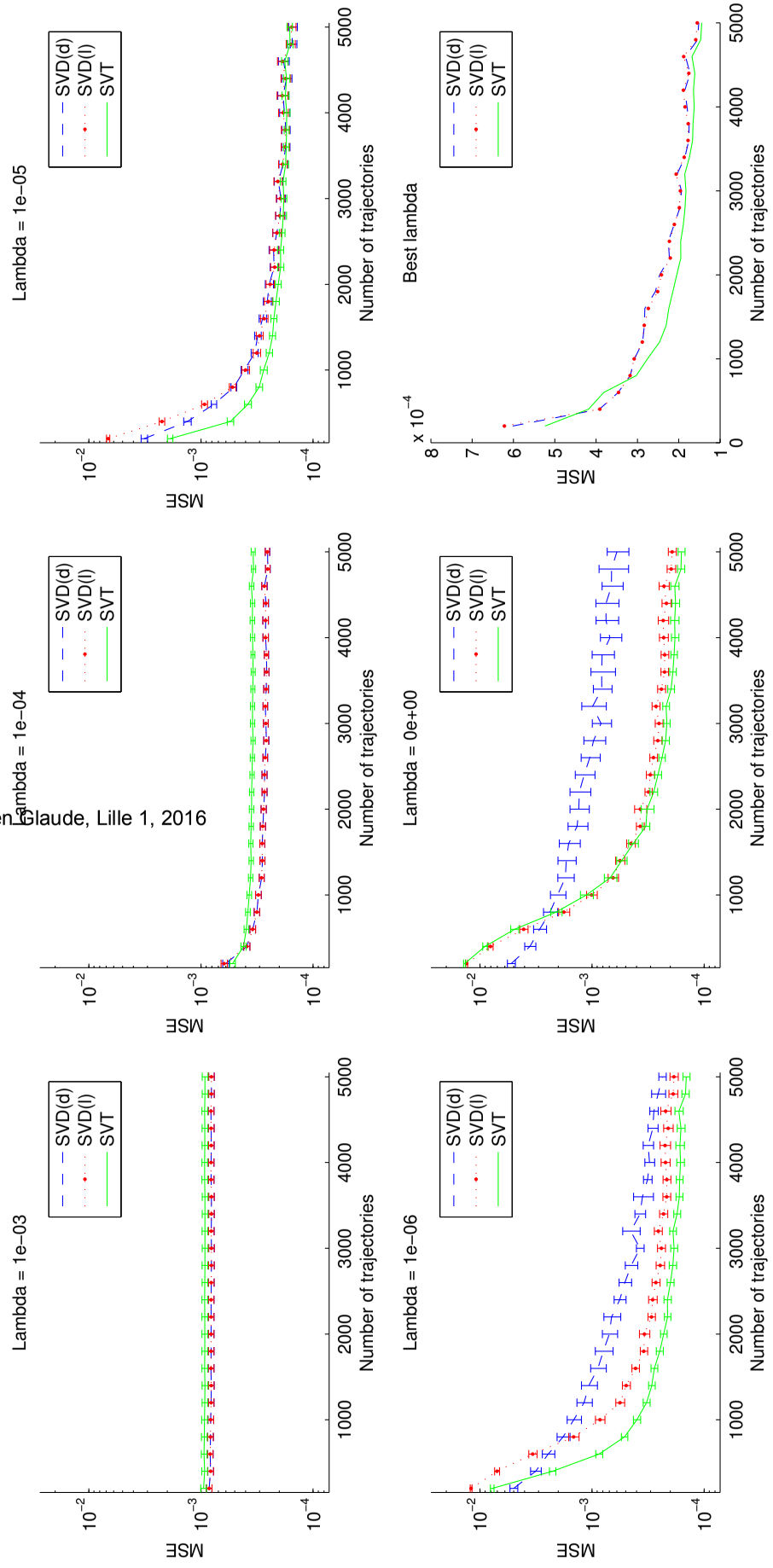


FIGURE 3.2 – Erreur de la prédiction à un pas pour les petits HMMs. Le vrai rang vaut 5.

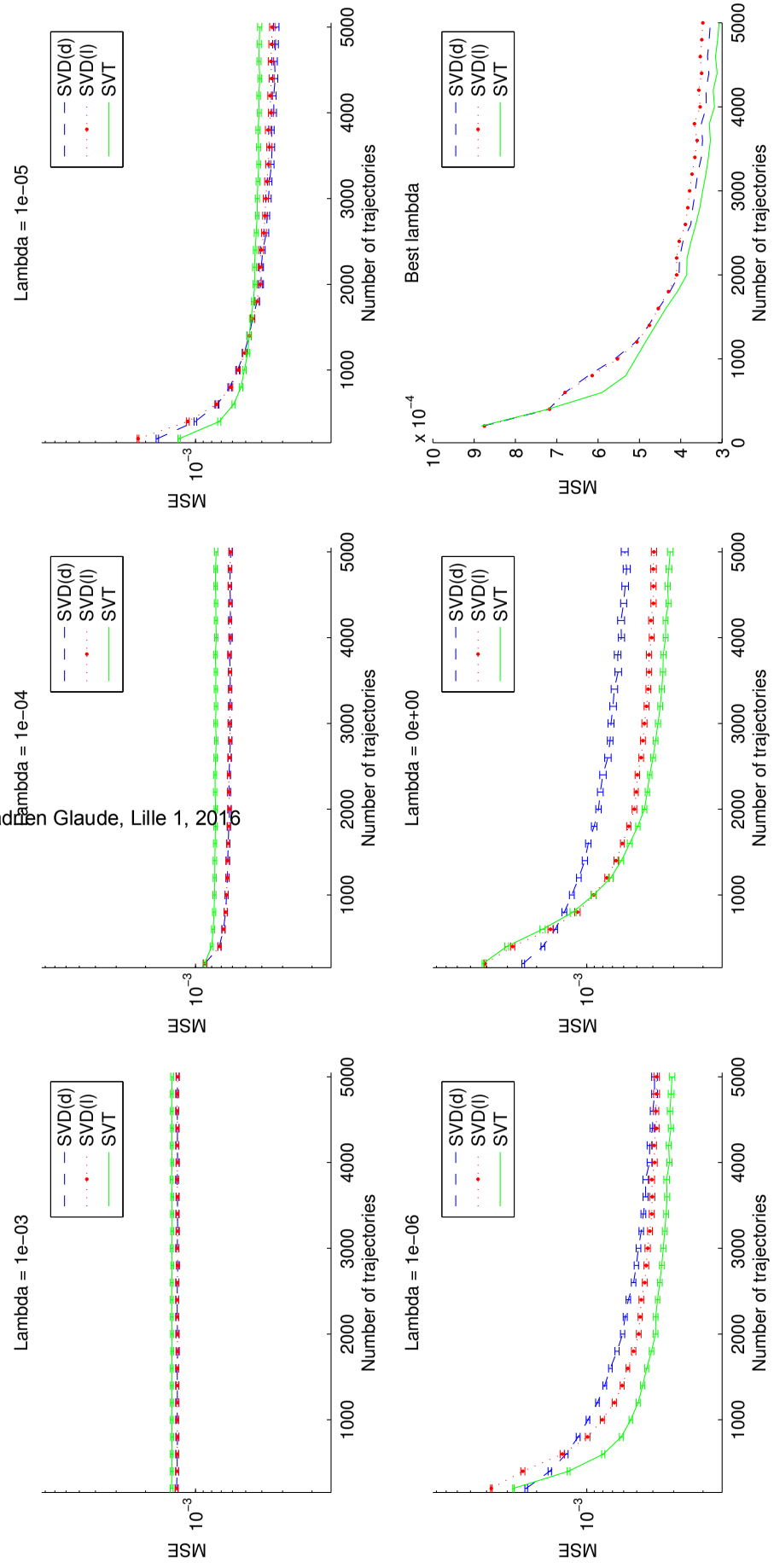


FIGURE 3.3 – Erreur de la prédiction à un pas pour les HMMs de taille moyenne. Le vrai rang vaut 10.

3.7 Conclusions

Dans ce chapitre nous avons étudié l'impact du choix de la dimension à la fois théoriquement et expérimentalement. La contribution de la partie théorique repose sur une analyse non-asymptotique de l'erreur ponctuelle de l'algorithme Spectral dans le cas des modèles compressés. À notre connaissance, c'est le seul travail traitant cette problématique dans le contexte de l'inférence. Cette analyse utilise les récents résultats sur la concentration des sommes de matrices aléatoires, permettant de donner un résultat indépendant de la dimension de la base utilisée. Cette caractéristique permet d'analyser la variante de l'algorithme Spectral travaillant à partir d'une matrice de Hankel infinie. Nous rappelons que ce n'est pas gênant pour l'inférence car la version empirique de cette matrice aura toujours un nombre fini de valeurs non nulles. La borne obtenue permet d'analyser l'effet de la dimension du modèle appris sur l'erreur et de conclure qu'il est important d'adapter celle-ci aux données.

Pour illustrer expérimentalement ce phénomène, nous proposons une variante de l'algorithme Spectral inspirée par les méthodes d'optimisation convexe permettant d'adapter la dimension de la représentation aux données. Cet algorithme nous permet aussi d'analyser l'effet de la régularisation à la fois lors de l'identification du sous-espace vectoriel et lors de la régression. De plus, l'algorithme proposé montre qu'il est préférable de chercher le sous-espace vectoriel généré par les $\{r_i\}$ contenant conjointement les $\{\dot{\sigma} r_i\}$ lorsque l'on travaille à partir d'une base finie.

Chapitre 4

Séquencement du récepteur superhétérodyne

Sommaire

4.1 Introduction

Dans le chapitre précédent, nous avons proposé un algorithme d'apprentissage régularisé qui automatiquement ajuste la dimension du modèle aux données en réalisant un compromis biais-variance. La régularisation intervient à deux endroits : lors de l'identification du sous-espace vectoriel au travers de la norme nucléaire et lors de la régression des poids de la représentation linéaire au travers de la norme euclidienne. Adapter la complexité du modèle en fonction des données est une problématique majeure dans les algorithmes d'apprentissage en ligne. Le but poursuivi est d'apprendre un modèle simple quand peu de données sont disponibles, afin de généraliser correctement, puis de complexifier celui-ci lorsque les exemples abondent.

Dans ce chapitre, nous allons nous intéresser à une application nécessitant l'apprentissage en ligne de processus contrôlés. La première partie de ce chapitre décrit le domaine des radars et les enjeux en guerre électronique. Puis, la tâche du séquencement du récepteur superhétérodyne est modélisée par l'apprentissage de processus contrôlés. De nombreux détails sont abordés comme : la mise à jour bayésienne en ligne de la croyance avec un modèle qui s'améliore au cours du temps ; l'utilisation de caractéristiques pour modéliser des préfixes et des suffixes longs sans faire exploser la dimension du problème ; et le compromis exploration-exploitation lors de la planification dans des processus contrôlés appris. Enfin, des expériences sur des données simulées montrent l'avantage des techniques d'apprentissage par rapport aux stratégies fixes conventionnelles.

Depuis l'invention du radar dans le premier tiers du XX^{e} siècle, ce système s'est imposé comme le moyen principal de détecter, de localiser et même parfois d'identifier les plateformes (véhicules, avions et bateaux). Le radar envoie une onde radio dirigée vers une zone de l'espace qui à l'encontre d'un objet réfléchissant retourne vers la plateforme émettrice. L'analyse de l'écho permet la détection, la localisation, la mesure de la vitesse et même l'identification de la cible réfléchissante. Dans cette thèse, nous nous concentrons sur l'usage militaire du radar. De plus, nous supposons ces radars monostatiques.

La guerre électronique [Wiley, 2006] consiste à exploiter les émissions radioélectriques (dont celles des radars) d'un adversaire et à l'empêcher de faire de même. Elle englobe donc tous les moyens qui cherchent à acquérir la maîtrise du spectre électromagnétique. Ainsi le général Jean-Paul Siffre intitule son livre [Siffre, 2003] sur la guerre électronique : « La guerre électronique : maître des ondes, maître du monde ». Parmi les moyens utilisés pour maîtriser le spectre électromagnétique, on trouve :

— des mesures de protection telles que les fuselages et matériaux furtifs qui sont très peu réfléchissants ;

— des mesures d'attaque telles que le brouillage qui consiste à envoyer du bruit ou de l'information faussée à l'adversaire sous forme d'onde radar pour le tromper ;

— des mesures de soutien qui exploitent les ondes radars adverses pour renseigner la situation tactique.

L'avantage des mesures de soutien en guerre électronique pour la détection, la localisation et l'identification est que l'horizon de vision est plus élevé que ceux des radars. En effet, l'onde radio doit effectuer un aller-retour pour être reçue par le radar alors que seul l'aller est nécessaire en guerre électronique. Le désavantage des mesures de soutien réside dans sa passivité. La détection n'est possible que si l'adversaire émet. Parmi les mesures de soutien on trouve trois grandes problématiques.

1. En Autoprotection ou *Radar Warning Receiver* (RWR), on souhaite détecter au plus vite les menaces pour lancer une action de protection comme le brouillage, le lancement de leurres ou l'exécution d'une manœuvre évasive permettant d'échapper au pistage adverse.
2. L'*Electronic Support Measures* (ESM) englobe l'autoprotection et a pour but d'établir la situation tactique. En plus de détecter, on souhaite localiser et identifier les menaces. Ces menaces sont *a priori* connues. Un retour visuel de la situation tactique après analyses est affiché au pilote. Cela lui permet de préciser sa connaissance de la situation tactique, d'évaluer le danger et d'adapter ses actions.
3. L'acquisition de Renseignements d'Origine Électro-magnétique ou *ELectronic INTelligence* (ELINT) a pour objectif de recueillir le maximum de renseignements électromagnétiques de sources potentiellement inconnues. Dans ce cadre, aucune action n'est engagée. Les délais de localisation et d'identification ne sont plus une contrainte.

On s'intéresse particulièrement aux problèmes rencontrés en ESM. On cherchera à optimiser le fonctionnement des capteurs pour renforcer la survie du pilote. La partie suivante fait une description technique du problème.

4.2 Description technique

Dans les systèmes d'écoute électromagnétique conçus par Thalès Systèmes Aéroportés (TSA), certains récepteurs sont à large bande et donc couvrent en fréquences entièrement le spectre de tous les émetteurs à surveiller mais sont moins sensibles que les récepteurs à bande étroite dits SHs, qui, quant à eux sont sélectifs en fréquences. En plus d'être sélectifs en fréquences, ils peuvent l'être spatialement (gisement / site), en polarisation et en dynamique. Ils sont donc contraints d'effectuer un balayage dans plusieurs dimensions à fin de couvrir tout le domaine de recherche. La problématique est de concevoir une stratégie d'écoute capable à la fois de détecter au plus tôt les signaux nouveaux et d'analyser ces signaux pour en connaître la nature. La première fois qu'un signal est détecté, une piste correspondant à ce signal est créée. Une fois créées, les pistes des signaux déjà détectés doivent aussi être entretenues. On suppose inscrite dans une bibliothèque avec une plus ou moins grande précision la forme (fréquence, période de répétition des impulsions, largeur d'impulsion, période de rotation de l'antenne, etc) des signaux radars que l'on cherche à intercepter ainsi que leur importance relative.

4.2.1 Réception du signal radar

4.2.1.1 Structure hiérarchique du signal

Un signal radar est composé d'impulsions de courte durée. Ces impulsions sont émises suivant un certain motif qui peut être déterministe ou stochastique. Ce motif se répète avec une plus ou moins longue période. Ceci constitue la première échelle de temps, appelée échelle de temps impulsion.

En plus de ne pas émettre en permanence, un radar émet uniquement dans une direction. Ainsi un récepteur reçoit le signal radar uniquement lorsque le lobe principal de l'antenne radar pointe vers le récepteur. On appelle ce moment un éclairement. Ces

éclairements, d'une durée souvent quasi-constante, se répètent. La répétition dépend du balayage radar qui peut être déterministe ou stochastique. Le balayage mécanique est toujours déterministe alors qu'un balayage électronique peut être stochastique. Ceci constitue la deuxième échelle de temps, appelée échelle de temps éclairement.

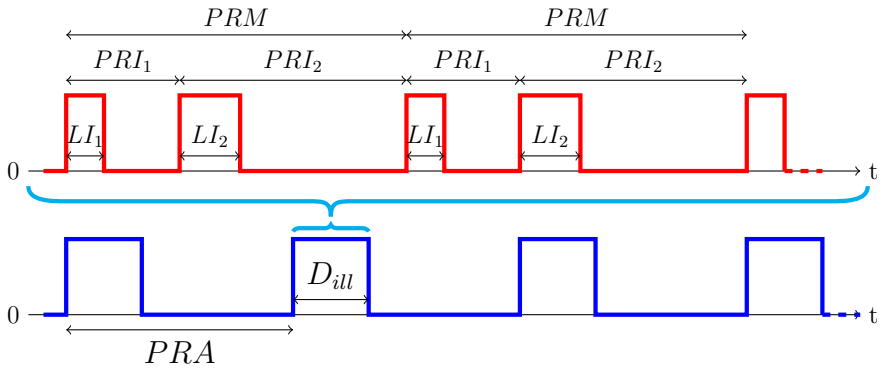


FIGURE 4.1 – En rouge l'échelle de temps impulsion. En bleu l'échelle de temps éclairement. Ici, nous considèrons un balayage mécanique périodique.

Un signal radar, lorsqu'il est reçu, se modélise par un motif répété, que l'on nomme train d'impulsions, enveloppé par le balayage radar comme le montre la Figure 4.1. Prendre en compte cette construction hiérarchique du signal radar est essentiel pour sa détection.

4.2.1.2 Paramètre du signal

On retrouve sur la Figure 4.1 les paramètres suivants,

— LI, la largeur d'impulsion,

— PRI, l'intervalle de répétition des impulsions,

— PRM, la période de répétition du motif,

— D_{ill}, la durée d'illumination ou d'éclairement,

— PRA, la période de rotation d'antenne,

auxquels s'ajoute la fréquence f. La LI, PRI et PRM constituent la Forme d'Onde (FO). L'ensemble de ces paramètres forme un mode. La période de rotation d'antenne ainsi que le type de balayage ne sont pas toujours renseignés dans un mode. De plus, un mode inclut une description qualitative de la menace qu'il représente. L'ensemble des modes constitue une bibliothèque.

4.2.1.3 Incertitude sur les paramètres

Les paramètres sont acquis lors de missions d'ELINT, qui sont soumises aux imprécisions et incertitudes des interceptions. Ainsi, les bibliothèques incluent souvent un intervalle plutôt qu'une unique valeur pour les paramètres. De plus, ces valeurs peuvent changer légèrement d'un radar à un autre du même modèle ou bien dans le temps.

Dans notre modèle, nous considérons que la bibliothèque contient une distribution de probabilité sur la valeur des paramètres radars. À défaut, nous pouvons toujours considérer une distribution uniforme pour un intervalle. L'incertitude sur la PRI peut être d'environ 10% de la PRI. Il en est de même sur la LI.

4.2.1.4 Agilité

En plus de l'incertitude, certains radars changent de fréquence et/ou de PRI plus ou moins régulièrement et plus ou moins aléatoirement. On parle d'agilité temporelle ou fréquentielle. De nombreux schémas d'agilité existent. Particulièrement, on peut distinguer sur la Figure 4.2 trois types d'agilité :

1. Le changement de PRI et/ou de fréquence à chaque impulsion de façon déterministe. Dans ce cas, on a souvent répétition d'un motif périodique. La période du motif s'appelle la PRM.
2. Le changement de PRI et/ou de fréquence à chaque impulsion de façon aléatoire. Dans ce cas, la PRI ne change pas beaucoup et on parle de jitter.
3. Le changement de PRI et/ou de fréquence après un train d'impulsions (64, 128, 256, etc, impulsions).

L'agilité en fréquence est souvent de l'ordre du dixième ou voir du cinquième de la bande passante instantanée du SH et donc peut être gérée facilement en adaptant les sous-bandes de fréquence.

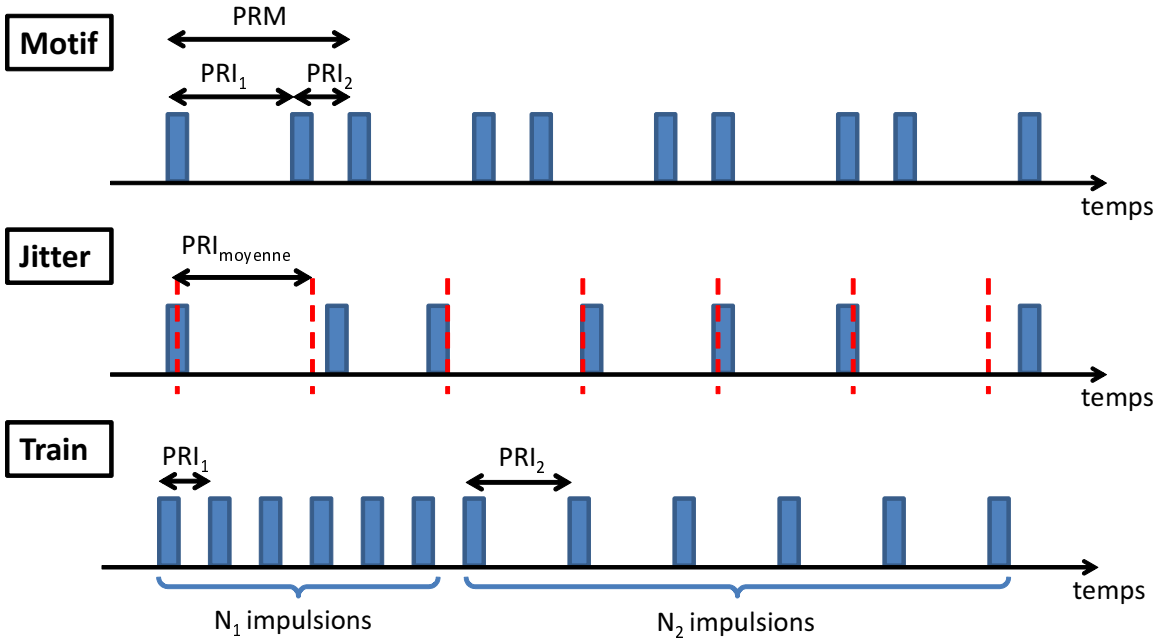


FIGURE 4.2 – Trois principaux types d'agilité temporelle.

4.2.1.5 Problèmes liés à la réception

Le signal radar arrive avec une puissance, mesurée en dB(Watt), plus ou moins élevée. La détection du signal par le récepteur est d'abord conditionnée par le rapport Signal sur Bruit ou *Signal over Noise* (S/N). En effet, si le signal est trop faible il sera confondu avec le bruit. Cela implique une distance maximale (fonction de la puissance d'émission) au-delà de laquelle un signal ne peut être détecté. De plus, la dynamique du récepteur autorise uniquement une certaine plage de puissance. Si la puissance du signal reçu est inférieure à cette plage, le signal est invisible pour le récepteur. Si la puissance est supérieure à cette plage, l'amplificateur du récepteur sature. On peut détecter la saturation mais on ne peut pas analyser le signal s'il y a saturation. En

général, on trouve quatre réglages pour la dynamique. La puissance reçue dépend de nombreux paramètres dont :

— la puissance d'émission,

— la distance entre le radar et le récepteur SH,

— la polarisation des antennes réceptrices,

— l'orientation des antennes réceptrices (gisement et site).

La polarisation des antennes radars favorise la réception des signaux avec la même polarisation et supprime les signaux qui possèdent une polarisation inverse. En général, on préfère une polarisation circulaire (droite ou gauche) qui n'implique pas de perte sur la réception d'un signal de polarisation circulaire (de même sens que celle de l'antenne), une perte de 3 dB pour les polarisations horizontale, verticale, ou diagonale mais une forte perte pour la polarisation circulaire de sens opposé. Donc, le choix de la polarisation influe sur la qualité de la réception.

De plus, le traitement effectué sur le signal après réception peut en diminuer l'intensité. C'est particulièrement le cas, pour ce que l'on appelle la gamme LI. En effet, selon la largeur d'impulsion, il est préférable d'augmenter ou de diminuer la durée des analyses Transformation Rapide de Fourier ou *Fast Fourier Transform* (FFT) du signal. Cela est dû au principe d'incertitude qui implique une borne inférieure à la résolution temps-fréquence. Ainsi, plus la durée d'analyse FFT est longue, plus la résolution fréquentielle est élevée et plus le S/N est fort. Ainsi avec une grande durée d'analyse FFT, on peut détecter des signaux plus faibles. Cependant, lorsque la résolution fréquentielle est élevée, les impulsions courtes sont plus étalées dans le spectre et ne forment plus un pic en intensité. Une grande durée d'analyse FFT ne permet pas toujours de mieux détecter des impulsions courtes qu'avec une résolution plus grossière. Il existe souvent deux configurations de gamme LI.

Enfin, il existe des antennes omnidirectionnelles et des antennes qui ne reçoivent les signaux que dans la direction sélectionnée. Dans le plan horizontal, le gisement correspond à l'angle d'arrivée. La gamme de site correspond au plan transverse (vertical). Les antennes directionnelles possèdent donc plusieurs secteurs.

4.2.1.6 Impact de l'altitude sur la réception

D'une façon générale, les avions ont tendance à voler bas pour ne pas se faire détecter, l'horizon les protégeant. Ainsi, le vol à très basse altitude concerne les missions de pénétration en territoire hostile. D'autre part la basse altitude est également défavorable aux radars de surface en raison de l'interférence destructive entre l'écho et sa réflexion sur le sol. Enfin, les arguments présentés ne concernent pas (ou moins) les radars aéroportés et spatiaux. Cependant à très haute altitude, un avion se met hors de portée des radars au sol car ceux-ci ne sont pas capables de détecter un porteur trop au-dessus. En effet, les radars ont un angle de vision maximum avec le sol. Les seuls radars alors capables (en gisement) de détecter un avion sont alors trop lointains.

4.2.2 Contrôle du récepteur superhétérodyne

Un récepteur SH est, en général, constitué de N voies. Une voie décrit la chaîne de traitement qui suit l'antenne jusqu'à la détection d'impulsion. Les antennes étant directionnelles, il est impératif de posséder plusieurs voies si l'on veut couvrir tout l'espace. Les voies sont affectées dynamiquement aux différents secteurs qui couvrent une partie du domaine spatial.

La chaîne de traitement du récepteur SH est décrite sur la Figure 4.3. Sur certains modèles de récepteur la commutation d'antenne permet de choisir la polarisation en combinant une antenne polarisée horizontale et une antenne polarisée verticale. L'Oscillateur Local (OL), souvent commun à toutes les voies pour des raisons de coût, permet de choisir la bande de fréquence écoutée. Dans la chaîne Radio Fréquence (RF), on peut ajuster la dynamique en modifiant le gain de l'amplificateur. Le signal analogique est ensuite converti en signal numérique. Lors de la FFT, on peut choisir la gamme LI. Enfin, les impulsions sont détectées et les paramètres primaires extraits constituent l'Ensemble des Descripteurs d'Impulsions ou *Pulse Description Word* (PDW). Pour chaque impulsion détectée, le PDW contient la fréquence, la polarisation, la LI, l'amplitude et le Date D'Arrivée ou *Time Of Arrival* (TOA). Les résultats de tous les canaux sont ensuite fusionnés et la Direction D'Arrivée ou *Direction Of Arrival* (DOA) est calculée pour chaque impulsion par goniométrie ou interférométrie.

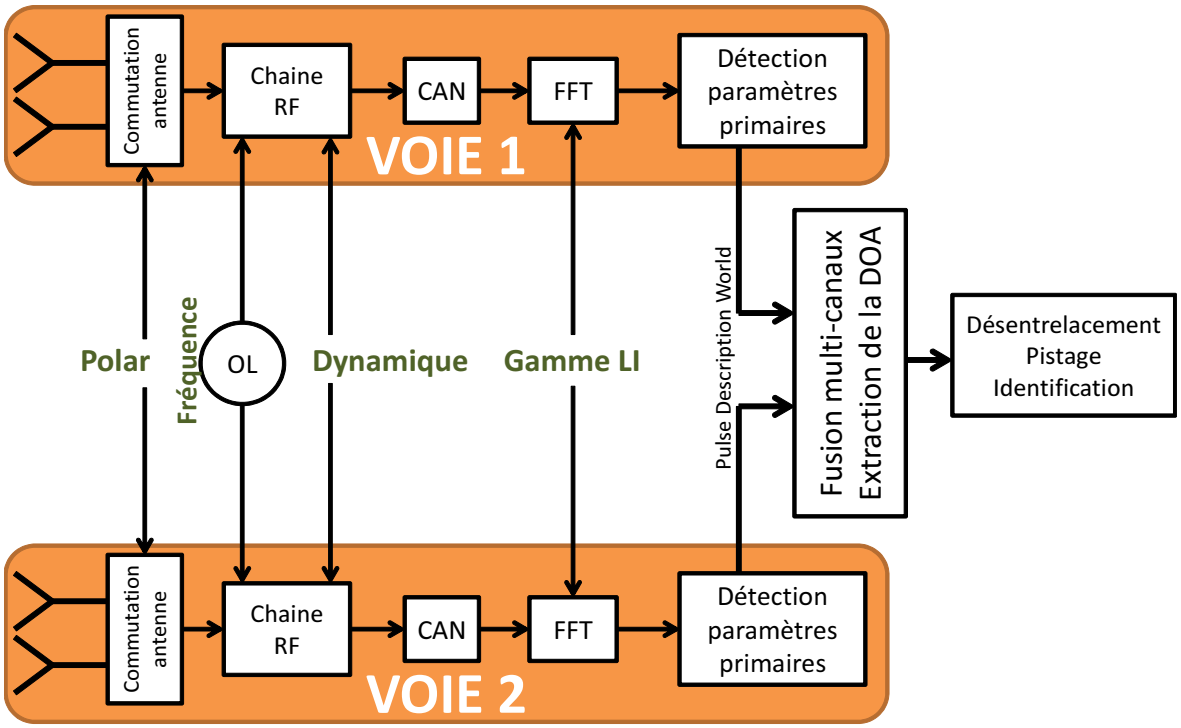


FIGURE 4.3 – Le récepteur SH

Le désentrelacement traite ensuite les impulsions appartenant à la même écoute et les regroupe par trains émis par le même émetteur. La plupart du temps, le critère séparateur est la DOA. Plus rarement, on utilise aussi la FO. Dès lors, des paramètres secondaires sont calculés, tels que la PRI ou la PRM.

A la fin de l'écoute en cours, les trains d'impulsions sont passés au pistage qui les ajoute aux pistes existantes ou en crée des nouvelles. Les pistes sont constituées des trains d'impulsions appartenant aux différentes écoutes passées. Enfin l'identification se charge de reconnaître les pistes grâce à la bibliothèque de modes embarquée.

4.2.2.1 Pilotage des voies

Le pilotage des voies peut donner lieu, en fonction de leur nombre à différentes configurations. Par exemple, il est possible d'obtenir une plus grande dynamique ou bien de faire une analyse de la polarisation.

Théoriquement, toutes les configurations sont possibles, mais des contraintes techniques peuvent limiter le nombre de configurations. On associe, dans la suite du document, une configuration du récepteur à un domaine de l'espace de recherche.

4.2.2.2 Séparation veille et analyse

Une bonne stratégie d'écoute doit donc permettre de balayer rapidement à travers le spectre pour détecter au plus tôt les menaces mais aussi de recueillir de l'information à l'aide d'écoutes longues en vue d'analyser ces signaux. Ainsi une stratégie d'écoute se sépare, usuellement, en deux parties : la veille et l'analyse.

La veille, constituée d'écoutes courtes, a pour but de balayer rapidement le domaine de recherche dans le but de détecter une impulsion. La détection d'une impulsion peut ensuite donner lieu à une analyse. Ainsi la veille est chargée de détecter toute nouvelle émission, attendue ou non, et d'assurer le suivi des pistes jusqu'à leur disparition.

L'analyse, suite à la détection d'une impulsion, a pour objectif de collecter les mesures complémentaires nécessaires à la caractérisation de l'émission (principalement la FO, éventuellement la DOA par interférométrie, le balayage radar). Elle est constituée d'écoutes plus longues. Une seule impulsion est suffisante pour intercepter la DOA. Pour mesurer la FO, la durée de l'écoute minimale est de l'ordre de $3 \times PRI$ ou, selon le système de désentrelacement, de l'ordre de $3 \times PRM$.

4.2.2.3 Activité préparatoire

La plupart des stratégies n'utilisent pas directement la bibliothèque fournie. Elles utilisent la bibliothèque pour définir leur stratégie d'écoute (définie par exemple en termes de contraintes sur les temps de revisite). L'intervention humaine est parfois nécessaire. On appelle cela l'activité préparatoire.

4.2.3 Besoin opérationnel

4.2.3.1 Critique des systèmes fonctionnant avec des bibliothèques

Les bibliothèques ont chacune un format qui dépend du système qui les utilise. Ceci rend leur élaboration très compliquée. De plus, de moins en moins d'experts sont formés pour construire correctement des bibliothèques. Ajoutés aux défauts des missions de renseignements, il en résulte des bibliothèques souvent imprécises, avec des modes manquants ou des valeurs très incertaines qui ne permettent plus de discriminer correctement les émissions.

4.2.3.2 Déroulement d'une mission

La supervision de la mission ou l'aide à cette supervision est une fonction majeure d'un système réactif en ambiance hostile complexe. Le minimum est de s'assurer que la mission peut être poursuivie malgré les aléas rencontrés et fournir le cas échéant des alternatives. Le problème général est

— la sélection d'une route optimale,

— la planification des actions de contre-mesures associées,

— l'optimisation des écoutes (mais aussi des analyses) pour garantir la détection et le traitement à temps des menaces létales.

Nous nous restreindrons à ce dernier problème déjà complexe. On suppose donc les deux premiers problèmes comme résolus.

4.2.3.3 Objectifs de mission

Une mission sert à remplir une fonction. On distingue plusieurs types de mission.

Dans les missions d'ELINT, les avions partent écouter les émissions des radars adverses sans se mettre en danger. Le but est de préciser la quantité, le type et la position des radars adverses.

Pour les missions Air-Air en territoire allié, l'avion vole en hippodrome pour contribuer à empêcher l'adversaire de rentrer. Il faut alors se défendre contre les attaques Air-Air et détecter les avions qui entrent dans la zone. La discrétion est moins importante dans ce type de mission. Mais le RWR est prioritaire.

Dans Les missions Air-Sol, une fois les missions d'ESM effectuées, on peut décider de détruire les systèmes d'armes au sol ou d'autres cibles. Le RWR est toujours très important dans ce type de mission mais celle-ci passe aussi par l'acquisition de renseignements sur la situation tactique. La situation tactique contribue en effet au RWR quand il s'agit de détecter les radars de veille afin de les éviter. Le but de la mission est souvent de se faufiler dans les lignes adverses entre les radars sans se faire voir et d'attaquer une cible après l'avoir localisée si besoin est.

Lors d'une mission un itinéraire favori et des itinéraires bis sont établis en préparation de mission. De même, en préparation de mission, l'opérateur met au point la bibliothèque qui va servir à définir la stratégie de séquencement.

4.2.3.4 Fonctionnement d'un radar

On distingue généralement trois régimes de fonctionnement pour un radar. Chaque régime autorise plusieurs modes d'émission. Les régimes se classent en trois grandes catégories :

— l'acquisition répond au besoin de veille (constitué de 2-3 modes radars) ;
— la poursuite correspond à la fonction de pistage (constitué de 4-5 modes radars) ;
— le tir, comme son nom l'indique, est chargé de guider un missile jusqu'à la cible. Ce régime inclut aussi les auto-directeurs de missiles.

Parfois, ces régimes de fonctionnement sont opérés par des radars différents mais qui fonctionnent de façon coopérative. Le régime de fonctionnement définit le niveau de menace d'un radar ou d'un système de radar.

4.3 État de l'art

4.3.1 Stratégie de veille

Dans cette partie, nous proposons une classification sous divers critères des stratégies de veille. Nous commençons par les méthodes les plus simples et les premières historiquement à être utilisées. Dans le balayage périodique simple, la stratégie balaye toutes les bandes de fréquence avec la même durée d'écoute dans le même ordre. Accessoirement, seules les bandes contenant les modes de la bibliothèque sont inclus dans la stratégie. Afin d'optimiser l'allocation du temps passé dans chaque bande de fréquence, des stratégies plus fines proposent de revenir plus souvent sur les émetteurs difficiles à intercepter.

Malheureusement, la périodicité des écoutes ne se marie pas bien avec la périodicité des trains d'impulsions et la stratégie de veille est sujette à des problèmes de synchronisations. L'analyse de l'interception de deux trains d'impulsions (le train d'écoute peut être considéré comme un train d'impulsion) a été étudiée depuis une soixantaine d'année, d'abord en discrétisant le temps. Richards [1948] a été le premier à publier une analyse détaillée de la probabilité d'interception entre deux trains périodiques. Miller et Schwarz [1953] puis Friedman [1954] et Hawkes [1983] ont analysé le temps d'interception moyen en fonction de la période des impulsions en utilisant des congruences linéaires. En supposant que l'interception entre deux trains est indépendante du pas de temps, Self et Smith [1985] donnent une approximation de la probabilité d'interception quand la phase d'un train est connu. Plus récemment, **?** proposent une analyse exacte de la probabilité d'interception quand une des phases est connue. Enfin, Clarkson et collab. [1996] ont analysé la probabilité d'interception et le temps moyen d'interception entre deux trains dans le cas continu. En formulant un problème d'approximation Diophantienne, ils établissent un lien avec les séries de Farey et généralisent les travaux précédents au cas où les phases sont aléatoires ou non. En utilisant des résultats sur les séries de Farey, Clarkson [2003] donne une méthode pour calculer les temps moyens d'interception du train d'écoute avec chacun des trains d'impulsions et une procédure pour qu'un opérateur choisisse la période de train d'écoute qui minimise le maximum des temps moyens d'interception des émetteurs. Ces résultats ont été étendus dans [Clarkson, 2005] où l'auteur propose d'optimiser aussi les durées d'écoute. Dans sa thèse, Koksal [2010] propose un algorithme pour optimiser un critère différent prenant en compte une probabilité d'interception requise pour chaque mode inférieur à 1. Cette stratégie permet de désengorger la séquence d'écoute de façon optimisée. Sa méthode est issue des travaux de Clarkson [2005].

Un autre type de stratégies ne cherche pas directement une séquence d'écoutes à répéter périodiquement mais plutôt définit des contraintes de revisites à respecter au mieux pour être sûr d'intercepter tous les modes. Nous appelons ces stratégies, balayages à périodes de revisites. En préparation de mission une « trame » de veille est calculée. Elle regroupe les modes nécessitant une configuration du récepteur similaire et associe à chaque groupe

— une configuration du récepteur et les modes qui peuvent être interceptés par celle-ci ;

— une période de revisite égale au minimum, sur les modes couverts par la configuration, de la durée d'illumination diminué de la durée nécessaire d'analyse ;

— une durée d'écoute égale au maximum, sur les modes couverts par la configuration, de la somme de la PRI et de la LI.

La durée d'écoute est donc choisie de façon à être sûr d'intercepter une impulsion si l'écoute est réalisée pendant un éclairement. La période de revisite, si elle est respectée, assure qu'au moins une écoute sera réalisée par éclairement. Nous sommes donc *a priori* sûr d'intercepter tous les modes, comme le montre la Figure 4.4. Le séquencement des écoutes est réalisé par un ordonnanceur du type *Earlier Deadline First* [Xun et collab., 2004]. Pour chaque configuration (ou groupe de modes), on calcule une date limite pour la prochaine écoute égale à la date de la dernière écoute augmentée de la période de revisite. La configuration du récepteur ainsi que la durée pour la prochaine écoute sont celles du groupe de modes dont la date limite est la plus faible. Lorsque que les contraintes de revisites ne peuvent pas être respectées (ce qui est souvent le cas), on dit que la trame de veille est surchargée. Quand c'est le cas, les performances se

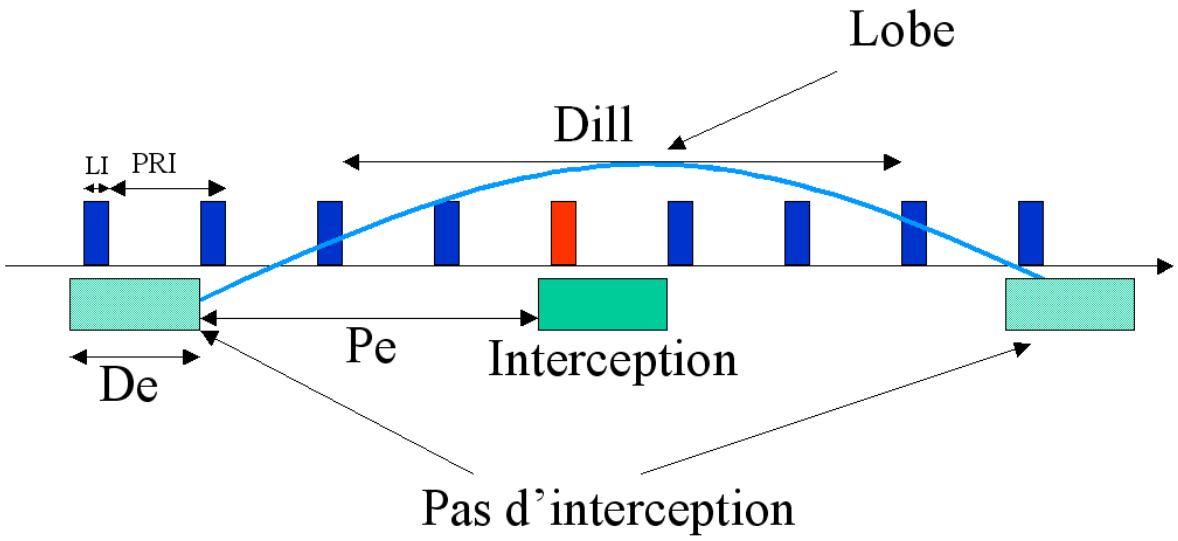


FIGURE 4.4 – Durée d'écoute et période de revisite. En bleu, la puissance du signal reçu.

dégradent arbitrairement selon les modes. En effet, ce genre de stratégies à mémoire finie induit des périodicités dans la séquence d'écoute. Bien que l'insertion d'une écoute d'analyse lors d'une interception casse en partie la périodicité, celle-ci subsiste à plus petite échelle. Dans [Xun et collab., 2004], un ordonnancement dynamique est proposé afin de relâcher certaines contraintes de revisite si la trame de veille est surchargée en fonction des interceptions précédentes. En optimisant en ligne la trame de veille, la méthode proposée permet de garantir une certaine qualité de service. D'autres travaux proposent de relâcher certaines contraintes de revisites pour alléger la trame de veille. Le taux de charge quantifie la charge de la trame de veille. C'est la somme sur les configurations du récepteur des rapports entre la durée d'écoute et la période de revisite. Si le taux de charge est supérieur à 1, la stratégie ne pourra respecter les contraintes de revisites. Par contre, un taux de charge inférieur à 1 ne garantii pas que les périodes de revisites seront respectées. Dans [Dutertre, 2002], les durées d'écoute sont fixées *a priori*, et les périodes de revisite sont choisie de façon à maximiser la moyenne des probabilités d'interception sous la contrainte de ne pas dépasser un taux de charge maximal calculé par simulation. Si la période de revisite est inférieure à la durée d'illumination, la probabilité d'interception vaut 1, sinon elle est supposée égale à la durée d'illumination divisée par la période de revisite. Le taux de charge maximale définit les instances (ensemble de contraintes) pour lesquelles on peut presque sûrement trouver un séquencement et, de plus, pour lesquelles la recherche de ce séquencement est facile. Le calcul du taux de charge maximale est réalisé en préparation de mission en générant des instances aléatoirement et en essayant de trouver un séquencement pour celles-ci. Le séquencement est trouvé grâce à un parcours de graphe en profondeur muni d'une heuristique. On observe une transition brusque entre les instances faisables et les instances sans solutions en fonction du taux de charge. On qualifie le taux de charge de critique quand la transition se réalise. En se plaçant légèrement en dessous du taux de charge critique, on peut efficacement chercher parmi les trames de veille faisables celles qui optimisent un certain critère. Des approches similaires sont proposées dans [Gopalakrishnan, 2012; Gopalakrishnan et collab., 2008].

Lorsque que la période des trains d'impulsion est inconnue, les performances des précédentes stratégies peuvent se dégrader arbitrairement. Ainsi, dans [El-Mahassni et collab., 2004], les auteurs définissent une chaine de Markov à temps continu ou

chaque état représente une configuration du récepteur. La matrice de transition est choisie de façon à obtenir un temps espéré maximal d'interception quasi-linéaire par rapport à la PRA. On suppose l'absence de bibliothèque. Dans ce cas, la linéarité est la propriété optimale pouvant être atteinte. De plus, cette propriété ne peut être atteinte que pour une stratégie stochastique [Clarkson et Pollington, 2007].

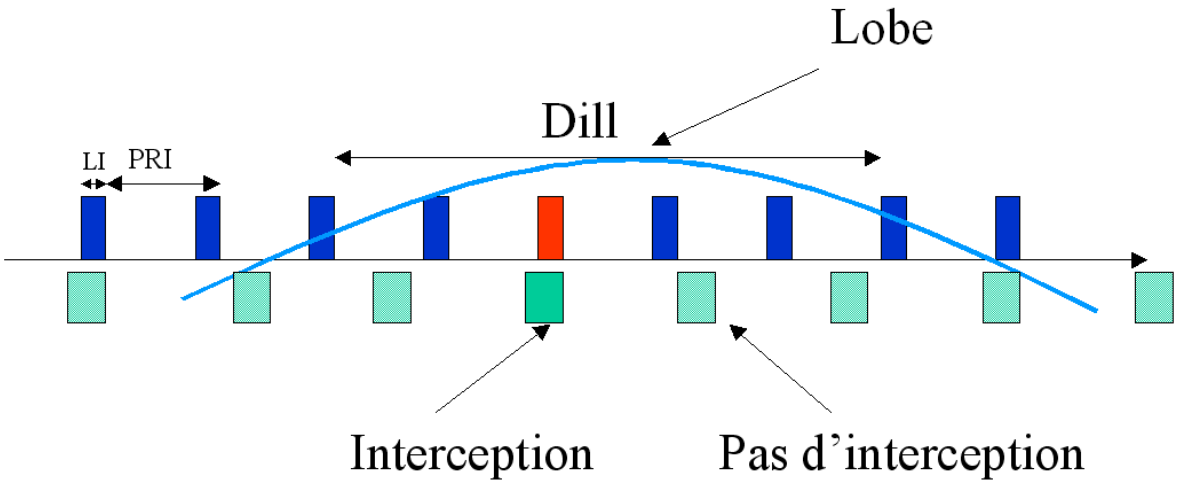


FIGURE 4.5 – Avec des écoutes de durée inférieure à $PRI + LI$, on est plus sûr d'intercepter au moins une impulsion dans un lobe.

Plutôt que d'augmenter la période de revisite, une autre possibilité est de réduire la durée d'écoute, comme le montre la Figure 4.5. Cette stratégie n'impose plus des durées d'écoutes supérieures aux PRI. On n'est donc plus sûr d'intercepter une impulsion par écoute programmée dans une illumination comme précédemment. Cependant, en jouant sur la répétition des écoutes, on peut être sûr d'intercepter une impulsion par plusieurs écoutes successives. Dans cette stratégie, à chaque fin d'écoute, l'algorithme trouve en fonction des écoutes déjà réalisées, la durée d'écoute et la configuration optimale du récepteur. La méthode repose sur un ensemble d'hypothèses définissant un modèle qui permet de calculer la probabilité d'interception (PoI) d'un mode. La PoI, pour une séquence donnée d'écoute et pour un mode, est une fonction du temps t. Elle est égale à la probabilité d'intercepter au moins une impulsion dans un éclairement qui finirait à la date t. De cette fonction on peut calculer l'espérance du pourcentage d'éclairements interceptés par mode. L'algorithme tente à chaque écoute de maximiser la moyenne pondérée des pourcentages d'éclairements interceptés. Cette approche probabiliste permet de définir de nouvelles stratégies de séquencement dont les performances sont meilleures que celles des stratégies actuelles. De plus, elle rend possible la prise en compte d'une importance relative entre les modes en fonction du danger qu'ils représentent. Cette stratégie a fait l'objet d'un brevet par l'auteur de cette thèse [Glaude et Boucard, 2015].

Enfin, de nombreux brevets traitent du séquencement du récepteur SH. La plupart exposent des variantes de la stratégie à périodes de revisite. En voici une liste non-exhaustive : [Gounalis, 2005, 2006a,b, 2007, 2008, 2010; Kunz et collab., 2010; Lewis et Kardatzke, 2000].

4.3.2 Stratégie d'analyse

A notre connaissance, les stratégies d'analyse ont toujours été très simples en se contentant de déclencher une analyse dès qu'une impulsion est interceptée. Ce principe de précaution est excessif car la plupart des passages de lobes n'ont qu'une importance très secondaire en ESM. En effet, une fois un radar détecté et analysé grâce à une première écoute d'analyse réussie, celui-ci doit être suivi régulièrement, uniquement si un changement de mode de sa part représente un danger immédiat ou une information supplémentaire pour la situation tactique. Prenons l'exemple des radars très puissants d'aéroports. Ceux-ci sont visibles de très loin mais ne représentent pas de menace ni d'information utile pour établir la situation tactique. Une fois interceptés, il n'est pas nécessaire de déclencher à nouveau des écoutes d'analyse. D'autres radars, qui peuvent représenter une menace, ne changent pas de mode. Ainsi, les analyser à nouveau n'apporte pas d'informations nouvelles. Tout ce temps gagné peut être réinvesti dans d'autres écoutes d'analyse et dans la stratégie de veille.

Afin d'éviter des écoutes d'analyses non informatives, nous proposons d'utiliser les répétitions prévisibles des illuminations de chaque émetteur pour décider le déclenchement d'une écoute d'analyse. Ce choix est justifié par les exemples suivants. Prenons pour simplifier un radar à balayage mécanique qui éclaire le récepteur de façon périodique dans une bande de fréquence. En apprenant la période du radar, nous pouvons prédire les intervalles de temps où le récepteur sera éclairé. Si l'émetteur est dangereux, nous pouvons déclencher directement une écoute d'analyse avec une configuration du récepteur adaptée à celui-ci. Si l'émetteur n'est pas dangereux, nous empêchons la veille de déclencher une écoute d'analyse ce qui permet de gagner du temps. Par émetteur dangereux, nous désignons aussi les émetteurs qui peuvent changer de mode et passer d'un mode non dangereux (veille) à un mode dangereux (conduite de tir). Nous savons aussi que pendant ce temps les signaux reçus dans cette bande de fréquence seront mélangés au signal de l'émetteur et donc plus difficiles à extraire et à analyser.

Les balayages radars sont souvent plus compliqués que le balayage périodique classique. Il existe par exemple des radars qui possèdent un balayage mécanique sectoriel ou d'autres dont le balayage est réalisé par électronique. Dans ce cas, les radars reviennent régulièrement sur le récepteur mais de façon plus ou moins aléatoire et plus ou moins prévisible. Enfin, même pour les balayages simples, la période de rotation d'antenne est souvent mal connue. Ainsi, pour prédire les intervalles de temps correspondant aux éclairements des différents émetteurs, nous proposons de pister les passages de lobes de chaque émetteur déjà identifié. Comme les séquences d'illuminations ne sont pas aussi régulières que les séquences d'impulsions et pour prendre en compte la variété des balayages radars, nous modélisons celles-ci par des processus contrôlés, comme le décrit la prochaine section.

4.4 Prédiction des prochains passages de lobe

4.4.1 Description du problème

Pour simplifier l'exposé, nous nous limitons au choix de la bande de fréquence pour la configuration du récepteur. Le temps est discrétisé en intervalles réguliers de durée approximativement égale à une illumination. Ainsi, soit N radars évoluant dans K bandes de fréquence, nous notons $a_t \in [1, K]$ la bande de fréquence écoutée à l'instant

t. Nous notons R_k l'ensemble des radars présents dans la bande k. De plus, pour tout $i \in [1, N]$, nous notons $o_t^i \in [0, 1]$ l'observation du radar i au temps t ($o_t^i = 1$ si une illumination a été interceptée, 0 sinon). De même, nous notons $a_t^i \in [0, 1]$ l'action pour le radar i au temps t ($a_t^i = 1$ si l'écoute au temps t aurait pu intercepter le radar i, 0 sinon). Nous avons ainsi, $a_t^i = \mathbb{1}_{R_{a_t}}(i)$, qui vaut 1 si le radar i est contenu dans l'ensemble R_{a_t} des radars couverts par l'écoute a_t. Le but est d'apprendre pour chaque radar i le processus contrôlé suivant $\mathbb{P}\left(o_t^i, o_{t-1}^i, \ldots \middle| a_t^i, a_{t-1}^i, \ldots\right)$. Nous pouvons ensuite prédire par raisonnement bayésien la probabilité d'intercepter une illumination du radar i si l'on écoute la bonne bande de fréquence à l'instant $t+1$ sachant les écoutes précédentes. Nous notons cette probabilité

$$\begin{aligned} p_{t+1}^i &= \mathbb{P}\left(O_{t+1}^i = 1 \middle| A_{t+1}^i = 1, a_t^i, a_{t-1}^i, \ldots, o_t^i, o_{t-1}^i, \ldots\right) \\ &= \frac{\mathbb{P}\left(O_{t+1}^i = 1, o_t^i, o_{t-1}^i, \ldots \middle| A_{t+1}^i = 1, a_t^i, a_{t-1}^i, \ldots\right)}{\mathbb{P}\left(o_t^i, o_{t-1}^i, \ldots | a_t^i, a_{t-1}^i, \ldots\right)}. \end{aligned}$$

Nous remarquons que les actions n'influent que sur l'observabilité du système. Celui-ci est intrinsèquement non contrôlé (les écoutes ne changent pas le balayage radar).

Afin de montrer qu'il est réalisable d'apprendre à prédire les prochains passages de lobes, nous nous concentrons sur l'objectif simplificateur consistant à intercepter le maximum d'éclairements. Le but est donc de concevoir une stratégie qui, en balayant les bandes de fréquences, apprend les récursivités dans les éclairements reçus et les exploite pour intercepter les futurs passages de lobes. Toujours dans un esprit de simplification, nous supposons qu'un éclairement est toujours intercepté si le récepteur est dans la bonne bande de fréquence au bon moment et que l'on sait associer parfaitement les éclairements reçus aux radars les ayant émis. Finalement, le récepteur ne réalise que des écoutes d'analyse de durée constante.

4.4.2 Lien avec la radio cognitive

On trouve en radio cognitive le problème dual. Des utilisateurs primaires, au nombre de N, d'un réseau radio communiquent aux travers de K canaux. Un groupe d'utilisateurs secondaires, pour simplifier au nombre de 1, cherchent à communiquer sur le réseau sans gêner les utilisateurs primaires. Pour ce faire, il doit émettre lorsque les utilisateurs primaires n'utilisent par leur canal. Ainsi, au lieu de repérer quand le spectre est occupé comme en guerre électronique, nous cherchons à prédire quand le spectre est inoccupé.

Ce problème peut être formulé comme un problème de bandit manchot [Anandkumar et collab., 2011; Liu et Zhao, 2010], où chaque canal est modélisé comme un processus où l'occupation est indépendante et identiquement distribuée dans le temps. Dans des travaux plus avancés [Tekin et Liu, 2012], les auteurs supposent que l'occupation d'un canal évolue selon une MC. Supposons que le score d'une stratégie soit égal au nombre d'émissions dans un canal non occupé. On peut alors mesurer la qualité d'une stratégie par le regret. Le regret d'une stratégie est défini par l'espérance de la différence entre le score associé à celle-ci et le score associé à une stratégie similaire calculée en ayant la connaissance parfaite de la statistique des canaux.

À première vue, on pourrait penser que les solutions développées pour la radio cognitive pourraient s'appliquer au séquencement du récepteur SH. Cependant, les premiers travaux se concentrent principalement sur les stratégies fixes qui choisissent le meilleur canal. Bien que la stratégie change de canal pour estimer leurs statistiques,

celle-ci finit par se concentrer vers le meilleur canal. Le regret est d'ailleurs calculé par rapport à la stratégie qui choisit le canal le moins occupé et émet en permanence dans celui-ci. De meilleurs résultats pourraient être obtenus par des stratégies dynamiques balayant les fréquences pour émettre à chaque pas de temps dans un canal libre. Ainsi, dans [Ortner et collab., 2014; Tekin et Liu, 2011], les auteurs développent la notion de regret fort correspondant à ces stratégies dynamiques. En supposant que l'occupation des canaux évolue de manière Markovienne, ils proposent des algorithmes ayant un faible regret. Cette fois le regret est mesuré par rapport à la stratégie dynamique calculée en ayant connaissance des paramètres des MCs de tous les canaux.

Bien qu'étroitement liés au séquencement du récepteur SH, ces travaux ne peuvent pas être appliqués à notre problème car, l'état d'un radar n'est que partiellement observable au travers de l'occupation des bandes de fréquence. Les canaux ne peuvent pas être modélisés par des MCs. Par exemple, pour un balayage mécanique, ne rien observer dans la bande de fréquence à un pas de temps ne permet pas de déduire la phase entre l'antenne du radar et le récepteur. Ainsi, les observations ne sont pas Markoviennes. Il convient alors de modéliser chaque canal par un processus stochastique partiellement observable, par exemple un HMM. Cette particularité rend en fait le problème bien plus difficile.

4.4.3 Apprentissage

4.4.3.1 Modélisation par des processus contrôlés

Nous supposons que les radars peuvent être modélisés par des processus contrôlés rationnels sur $\mathbb{R}$, autrement dit des PSRs. Soit $(\boldsymbol{\alpha}_0^i, A^i, \boldsymbol{\alpha}_\infty^i)$ la représentation linéaire du PSR associé au radar i, nous avons alors que

$$\mathbb{P}\left(o_t^i, \dots, o_1^i \middle| a_t^i, \dots, a_1^i\right) = {\boldsymbol{\alpha}_0^i}^\top A^i_{a_1^i, o_1^i} \dots A^i_{a_t^i, o_t^i} \boldsymbol{\alpha}_\infty^i.$$

Afin d'obtenir p_{t+1}^i pour tout t, nous procédons par filtrage bayésien. Nous notons $\boldsymbol{\alpha}_t^i$ la croyance associée au radar i au temps t et définie récursivement par

$${\boldsymbol{\alpha}_t^i}^\top = \frac{{\boldsymbol{\alpha}_{t-1}^i}^\top A^i_{a_t^i, o_t^i}}{{\boldsymbol{\alpha}_{t-1}^i}^\top A^i_{a_t^i, o_t^i} \boldsymbol{\alpha}_\infty^i}.$$

Nous pouvons alors calculer

$$p_{t+1}^i = {\boldsymbol{\alpha}_t^i}^\top A^i_{1,1} \boldsymbol{\alpha}_\infty^i.$$

De cette façon, nous pourrions efficacement calculer p_{t+1}^i pas à pas. Pour l'apprentissage, nous proposons d'utiliser l'algorithme Spectral régularisé proposé au Chapitre 3 basé sur le SVT. En effet, celui-ci permet d'adapter la taille du modèle appris aux données. De plus, utilisant la régularisation, nous pouvons rendre l'inférence plus robuste quand peu de données sont disponibles, c'est-à-dire au début de l'apprentissage. Dans les expériences, la valeur singulière de seuillage μ est choisie par rapport à la seconde plus grande valeur singulière σ_2. Plus précisément, $\mu = \frac{\sigma_2}{50}$. Le paramètre de régularisation λ lors de la régression linéaire a été choisie par validation croisée.

4.4.3.2 Utilisation de caractéristiques

Détaillons la première étape de l'algorithme consistant à l'estimation des matrices de Hankel. Dans cette partie, pour alléger les notations, nous n'utilisons pas l'indice i

désignant un radar particulier. Afin d'obtenir une base complète, utiliser des préfixes et des suffixes dont la longueur est plus grande que la dimension du système est suffisant comme le souligne la Proposition 3. Or un radar à balayage mécanique parfait peut être modélisé par un HMM dont le nombre d'états cachés correspond à la période de celui-ci. Ainsi prendre des préfixes et des suffixes supérieurs à cette période est une bonne façon d'obtenir une base complète. Cependant le nombre de séquences possibles d'une certaine longueur augmente exponentiellement avec celle-ci. Ainsi plutôt que d'estimer les probabilités conditionnelles, nous choisissons de travailler à partir de caractéristiques. En effet, Boots et collab. [2010b] montrent que l'on peut aussi bien travailler avec une matrice de Hankel estimée à partir des caractéristiques des séquences observées. Nous notons $\tilde{H}$ cette nouvelle matrice telle que,

$$\tilde{H} = {\Phi^{\mathcal{P}}}^{\top} H \Phi^{\mathcal{S}},$$

où $\Phi^{\mathcal{P}} \in \mathbb{R}^{\mathcal{P}\times p}$ et $\Phi^{\mathcal{S}} \in \mathbb{R}^{\mathcal{S}\times s}$, contiennent en ligne les caractéristiques de taille respectives p et s associées respectivement aux préfixes et aux suffixes. Par commodité, nous utilisons des préfixes et des suffixes de longueur fixe. Comme les actions n'apportent que peu d'informations sur la dynamique du système, nous choisissons de ne pas les utiliser pour construire les caractéristiques. Plus précisément, une interception implique qu'une écoute a été réalisée. De plus, les actions ne changent pas l'état des radars. Ainsi, un préfixe ou un suffixe de longueur l $(a_1, o_1)\dots(a_l, o_l)$ est décrit par le vecteur,

$$\phi = (o_1, \dots, o_l)^{\top}.$$

Soit $l = p = s$ la longueur des suffixes et des préfixes utilisés par l'algorithme, nous notons au temps t

$$\begin{aligned}\phi_t^{\mathcal{P}} &= (o_{t-l}, \dots, o_t)^{\top},\\ \phi_t^{\mathcal{S}} &= (o_{t+1}, \dots, o_{t+l+1})^{\top}.\end{aligned}$$

Ainsi, nous avons

$$\begin{aligned}\tilde{H}_{\mathcal{B}} &= \mathbb{E}\left[\phi_t^{\mathcal{P}} {\phi_t^{\mathcal{S}}}^{\top} \middle| a_{t-l}, \dots, a_{t+l+1}\right],\\ \tilde{H}_{\mathcal{B}}^{a,o} &= \mathbb{E}\left[\phi_t^{\mathcal{P}} {\phi_{t+1}^{\mathcal{S}}}^{\top} \mathbb{1}_{o_t}(o) \middle| a_{t-l}, \dots, a_{t+l+2}\right],\\ \mathbf{h}^{\mathcal{P}} &= \mathbb{E}\left[\phi_t^{\mathcal{P}} \middle| a_{t-l}, \dots, a_t\right],\\ \mathbf{h}^{\mathcal{S}} &= \mathbb{E}\left[\phi_t^{\mathcal{S}} \middle| a_{t+1}, \dots, a_{t+l+1}\right].\end{aligned}$$

En général, la plupart des travaux sur l'apprentissage de PSRs envisagent un apprentissage hors-ligne et supposent que la politique ayant généré les actions est aléatoire et uniforme. Dans notre cas, nous ne pouvons pas faire cette hypothèse et l'estimation des matrices de Hankel doit être adaptée comme le remarquent Bowling et collab. [2006]. Nous notons $\pi_t = \mathbb{P}(A_t = 1)$ la probabilité à l'instant t qu'une écoute soit réalisée. Étant donné que le système est non-contrôlé, les observations sont indépendantes des actions prises à des temps différents. De plus, comme l'apprentissage est réalisé à partir d'une trajectoire unique de longueur non bornée, nous utilisons l'algorithme *suffix-history* de Wolfe et collab. [2005]. Ainsi, soit une trajectoire de

longueur T, en notant

$$\hat{\phi}_t^{\mathcal{P}} = \left(\frac{o_{t-l}}{\pi_{t-l}}, \dots, \frac{o_t}{\pi_t} \right)^\top,$$
$$\hat{\phi}_t^{\mathcal{S}} = \left(\frac{o_{t+1}}{\pi_{t+1}}, \dots, \frac{o_{t+l+1}}{\pi_{t+l+1}} \right)^\top,$$

on obtient les estimateurs non-biaisés suivant

$$\hat{H}_{\mathcal{B}} = \sum_{t=l+1}^{T-l-1} \hat{\phi}_t^{\mathcal{P}} \hat{\phi}_t^{\mathcal{S}\top},$$
$$\hat{H}_{\mathcal{B}}^{a,o} = \sum_{t=l+1}^{T-l-2} \hat{\phi}_t^{\mathcal{P}} \hat{\phi}_{t+1}^{\mathcal{S}\top} \mathbb{1}_{o_t}(o),$$
$$\hat{\mathbf{h}}^{\mathcal{P}} = \sum_{t=l+1}^{T} \hat{\phi}_t^{\mathcal{P}},$$
$$\hat{\mathbf{h}}^{\mathcal{S}} = \sum_{t=0}^{T-l-1} \hat{\phi}_t^{\mathcal{S}}.$$

Notons que, comme nous avons pris $p = s = l$, nous obtenons $\mathbf{h}^{\mathcal{P}} = \mathbf{h}^{\mathcal{S}}$.

4.4.4 Apprentissage en ligne

Cette partie aborde les problèmes liés à l'apprentissage en ligne. Comme pour la section précédente, l'indice i indiquant un radar particulier n'est pas utilisé. Premièrement, à cause de la longueur des préfixes et des suffixes, une phase d'initialisation est nécessaire avant de récolter suffisamment de données pour commencer l'apprentissage. Pendant cette phase, une stratégie d'écoute aléatoire et uniforme est utilisée. Cette phase d'initialisation pourrait être éventuellement remplacée en utilisant un modèle *a priori* pour le radar.

Ensuite, pour des raisons de complexité calculatoire, les paramètres du modèle sont régulièrement mis à jour avec les nouvelles observations après un certain nombre de pas de temps. La mise à jour des matrices de Hankel estimées est immédiate. Pour le reste des paramètres, une procédure similaire à celle proposée dans [Boots et Gordon, 2011] peut être utilisée pour alléger la complexité calculatoire.

À chaque fois que les paramètres sont mis à jour, la croyance courante $\boldsymbol{\alpha}_t$ doit être projetée dans le nouveau sous espace vectoriel appris. Soit $(\hat{\boldsymbol{\alpha}}_0^{(t)}, \hat{A}^{(t)}, \hat{\boldsymbol{\alpha}}_\infty^{(t)})$ la représentation linéaire apprise et $\hat{V}^{(t)}$ les vecteurs singuliers de droite identifiés par SVT au temps t, alors nous pouvons réécrire l'équation de mise à jour de la croyance

$$\boldsymbol{\alpha}_t{}^\top = \frac{\boldsymbol{\alpha}_{t-1}^\top \hat{V}^{(t-1)\top} \hat{V}^{(t)} A_{a_t,o_t}^{(t)}}{\boldsymbol{\alpha}_{t-1}^\top \hat{V}^{(t-1)\top} \hat{V}^{(t)} A_{a_t,o_t}^{(t)} \boldsymbol{\alpha}_\infty^{(t)}}.$$

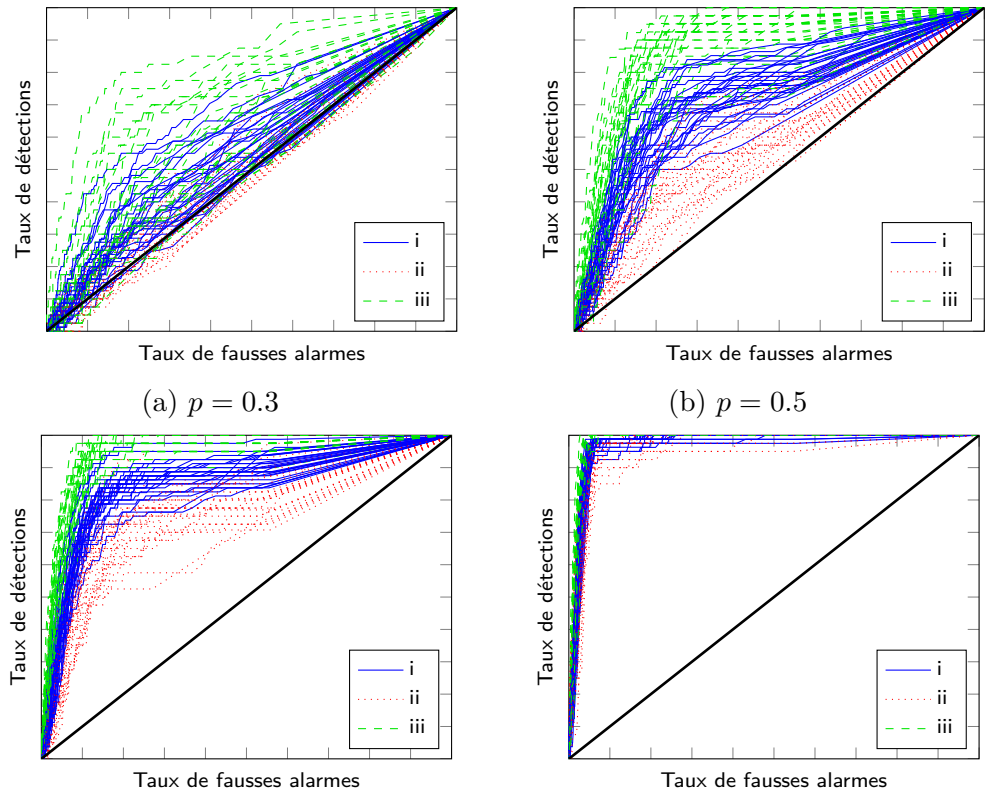
4.5 Expériences

4.5.1 Prédiction pour un radar

Afin de tester la capacité de prédiction du modèle appris, nous travaillons avec un seul radar. Le signal radar est quasi-périodique, comme pour un balayage mécanique.

Plus exactement, les passages de lobes sont espacés d'une période moyenne plus un bruit de 10%. La période moyenne est de 50. Les éclairements du radar sont interceptés à travers un processus de Bernoulli de paramètre p. Ainsi, à chaque pas de temps, une écoute est réalisée avec une probabilité p.

Après une période d'initialisation de 300 pas, les paramètres du PSR sont mis à jour tous les 50 pas de temps. Dans cette expérience nous évaluons la qualité de la prédiction à un pas sachant que l'on écoute. Les résultats sont donnés sous forme de Fonction d'Efficacité du Récepteur ou *Receiver Operating Characteristic* (ROC) représentant le taux de détections en fonction du taux de fausses alarmes. Ces taux sont calculés sur trois parties d'une trajectoire de 4000 pas : (i) sur toute la trajectoire, (ii) sur la première moitié, (iii) sur la seconde moitié. Les résultats, présentés Figure 4.6, montrent que le taux de détections s'améliore avec le nombre d'éclairements déjà interceptés (que ce soit entre les trajectoires (i) et (iii) ou en fonction du paramètre p). D'une part, cela montre que le modèle est capable d'apprendre les incertitudes dans le balayage radar. D'autre part, lorsque plusieurs passages de lobes successifs ne sont pas interceptés le modèle est quand même capable de prédire les suivants.



(a) $p = 0.3$ (b) $p = 0.5$

(c) $p = 0.8$ (d) $p = 1$

FIGURE 4.6 – Courbes ROC pour différents paramètres p. Chaque courbe représente une simulation sur les 30 réalisée.

4.5.2 Stratégie d'écoute

Pour la deuxième expérience nous avons évalué la planification et l'apprentissage en ligne. Neuf radars sont simulés dans trois bandes de fréquences différentes. La période moyenne de chaque radar est tirée uniformément entre 40 et 80. Le temps entre deux passages de lobes successifs est tiré uniformément dans un intervalle de 5% autour de

la période moyenne. Pour l'apprentissage des PSRs associés aux radars, la taille des préfixes et des suffixes est de 240 et la période d'initialisation est de 480. Pendant cette période d'initialisation la stratégie d'écoute est aléatoire et uniforme. Puis, les modèles sont mis à jour tous les 50 pas avec les nouvelles observations collectées.

Après la période d'initialisation, nous proposons d'utiliser une stratégie d'écoute inspirée des travaux sur les bandits manchots [Auer et collab., 2002] qui réalise un compromis entre exploration et exploitation. En effet, cette stratégie mélange, selon le paramètre γ, une exploration uniforme des bandes de fréquences et une politique d'exploitation à un pas de Gibbs de paramètre T. Ainsi, nous avons

$$\mathbb{P}\left(A_t = k\right) = \frac{(1-\gamma)\exp\left(\frac{\gamma}{T}\sum_{i\in R_k} p_t^i\right)}{\sum_{j=1}^{K}\exp\left(\frac{\gamma}{T}\sum_{i\in R_j} p_t^i\right)} + \frac{\gamma}{K}.$$

Le but étant d'intercepter le plus d'illuminations, nous mesurons le taux d'interception à chaque pas de temps depuis le début de la simulation. Nous ne pouvons ainsi qu'espérer au mieux une convergence asymptotique vers 1 si les éclairements dans des bandes différentes n'ont jamais lieu en même temps. Chacune des 30 simulations réalisées dure 10000 pas de temps. La Figure 4.7 montre le taux d'interception moyen et un intervalle de confiance à 95% calculé à partir d'une distribution Normale. La ligne en pointillés rouges correspond aux performances moyennes de la stratégie d'écoute uniforme. Les paramètres utilisés pour la politique sont $\gamma = 0.2$ et $T = 0.001$.

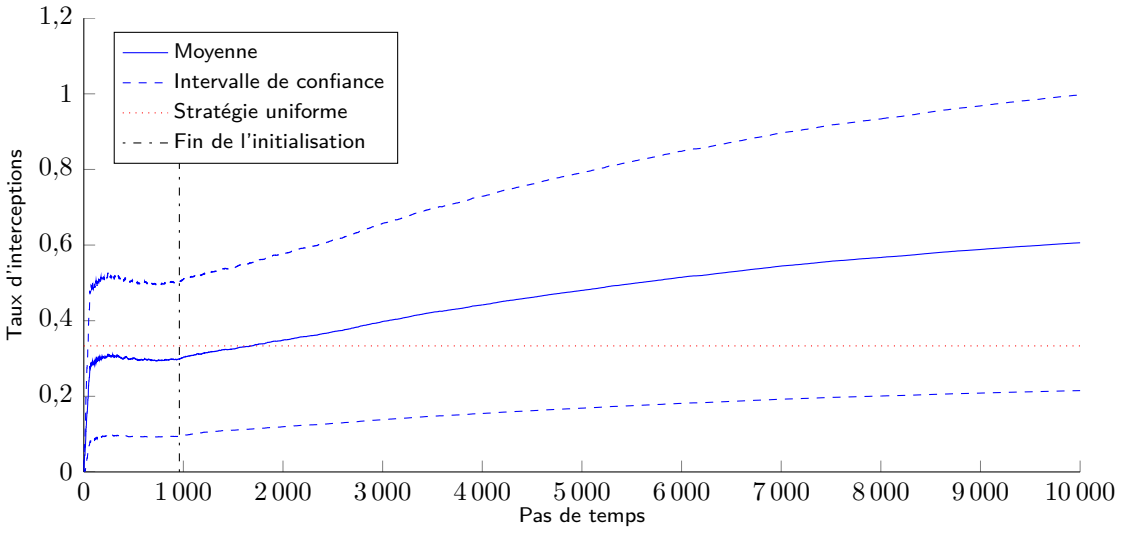


FIGURE 4.7 – Taux d'interception en fonction de temps.

Ces résultats montrent que l'apprentissage en ligne est efficace et permet d'obtenir une stratégie en boucle fermée qui utilise les interceptions passées pour intercepter plus d'éclairements dans le futur. Bien que notre expérience consiste en un scénario très simple, elle permet de montrer que les performances s'améliorent avec le temps grâce à l'apprentissage, par rapport à une stratégie en boucle ouverte (ici, la stratégie uniforme).

Troisième partie

Apprentissage par enveloppe convexe

Chapitre 5

Apprentissage spectral non-négatif

Sommaire

5.1 Introduction

Au chapitre 3, nous avons vu que l'algorithme Spectral et plusieurs de ses variantes étaient consistants pour l'inférence de langages stochastiques rationnels sur $\mathbb{R}$. De plus, les algorithmes proposés réalisent un apprentissage PAC. Ces garanties s'étendent naturellement aux processus stochastiques. Pour les processus contrôlés, en fonction de la politique utilisée pour générer les exemples d'apprentissage, nous pouvons obtenir le même genre de garanties. Malheureusement, nous avons aussi remarqué que pour ces algorithmes l'apprentissage est impropre. C'est-à-dire que les valeurs retournées par la série apprise ne sont pas forcément positives et ne somment pas à 1 comme pour une distribution de probabilité. Ce défaut, détaillé à la Section 1.4.4, provient du fait qu'il est indécidable de savoir si un $\mathbb{R}$-MA réalise une série positive. Dans cette introduction, nous allons détailler deux propriétés qui peuvent causer une erreur mal contrôlée dans la série estimée. Nous verrons que la divergence de la série estimée peut causer une erreur absolue exponentiellement grande mais que celle-ci peut être contrôlée par normalisation. Puis, nous expliquerons qu'une erreur relative peut néanmoins subsister à cause de la non-robustesse de la représentation linéaire des $\mathbb{R}$-MA stochastiques.

Analysons d'abord la probabilité jointe d'une séquence. Celle-ci est un polynôme des coefficients de la représentation linéaire. Lorsque les coefficients sont estimés, l'erreur sur la probabilité d'une séquence de longueur l augmente exponentiellement avec l dans le cas général. Cette propriété est due à la divergence probable de la série estimée. Au Chapitre 3, nous avons expliqué que si le nombre d'exemples servant à l'apprentissage était suffisamment grand, la série estimée était néanmoins absolument convergente avec forte probabilité. Cependant, ces garanties théoriques ne sont pas suffisantes en pratique car l'on ne dispose pas forcément d'un nombre d'exemples suffisants. De plus, ce nombre d'exemples n'est pas toujours calculable.

Pire, dans le cas des distributions conditionnelles, l'erreur peut devenir très grande, dès lors que l'on conditionne par rapport à des évènements très peu probables. Prenons, l'exemple des processus stochastiques ou des processus contrôlés pour lesquels le filtrage bayésien est très utilisé dans les applications. Dans les applications, on souhaite souvent prédire la prochaine observation sachant les observations précédentes (et les actions prises dans le cas des processus contrôlés). Le calcul de la distribution conditionnelle se fait naturellement par la règle de Bayes. Soit $o_{1:t}$ la séquence d'observations jusqu'au temps t, et $(\boldsymbol{\alpha}_0, A, \boldsymbol{\alpha}_\infty)$ une représentation linéaire apprise du processus stochastique observé, alors nous avons

$$\mathbb{P}\left(o_{t+1}|o_{1:t}\right) = \frac{\mathbb{P}\left(o_{1:t+1}\right)}{\mathbb{P}\left(o_{1:t}\right)},$$

avec $\mathbb{P}\left(o_{1:t+1}\right) = \boldsymbol{\alpha}_0^\top A_{o_{1:t+1}} \boldsymbol{\alpha}_\infty$ et $\mathbb{P}\left(o_{1:t}\right) = \boldsymbol{\alpha}_0^\top A_{o_{1:t}} \boldsymbol{\alpha}_\infty$. Ainsi, les erreurs d'estimation de la représentation linéaire se répercutant sur l'estimation du dénominateur peuvent causer une forte instabilité dans la distribution conditionnelle. Si $\mathbb{P}\left(o_{1:t}\right)$ est très faible, l'erreur d'estimation peut facilement rendre le dénominateur négatif ou, pire, nul. Du côté des analyses non-asymptotiques, Hsu et collab. [2012] relèvent des difficultés pour borner l'erreur sur la distribution conditionnelle, en particulier à cause des séquences d'observations peu probables. Afin de minorer le dénominateur, les auteurs font l'hypothèse que les probabilités de transitions sont elles aussi minorées.

Afin d'obtenir des probabilités à partir de valeurs potentiellement négatives et ne sommant pas à 1, plusieurs heuristiques ont été proposées pour obtenir des distributions de probabilités sur des ensembles finis d'événements. On peut, par exemple, remplacer

les valeurs négatives par zéro, ou bien prendre la valeur absolue, puis normaliser localement. Dans la plupart des cas, obtenir des distributions pour des ensembles finis d'événements, par exemple sur le prochain symbole dans la séquence, est suffisant. En fait, cette normalisation locale permet de s'affranchir du problème de la divergence de la série.

Cependant, nous constatons que ces heuristiques ne sont pas entièrement satisfaisantes pour la raison suivante. Pour les $\mathbb{R}$-MA, comme les coefficients des matrices de transitions peuvent être négatifs ou positifs, une erreur, même très faible, peut causer un changement de signe entrainant, par multiplications et additions successives, une erreur relative très grande entre les probabilités des évènements de la distribution bien que celles-ci soient normalisées. Autrement dit, la représentation n'est pas robuste, comme précisé en Section 1.4.4. Pour résumer, ce n'est pas un problème de divergence car nous effectuons une normalisation locale mais plutôt un problème d'erreur relative.

Partant de ce constat, nous proposons de se limiter à l'inférence de langages stochastiques rationnels sur $\mathbb{R}^+$-MA. En proposant, un algorithme d'apprentissage, appelé NNSpectral, retournant uniquement des $\mathbb{R}^+$-MA, nous condamnons tout changement de signe dans les paramètres estimés. De même, nous pouvons alors retrouver des distributions sur des ensembles finis d'événements par normalisation locale. L'espoir est qu'en bannissant les changements de signes des coefficients, l'erreur relative soit quelque peu contrôlée. L'ensemble des $\mathbb{R}^+$-MA, dont fait l'objet ce chapitre, est représenté dans la hiérarchie d'automates établie au Chapitre 1 sur la Figure 5.1.

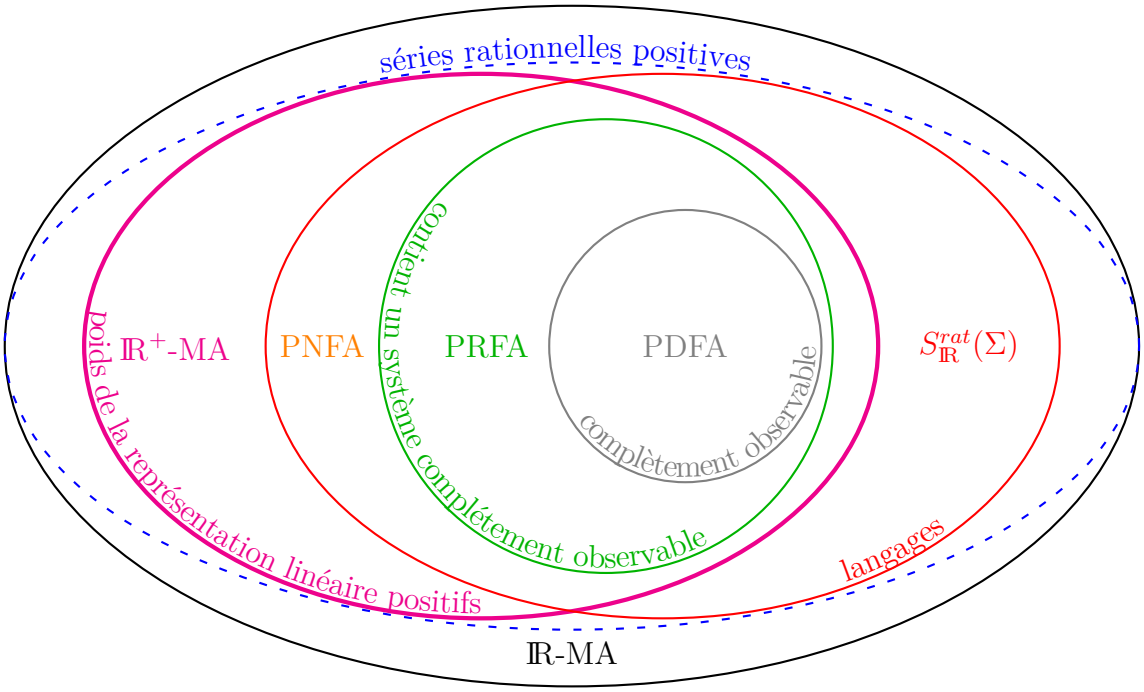


Figure 5.1 – Hiérarchie entres les classes d'automates. Dans ce chapitre, nous nous intéressons à la classe d'automate en magenta épais.

Enfin, nous mentionnons ici que l'algorithme NNSpectral est conçu pour apprendre indifféremment tout type de séries formelles rationnelles sur $\mathbb{R}^+$, en particulier les processus stochastiques et les processus contrôlés. Nous limitons l'explication et les expériences aux langages stochastiques mais le principe est strictement le même.

5.2 Recherche de sous semi-modules

5.2.1 Algorithme NNSpectral

Dans cette section, nous présentons l'algorithme NNSpectral pour l'inférence de langages stochastiques rationnels sur $\mathbb{R}^+$. NNSpectral est inspiré de l'algorithme Spectral, et plus généralement de la MoM, dans le sens où il calcule d'abord une estimation $\hat{H}_{\mathcal{B}}$ de la matrice de Hankel $H_{\mathcal{B}}$ définie sur une base finie. L'algorithme pourrait très bien être dérivé pour une matrice de Hankel infinie comme l'algorithme Spectral. Cependant, pour les applications, il est souvent pratique, voire nécessaire, de limiter la dimension de H en prenant une base finie. De plus, pour simplifier la présentation de l'algorithme sans vraiment contraindre les applications, nous supposons que la base contient le mot vide ($(\varepsilon, \varepsilon) \in \mathcal{B}$).

À la différence de Spectral qui cherche une décomposition en facteurs de faible dimension à coefficients dans $\mathbb{R}$ proche de $\hat{H}_{\mathcal{B}}$ en norme de Frobenius, NNSpectral contraint les coefficients de la décomposition à être non-négatifs. Autrement dit, nous cherchons une décomposition en facteurs de faible dimension non-négatifs approchant $\hat{H}_{\mathcal{B}}$ en norme de Frobenius. Trouver une telle décomposition est un problème bien connu, appelé NMF, qui a reçu beaucoup d'intérêt depuis son introduction par Lee et Seung [1999]. Les algorithmes de NMF et leurs propriétés seront décrites dans la section suivante. À partir de la factorisation $\tilde{H}_{\mathcal{B}} = PS \approx \hat{H}_{\mathcal{B}}$, nous pouvons former une famille finie $\{r_1, \ldots, r_d\}$ avec les lignes de S, où d est la dimension de la factorisation. Par la non-négativité de P et S, les $\{r_1, \ldots, r_d\}$ génèrent un sous semi-module de $\mathbb{R}^{+\mathcal{S}}$ contenant $\tilde{p} = \mathbf{1}_\varepsilon^\top \tilde{H}_{\mathcal{B}}$ la série estimée.

Afin d'appliquer la transformation décrite par la Proposition 4, il faut estimer $\{\dot{\sigma} r_1, \ldots, \dot{\sigma} r_d\}$. Avant de décrire comment NNSpectral procède, nous mettons en avant une différence fondamentale avec le cas des décompositions sur $\mathbb{R}$.

Nous sortons du cadre de l'inférence pour se placer dans le cas idéal d'une matrice de Hankel H infinie parfaitement connue représentant une série r. Nous posons $PS = H$ une décomposition exacte sur $\mathbb{R}^+$. Soit $\{r_1, \ldots, r_d\}$ les séries représentées par les lignes de S, alors $\{r_1, \ldots, r_d\}$ génère bien un sous semi-module contenant la série r. Cependant, rien ne garantit que les séries de $\{\dot{\sigma} r_1, \ldots, \dot{\sigma} r_d\}$ soient contenues dans $[[\{r_1, \ldots, r_d\}]]$ comme pour les décompositions sur $\mathbb{R}$. Bien que, la linéarité de $\dot{\sigma}$ implique que les séries de $\{\dot{\sigma} r_1, \ldots, \dot{\sigma} r_d\}$ soient contenues dans l'espace vectoriel généré par $\{r_1, \ldots, r_d\}$, le cône décrit par $[[\{\dot{\sigma} r_1, \ldots, \dot{\sigma} r_d\}]]$ n'est pas forcément contenu dans le cône $[[\{r_1, \ldots, r_d\}]]$. Autrement dit, le sous semi-module $[[\{r_1, \ldots, r_d\}]]$ n'est pas nécessairement stable.

Revenons à l'estimation de $\{\dot{\sigma} r_1, \ldots, \dot{\sigma} r_d\}$ dans le cas des matrices de Hankel finies. Pour l'inférence de $\mathbb{R}$-MA, nous aurions estimé $\{\dot{\sigma} r_1, \ldots, \dot{\sigma} r_d\}$ par régression linéaire au sens des moindres carrés entre $\hat{H}^\sigma_{\mathcal{B}}$ et $\hat{P}$ (voir Chapitre 2). Ici, pour s'assurer que les $\{\dot{\sigma} r_1, \ldots, \dot{\sigma} r_d\}$ sont bien des séries sur $\mathbb{R}^+$, nous procèdons par Moindres Carrés Non-Negatifs, ou *Non-Negative Least Squares* (NNLS).

Enfin, la dernière étape consiste à estimer la représentation linéaire. Comme pour tout $\sigma \in \Sigma$, $\{\dot{\sigma} r_1, \ldots, \dot{\sigma} r_d\}$ n'a pas la garantie d'appartenir à $[[\{r_1, \ldots, r_d\}]]$, il faut une fois de plus procéder par NNLS pour retrouver $\hat{A}$. Pour les poids initiaux, le résultat est donné directement par $\mathbf{1}_\varepsilon^\top P$ car $\varepsilon \in \mathcal{S}$. Les poids finaux sont donnés par $(r_1(\varepsilon), \ldots, r_d(\varepsilon))$. Finalement, nous avons bien la non-négativité de tous les poids. Les étapes de l'algorithme NNSpectral sont résumées dans Algorithme 2.

Algorithme 2 Algorithme NNSpectral

Entrées Un alphabet Σ, un ensemble de mots d'apprentissage, un rang estimé d
Sortie Un $\mathbb{R}^+$-MA $\langle \Sigma, \{1..d\}, \left\{\hat{A}_\sigma\right\}_{\sigma\in\Sigma}, \hat{\boldsymbol{\alpha}}_0, \hat{\boldsymbol{\alpha}}_\infty\rangle$

1: Choisir une base de préfixes $\mathcal{P} \subset \Sigma^\star$ et de suffixes $\mathcal{S} \subset \Sigma^\star$ contenant ε
2: Estimer $\hat{H}_\mathcal{B}$ et $\forall\sigma \in \Sigma$, $\hat{H}^\sigma_\mathcal{B}$
3: $\hat{P}, \hat{S} \leftarrow \operatorname{argmin}_{\substack{\hat{P}\in{\mathbb{R}^+}^{\mathcal{P}\times d}\\ \hat{S}\in{\mathbb{R}^+}^{d\times\mathcal{S}}}} \left\|\hat{H}_\mathcal{B} - \hat{P}\hat{S}\right\|_F$
4: $\hat{\boldsymbol{\alpha}}_0^\top \leftarrow \mathbf{1}_\varepsilon^\top \hat{P}$
5: $\hat{\boldsymbol{\alpha}}_\infty \leftarrow \hat{S}\mathbf{1}_\varepsilon$
6: **pour tout** $\sigma \in \Sigma$ **faire**
7: $\quad \hat{R}_\sigma \leftarrow \operatorname{argmin}_{R_\sigma\in{\mathbb{R}^+}^{d\times\mathcal{S}}} \left\|\hat{P}R_\sigma - \hat{H}^\sigma_\mathcal{B}\right\|_F$
8: $\quad \hat{A}_\sigma \leftarrow \operatorname{argmin}_{A_\sigma\in{\mathbb{R}^+}^{d\times d}} \left\|A_\sigma\hat{S} - \hat{R}_\sigma\right\|_F$
9: **fin pour**

5.2.2 Factorisation en matrices non-négatives

La NMF est un outil largement utilisé pour réduire la dimension d'attributs non-négatifs. L'objectif est de décomposer une matrice non-négative $X \in \mathbb{R}^{n\times m}$ telle que $X \approx WH$ où $W \in \mathbb{R}^{n\times r}, W \geq 0$ et $H \in \mathbb{R}^{r\times m}, H \geq 0$. À la différence de la SVD, la NMF est reconnue pour extraire des attributs parcimonieux et significatifs. Ces propriétés ont prouvé leur utilité dans le traitement d'images pour l'extraction d'attribut du visage [Lee et Seung, 1999] et dans l'exploration de textes pour la classification des documents [Xu et collab., 2003]. Pour la NMF, plusieurs fonctions de perte ont été envisagées, comme la distance KL, où la norme de Frobenius. Ici, nous nous intéressons au problème suivant,

$$\min_{W\in\mathbb{R}^{n\times r}, H\in\mathbb{R}^{r\times m}} \|X - WH\|_F \text{ tels que } W \geq 0, H \geq 0.$$

Malheureusement, la NMF est un problème **NP**-dur en général [Vavasis, 2009] et la décomposition non unique en fait un problème mal posé. En pratique, on utilise des heuristiques qui, au mieux, ont la garantie de converger vers un point stationnaire. La plupart d'entre elles s'exécutent en $\mathcal{O}(nmr)$ pas de temps. Le lecteur peut se référer à [Gillis, 2014b] pour un comparatif des heuristiques les plus classiques. Dans nos expériences, nous avons utilisé les NNLS alternés avec projection du gradient de Lin [2007]. Cette méthode présente un bon compromis entre qualité de la solution et performance en optimisant de façon alternée W et H par NNLS. Plusieurs méthodes d'initialisation ont été proposées. Nous avons choisi dans les expériences une initialisation déterministe basée sur la SVD proposée par Boutsidis et Gallopoulos [2008].

5.2.3 Moindres carrés non-négatifs

Les NNLS sont une version contrainte du problème des moindres carrés, définie par

$$\operatorname*{argmin}_{X\in\mathbb{R}^{+n\times m}} \|AX - B\|_F,$$

où $A \in \mathbb{R}^{p\times n}$ et $B \in \mathbb{R}^{m\times p}$. Étant équivalent à un problème de programmation quadratique, le problème est convexe et les contraintes forment un ensemble convexe,

ce qui garantit la convergence en temps polynomial vers un point stationnaire. De plus tous les points stationnaires sont de valeurs égales [Lőtstedt, 1983]. Si A n'est pas de rang de plein, on peut à partir de n'importe quelle solution calculer la solution de norme minimale par programmation quadratique. Cette solution est unique et donc le problème est bien-défini. Nous utilisons le solveur quadratique Mosek [MOSEK, 2015] pour trouver la solution. Bien que un peu plus lent que d'autres algorithmes spécialisés pour les NNLS, les résultats étaient très légèrement meilleurs pour les systèmes mal conditionnés.

5.3 Expériences

Dans cette section, nous détaillons les ensembles de données et les critères sur lesquels sont évalués les algorithmes proposés dans les Chapitres 5 à 7.

5.3.1 Trois ensembles de données

5.3.1.1 Compétition PAutomaC

La compétition PAutomaC [Verwer et collab., 2012], proposée dans la conférence ICGI 2012, propose de résoudre un ensemble de 48 problèmes dont le but est d'apprendre un modèle génératif de séquences de symboles, celles-ci ayant été générées par différents types de systèmes séquentiels linéaires. Ces modèles peuvent être de trois types : PNFA, HMM et PDFA. Afin de produire une comparaison objective, nous avons sélectionné les douze mêmes problèmes que dans [Balle et collab., 2014b]. Cet ensemble est composé de quatre modèles de chaque type.

5.3.1.2 Corpus Penn-Treebank

Le corpus Penn-Treebank [Marcus et collab., 1993] pour l'étiquetage morpho-syntaxique ou *part-of-speech tagging* contient un ensemble de 48825 phrases en anglais. À chaque mot d'une phrase est associé sa nature grammaticale parmi 11 grandes classes. L'ensemble d'entrainement est constitué des séquences de natures grammaticales. La tâche consiste à apprendre le langage stochastique ayant généré ces séquences.

5.3.1.3 Wikipédia

Wikipédia est un projet d'encyclopédie universelle, libre et multilingue. Nous avons utilisé un bloc, tiré au hasard, des 2 Go de texte brut extrait par Sutskever et collab. [2011] des pages anglaises de Wikipédia. Puis nous avons partitionné ce bloc en séquences de 250 caractères chacune. Le nombre de symboles différents a été réduit à 85 en prenant les lettres, majuscules et minuscules, les nombres et la ponctuation. Cet alphabet contient aussi un symbole représentant les caractères du corpus non-inclus dans l'alphabet.

Pour l'apprentissage, deux blocs ont été extraits : un petit de 409 séquences et un plus grand de 41943 séquences. Pour des blocs plus grands, nous n'avons pas observés de changements significatifs dans les performances. Pour l'évaluation, nous avons un même bloc composé de 5188 séquences. La tâche consiste à apprendre le processus stochastique ayant générés les séquences supposées extraites d'une séquence de taille infinie (contrairement à l'apprentissage de langages stochastiques). Nous supposons donc que les distributions stationnaire et initiale du processus stochastique sont égales.

5.3.2 Critères d'évaluation

5.3.2.1 Perplexité

La qualité d'un automate M peut être mesurée par la distribution de probabilité qu'il réalise p_M. L'objectif est alors d'apprendre un automate M dont la série est proche de la distribution, notée p, ayant généré l'ensemble d'apprentissage, noté $\mathcal{T}$. La qualité de p_M peut être mesurée par la perplexité. Celle-ci correspond au nombre moyen de bits nécessaire pour représenter un mot en utilisant le code donné par p_M. La définition exacte est la suivante,

$$\text{Perplexité}(\mathcal{M}) = 2^{-\sum_{u\in\mathcal{T}} p(u)\log(p_M(u))}.$$

Une faible perplexité signifie que M prédit bien les exemples générés par p.

5.3.2.2 Taux d'erreur de mots

La qualité d'un modèle peut aussi être évaluée par sa capacité à prédire le prochain symbole dans un mot. Le critère que l'on utilise est alors le WER. L'appellation peut porter à confusion dans notre cas, car ici les mots sont les symboles de l'alphabet. Le WER est le taux d'erreur sur la prédiction du prochain symbole sachant les précédents. La moyenne est calculée sur tous les symboles de tous les mots de $\mathcal{T}$. Soit W le WER, on a

$$W = \frac{1}{|\mathcal{T}|}\sum_{u\in\mathcal{T}}\frac{1}{|u|}\sum_{t=1}^{|u|}\left(1 - \mathbb{1}_{\text{argmax}_\sigma\, p_M(\sigma|\sigma_{1:t-1})}(\sigma_t)\right).$$

5.3.2.3 Vraisemblance conditionnelle

Un critère très proche du précédent est la vraisemblance moyenne de la prochaine observation conditionnée au passé. Celle-ci est donnée par

$$L = \frac{1}{|\mathcal{T}|}\sum_{u\in\mathcal{T}}\frac{1}{|u|}\sum_{t=1}^{|u|} p_M(\sigma_t|\sigma_{1:t-1}).$$

Nous remarquons que si p_M est une distribution de probabilité alors

$$\mathbb{E}\left[\mathbb{1}_{\text{argmax}_\sigma\, p_M(\sigma|\sigma_{1:t-1})}(\sigma_t)\right] = p_M(\sigma_t|\sigma_{1:t-1}),$$

et donc $L = 1 - \mathbb{E}[W]$. Nous utilisons la vraisemblance pour évaluer les algorithmes sur Wikipédia comme cela a déjà été fait dans [Kulesza et collab., 2015b]. Dans chaque séquence de test de 250 symboles, les 50 premiers symboles ne sont pas utilisés pour évaluer la vraisemblance conditionnelle mais servent au calcul de la vraisemblance conditionnelle des observations suivantes.

5.3.2.4 Nombre moyen de bits par caractère

Toujours dans l'idée de mesurer la qualité de la distribution conditionnelle sur le prochain symbole, on peut aussi utiliser une mesure d'information. Le nombre de BPC B est défini par

$$B = \frac{1}{|\mathcal{T}|}\sum_{u\in\mathcal{T}}\frac{1}{|u|}\sum_{t=1}^{|u|} -\log_2\left(p_M(\sigma_t|\sigma_{1:t-1})\right).$$

A la différence de la vraisemblance conditionnelle qui effectue une moyenne arithmétique, le BPC est calculée en prenant la moyenne des logarithmes qui représente le logarithme de la moyenne géométrique. Celle-ci est moins sensible aux fortes valeurs que la moyenne arithmétique. Ainsi, la qualité de prédiction des symboles rares influera plus le BPC que la vraisemblance conditionnelle. Nous utilisons le BPC pour évaluer les algorithmes sur Wikipédia comme cela a déjà été fait dans [Sutskever et collab., 2011]. Dans chaque séquence de test de 250 symboles, les 50 premiers symboles ne sont pas utilisés pour évaluer le BPC mais servent au calcul du BPC des observations suivantes.

5.3.3 Algorithmes utilisés en comparaison

Dans les expériences, nous avons comparé NNSpectral à d'autres algorithmes basés sur la MoM, ainsi qu'à l'algorithme de BW.

En plus de l'algorithme Spectral, les algorithmes basés sur la MoM utilisés sont celui de Balle et collab. [2012b] basé sur l'optimisation convexe et noté CO et celui de Anandkumar et collab. [2012a] basé sur la décomposition de tenseurs, noté Tenseur. L'algorithme Tenseur a d'abord été formulé pour les processus stochastiques mais Balle et collab. [2014b] l'a étendu pour travailler avec des langages stochastiques. Pour l'algorithme CO et Spectral, les modifications entre la version pour les processus stochastiques et la version pour les langages stochastiques sont mineures. Pour l'algorithme Tenseur et CO, une partie du code de Balle et collab. [2014b] a été réutilisé.

Pour l'algorithme de BW, nous avons utilisé une implémentation très optimisée disponible dans la bibliothèque treba. L'algorithme de BW accroit à chaque itération la vraisemblance de la représentation linéaire sachant les observations. Étant sujet à rester coincé dans un maximum local, l'algorithme de BW est d'abord lancé 3 fois à partir d'initialisations aléatoires pour 3 itérations. Ensuite, le meilleur modèle est sélectionné et jusqu'à 30 itérations de l'algorithme sont exécutées.

Dans les expériences conduites, les critères évalués ne sont pas la vraisemblance des paramètres. Ainsi, nous observons parfois que ces critères se dégradent après un certain nombre d'itérations de BW bien que la vraisemblance des paramètres ne fasse que de croitre. Ainsi, entre chaque itération, nous vérifions si celles-ci améliorent bien le score final. Si après 4 itérations le score n'a fait que de se dégrader, l'algorithme s'arrête et renvoie le meilleur modèle trouvé jusque-là. En pratique, l'algorithme n'exécute que très rarement les 30 itérations qui lui sont autorisées ce qui indique qu'il a convergé vers un des optimums.

5.4 Implémentation

5.4.1 Estimation de séries auxiliaires

De nombreux algorithmes d'apprentissage, comme Spectral, NNSpectral, Tenseur et CO sont capables d'apprendre n'importe quel type de séries formelles rationnelles, en particulier des langages stochastiques rationnels. Les algorithmes issus de la MoM, se basent sur l'estimation de la matrice de Hankel de la série à partir des mots servant à l'apprentissage. Pour les langages stochastiques, il faut donc estimer pour chaque mot u se décomposant sur la base de préfixes et de suffixes ($\exists v \in \mathcal{P}$, $w \in \mathcal{S}$, $vw = u$), sa probabilité d'occurrence $p(u)$. Pour l'estimer, on compte

le nombre de fois où u est présent dans l'ensemble d'apprentissage $\mathcal{T}$ et on divise par la taille de l'ensemble. On obtient ainsi,

$$\hat{p}(u) = \frac{1}{|\mathcal{T}|} \sum_{v \in \mathcal{T}} \mathbb{1}_u(v) .$$

On remarque qu'à partir de p, on peut définir d'autres séries dont la matrice de Hankel est parfois plus simple à estimer à partir des données. Le $\mathbb{R}$-MA appris modélisant la série dérivée peut ensuite être transformée pour réaliser la série originale p. Ainsi, Balle [2013] propose deux autres séries dérivées de p. La série basée sur les préfixes est définie par $p^{prefixe}(u) = p(u\Sigma^\star) = \sum_{v \in \Sigma^\star} p(uv)$. La série basée sur les sous-séquences est définie par $p^{sous}(u) = \sum_{w,v \in \Sigma^\star} p(wuv)$. Dans [Balle, 2013, Lemme 6.1.1], l'auteur montre que lorsque p est un langage stochastique rationnel dont la représentation linéaire $(\boldsymbol{\alpha}_0, A, \boldsymbol{\alpha}_\infty)$ vérifie $\|A_\Sigma\| < 1$, où $\|\cdot\|$ est sous-multiplicative, alors $p^{prefixe}$ et p^{sous} sont rationnelles. De plus, il fournit une conversion explicite qui préserve le nombre d'états entre les $\mathbb{R}$-MA qui réalisent les deux séries dérivées et la série originale. Par conséquent, pour l'apprentissage, on peut travailler avec l'une de ces trois séries. Bien que les séries apprises par Spectral, NNSpectral, CO et Tenseur, ne vérifient pas en général $\|A_\Sigma\| < 1$, l'application de la transformation donne quand même de bons résultats. Pour NNSpectral, lorsque l'apprentissage est réalisé par estimation d'une série auxiliaire, le modèle retourné n'a plus la garantie d'être un $\mathbb{R}^+$-MA.

Dans les expériences, nous avons utilisé $\hat{p}$ (la série originalement décrite par le langage stochastique) et $\hat{p}^{sous}$ (la série décrivant les statistiques des sous-séquences). Travailler avec $\hat{p}^{sous}$ permet une meilleure efficacité statistique car le nombre de sous-séquences est plus grand que le nombre de séquences. Cependant, l'estimation est plus longue et la matrice de Hankel est moins parcimonieuse ce qui alourdit les calculs subséquents.

5.4.2 Choix de la base

Il existe de nombreuses approches pour choisir une base de préfixes et de suffixes en espérant quelle soit suffisamment riche pour identifier un sous semi-module stable contenant la série cible. Par exemple, on peut prendre tous les préfixes et suffixes qui apparaissent dans l'ensemble d'entrainement. Lorsque l'ensemble d'entrainement est trop grand, on peut limiter la taille de la base en limitant la longueur maximale des préfixes et suffixes. Cependant, utiliser des préfixes et des suffixes courts peut conduire à une perte d'information sur la dynamique à long terme de la série. En effet, dans ce cas, les séquences longues ne sont pas comptabilisés. Une autre possibilité est de sélectionner les préfixes et les suffixes les plus fréquents. De cette façon, chaque séquence de symboles contribue, en moyenne, à l'estimation d'un nombre plus élevé de coefficients dans la matrice $\hat{H}_\mathcal{B}$, renforçant la structure dans celle-ci. Mais cela peut conduire à l'utilisation de préfixes et de suffixes trop courts car beaucoup plus fréquents que les longs. Néanmoins cette heuristique donne de bons résultats en pratique, c'est pourquoi nous l'avons utilisée dans nos expériences. Lorsque l'on travaille à partir des statistiques des sous-séquence (voir section précédente), nous utilisons la même base pour préfixes et les suffixes. Celles-ci sont constituées à partir des sous-séquences les plus fréquentes. Dans tous les cas, Nous avons à chaque fois inclus dans la base, le suffixe et le préfixe vide ε.

Des bases plus ou moins grandes ont été utilisées en fonction de l'algorithme d'apprentissage, de l'ensemble d'entrainement et de la série estimée. Dans toutes les

expériences les bases ont été constituées d'un même nombre de suffixes et de préfixes. Nous nous référons à ce nombre en tant que taille de la base.

Pour PAutomaC et Penn-Treebank, nous avons utilisé pour SPECTRAL et TENSEUR des bases de tailles $\{200, 400, 1000, 4000, 10000\}$ pour $\hat{p}$ et $\{50, 100, 150, 200, 500\}$ pour $\hat{p}^{sous}$. Afin de limiter le temps de calcul pour NNSPECTRAL des bases plus petites ont été utilisées, de taille $\{50, 100, 150, 200, 500\}$ pour $\hat{p}$ et $\hat{p}^{sous}$. Pour CO, des bases encore plus petites ont été utilisées à la fois pour des raisons calculatoires et pour des raisons de performances. L'ensemble des tailles utilisées pour CO est $\{50, 100, 150, 200\}$ pour $\hat{p}$ et $\hat{p}^{sous}$.

Pour Wikipédia, nous avons sélectionné moins de tailles différentes pour les bases. Pour SPECTRAL et TENSEUR, des bases de 500 et de 10000 mots ont été utilisées. Pour NNSPECTRAL, une base de 500 mots offre un bon compromis performance versus temps de calcul. Pour CO, le temps de calcul devenait prohibitif pour des bases de tailles supérieures à 200. Nous avons donc utilisé une base de 200 mots.

5.4.3 Normalisation de la variance

En suivant les conseils de Cohen et collab. [2013], nous avons normalisé la variance des probabilités estimées en multipliant les lignes et les colonnes de $\hat{H}_{\mathcal{B}}$ par un facteur $c_u = \sqrt{|\mathcal{S}| / (\#(u) + 5)}$, où $\#(u)$ est le nombre d'occurrences de u dans l'ensemble de la formation. Cette procédure améliore légèrement les performances. Nous avons appliqué cette normalisation uniquement pour les algorithmes SPECTRAL et NNSPECTRAL. Comme cette normalisation ajoute un léger biais dans l'estimation, nous ne l'avons pas appliqué sur Wikipédia car la variance des préfixes et des suffixes était déjà bien équilibrée et assez faible.

5.4.4 Taille des modèles

Pour chaque expérience et chaque algorithme, nous avons appris des modèles de dimensions différentes. Le résultat reporté est celui du meilleur modèle. Les dimensions utilisées sur PAutomaC et Penn-Treebank sont $\{2, 6, 10, 14, 18, 22, 26, 30, 34, 38, 42, 46\}$. Pour Wikipédia, les résultats sont reportés pour toutes les dimensions allant de 10 à 100. Pour TENSEUR, l'algorithme échoue pour les dimensions supérieures à 30. Pour l'algorithme de BW, le temps de calcul devient prohibitif pour des modèles de dimensions supérieures à 60.

Pour CO, le modèle possède la même taille que la base. Nous avons alors essayé différentes valeurs pour le paramètre de régularisation τ. Ce paramètre ajuste la complexité du modèle dans l'optimisation suivante (voir Chapitre 2) :

$$\hat{A}_{\Sigma} = \operatorname*{argmin}_{A} \left\| \hat{H}_{\mathcal{B}} A - \hat{H}^{\Sigma}_{\mathcal{B}} \right\|_F + \tau \|A\|_*.$$

Les valeurs de τ utilisés pour PAutomaC et Penn-Treebank sont répertoriées dans laTableau 5.1.

$6.31 \cdot 10^{-2}$	$2.51 \cdot 10^{-2}$	$1 \cdot 10^{-2}$	$3.98 \cdot 10^{-3}$	$1.58 \cdot 10^{-3}$	$6.31 \cdot 10^{-4}$
$2.51 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$3.98 \cdot 10^{-5}$	$1.58 \cdot 10^{-5}$	$6.31 \cdot 10^{-6}$	$2.51 \cdot 10^{-6}$

TABLEAU 5.1 – Valeurs de τ, dans CO ; pour PAutomaC et Penn-Treebank.

$6.31 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1.58 \cdot 10^{-5}$	$2.51 \cdot 10^{-6}$
$3.98 \cdot 10^{-7}$	$6.31 \cdot 10^{-8}$	$1 \cdot 10^{-8}$	$1.59 \cdot 10^{-9}$

TABLEAU 5.2 – Valeurs de τ, dans CO pour Wikipédia.

Pour Wikipédia, nous avons utilisé les valeurs de la Tableau 5.2

Pour l'algorithme SPECTRAL, la SVD n'est calculée qu'une seule fois avec la dimension maximale puis tronquée pour les dimensions inférieures. Malheureusement, les algorithmes NNSPECTRAL, CO et TENSEUR ne permettent pas d'utiliser une astuce similaire et le sous semi-module doit être estimée pour chaque taille de modèle.

5.4.5 Mesure du temps de calcul

Pour évaluer la complexité algorithmique des différents algorithmes, nous avons mesuré le temps de calcul de chacune des trois étapes. D'abord, le temps nécessaire à l'estimation de la matrice de Hankel à partir des données a été mesuré pour chaque algorithme en fonction de la série utilisée. Puis, nous avons mesuré le temps pris par l'identification du sous semi-module. Enfin, nous avons noté le temps pris par l'estimation des paramètres par régression.

Ainsi, l'étape 2 correspond à la SVD pour l'algorithme SPECTRAL, à la décomposition du Tenseur pour TENSEUR, à la NMF pour NNSPECTRAL. Pour CO, nous avons considéré que le temps de calcul de l'algorithme était réparti uniquement sur l'étape 1 et 3. Pour l'algorithme de BW, le temps de calcul est entièrement inclus dans l'étape 3. Nous avons moyenné les temps de calculs sur les différents problèmes pour PAutomaC et sur les différentes tailles de modèles. Ainsi les temps donnés correspondent à une exécution complète de l'algorithme sur un problème moyen. Lorsque l'identification du sous semi-module peut être réalisée en une seule fois pour toutes les tailles de modèle (comme pour SPECTRAL, voir section précédente), nous avons pris comme temps de référence celui-là.

5.4.6 Apprentissage de processus stochastiques

Pour le problème d'inférence du texte brute extrait de Wikipédia, un processus stochastique est mieux adaptée qu'un langage stochastique car la séquence de symboles est supposée infinie. Pour les algorithmes utilisés dans ce chapitre, seul l'estimation de matrice de Hankel doit être adaptée. Nous rappelons que l'apprentissage est réalisé à partir de séquences de 250 symboles. Ce découpage, bien que non nécessaire, permet une implémentation, plus simple et plus efficace en mémoire. De plus, il permet d'ajouter simplement de nouvelles données incrémentalement.

Comme nous travaillons avec de longues séquences mais de longueur finie, nous proposons d'utiliser l'algorithme *suffix-history* [Wolfe et collab., 2005] pour estimer la série. Cette méthode consiste à traiter de longues séquences comme un ensemble de sous-séquences. Elle est adaptée si la distribution initiale et stationnaire sont égales. L'estimateur $\hat{p}^{history}$ est donné par

$$\hat{p}^{sous}(u) = \frac{1}{|\mathcal{T}|} \sum_{v \in \mathcal{T}} \sum_{w \in \mathcal{S}(v)} \frac{\mathbb{1}_w(u)}{|v| - |u| + 1},$$

où $\mathcal{S}(v)$ est l'ensemble des sous-séquences de v. Autrement dit, $\sum_{w \in \mathcal{S}(v)} \mathbb{1}_w(u)$ est le nombre de fois que u apparait dans v. Cet estimateur est non-biaisé.

5.5 Résultats

5.5.1 PAutomaC

Nous commençons par la Perplexité. La Tableau 5.3 donne les résultats de chaque algorithme en fonction de la série utilisée pour l'apprentissage. Le résultat reporté dans chaque cellule est le meilleur parmi toutes les dimensions et tailles de base. La dimension du meilleur modèle est indiquée entre parenthèses. En cas d'égalité, c'est la dimension du plus petit modèle qui est indiquée. Pour Tenseur, les modèles appris sont plus petits que pour les autres algorithmes à cause de la nécessité pour la matrice d'observation d'être de rang plein (voir Chapitre 2). Pour comparer, la perplexité et la dimension du vrai modèle figurent aussi dans le tableau. Le meilleur résultat pour chaque problème est indiqué en gras. Premièrement, nous remarquons que l'utilisation de la série auxiliaire basée sur les sous-séquences est profitable à tous les algorithmes et qu'aucune des deux séries ne domine l'autre sur tous les problèmes.

Deuxièmement, l'algorithme NNSpectral se classe premier sur une majorité de problèmes et est globalement très bon. Nous remarquons que l'algorithme de BW donne de très bons résultats sur certains problèmes, où il se classe premier. Par contre, sur d'autres problèmes (1, 27, 42, 43), l'algorithme de BW produit une solution très sous-optimale. L'algorithme Spectral peut parfois produire de très bonnes solutions (comme sur le problème 29) mais reste en général moins bon que NNSpectral et BW. L'algorithme CO ne se place qu'une seule fois (au problème 1) sur le podium des trois premiers. Pourtant ces performances sont globalement bonnes. Les performances de l'algorithme Tenseur sont plutôt mauvaises.

Troisièmement, afin de juger des performances globales de chaque algorithme nous avons sélectionné par algorithme le meilleur résultat entre celui calculé à partir de la série originale et celui calculé à partir de la série auxiliaire. Puis, nous avons calculé la moyenne logarithmique sur les 12 problèmes, définie par,

$$\text{MoyenneLogarithmique}(x_1, \dots, x_n) = 2^{\frac{1}{N}\sum_{i=1}^{n} \log_2(x_i)}.$$

Prendre la moyenne logarithmique est justifié car celle-ci est proportionnelle à la perplexité d'un mélange probabiliste des modèles de chaque problème à pondération égale. Cette moyenne nous permet de classer les algorithmes selon leur performance moyenne. Ce classement confirme les très bonnes performances de l'algorithme NNSpectral. L'algorithme CO et de BW ont des performances globales similaires. Alors que CO est moyen partout, l'algorithme de BW est parfois très bon et parfois mauvais. L'algorithme Spectral s'en sort honorablement alors que Tenseur est assez mauvais.

Enfin, analysons l'impact de la taille de la base sur le résultat. Contrairement à l'intuition, les meilleurs résultats ne sont pas toujours atteints sur les bases les plus grandes. Nous remarquons aussi que la taille de la base ne joue pas beaucoup sur la qualité de l'apprentissage. Pour illustrer ce phénomène, pour chaque problème, nous avons pris le meilleur résultat parmi les différentes tailles de modèles pour chaque taille de base. Puis, nous avons calculé la l'écart-type logarithmique des perplexités, définit par

$$\text{EcartTypeLogarithmique}(x_1, \dots, x_n) = 2^{\sqrt{\frac{1}{N}\sum_{i=1}^{n}\left(\log_2(x_i) - \sum_{i=1}^{n}\log_2(x_i)\right)^2}}.$$

Nous observons que l'écart-type logarithmique est très faible par rapport à la moyenne logarithmique. La taille de la base a cependant un fort impact sur le temps de calcul. Ces résultats sont synthétisés Tableau 5.5.

		CO		NNSpectral		BW
type	n°	séquence	sous-séquence	séquence	sous-séquence	
HMM	1	40.222 ($1.58 \cdot 10^{-5}$)	43.391 ($6.31 \cdot 10^{-6}$)	**30.267 (34)**	62.000 (6)	56.325 (2)
	14	130.797 ($2.51 \cdot 10^{-6}$)	162.778 ($1.58 \cdot 10^{-5}$)	116.897 (10)	117.139 (14)	**116.847 (14)**
	33	32.840 ($3.98 \cdot 10^{-5}$)	60.251 ($2.51 \cdot 10^{-6}$)	31.971 (6)	32.708 (6)	**31.876 (26)**
	45	24.580 ($2.51 \cdot 10^{-6}$)	25.695 ($1.58 \cdot 10^{-5}$)	24.057 (2)	**24.047 (2)**	24.207 (2)
PDFA	6	250.544 ($2.51 \cdot 10^{-6}$)	98.399 ($2.51 \cdot 10^{-6}$)	**67.134 (26)**	70.202 (14)	67.196 (26)
	7	62.145 ($6.31 \cdot 10^{-6}$)	52.915 ($2.51 \cdot 10^{-6}$)	51.262 (14)	51.254 (14)	**51.250 (26)**
	27	118.878 ($2.51 \cdot 10^{-6}$)	58.825 ($1.58 \cdot 10^{-5}$)	**42.664 (46)**	74.088 (26)	69.462 (6)
	42	23.521 ($2.51 \cdot 10^{-6}$)	19.122 ($2.51 \cdot 10^{-2}$)	16.010 (6)	**16.010 (6)**	27.899 (6)
PNFA	29	57.893 ($2.51 \cdot 10^{-6}$)	30.405 ($6.31 \cdot 10^{-6}$)	24.170 (34)	25.380 (34)	24.971 (46)
	39	13.939 ($2.51 \cdot 10^{-6}$)	10.829 ($2.51 \cdot 10^{-2}$)	10.009 (6)	**10.004 (6)**	11.823 (6)
	43	34.907 ($2.51 \cdot 10^{-6}$)	36.377 ($2.51 \cdot 10^{-2}$)	**32.901 (6)**	34.646 (14)	37.930 (2)
	46	20.227 ($3.98 \cdot 10^{-5}$)	14.698 ($2.51 \cdot 10^{-6}$)	12.075 (46)	24.093 (2)	**12.017 (46)**

		Spectral		Tenseur		solution
type	n°	séquence	sous-séquence	séquence	sous-séquence	
HMM	1	43.408 (22)	98.695 (6)	41.781 (3)	65.219 (2)	29.898 (63)
	14	116.899 (42)	119.422 (10)	119.535 (6)	250.249 (8)	116.792 (15)
	33	32.031 (6)	32.817 (30)	33.273 (5)	59.445 (4)	31.865 (13)
	45	32.611 (42)	7974.763 (42)	24.265 (3)	39.753 (4)	24.042 (14)
PDFA	6	78.681 (10)	231.353 (10)	295.022 (3)	567.538 (5)	66.985 (36)
	7	51.259 (14)	51.255 (14)	357.544 (7)	1000.000 (5)	51.224 (6)
	27	55.319 (10)	146.136 (10)	139.871 (8)	308.131 (5)	42.427 (67)
	42	26.499 (2)	32.838 (18)	20.973 (5)	37.784 (2)	16.004 (19)
PNFA	29	**24.162 (46)**	25.825 (26)	72.119 (5)	113.441 (2)	24.031 (10)
	39	10.011 (18)	10.012 (6)	12.562 (4)	20.954 (5)	10.002 (12)
	43	151.580 (6)	36822.163 (14)	36.262 (2)	36.730 (2)	32.637 (19)
	46	15.552 (14)	20.293 (14)	15.050 (3)	30.294 (7)	11.982 (6)

Tableau 5.3 – Perplexité sur douze problèmes de PAutomaC.

	BW	CO	NNSpectral	Spectral	Tenseur
moyenne	35.886	35.641	30.364	40.210	54.015
classement	3	2	1	4	5

Tableau 5.4 – Moyenne logarithmique de la perplexité sur douze problèmes de PAutomaC et classement.

		CO		NNSpectral		Spectral		Tenseur	
type	n°	séq.	ss-séq.	séq.	ss-séq.	séq.	ss-séq.	séq.	ss-séq.
HMM	1	200	100	200	50	10000	50	10000	500
	14	200	200	500	150	1000	100	400	200
	33	100	200	500	100	10000	500	400	50
	45	150	50	500	100	10000	150	400	100
PDFA	6	200	200	500	500	10000	500	1000	150
	7	150	150	500	150	1000	500	1000	50
	27	50	50	500	500	400	50	10000	500
	42	200	50	100	500	200	500	10000	500
PNFA	29	150	150	500	500	1000	150	1000	500
	39	150	200	150	150	1000	200	1000	100
	43	200	50	100	150	10000	500	400	100
	46	200	200	200	200	10000	500	200	200
écart-type (per.)		1.222	1.037	1.192	1.230	1.233	1.546	1.100	1.037

Tableau 5.5 – Taille de base optimale par problème et écart-type logarithmique pour la perplexité sur douze problèmes de PAutomaC.

Analysons maintenant le WER. Premièrement, nous remarquons que l'utilisation de la série auxiliaire basée sur les sous-séquences est profitable à tous les algorithmes sauf Tenseur et qu'aucune des deux séries ne domine l'autre sur tous les problèmes. Les performances de Tenseur avec la série auxiliaire sont étrangement très mauvaises.

Deuxièmement, par rapport à la perplexité, les performances sont légèrement plus nuancées. L'algorithme NNSpectral se classe premier sur 5 problèmes et parmi les 2 premiers sur 11 problèmes. L'algorithme Spectral est nettement meilleur que pour la perplexité et rivalise avec NNSpectral et BW. Nous remarquons que l'algorithme de BW donne des résultats légèrement meilleurs que pour la perplexité. L'algorithme CO globalement mauvais. Enfin, les performances de l'algorithme Tenseur sont très mauvaises.

Pour évaluer les performances globales, nous avons cette fois effectué une moyenne arithmétique. Celle-ci nous permet de donner un classement des algorithmes en fonction de leur performance moyenne. Comme pour la perplexité NNSpectral se place premier. À la différence des résultats pour la perplexité, CO et Spectral échangent leur place.

Enfin, les conclusions sur l'impact de la taille de la base sur le WER sont les mêmes que pour la perplexité. Nous pouvons donc conclure que la taille de la base ne joue pas beaucoup sur les performances. Bien que les grandes bases donnent en moyenne des résultats légèrement meilleurs, ce n'est pas systématique.

Nous terminons par une comparaison du temps de calcul de chaque algorithme. Sur la Section 5.5.1, nous avons représenté le temps de calcul sous la forme d'un histogramme empilé. Nous remarquons que l'utilisation de la série auxiliaire basée sur les sous-séquences augmente de beaucoup le temps nécessaire à l'estimation de la matrice de Hankel. Les algorithmes Spectral et Tenseur sont les plus rapides. Nous remarquons ensuite que l'estimation des paramètres pour les algorithmes CO et NNSpectral prend beaucoup de temps. Pour CO, la complexité de cette étape dépend de la taille de la base, ce qui limite son utilisation. Pour NNSpectral, le temps passé à estimer les paramètres ne dépend pas de la taille de la base mais uniquement

		CO		NNSpectral	
type	n°	séquence	sous-séquence	séquence	sous-séquence
HMM	1	83.584 ($6.31 \cdot 10^{-6}$)	78.396 ($6.31 \cdot 10^{-2}$)	76.731 (10)	**72.729 (38)**
	14	76.902 ($2.51 \cdot 10^{-6}$)	74.907 ($1.58 \cdot 10^{-3}$)	69.316 (6)	68.594 (10)
	33	74.219 ($2.51 \cdot 10^{-6}$)	77.772 ($3.98 \cdot 10^{-3}$)	76.562 (30)	74.431 (6)
	45	88.650 ($6.31 \cdot 10^{-6}$)	80.175 ($6.31 \cdot 10^{-4}$)	78.240 (2)	78.195 (2)
PDFA	6	59.653 ($6.31 \cdot 10^{-4}$)	54.128 ($2.51 \cdot 10^{-2}$)	48.316 (14)	**47.204 (34)**
	7	61.837 ($2.51 \cdot 10^{-6}$)	70.593 ($6.31 \cdot 10^{-2}$)	**45.940 (42)**	48.127 (22)
	27	87.685 ($6.31 \cdot 10^{-6}$)	85.489 ($1.58 \cdot 10^{-5}$)	78.691 (22)	73.837 (18)
	42	65.396 ($2.51 \cdot 10^{-6}$)	61.749 ($2.51 \cdot 10^{-2}$)	56.567 (6)	56.567 (6)
PNFA	29	59.562 ($1.58 \cdot 10^{-5}$)	61.230 ($2.51 \cdot 10^{-2}$)	**44.254 (42)**	45.466 (26)
	39	78.638 ($2.51 \cdot 10^{-6}$)	64.537 ($2.51 \cdot 10^{-2}$)	59.303 (6)	59.153 (6)
	43	78.237 ($6.31 \cdot 10^{-6}$)	78.687 ($2.51 \cdot 10^{-6}$)	**76.574 (22)**	76.939 (14)
	46	91.851 ($2.51 \cdot 10^{-4}$)	84.370 ($6.31 \cdot 10^{-4}$)	86.865 (18)	77.943 (22)

		Spectral		Tenseur		BW
type	n°	séquence	sous-séquence	séquence	sous-séquence	
HMM	1	74.367 (42)	74.606 (14)	79.857 (3)	87.958 (5)	77.138 (34)
	14	68.912 (38)	70.164 (6)	76.318 (5)	84.414 (8)	**68.414 (14)**
	33	**74.181 (6)**	74.299 (18)	74.727 (4)	83.616 (9)	74.228 (30)
	45	78.349 (14)	78.593 (2)	78.575 (2)	86.371 (10)	**78.095 (18)**
PDFA	6	47.423 (46)	47.546 (34)	59.653 (1)	88.832 (5)	59.653 (2)
	7	48.298 (34)	48.260 (14)	80.681 (7)	87.269 (7)	48.289 (34)
	27	75.090 (22)	74.499 (22)	88.499 (2)	91.797 (6)	**73.030 (30)**
	42	**56.507 (30)**	57.166 (6)	68.436 (3)	92.512 (1)	56.582 (30)
PNFA	29	48.646 (42)	49.708 (26)	75.872 (3)	92.518 (1)	48.504 (42)
	39	**59.058 (34)**	59.623 (6)	75.228 (3)	87.417 (6)	59.174 (38)
	43	77.936 (2)	78.162 (6)	78.955 (1)	81.165 (3)	77.314 (26)
	46	85.450 (22)	78.204 (22)	90.562 (5)	94.654 (11)	**77.357 (30)**

Tableau 5.6 – WER en pourcentage sur douze problèmes de PAutomaC.

	BW	CO	NNSpectral	Spectral	Tenseur
moyenne	66.482	71.467	64.618	65.529	77.280
classement	3	4	1	2	5

Tableau 5.7 – Moyenne du WER en pourcentage sur douze problèmes de PAutomaC et classement.

		CO		NNSpectral		Spectral		Tenseur	
type	n°	séq.	ss-séq.	séq.	ss-séq.	séq.	ss-séq.	séq.	ss-séq.
HMM	1	200	50	500	500	1000	500	400	100
	14	50	50	500	150	4000	100	4000	100
	33	200	200	500	200	10000	500	200	150
	45	150	200	150	50	400	500	10000	150
PDFA	6	50	100	500	500	400	500	200	100
	7	50	200	150	50	200	200	4000	50
	27	200	50	500	200	1000	500	10000	150
	42	50	200	200	200	400	500	200	50
PNFA	29	100	200	150	500	1000	500	200	50
	39	150	50	500	100	4000	150	1000	50
	43	100	50	50	200	10000	500	200	200
	46	50	50	200	200	1000	500	200	150
écart-type		0.350	0.555	1.937	0.431	0.266	0.902	1.130	1.616

Tableau 5.8 – Taille de base optimale par problème et écart-type pour le WER en pourcentage sur douze problèmes de PAutomaC.

de la dimension et linéairement de la taille de l'alphabet. De plus, nous pensons que ce temps peut être réduit en utilisant un algorithme spécialisé pour la NNLS plutôt que les méthodes traditionnelles de résolution de problèmes de programmation quadratique. L'algorithme de BW est celui qui prend le plus de temps.

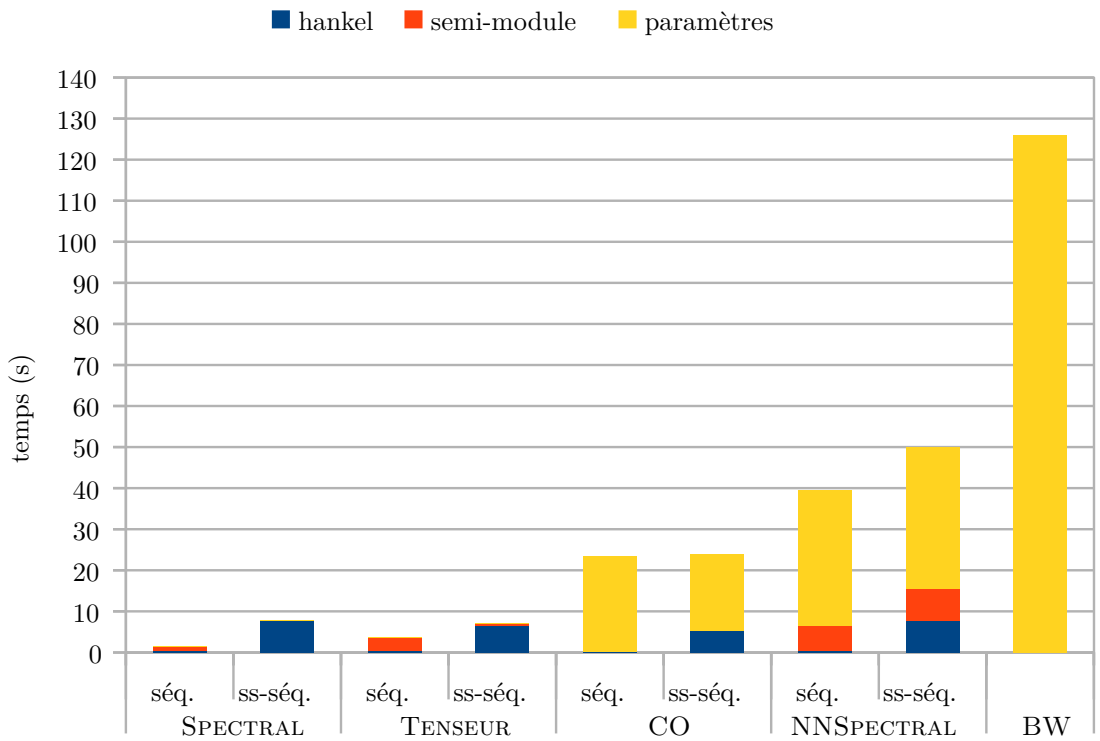


FIGURE 5.2 – Temps d'exécution moyen par algorithme sur douze problèmes de PAutomaC.

5.5.2 Penn-Treebank

Pour Penn-Treebank, les résultats sont rassemblés Tableau 5.9. Seul le WER a été évalué sur ce problème car la perplexité pour tous les algorithmes était très mauvaise et ne permettait donc pas de les comparer. L'algorithme de BW obtient le meilleur score, suivit de près par l'algorithme SPECTRAL et NNSPECTRAL, dont les performances sont très proches. L'algorithme CO est un peu moins performant et l'algorithme TENSEUR est assez mauvais. Pour ce problème, nous observons que l'utilisation d'une série auxiliaire est très profitable pour tous les algorithmes sauf pour TENSEUR, comme nous l'avons déjà observé sur PAutomaC pour le WER. Les temps de calcul sont cohérents avec ceux observés sur PAutomaC.

	BW	CO		NNSPECTRAL		SPECTRAL		TENSEUR	
		séq.	ss-séq.	séq.	ss-séq.	séq.	ss-séq.	séq.	ss-séq.
WER (%)	60.774	70.790	66.070	66.424	61.413	66.256	61.133	70.790	86.330
dimension	30	$2.51 \cdot 10^{-6}$	$3.98 \cdot 10^{-5}$	30	34	46	18	1	7
taille de la base	N/A	50	150	500	500	1000	200	200	150
classement	1	7	4	6	3	5	2	7	9
hankel (s)	0.00	0.73	33.37	0.85	50.87	1.12	50.87	1.10	43.24
semi-module	0.00	0.00	0.00	5.40	7.89	0.82	0.02	2.07	0.29
paramètre	1335.99	6.67	16.19	30.23	30.32	0.19	0.13	0.09	0.09
total	1335.99	7.41	49.57	36.49	89.08	2.14	51.03	3.25	43.63

TABLEAU 5.9 – Résultats pour Penn-Treebank.

5.5.3 Wikipédia

Nous intéressons maintenant à l'évaluation des algorithmes sur l'ensemble des séquences extraites du texte brut de Wikipédia. À la différence des deux précédents ensembles de données, le modèle appris modélise un processus stochastique. De plus, l'apprentissage est réalisé uniquement à partir de la série originale, estimée grâce à l'algorithme *suffix-history*.

Sur cet ensemble de données, nous évaluons les performances de chaque algorithme en fonction de la dimension du modèle. Pour certains algorithmes, nous comparons les résultats pour différentes tailles de base. Comme, pour l'algorithme CO, la dimension du modèle appris est égale à la taille de la base, nous traçons séparément ses performances en fonction du paramètre τ de régularisation. Pour Tenseur, à partir d'une certaine dimension, la matrice d'observation n'est plus de rang plein et l'algorithme échoue. Pour des modèles de grandes dimensions, l'algorithme de BW est trop lent. Ainsi, les dimensions évaluées pour Tenseur et l'algorithme de BW sont inférieures que pour Spectral, NNSpectral et l'algorithme de BW.

Nous commençons par donner les résultats pour la vraisemblance conditionnelle. La Figure 5.3 donne les performances de l'algorithme CO sur les deux ensembles d'apprentissage de taille différentes. Les Figures 5.4 et 5.5 représentent les performances des autres algorithmes. Nous remarquons que l'algorithme CO et Tenseur, ont des performances assez faibles pour la vraisemblance conditionnelle et que le nombre de mots utilisés pour l'apprentissage ne joue pas beaucoup. Alors que pour CO, la complexité du modèle n'influe pas sur les performances, pour Tenseur les performances se dégradent quand la dimension augmente. Ces faibles performances peuvent s'expliquer par la nécessite d'utiliser une trop petite base pour l'algorithme CO. Pour Tenseur, la condition de rang plein limite la richesse de la représentation apprise.

Pour l'algorithme NNSpectral, les performances sont assez irrégulières. Cette variance est due à l'utilisation d'une initialisation aléatoire de la NMF. En moyenne, les performances augmentent naturellement avec la dimension. Par rapport à Spectral et l'algorithme de BW, NNSpectral est moins performant. Cela est dû à la taille de la base utilisée par NNSpectral, qui trop faible (pour des raisons de temps de calcul), ne peut pas capturer au moins une partie suffisante de la dynamique du système. Enfin, Spectral affiche des performances légèrement supérieures à l'algorithme de BW. Lorsque plus de mots sont utilisés pour l'apprentissage ou bien que la taille de la base croît, les performances de Spectral augmentent légèrement.

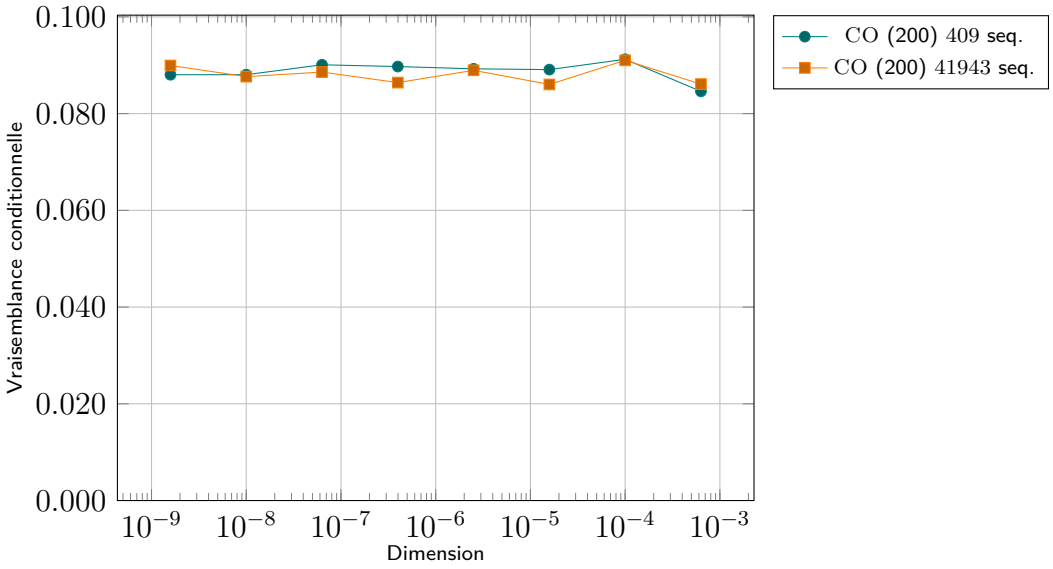


FIGURE 5.3 – Performances de CO pour la vraisemblance conditionnelle sur Wikipédia en fonction de la dimension et de la taille de la base (indiquée entre parenthèses).

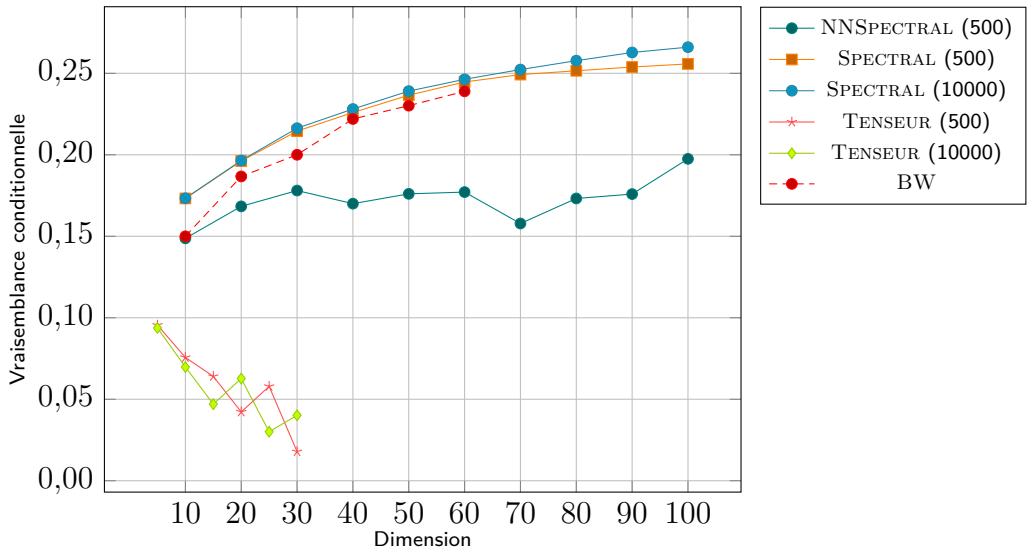


FIGURE 5.4 – Performances de SPECTRAL, NNSPECTRAL, TENSEUR et l'algorithme de BW pour la vraisemblance conditionnelle sur 409 séquences de Wikipédia en fonction de la dimension et de la taille de la base (indiquée entre parenthèses).

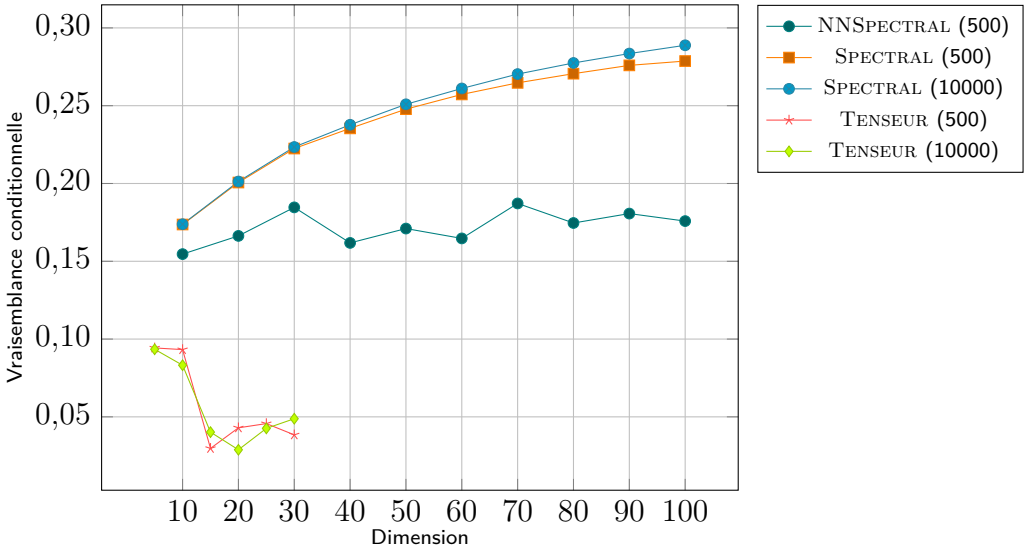


FIGURE 5.5 – Performances de SPECTRAL, NNSPECTRAL, TENSEUR et l'algorithme de BW pour la vraisemblance conditionnelle sur 41943 séquences de Wikipédia en fonction de la dimension et de la taille de la base (indiquée entre parenthèses).

Passons maintenant à l'évaluation des algorithmes pour le BPC. Les résultats sont affichés sur les Figures 5.6 à 5.8. Comme pour la vraisemblance conditionnelle, les performances de CO et TENSEUR sont très moyennes, dans toutes les configurations de bases et d'ensembles d'apprentissage. De plus, CO est légèrement meilleur que TENSEUR. De façon surprenante, l'algorithme SPECTRAL est très mauvais sur cette tâche. De plus, les performances se dégradent avec la dimension du modèle. Cet effet est moins marqué lorsque l'ensemble d'apprentissage et le nombre de mots dans la base sont plus grands. Si bien que l'effet est inversé lorsque 10000 mots sont utilisés dans la base et 41943 séquences sont utilisées pour l'apprentissage. Cela montre que l'algorithme SPECTRAL est beaucoup moins bon lorsqu'il s'agit d'estimer les probabilités d'événements peu probables. En effet, les préfixes et les suffixes peu probables ne sont pas présents dans une base trop petite et sont mal estimés si l'ensemble d'apprentissage est petit. Ainsi lorsque les statistiques des événements peu probables sont absentes ou mal estimées, l'algorithme SPECTRAL se concentre sur les événements les plus probables. Comme, cet effet est absent pour NNSPECTRAL et que de plus, les performances de NNSPECTRAL sont bien meilleures, nous pouvont suspecter que l'erreur excessive de SPECTRAL sur les événements peu probables provient des changements de signes. En effet, ceux-ci se produisent avec une plus forte probabilité pour les faibles valeurs. Ces résultats confirment ainsi l'intuition donnée en début de chapitre. Enfin, l'algorithme de BW affiche les meilleures performances.

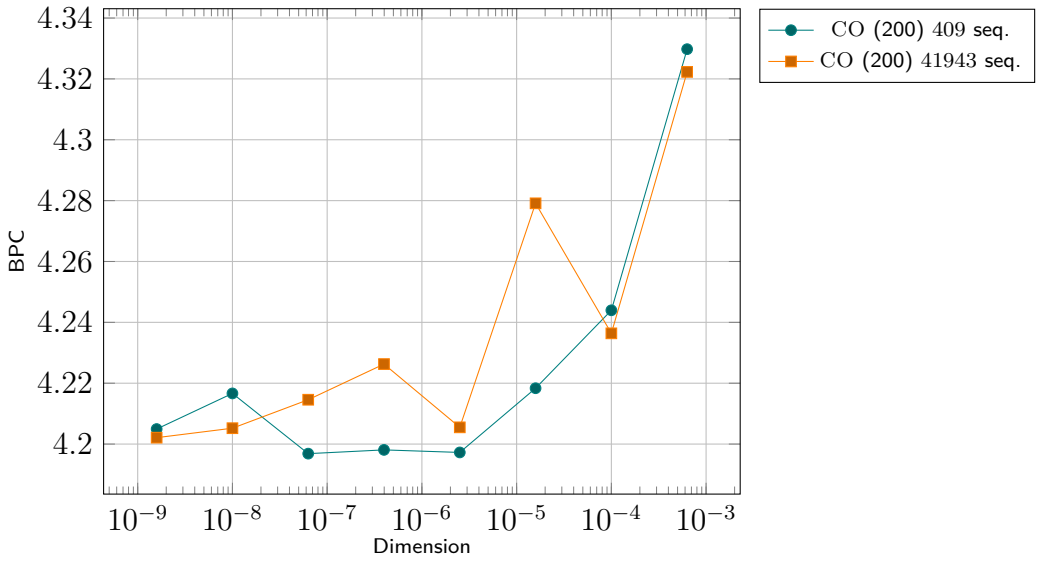


FIGURE 5.6 – Performances de CO pour le BPC sur 500 séquences de Wikipédia en fonction de la dimension et de la taille de la base (indiquée entre parenthèses).

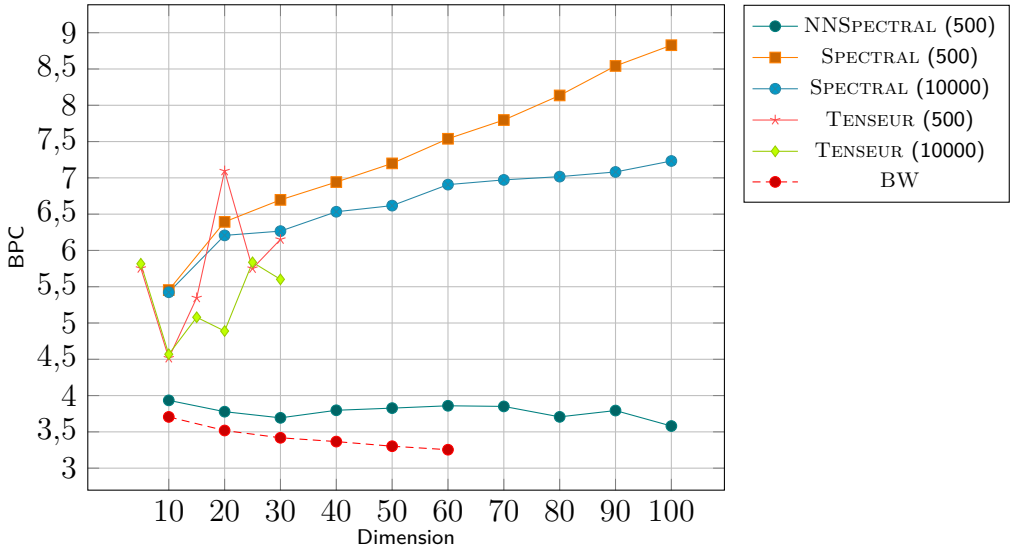


FIGURE 5.7 – Performances de SPECTRAL, NNSPECTRAL, TENSEUR et l'algorithme de BW pour le BPC sur 409 séquences de Wikipédia en fonction de la dimension et de la taille de la base (indiquée entre parenthèses).

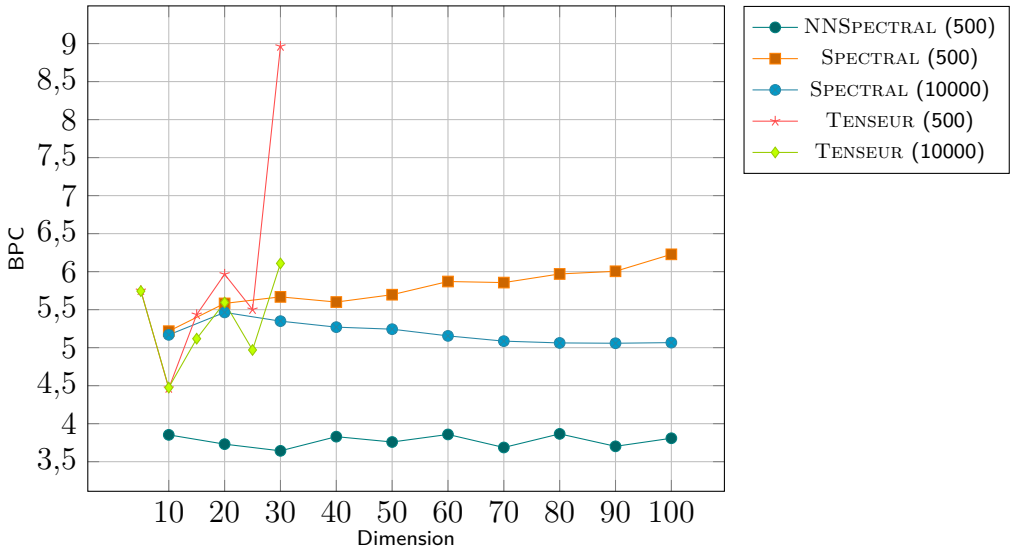


FIGURE 5.8 – Performances de SPECTRAL, NNSPECTRAL, TENSEUR et l'algorithme de BW pour le BPC sur 41943 séquences de Wikipédia en fonction de la dimension et de la taille de la base (indiquée entre parenthèses).

5.6 Comparaison à l'état de l'art

Plusieurs travaux se sont déjà intéressés à l'inférence propre de séries formelles rationnelles à valeurs non-négatives. Comme il est indécidable si une série formelle rationnelle quelconque est non-négative, l'ensemble de ces travaux s'attaque à l'inférence de modèles moins riches que les $\mathbb{R}$-MA.

Les travaux les plus connexes à notre approche, ont déjà été présentés Section 2.2.3.3. Nous revenons en particulier sur les travaux de Cybenko et Crespi [2011]; Finesso et collab. [2010] proposant une décomposition de matrices similaires à la matrice de Hankel en deux facteurs non-négatifs. NNSpectral se différencie de ces précédentes approches en plusieurs points.

D'une part, la présentation de notre algorithme est très générale, ce qui le rend applicable à l'inférence de différents types de séries formelles comme les langages stochastiques, les processus stochastiques et les processus contrôlés sans modification particulière outre l'estimation de la matrice de Hankel.

D'autre part, nous utilisons la distance ℓ_2 à la fois pour l'approximation en facteurs de faible dimension et pour la régression des paramètres. Dans [Finesso et collab., 2010], les auteurs utilisent la divergence KL. Dans [Cybenko et Crespi, 2011], la divergence KL est utilisée pour la décomposition et la distance ℓ_1 pour la régression des paramètres. De plus, dans ces deux travaux, les auteurs intègrent des contraintes dans la décomposition et la régression afin de retrouver exactement un HMM. Bien qu'il puisse sembler souhaitable de faire de même, il est difficile d'intégrer ces contraintes dans la NMF lorsque la distance ℓ_2 est utilisée. Que ce soit pour la divergence KL où la distance ℓ_2, les algorithmes de NMF sont construits sur une minimisation alternée de chaque facteur. Si la divergence KL est utilisée, la minimisation alternée sous contraintes est convergente tant que les facteurs appartiennent à des ensembles convexes [Csiszár et Tusnády, 1984]. Cette propriété rend l'algorithme EM, et nombreuses de ces variantes, convergents dans des cas très variés. Lorsque la distance ℓ_2 est utilisée, la minimisation alternée sous contrainte n'est pas convergente en général. Ainsi, nous n'avons pas inclus de contrainte particulière dans l'étape de NMF. L'utilisation de la distance ℓ_2 a tout de même l'avantage d'être plus régulière que la divergence KL qui possède des singularités. De plus, comme expliqué au Chapitre 2, la distance ℓ_2 est adaptée au bruit d'échantillonnage presque Gaussien.

Enfin, nous pouvons comparer NNSpectral à l'algorithme de Bailly [2011b] pour apprendre des QWA (voir Chapitre 2). Les QWA sont un type d'automates réalisant des séries non-négatives. Leur algorithme possède des garanties non asymptotiques. En contrepartie, les QWA ne sont pas capables de modéliser tous les HMMs. Pour cette raison, nous n'avons pas inclus QWA dans nos expériences. Enfin, les performances de leur algorithme sont mitigées. En effet, les auteurs complètent celui-ci par une descente de gradient sur la vraisemblance pour obtenir des résultats très compétitifs. Malheureusement, cette descente de gradient annule les garanties non-asymptotiques. Pour plus de détails, le lecteur peut se reporter à la Section 2.2.3.5.

5.7 Conclusions

Les expériences confirment l'intuition donnée en introduction. En retournant des $\mathbb{R}^+$-MA, NNSpectral évite les erreurs introduites par les heuristiques pour obtenir une probabilité et repose uniquement sur une normalisation naturelle. Évalué sur Wikipédia, cela permet à NNSpectral d'être plus performant pour le BPC que

Spectral car pour les événements peu probables les changements de signes dans la représentation linéaire apprise sont plus fréquents. Au niveau théorique, les $\mathbb{R}^+$-MA sont moins expressifs que les $\mathbb{R}$-MA et aussi moins compacts. En pratique, les $\mathbb{R}^+$-MA restent suffisamment expressifs pour de nombreuses applications. Dans les expériences sur PAutomaC et Penn-Treebank, la dimension des représentations linéaires apprises par NNSpectral n'est pas spécialement plus grande que pour l'algorithme Spectral. Sur Wikipédia, l'expressivité des $\mathbb{R}$-MA donne un avantage à Spectral.

Malheureusement, NNSpectral ne bénéficie de garanties théoriques comme Spectral. La raison est double. D'une part, comme la complexité calculatoire requise pour résoudre exactement la NMF est trop élevée, NNSpectral repose sur des heuristiques ne bénéficiant que d'une convergence locale. Ce problème est à rapprocher des difficultés intrinsèques à l'apprentissage de PNFA expliquées au Chapitre 2. D'autre part, comme mentionnée dans la Section 2.2.3.3, une décomposition en facteurs non-négatifs ne garantit pas d'identifier un sous semi-module stable, même dans le cas d'une matrice de Hankel infinie.

En conclusion, comme d'autres algorithmes basés sur la MoM, l'algorithme NNSpectral est en mesure de traiter des problèmes de taille importante plus rapidement que BW. Bien que les complexités NNSpectral et Spectral soit sensiblement les mêmes ($\mathcal{O}(mnr)$), le temps d'exécution de NNSpectral sur les problèmes testés est environ 100 fois supérieur à Spectral mais reste toutefois largement inférieur à BW. Ainsi, NNSpectral est une bonne alternative à l'algorithme BW avec des performances accrues et un coût de calcul beaucoup plus faible. En particulier, NNSpectral semble bien moins sujet à rester coincé dans des minimaux locaux très mauvais.

Chapitre 6

Apprentissage par enveloppe convexe

Sommaire

6.1 Introduction

Ce chapitre est motivé par les résultats négatifs présentés au Chapitre 1 dans la Section 1.4.4 concernant les $\mathbb{R}$-MA. En effet, ils condamnent toute tentative d'algorithme propre pour l'apprentissage de langages stochastiques rationnel sur $\mathbb{R}$. Nous avons vu que les $\mathbb{R}$-MA avaient deux grands défauts pour l'inférence de langages stochastiques rationnels. Premièrement, il est indécidable de savoir si un $\mathbb{R}$-MA réalise une série positive. Cette propriété ruine la possibilité d'obtenir des algorithmes propres identifiant à la limite avec probabilité 1 les séries formelles rationnelles positives. Deuxièmement, les $\mathbb{R}$-MA réalisant des langages stochastiques, et plus généralement des séries formelles rationnelles positives, ne forment pas une classe robuste. Autrement dit, un changement infime dans les paramètres de la représentation linéaire d'une série positive permet d'obtenir une représentation réalisant une série ayant des valeurs négatives. En contre-partie, il existe des algorithmes propres d'inférence de langages stochastiques rationnel sur $\mathbb{R}^+$. Ainsi, les PNFA semble être une classe d'automates adaptée à l'apprentissage.

Dans le précédent chapitre, nous avons utilisé la caractérisation des séries formelles rationnelles sur $K = \mathbb{R}^+$ par le sous semi-module stable établi au Chapitre 1. En s'inspirant de l'algorithme Spectral, nous avons proposé d'identifier le sous semi-modules par décomposition en facteurs non-négatifs de la matrice de Hankel. Dans ce chapitre, nous mettons en œuvre une approche similaire mais utilisant une caractérisation spécifique des langages stochastiques rationnels sur $\mathbb{R}^+$. Ainsi, l'algorithme CH-PNFA, proposé dans ce chapitre, retourne exclusivement des PNFA. L'ensemble des PNFA, dont fait l'objet ce chapitre, est représenté dans la hiérarchie d'automates établie au Chapitre 1 sur la Figure 6.1.

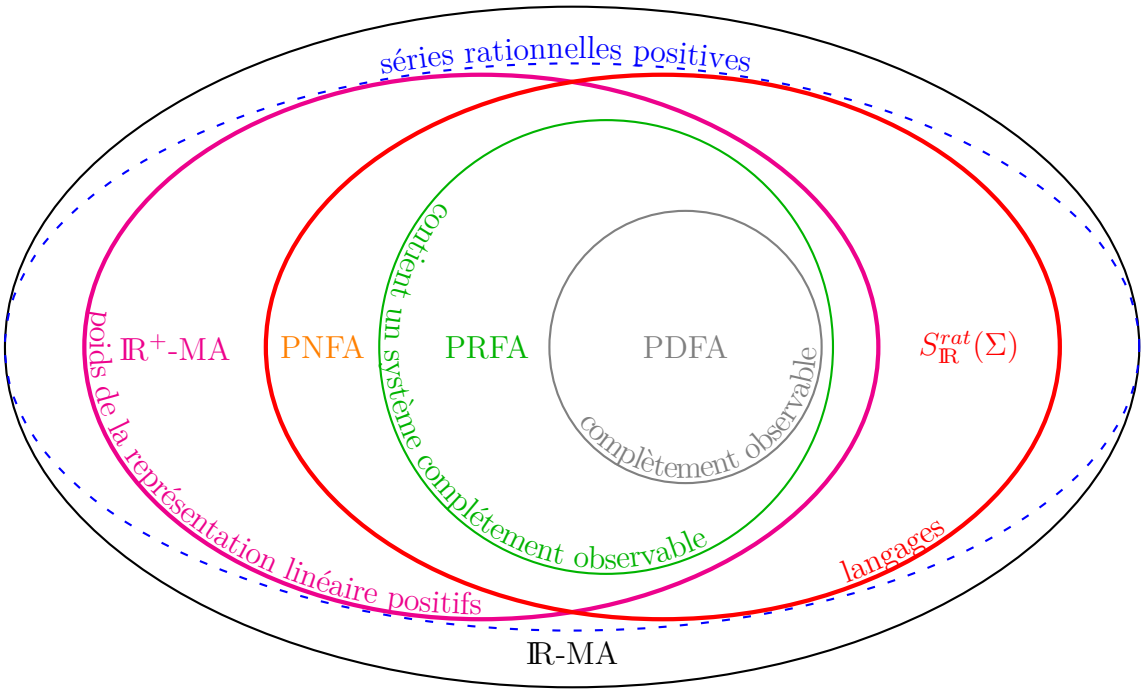


Figure 6.1 – Hiérarchie entres les classes d'automates. Dans ce chapitre, nous nous intéressons à la classe d'automate en orange, définie par l'intersection de celle des langages, en rouge épais, et celle des des $\mathbb{R}^+$-MA, en magenta épais.

Cela a pour avantage de rendre possible l'initialisation d'algorithme itératifs comme l'algorithme de BW, ce que ne peut pas faire NNSpectral. Nous verrons dans les

expériences que utiliser CH-PNFA pour initialiser l'algorithme de BW fournit de meilleurs résultats que de lancer plusieurs fois une initialisation aléatoire.

6.2 Recherche d'enveloppe convexe de langages stochastiques

6.2.1 Algorithme CH-PNFA

Dans cette section, nous détaillons l'algorithme CH-PNFA pour l'inférence propre de langages stochastiques rationnels sur $\mathbb{R}^+$. L'algorithme CH-PNFA est très similaire à NNSpectral, lui-même dérivé de Spectral. L'algorithme pourrait très bien être dérivé pour une matrice de Hankel infinie comme l'algorithme Spectral. Cependant, pour les applications, il est souvent pratique, voire nécessaire, de limiter la dimension de H en prenant une base finie. De plus, pour simplifier la présentation de l'algorithme sans vraiment contraindre les applications, nous supposons que la base contient le mot vide ($(\varepsilon, \varepsilon) \in \mathcal{B}$).

La différence fondamentale avec les précédents algorithmes est que CH-PNFA se base sur une caractérisation spécifique aux langages stochastiques. Cette caractérisation, donnée par le Théorème 6 au Chapitre 1, établit pour un langage stochastique p l'équivalence entre la rationalité de p et l'existence d'une famille finie de langages stochastiques générant une enveloppe convexe contenant p. La Proposition 11 établit que la transformation donnée dans la Proposition 4 retourne bien un PNFA.

Alors que, dans le chapitre précédent, nous cherchions une famille génératrice de séries formelles quelconques, nous cherchons maintenant une famille génératrice de langages stochastiques. Dans la suite, nous notons p le langage stochastique cible. Dans leur article, Denis et Esposito [2008, Proposition 13] montrent qu'à partir d'une famille de séries formelles de $\mathbb{R}^+\langle\langle\Sigma\rangle\rangle$ génératrice d'un sous semi-module stable contenant un p, nous pouvons construire une famille de langages stochastiques génératrice d'un sous semi-module stable dont l'enveloppe convexe contient p. Pour ce faire, il suffit d'éliminer les séries formelles non convergentes puis de normaliser les autres pour qu'elles convergent vers 1. C'est exactement ce que va faire CH-PNFA. L'algorithme utilise d'abord la NMF pour trouver une famille finie $\{r_1, \ldots, r_d\}$ sous semi-modules contenant $\hat{p}$. Soit $\hat{P} \in {\mathbb{R}^+}^{\mathcal{P}\times d}$ et $\hat{S} \in {\mathbb{R}^+}^{d\times\mathcal{S}}$, nous notons $\tilde{H} = \hat{P}\hat{S}$ la décomposition en facteurs non-négatifs de faible dimension approchant en norme de Frobenius $\hat{H}_{\mathcal{B}}$. Sans nuire à la généralité, nous pouvons supposer que les lignes de $\hat{S}$ ne représentent, sur la base des suffixes $\mathcal{S}$, que des séries formelles convergentes. Nous notons ces séries $\{r_1, \ldots, r_d\}$. Il reste à normaliser pour obtenir $\{p_1, \ldots, p_d\}$. Nous pourrions être tenté de normaliser les lignes de $\hat{S}$ à 1. Si l'on travaillait à partir d'une matrice de Hankel infinie, cette normalisation serait correcte mais, lorsque l'on travaille sur une base finie, la limite d'une série ne peut pas être estimée par la somme de ses valeurs sur cette base. Dans ce cas, comment estimer pour tout $i \in [1, d]$, $r_i(\Sigma^\star)$? Quittons le domaine de l'inférence et travaillons avec les statistiques connues. Comme au Chapitre 5, supposons que $\left[\!\left[S^\top\right]\!\right]$ soit stable. Dans ce cas, il existe une matrice $M \in {\mathbb{R}^+}^{d\times\Sigma^\star}$ telle que, soit $G = PSM$, $G[u, v] = r_d(v) = \dot{v}r_d(\varepsilon)$. De cette façon, nous avons que

$$G\mathbf{1} = \boldsymbol{\gamma} = \begin{pmatrix} r_1(\Sigma^\star) \\ \vdots \\ r_d(\Sigma^\star) \end{pmatrix},$$

De plus, nous pouvons vérifier que $\boldsymbol{\rho} = (p(u\Sigma^\star))_{u\in\mathcal{P}} = PSM\mathbf{1} = P\boldsymbol{\gamma}$. Nous notons X la matrice normalisée dont les lignes correspondent aux séries $\{p_1, \ldots, p_d\}$,

$$\begin{aligned} \operatorname{diag}(\boldsymbol{\gamma})X &= S, \\ X &= \operatorname{diag}(\boldsymbol{\gamma})^{-1}S. \end{aligned}$$

Or $\boldsymbol{\rho}$ est simple à estimer par comptage des séquences commençant par $u \in \mathcal{P}$ dans l'ensemble d'entrainement. Nous notons $\hat{\boldsymbol{\rho}}$ cet estimateur et l'on cherche $\hat{\boldsymbol{\gamma}}$ tel que $\hat{\boldsymbol{\rho}} \approx \hat{P}\hat{\boldsymbol{\gamma}}$ au sens des moindres carrés sous la contrainte que $\hat{\boldsymbol{\gamma}}$ soit non-négative. Cette résolution est conduite par NNLS. Enfin, nous nous servons de $\hat{\boldsymbol{\gamma}}$ pour normaliser les lignes de $\hat{S}$ pour obtenir $\hat{X} = \operatorname{diag}(\hat{\boldsymbol{\gamma}})^{-1}\hat{S}$, où les lignes de $\hat{X}$ représentent les séries $\{p_1, \ldots, p_d\}$. Nous remarquons que l'inversion de $\operatorname{diag}(\hat{\boldsymbol{\gamma}})$ nécessite que $\hat{\boldsymbol{\gamma}}$ soit strictement positif. À cause des erreurs d'estimation, $\hat{\boldsymbol{\gamma}}$ peut avoir des coefficients nuls. Dans ce cas, $\operatorname{diag}(\hat{\boldsymbol{\gamma}})$ n'est pas inversible. Il convient de rajouter un terme de régularisation. Par exemple, $\hat{X} = \operatorname{diag}(\hat{\boldsymbol{\gamma}} + \lambda I)^{-1}\hat{S}$, où λ est le paramètre de régularisation. Dans les expériences nous n'en n'avons pas eu besoin.

Nous passons maintenant à l'estimation pour tout $\sigma \in \Sigma$ des séries $\{\dot{\sigma}p_1, \ldots, \dot{\sigma}p_d\}$ afin de pouvoir appliquer la Proposition 11. Nous commençons par estimer $\{\dot{\sigma}r_1, \ldots, \dot{\sigma}r_d\}$ par NNLS de la même façon qu'au Chapitre 5, pour obtenir les matrices non-négatives $\hat{R}_\sigma$ telles que $\hat{P}\hat{R} \approx \hat{H}^\sigma_{\mathcal{B}}$ au sens des moindres carrés. Pour obtenir les $\{\dot{\sigma}p_1, \ldots, \dot{\sigma}p_d\}$, nous normalisons $\hat{R}_\sigma$ par $\hat{\boldsymbol{\gamma}}$. Nous notons pour $\sigma \in \Sigma$, $\hat{Y}_\sigma$ la matrice sont les lignes représentent les séries $\{\dot{\sigma}p_1, \ldots, \dot{\sigma}p_d\}$ et telle que $\hat{Y}_\sigma = \operatorname{diag}(\boldsymbol{\gamma})^{-1}\hat{R}_\sigma$.

Finalement, il reste à retrouver la représentation linéaire par application de la Proposition 11. Les poids finaux sont donnés par $(p_1(\varepsilon), \ldots, p_d(\varepsilon))$. Pour retrouver $\hat{A}$, comme $\{\dot{\sigma}p_1, \ldots, \dot{\sigma}p_d\}$ n'appartient pas forcément à $\operatorname{conv}(\{p_1, \ldots, p_d\})$, nous résolvons un problème de NNLS sous les contraintes linéaires définies Équation (1.1). De même pour les poids initiaux, $\operatorname{conv}(\{p_1, \ldots, p_d\})$ ne contient pas forcément $\hat{H}^\top_{\mathcal{B}}\mathbf{1}_\varepsilon$, nous procédons donc par NNLS sous la contrainte que $\hat{\boldsymbol{\alpha}}_0^\top\mathbf{1} = 1$. Finalement, l'automate appris satisfait bien les contraintes inhérentes aux PNFA. Les étapes de l'algorithme CH-PNFA sont résumées dans Algorithme 3.

Algorithme 3 Algorithme CH-PNFA

Entrées Un alphabet Σ, un ensemble de mots d'apprentissage, un rang estimé d
Sortie Un PNFA $\langle \Sigma, \{1..d\}, \left\{\hat{A}_\sigma\right\}_{\sigma\in\Sigma}, \hat{\boldsymbol{\alpha}}_0, \hat{\boldsymbol{\alpha}}_\infty \rangle$

1: Choisir une base de préfixes $\mathcal{P} \subset \Sigma^\star$ et de suffixes $\mathcal{S} \subset \Sigma^\star$ contenant ε.
2: Estimer $\hat{H}_{\mathcal{B}}$, $\hat{\boldsymbol{\rho}}$ et $\forall\sigma \in \Sigma$, $\hat{H}^\sigma_{\mathcal{B}}$.
3: $\hat{P}, \hat{S} \leftarrow \operatorname{argmin}_{\substack{\hat{P}\in\mathbb{R}^{+\mathcal{P}\times n} \\ \hat{S}\in\mathbb{R}^{+n\times\mathcal{S}}}} \left\|\hat{H}_{\mathcal{B}} - \hat{P}\hat{S}\right\|_F$
4: $\hat{\boldsymbol{\gamma}} \leftarrow \operatorname{argmin}_{\gamma\in\mathbb{R}^{+d}} \left\|\hat{\boldsymbol{\rho}} - \hat{P}\boldsymbol{\gamma}\right\|_2$
5: $\hat{X} \leftarrow \operatorname{diag}(\hat{\boldsymbol{\gamma}})^{-1}\hat{S}$
6: $\hat{\boldsymbol{\alpha}}_\infty \leftarrow \hat{X}\mathbf{1}_\varepsilon$
7: $\hat{\boldsymbol{\alpha}}_0 \leftarrow \operatorname{argmin}_{\boldsymbol{\alpha}_0\in\mathbb{R}^{+d}} \left\|\boldsymbol{\alpha}_0^\top\hat{X} - \mathbf{1}_\varepsilon^\top\hat{H}_{\mathcal{B}}\right\|_2$ tel que $\boldsymbol{\alpha}_0^\top\mathbf{1} = 1$
8: **pour tout** $\sigma \in \Sigma$ **faire**
9: $\quad\hat{R}_\sigma \leftarrow \operatorname{argmin}_{R_\sigma\in\mathbb{R}^{+d\times\mathcal{S}}} \left\|\hat{P}R_\sigma - \hat{H}^\sigma_{\mathcal{B}}\right\|_F$
10: $\quad\hat{Y}_\sigma \leftarrow \operatorname{diag}(\hat{\boldsymbol{\gamma}})^{-1}\hat{R}_\sigma$
11: **fin pour**
12: $\left\{\hat{A}_\sigma\right\}_{\sigma\in\Sigma} \leftarrow \operatorname{argmin}_{\left\{A_\sigma\in\mathbb{R}^{+d\times d}\right\}_{\sigma\in\Sigma}} \sum_{\sigma\in\Sigma}\left\|A_\sigma\hat{X} - \hat{Y}_\sigma\right\|_F$ tel que $\hat{\boldsymbol{\alpha}}_\infty + A_\Sigma\mathbf{1} = \mathbf{1}$

Nous remarquons qu'à l'étape 5 de l'Algorithme 3, si $\hat{\boldsymbol{\gamma}}$ contient des valeurs nulles, alors $\text{diag}(\hat{\boldsymbol{\gamma}})$ n'est pas inversible. Dans ce cas, il convient de rajouter un terme de régularisation. Par exemple, $\hat{X} \leftarrow \text{diag}(\hat{\boldsymbol{\gamma}} + \lambda I)^{-1}\hat{S}$, où λ est le paramètre de régularisation.

6.2.2 Moindres carrés non-négatifs sous contraintes

Dans l'algorithme CH-PNFA la résolution des problèmes de NNLS sous contraintes linéaires se fait par programmation quadratique. La forme classique des problèmes de minimisation quadratique sous contraintes linéaires est la suivante,

$$\begin{aligned} &\underset{\mathbf{x}}{\text{argmin}} \frac{1}{2}\mathbf{x}^\top Q\mathbf{x} + \mathbf{q}^\top\mathbf{x} \\ &\text{tel que } \begin{cases} B\mathbf{x} + \mathbf{b} \geq 0, \\ C\mathbf{x} + \mathbf{c} = 0 \end{cases}. \end{aligned}$$

Le problème,

$$\underset{\boldsymbol{\alpha}_0 \in \mathbb{R}^{+d}}{\text{argmin}} \left\| \boldsymbol{\alpha}_0^\top \hat{X} - \mathbf{1}_\varepsilon^\top \hat{H}_{\mathcal{B}} \right\|_2 \text{ tel que } \boldsymbol{\alpha}_0^\top \mathbf{1} = 1,$$

peut être écrit dans la forme précédente, où $\mathbf{x}$ représente $\boldsymbol{\alpha}_0$ et les autres paramètres sont calculés par,

$$\begin{aligned} Q &= \hat{X}\hat{X}^\top, \\ \mathbf{q}^\top &= -\mathbf{1}_\varepsilon^\top \hat{H}_{\mathcal{B}} \hat{X}^\top, \\ B &= I, \\ \mathbf{b} &= 0, \\ C &= \mathbf{1}^\top, \\ \mathbf{c} &= -1. \end{aligned}$$

Nous donnons maintenant la forme en programmation quadratique du problème suivant,

$$\underset{A_\sigma \in \mathbb{R}^{+d\times d}}{\text{argmin}} \sum_{\sigma\in\Sigma} \left\| A_\sigma \hat{X} - \hat{Y}_\sigma \right\|_F \text{ tel que } \hat{\boldsymbol{\alpha}}_\infty + A_\Sigma \mathbf{1} = \mathbf{1}.$$

Pour plus de clarté, nous associons un index à chaque symbole de Σ. Nous notons $\mathbf{x}$ le vecteur de taille $d^2|\Sigma|$ tel que $\mathbf{x}[i + d(j-1) + d^2(\sigma - 1)] = A_\sigma[i,j]$. Autrement dit, $\mathbf{x}^\top$ est la concaténation horizontale des lignes $A_1, A_2, \ldots, A_{|\Sigma|}$,

$$\mathbf{x}^\top = \underbrace{\begin{pmatrix} A_1[1,:] & \ldots & A_1[d,:] & \ldots\ldots & A_{|\Sigma|}[1,:] & \ldots & A_{|\Sigma|}[d,:] \end{pmatrix}}_{d^2|\Sigma|}.$$

Les autres paramètres sont alors donnés par,

$$Q = \underbrace{\begin{pmatrix} \hat{X}\hat{X}^\top & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \hat{X}\hat{X}^\top \end{pmatrix}}_{d^2|\Sigma|},$$

$$\mathbf{q}^\top = -\underbrace{\begin{pmatrix} \hat{Y}_1[1,:]\hat{X}^\top & \ldots & \hat{Y}_1[d,:]\hat{X}^\top & \ldots\ldots & \hat{Y}_{|\Sigma|}[1,:]\hat{X}^\top & \ldots & \hat{Y}_{|\Sigma|}[d,:]\hat{X}^\top \end{pmatrix}}_{d^2|\Sigma|},$$

$$B = I,$$
$$\mathbf{b} = 0,$$
$$\mathbf{c} = \hat{\boldsymbol{\alpha}}_\infty - \mathbf{1}.$$

$$C = \underbrace{\underbrace{\underbrace{\left(\begin{array}{ccc:ccc:ccc:ccc:ccc} 1 & \dots & 1 & 0 & \dots & 0 & 1 & \dots & 1 & 0 & \dots & 0 \\ 0 & \dots & 0 & 0 & \dots & 0 & 0 & \dots & 0 & 0 & \dots & 0 \\ \vdots & \dots & \vdots & \vdots & \dots & \vdots & \vdots & \dots & \vdots & \vdots & \dots & \vdots \\ 0 & \dots & 0 & 0 & \dots & 0 & 0 & \dots & 0 & 0 & \dots & 0 \\ 0 & \dots & 0 & 1 & \dots & 1 & 0 & \dots & 0 & 1 & \dots & 1 \end{array}\right)}_{d} \cdot \Big\} d}_{d^2}}_{d^2|\Sigma|}$$

Comme Q est symétrique semi-définie positive, le problème est convexe avec des contraintes linéaires et la convergence a lieu en temps polynomial vers un point stationnaire. De plus, tous les points stationnaires sont de valeurs égales [Lőtstedt, 1983]. Si Q est une matrice de rang plein, le problème est fortement convexe et il existe un unique point stationnaire. Si Q n'est pas de rang de plein, nous pouvons à partir de n'importe quelle solution calculer la solution de norme minimale par programmation quadratique. Cette solution est unique et donc le problème est bien-défini. Dans les expériences, nous utilisons le solveur [MOSEK, 2015].

6.3 Expériences

6.3.1 Initialisation d'algorithmes itératifs

Comme l'algorithme CH-PNFA renvoie un PNFA, le modèle appris peut servir à initialiser un algorithme itératif, tel que l'algorithme BW. Ainsi, la capacité de CH-PNFA à éviter les mauvais optimums locaux, permet d'obtenir une initialisation intéressante. C'est aussi le cas de l'algorithme Tenseur. Afin d'utiliser l'algorithme de BW de treba, nous devons transformer la représentation linéaire d'initialisation pour qu'elle ne possède qu'un seul état initial. Cette transformation élémentaire augmente la dimension de 1. À part pour l'initialisation, la procédure d'apprentissage de l'algorithme de BW est exactement la même que précédemment. Enfin, nous notons ces algorithmes Tenseur+BW et CH-PNFA+BW.

6.3.2 Paramètres pour CH-PNFA

Pour l'algorithme CH-PNFA, seule la série originale a été utilisée. De plus, nous n'avons pas normalisé la variance comme pour Spectral et NNSpectral. Le nombre de préfixes et de suffixes dans la base a été limité à 500 afin d'offrir un bon compromis entre performances et temps d'apprentissage. La dimension du modèle varie de la même façon que pour NNSpectral.

6.4 Résultats

Les résultats reportés ici complètent ceux du Chapitre 5. Le détail des expériences et les résultats déjà exposés ne sont pas répétés dans ce chapitre. En conclusion de cette thèse, une comparaison de tous les algorithmes présentés est proposée.

type	n°	BW	CH-PNFA	CH-PNFA+BW	Tenseur	Tenseur+BW
HMM	1	56.325 (2)	**31.805 (10)**	**31.805 (11)**	42.408 (3)	42.408 (4)
	14	**116.847 (14)**	117.495 (6)	117.495 (7)	119.719 (6)	117.397 (7)
	33	**31.876 (26)**	32.431 (6)	32.202 (7)	33.248 (5)	32.045 (6)
	45	24.207 (2)	**24.080 (2)**	**24.080 (3)**	24.311 (2)	24.311 (3)
PDFA	6	**67.196 (26)**	69.353 (22)	67.575 (23)	320.415 (3)	290.567 (4)
	7	**51.250 (26)**	217.107 (10)	72.995 (11)	291.291 (7)	95.951 (8)
	27	69.462 (6)	**46.556 (14)**	**46.556 (15)**	139.871 (8)	139.871 (9)
	42	27.899 (6)	**18.803 (6)**	**18.803 (7)**	21.986 (3)	21.986 (4)
PNFA	29	**24.971 (46)**	152.996 (6)	54.912 (7)	77.588 (5)	62.063 (6)
	39	11.823 (6)	**10.565 (6)**	**10.565 (7)**	12.562 (4)	12.562 (5)
	43	37.930 (2)	**32.907 (14)**	**32.907 (15)**	36.262 (2)	36.262 (3)
	46	**12.017 (46)**	12.781 (6)	12.781 (7)	14.768 (3)	14.591 (4)
moyenne		35.886	41.507	**34.705**	54.000	47.655

Tableau 6.1 – Perplexité sur douze problèmes de PAutomaC de CH-PNFA, CH-PNFA+BW, Tenseur, Tenseur+BW et l'algorithme de BW.

6.4.1 PAutomaC

Comme au chapitre précédent, nous commençons par commenter les résultats pour la perplexité. D'abord, CH-PNFA apprend en moyenne des modèles dont la perplexité est comparable à celle de l'algorithme de BW. Malheureusement, nous remarquons que l'utilisation de CH-PNFA ou Tenseur pour initialiser l'algorithme de BW n'améliore pas systématiquement les performances par rapport à respectivement CH-PNFA et Tenseur. De plus, les performances peuvent être aussi inférieures à celle de l'algorithme de BW initialisé aléatoirement. Nous concluons que l'algorithme Tenseur et l'algorithme CH-PNFA peuvent parfois fournir une mauvaise initialisation à l'algorithme de BW. Cette propriété peut être expliquée par le fait que CH-PNFA et Tenseur ne sont pas consistants comme nous l'expliquerons en conclusions. Néanmoins, l'algorithme CH-PNFA+BW se classe en premier suivit de près par l'algorithme de BW. Les performances de l'algorithme Tenseur et Tenseur+BW sont très inférieures aux autres algorithmes.

Pour le WER, l'utilisation de l'algorithme de BW en complément de CH-PNFA et Tenseur améliore légèrement les performances. Néanmoins, celles-ci restent inférieures à celle atteintes par une initialisation aléatoire de l'algorithme de BW. Une fois de plus, CH-PNFA et Tenseur ne fournissent pas de bonnes initialisations pour l'algorithme de BW. Finalement, l'algorithme de BW surpasse les autres.

type	n°	BW	CH-PNFA	CH-PNFA+BW	Tenseur	Tenseur+BW
HMM	1	77.138 (34)	80.352 (10)	**75.288 (11)**	79.077 (3)	78.254 (4)
	14	**68.414 (14)**	71.056 (6)	69.910 (7)	76.806 (5)	69.708 (6)
	33	**74.228 (30)**	75.334 (6)	74.487 (7)	74.501 (4)	74.261 (5)
	45	**78.095 (18)**	78.394 (2)	78.141 (3)	78.448 (3)	78.141 (4)
PDFA	6	59.653 (2)	51.332 (22)	**49.133 (23)**	59.653 (1)	59.653 (2)
	7	**48.289 (34)**	58.167 (10)	53.613 (11)	82.383 (6)	62.731 (7)
	27	**73.030 (30)**	83.120 (14)	79.685 (15)	88.499 (2)	88.103 (3)
	42	**56.582 (30)**	67.598 (6)	62.977 (7)	68.436 (3)	62.850 (4)
PNFA	29	**48.504 (42)**	75.161 (6)	68.749 (7)	76.313 (3)	74.929 (4)
	39	**59.174 (38)**	64.924 (6)	64.332 (7)	75.228 (3)	68.314 (4)
	43	**77.314 (26)**	79.213 (14)	78.376 (15)	78.955 (1)	77.958 (2)
	46	**77.357 (30)**	88.818 (6)	87.342 (7)	90.900 (5)	87.624 (6)
moyenne		**66.482**	72.789	70.169	77.433	73.544

Tableau 6.2 – WER en pourcentage sur douze problèmes de PAutomaC de CH-PNFA, CH-PNFA+BW, Tenseur, Tenseur+BW et l'algorithme de BW.

6.4.2 Penn-Treebank

Sur l'ensemble de donnée Penn-Treebank, la performance de CH-PNFA, donnée Tableau 6.3, est consistante avec celles des autres algorithmes qui utilisent la série originale pour l'apprentissage. Nous avons vu au Chapitre 5 précédent que l'utilisation d'une série auxiliaire basée sur les sous-séquences permettait d'accroître davantage les performances. Dans notre version de l'algorithme, CH-PNFA ne permet que d'utiliser la série originale, ce qui explique son écart de performance avec l'algorithme de BW. Cette fois l'algorithme CH-PNFA semble fournir une initialisation intéressante pour l'algorithme de BW car le meilleur résultat est obtenu par CH-PNFA+BW. Nous notons que ce n'est pas le cas de Tenseur.

Nous terminons en commentant les temps d'exécution. L'algorithme CH-PNFA reste bien plus rapide que l'algorithme de BW. Une majeure partie du temps est consacrée à l'estimation des paramètres par programmation quadratique. En somme, bien que au moins deux fois plus lent, le temps d'exécution est comparable a celui des algorithmes basés sur la MoM et est très inférieur à celui de l'algorithme de BW.

6.4.3 Wikipédia

Pour l'ensemble de donnée Wikipédia, les Figures 6.2 et 6.3 donnent les performances de CH-PNFA pour des ensembles d'apprentissages de tailles différentes. Premièrement, nous remarquons que comme pour NNSpectral, la variance des perfor-

	BW	CH-PNFA	CH-PNFA+BW	Tenseur	Tenseur+BW
WER (%)	60.774	66.581	**59.045**	70.790	64.428
dimension	30	46	47	1	9
taille de la base	*N/A*	500	500	200	200
hankel (s)	0.00	0.82	0.89	1.10	0.87
semi-module	0.00	7.32	142.22	2.07	0.41
paramètre	1335.99	28.13	2240.62	0.09	12.48
total	1335.99	36.27	2383.73	3.25	13.76

Tableau 6.3 – Résultats pour Penn-Treebank de CH-PNFA, CH-PNFA+BW, Tenseur, Tenseur+BW et l'algorithme de BW.

mances en fonction de la dimension est plus élevées que pour les autres algorithmes. Cela est dû à l'utilisation d'une heuristique pour la résolution de la NMF. Malgré cela, nous percevons qu'une amélioration moyenne très légère lorsque la taille de l'ensemble d'apprentissage augmente. Les Figures 6.4 et 6.5 rapportent les résultats liés à l'utilisation de TENSEUR et de CH-PNFA pour l'initialisation de l'algorithme de BW. L'utilisation de TENSEUR pour initialiser BW est à proscrire car les performances à la fois pour la vraisemblance et le BPC sont inférieures en comparaison à une initialisation aléatoire. Bien que, l'utilisation de CH-PNFA n'apporte pas de gains sensibles elle permet à l'algorithme de BW de converger plus rapidement. En comparaison avec les autres algorithmes, nous remarquons que pour le BPC, CH-PNFA obtient de meilleures performances que CO, TENSEUR et SPECTRAL mais n'atteint pas celles de l'algorithme de BW, ni celles de NNSPECTRAL. Pour la vraisemblance conditionnelle, les résultats de CH-PNFA sont bien que meilleurs à ceux de TENSEUR et CO.

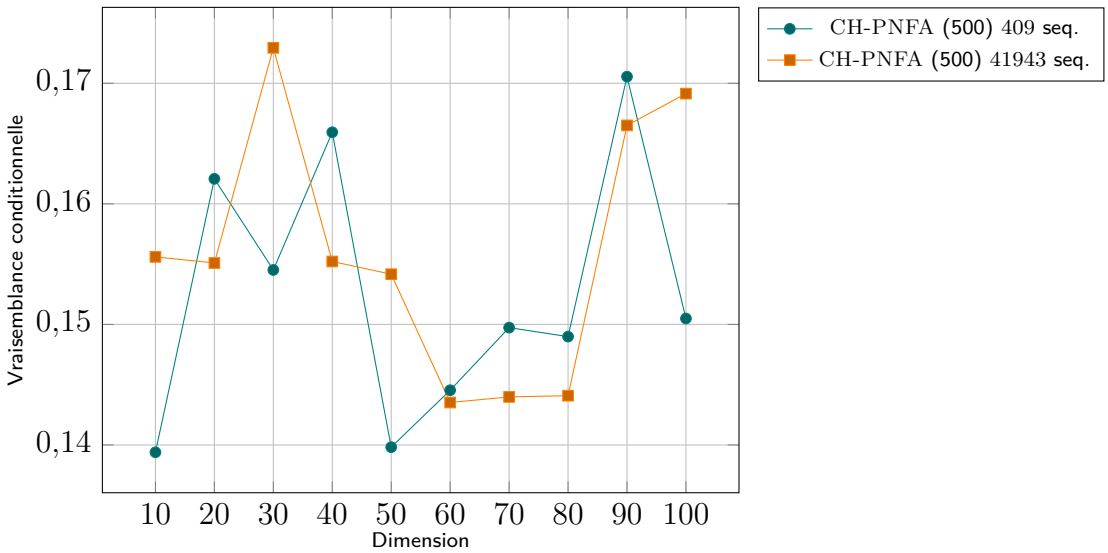


FIGURE 6.2 – Performances de CH-PNFA pour la vraisemblance conditionnelle sur 409 séquences extraites de Wikipédia en fonction de la dimension.

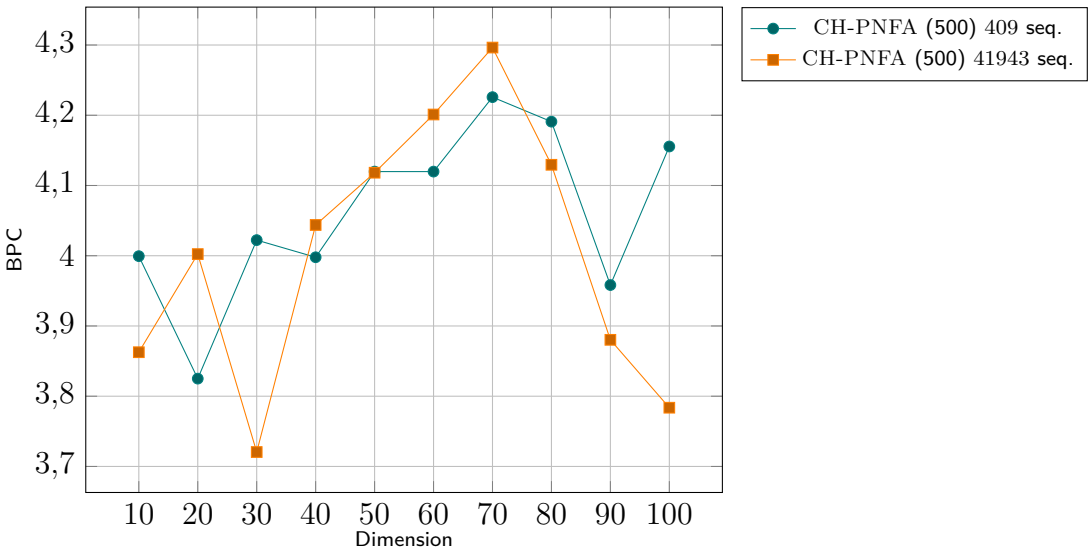


FIGURE 6.3 – Performances de CH-PNFA et NNSPECTRAL pour le BPC sur 41943 séquences extraites de Wikipédia en fonction de la dimension.

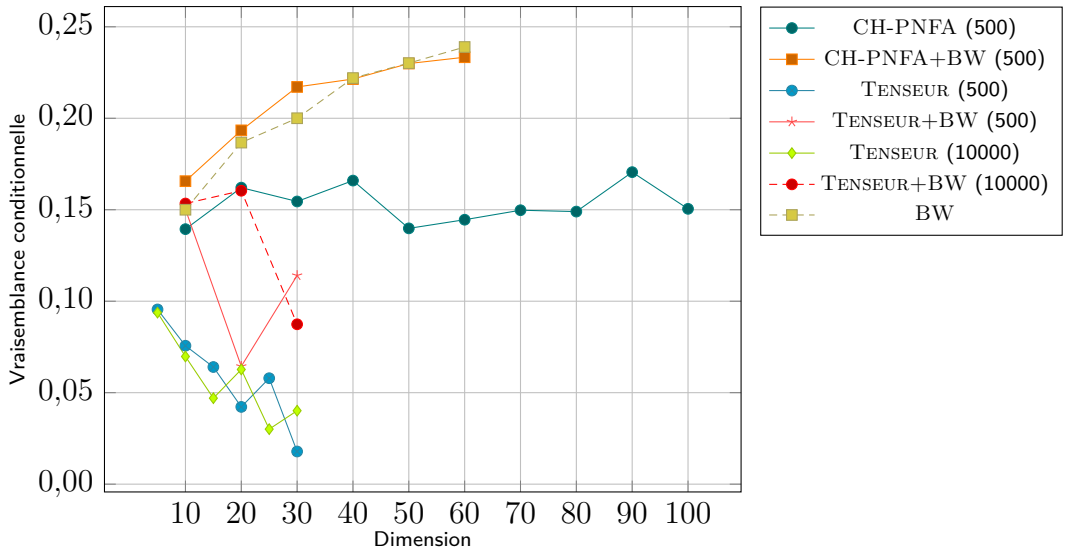


FIGURE 6.4 – Performances de CH-PNFA, CH-PNFA+BW, TENSEUR, TENSEUR+BW et l'algorithme de BW pour la vraisemblance conditionnelle sur 409 séquences de Wikipédia en fonction de la dimension et de la taille de la base (indiquée entre parenthèses).

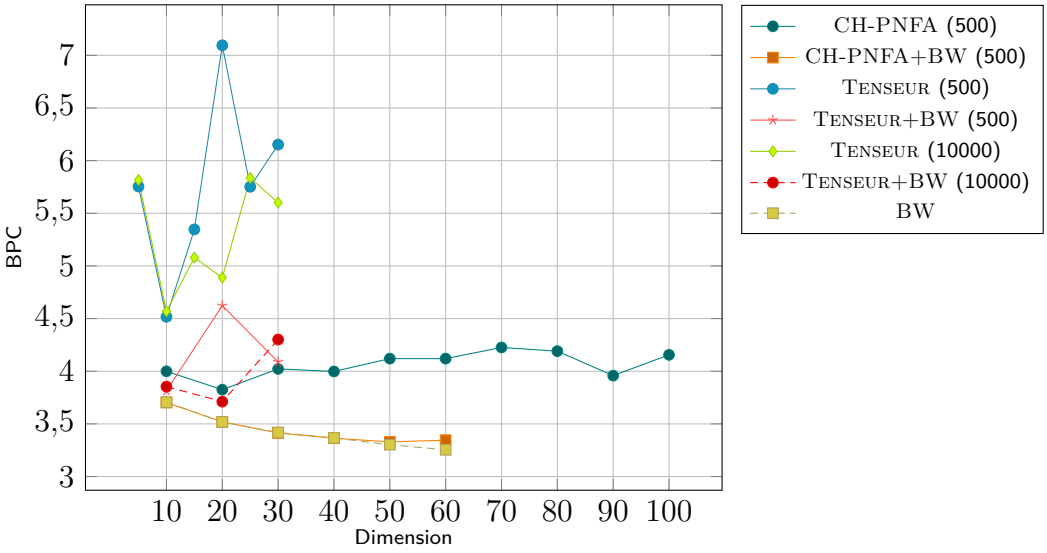


FIGURE 6.5 – Performances de CH-PNFA, CH-PNFA+BW, TENSEUR, TENSEUR+BW et l'algorithme de BW pour le BPC sur 409 séquences de Wikipédia en fonction de la dimension et de la taille de la base (indiquée entre parenthèses).

6.5 Conclusions

Dans ce chapitre, nous avons proposé l'algorithme CH-PNFA pour l'apprentissage de PNFA. Comme pour l'algorithme NNSpectral, l'algorithme s'exécute en temps polynomial mais n'est pas consistant comme le prévoient les résultats de complexité présentés au Chapitre 2. Tout comme l'algorithme NNSpectral, CH-PNFA identifie d'abord une enveloppe convexe générée par des séries formelles contenant le langage cible estimé. La différence avec l'algorithme NNSpectral tient dans la normalisation des séries afin d'obtenir une enveloppe convexe de langages stochastiques. Cette normalisation permet, par résolution d'un problème de NNLS sous contraintes linéaires, de retourner un PNFA. Comme pour NNSpectral, l'identification de l'enveloppe convexe repose sur la résolution d'un problème de NMF par des heuristiques qui peuvent converger vers un minimum local. De même, l'enveloppe convexe identifiée n'a aucune garantie d'être stable comme le requière la théorie. Cependant, le résultat de l'algorithme CH-PNFA permet d'initialiser une recherche locale par des algorithmes itératifs comme l'algorithme de BW. Cette possibilité permet parfois d'améliorer les performances de l'algorithme de BW par rapport à une initialisation aléatoire.

Du point de vue des performances, les résultats sont mitigés. En particulier, les performances sont moins bonnes que NNSpectral et que l'algorithme de BW. De plus, CH-PNFA, tout comme Tenseur, ne semble pas fournir de bonnes initialisations pour l'algorithme de BW. Dans le cas d'un apprentissage à partir d'un grand nombre de données, comme le temps d'exécution est proportionnel au nombre d'exemple, NNSpectral est bien plus rapide que l'algorithme de BW. Ainsi, lorsque NNSpectral est utilisé en initialisation, il permet néanmoins de réduire le temps de convergence de l'algorithme de BW.

Chapitre 7

Apprentissage par enveloppe convexe résiduelle

Sommaire

7.1 Introduction

Dans le chapitre précédent, nous avons proposé un algorithme d'apprentissage propre pour les PNFA. Malheureusement, la complexité calculatoire inhérente à l'inférence de langages stochastiques rationnels sur $\mathbb{R}^+$ nous contraint à l'utilisation d'heuristiques qui empêchent d'obtenir la garantie d'une convergence globale.

Motivé par la recherche d'un algorithme d'inférence PAC renvoyant un modèle probabiliste, ce chapitre est inspiré des travaux de Esposito et collab. [2002]. Les auteurs proposent un algorithme propre d'identification à la limite avec probabilité 1 des langages stochastiques de $S_{\mathbb{R}^+}^{[[Res]]}(\Sigma)$. Leur algorithme est basé sur la fusion d'états. Motivé par les performances des algorithmes issus de la MoM, nous proposons dans ce chapitre un algorithme CH-PRFA d'inférence pour les langages stochastiques de $S_{\mathbb{R}^+}^{[[Res]]}(\Sigma)$ renvoyant un PNFA et possédant des garanties PAC. Bien que l'algorithme ne soit pas propre car il ne renvoie pas un PRFA, le modèle renvoyé réalise bien une distribution et peut servir d'initialisation pour les algorithmes itératifs. Afin d'obtenir un algorithme propre, nous montrerons que l'algorithme peut facilement être modifié afin de renvoyer un PRFA préfixe (voir Définition 51). Malheureusement, la dimension du modèle renvoyé par un tel algorithme est bien plus élevée que celle du modèle renvoyé par CH-PRFA. Cela rend l'algorithme inutilisable en pratique. L'ensemble des PRFA, dont fait l'objet ce chapitre, est représenté dans la hiérarchie d'automates établie au Chapitre 1 sur la Figure 7.1.

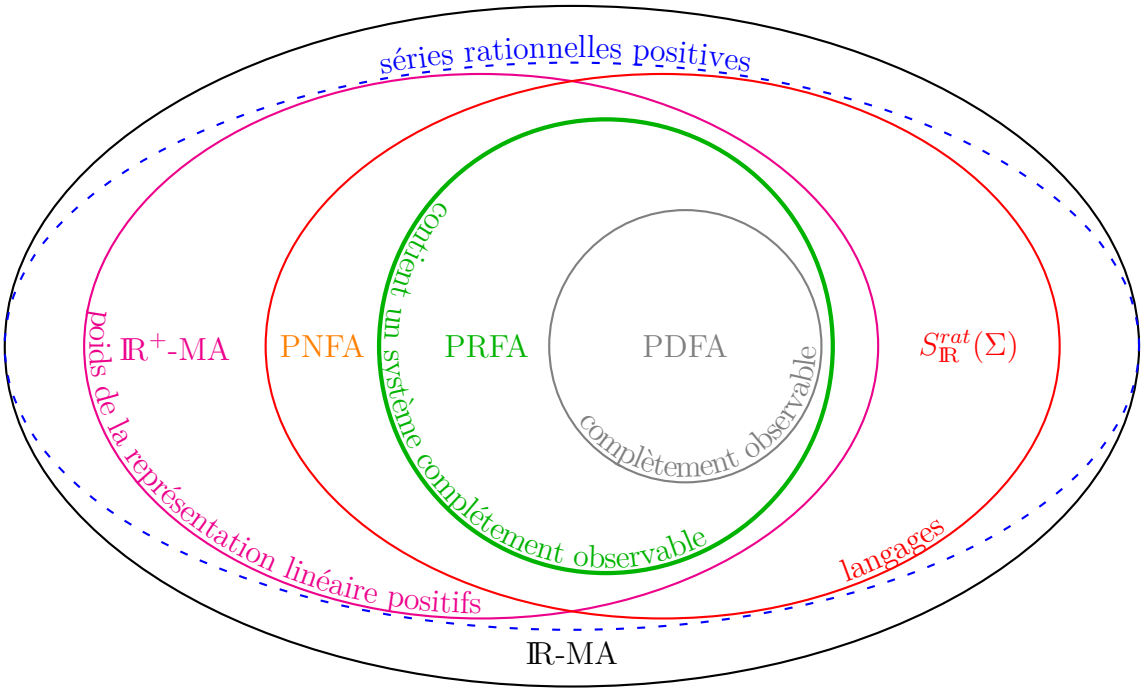


FIGURE 7.1 – Hiérarchie entres les classes d'automates. Dans ce chapitre, nous nous intéressons à la classe d'automate en vert épais.

7.2 Identification de l'enveloppe convexe générée par les langages résiduels

7.2.1 Matrices séparables et point de vue géométrique

Nous commençons par démontrer des résultats établissant la caractérisation des langages stochastiques de $S_{\mathbb{R}^+}^{[[Res]]}(\Sigma)$ par certaines matrices Hankel dites séparables. Cette section présente la notion de séparabilité. Celle-ci a été introduite par Donoho et Stodden [2003] comme une des conditions suffisantes pour que le problème de NMF soit bien posé (admette une unique factorisation). Cette condition a ensuite été utilisé par de nombreux auteurs, sous plusieurs variantes, pour proposer des algorithmes robustes de résolution de NMF.

Définition 53 (Matrice séparable)**.**
Une matrice $M \in {\mathbb{R}^+}^{m\times n}$ est dite r-séparable si elle peut s'écrire sous la forme suivante

$$M = \Pi \begin{pmatrix} I_r \\ H \end{pmatrix} W,$$

où $W \in {\mathbb{R}^+}^{r\times n}$, $H \in {\mathbb{R}^+}^{(m-r)\times r}$, Π est une matrice de permutation et $I_r \in \mathbb{R}^{r\times r}$ est la matrice identité.

On peut donner une interprétation géométrique de la séparabilité. Une matrice M est r-séparable si et seulement si le cône généré par un sous-ensemble de r lignes de M contient toutes les lignes de M. La Figure 7.2 montre l'enveloppe conique (en bleu) d'un ensemble de points (en rouge). Les croix symbolisent les lignes de M qui génèrent le cône. Les points rouges symbolisent les autres lignes de M contenues dans le cône.

Certains algorithmes nécessitent des hypothèses supplémentaires qui peuvent être satisfaites sans nuire à la généralité par une simple normalisation. Par exemple, on peut supposer que les lignes de H somment au plus à 1 et même exactement à 1. Géométriquement, $H\mathbf{1} \leq \mathbf{1}$ signifie que les lignes de M sont contenus dans l'enveloppe convexe du sous-ensemble de r lignes de M et de l'origine. Cette situation est représentée Figure 7.3, où les croix marrons sont les lignes de M générant l'enveloppe convexe en bleu. Les points marrons sont les autres lignes de M contenues dans l'enveloppe convexe.

Cette condition sur H peut être satisfaite très simplement. Sans nuire à la généralité, réordonnons les lignes de M pour faire disparaître la matrice de permutation Π. Normalisation les lignes de M non nulles de façon à ce qu'elles somment à 1. On note D_M (resp. D_W)la matrice diagonale telle que les lignes non nulles de $D_M^{-1}M$ (resp. $D_W^{-1}W$) somment exactement à 1. On obtient alors

$$D_M^{-1}M = \begin{pmatrix} D_M^{-1}I_rD_W \\ D_M^{-1}HD_W \end{pmatrix} D_W^{-1}W = \begin{pmatrix} I_r \\ D_WHD_M^{-1} \end{pmatrix} D_W^{-1}W,$$

où la seconde égalité découle du fait que W représente les r premières lignes de M. Ainsi par construction, les lignes $D_WHD_M^{-1}$ doivent sommer exactement à un sauf celles correspondantes aux lignes nulles de M qui sont, du coup, nulles aussi. Géométriquement, si l'on enlève les lignes nulles de M, cette normalisation revient à se placer dans le simplexe comme l'illustre la Figure 7.2. Les lignes représentées par des symboles rouges sont projetées sur le simplexe en vert. Les points projetés sont en marrons. Dans le simplexe, les r premières lignes de MD_M^{-1} forment un polytope convexe contenant les autres lignes de MD_M^{-1}. Cette situation est représentée sur la Figure 7.4.

7.2.2 Caractérisation par la matrice de Hankel

Au Chapitre 1, nous avons caractérisé les langages stochastiques de $S_{\mathbb{R}^+}^{[[Res]]}(\Sigma)$ par leur semi-module résiduel dans la Définition 26. Dans cette partie, nous allons utiliser une caractérisation à partir de la matrice de Hankel. La notion de séparabilité va nous permettre d'établir cette caractérisation.

Proposition 36 (Séparabilité de la matrice de Hankel d'un langage stochastique résiduel)**.**
Soit un langage stochastique p et H sa matrice de Hankel, si p est réalisée par un PRFA de dimension d alors H est d-séparable.

Démonstration. Soit un PRFA $M = (\Sigma, Q, \iota, \varphi, \tau)$ tel que $p_M = p$. Pour chaque état $q \in Q$, nous notons u_q, le plus petit mot (dans l'ordre lexicographique) vérifiant $u_q^{-1}p_M = p_{M,q}$. D'après la Définition 28, u_q existe. On note $\mathcal{R} = \{u_q | q \in Q\}$ l'ensemble de ces mots. Sans nuire à la généralité réordonnons les lignes de H tel que les d premières lignes soient associées aux mots de $\mathcal{R}$. On note S le sous bloc supérieur de H associé aux d premières lignes. D'après la Proposition 18, les autres lignes de H sont contenues dans $[[S]]$, il existe donc une matrice Q telle que,

$$H = \begin{pmatrix} S \\ QS \end{pmatrix} = \begin{pmatrix} I_r \\ Q \end{pmatrix} S.$$

H est donc d-séparable. □

Proposition 37 (Passage d'une matrice de Hankel à la représentation linéaire pour les PRFA)**.**
Soit $H \in \mathbb{R}^{\Sigma^\star \times \Sigma^\star}$ une matrice de Hankel d-séparable, on note $\mathcal{R}$ l'ensemble des d mots associés aux lignes générant le cône contenant les lignes de H. On note $S \in \mathbb{R}^{\mathcal{R}\times\Sigma^\star}$ la matrice construite à partir des lignes de H correspondant aux mots de $\mathcal{R}$. De même, on note $S_\sigma \in \mathbb{R}^{\mathcal{R}\times\Sigma^\star}$ la matrice construite à partir des lignes de H correspondant aux mots de $\mathcal{R}\sigma$. Soit un langage stochastique p tel que $H[u, v] = p(uv)$, alors $p \in S_{\mathbb{R}^+}^{[[Res]]}(\Sigma)$ et est réalisée par un PRFA dont la représentation linéaire $(\boldsymbol{\alpha}_0, A, \boldsymbol{\alpha}_\infty)$ de dimension d vérifie

(i) $Z_\sigma = A_\sigma Z$,

(ii) $\boldsymbol{\alpha}_0^\top Z = \mathbf{1}_\varepsilon^\top H$,

(iii) $\boldsymbol{\alpha}_\infty = Z\mathbf{1}_\varepsilon$,

où $Z = \mathrm{diag}(S\mathbf{1})^{-1}S$ *et* $Z_\sigma = \mathrm{diag}(S\mathbf{1})^{-1}S_\sigma$.

Démonstration. Comme les lignes de S sont les $\{\dot{u}p | u \in \mathcal{R}\}$ alors $S\mathbf{1} = (\dot{u}p(\Sigma^\star))_{u\in\mathcal{R}}$. Nous identifions donc les lignes de Z aux langages stochastiques résiduels $R = \{u^{-1}p\}_{u\in\mathcal{R}}$. Ainsi, ces lignes définissent une famille finie R de résiduels de p générant un semi-module contenant toutes les lignes de H identifiables à $\{\dot{u}p | u \in \Sigma^\star\}$. Comme $[[\{\dot{u}p | u \in \Sigma^\star\}]]$ et $[[\{\mathrm{Res}(p)\}]]$ coïncident, nous avons que $[[\{\mathrm{Res}(p)\}]] \subset [[R]]$. Comme R et $\mathrm{Res}(p)$ sont des familles de langages stochastiques, nous avons même $[[\{\mathrm{Res}(p)\}]] \subset \mathrm{conv}(R)$. Nous pouvons dès lors appliquer la Proposition 19 en remarquant que les lignes de Z_σ sont identifiables à $\dot{\sigma}R = \{\dot{\sigma}(u^{-1}p) | u \in \mathcal{R}\}$ et que $\mathbf{1}_\varepsilon^\top H$ est identifiable à p. Nous obtenons ainsi que la représentation linéaire définie par $Z_\sigma = A_\sigma Z$, $\boldsymbol{\alpha}_0^\top Z = \mathbf{1}_\varepsilon^\top H$ et $\boldsymbol{\alpha}_\infty = Z\mathbf{1}_\varepsilon$ définit un PRFA. □

Théorème 17 (Caractérisation des langages stochastiques de $S_{\mathbb{R}^+}^{[[Res]]}(\Sigma)$ par la matrice de Hankel)**.**
Soit p un langage stochastique, et $H[u,v] = r(uv)$ sa matrice de Hankel, alors p est réalisée par un PRFA de dimension d si et seulement si H est d-séparable.

Démonstration. Conséquence directe des Propositions 36 et 37. □

7.2.3 Caractérisation par les matrices de Hankel finies

Dans le cas où l'on travaille à partir d'une base finie, il est possible d'obtenir un résultat similaire si la base remplit certaines conditions. Dans la suite, nous supposons pour simplifier que les bases finies considérées contiennent le mot vide $(\varepsilon, \varepsilon)$.

Définition 54 (Base résiduelle)**.**
Soit un langage stochastique $p \in S_{\mathbb{R}^+}^{[[Res]]}(\Sigma)$, et $\mathcal{B} = (\mathcal{P}, \mathcal{S})$ une base, nous notons $\mathrm{Res}_{\mathcal{B}}(p)$ l'ensemble des résidus de p sur $\mathcal{B}$ défini par $\mathrm{Res}_{\mathcal{B}}(p) = \{(u^{-1}p(v))_{v\in\mathcal{S}} \,|u \in \mathcal{P} \cap \mathrm{res}(p)\}$. Une base est résiduelle si la projection des arêtes du cône $[[\mathrm{Res}(p)]]$ sur $\mathcal{S}$ coïncident avec les arêtes du cône $[[\mathrm{Res}_{\mathcal{B}}(p)]]$.

Si, une base résiduelle est de plus complète alors les cônes $[[\mathrm{Res}(p)]]$ et $[[\mathrm{Res}_{\mathcal{B}}(p)]]$ ont le même nombre d'arêtes. Ainsi, les $\{(u^{-1}p(v))_{v\in\mathcal{S}} \,|u \in \mathcal{P} \cap \mathrm{res}(p)\}$ suffisent pour identifier les préfixes générant les arêtes de $[[\mathrm{Res}(p)]]$.

Alors que la Proposition 36 se généralise sans difficulté aux bases finies quelconques, la Proposition 37 doit être adaptée aux matrices de Hankel définies sur une base résiduelle.

Proposition 38 (Passage des matrices de Hankel finies à la représentation linéaire pour les PRFA)**.**
Soit une base $\mathcal{B} = (\mathcal{P}, \mathcal{S})$, p un langage stochastique, on note $H_{\mathcal{B}} \in \mathbb{R}^{\mathcal{P}\times\mathcal{S}}$ sa matrice de Hankel telle que

$$H_{\mathcal{B}}[u,v] = p(uv),$$

pour tout $\sigma \in \Sigma$, on note $H_{\mathcal{B}}^{\sigma} \in \mathbb{R}^{\mathcal{P}\times\mathcal{S}}$ la matrice telle que

$$H_{\mathcal{B}}^{\sigma}[u,v] = p(u\sigma v)$$

et on note $\mathbf{h}_{\mathcal{P}}^{\Sigma^\star} \in \mathbb{R}^{\mathcal{P}}$ le vecteur défini par

$$\mathbf{h}_{\mathcal{P}}^{\Sigma^\star}[u] = p(u\Sigma^\star).$$

Soit $\mathcal{R}$ l'ensemble des d mots associés aux lignes générant le cône contenant les lignes de H, on note $S \in \mathbb{R}^{\mathcal{R}\times\mathcal{S}}$ la matrice construite à partir des lignes de $H_{\mathcal{B}}$ correspondantes aux mots de $\mathcal{R}$ et $S_\sigma \in \mathbb{R}^{\mathcal{R}\times\mathcal{S}}$ la matrice construite à partir des lignes de $H_{\mathcal{B}}^{\sigma}$ correspondantes aux mots de $\mathcal{R}$. Si $\mathcal{B}$ est complète et résiduelle, alors $p \in S_{\mathbb{R}^+}^{[[Res]]}(\Sigma)$ et est réalisé par un PRFA dont la représentation linéaire $(\boldsymbol{\alpha}_0, A, \boldsymbol{\alpha}_\infty)$ de dimension d vérifie

(i) $Z_\sigma = A_\sigma Z$,

(ii) $\boldsymbol{\alpha}_0^\top Z = \mathbf{1}_\varepsilon^\top H_{\mathcal{B}}$,

(iii) $\boldsymbol{\alpha}_\infty = Z\mathbf{1}_\varepsilon$.

où $Z = \mathrm{diag}(\mathbf{h}_{\mathcal{P}}^{\Sigma^\star})^{-1}S$ et $Z_\sigma = \mathrm{diag}(\mathbf{h}_{\mathcal{P}}^{\Sigma^\star})^{-1}S_\sigma$.

Démonstration. Premièrement, nous notons R la famille finie formée par $\{u^{-1}p | u \in \mathcal{R}\}$ et R_σ la famille finie formée par $\{u^{-1}p | u \in \mathcal{R}\}$. Nous vérifions que les lignes de Z sont identifiables à la famille finie de langages stochastiques de R projetées sur $\mathcal{S}$. De même, les lignes de Z_σ sont identifiables à la famille finie de langages stochastiques de R_σ projetées sur $\mathcal{S}$.

Comme la base est complète et résiduelle, nous avons que $[[R]] = [[\mathrm{Res}(p)]]$. Ainsi, d'après la Proposition 19, p est réalisé par un PRFA de dimension d.

Nous notons alors Z' et Z'_σ les matrices de la Proposition 37 extraites de la matrice de Hankel infinie H. Nous remarquons que Z et Z_σ sont des sous-blocs de Z' et Z'_σ respectivement. Comme la base est complète, ce sont des sous-blocs de rang plein. De plus, nous remarquons que $ZZ^\dagger$ est un projecteur orthogonal sur l'image de Z. Comme la base est complète Z et Z' ont la même image et donc $ZZ^\dagger Z' = Z'$. Par le même raisonnement, nous montrons que pour tout $\sigma \in \Sigma$, $Z_\sigma {Z_\sigma}^\dagger Z'_\sigma = Z'_\sigma$.

Comme Z et Z_σ sont des sous-blocs de rang plein de Z' et Z'_σ respectivement et que l'opération $\dot{\sigma}$ est linéaire, alors il existe une matrice $N \in \mathbb{R}^{\mathcal{S} \times (\Sigma^\star \setminus \mathcal{S})}$ telle que

$$Z' = \begin{pmatrix} Z & ZN \end{pmatrix},$$
$$Z'_\sigma = \begin{pmatrix} Z_\sigma & Z_\sigma N \end{pmatrix}.$$

Cela permet de montrer que,

$$A_\sigma Z {Z_\sigma}^\dagger Z'_\sigma = \begin{pmatrix} A_\sigma Z {Z_\sigma}^\dagger Z_\sigma & A_\sigma Z {Z_\sigma}^\dagger Z_\sigma N \end{pmatrix} = \begin{pmatrix} A_\sigma Z & A_\sigma Z N \end{pmatrix} = A_\sigma Z'.$$

La deuxième égalité provient du fait que ${Z_\sigma}^\dagger Z_\sigma$ et $\left({Z_\sigma}^\dagger Z_\sigma\right)^\top$ sont un projecteur orthogonal (donc auto-adjoint) sur l'image de $Z_\sigma^\top$ et que l'image de $Z_\sigma^\top$ et $(A_\sigma Z)^\top$ coïncident. Ainsi, nous avons la relation $\left({Z_\sigma}^\dagger Z_\sigma\right)^\top (A_\sigma Z)^\top = (A_\sigma Z)^\top$ qui, transposé, donne $A_\sigma Z {Z_\sigma}^\dagger Z_\sigma = A_\sigma Z$.

Ces trois relations vont nous permettre de montrer le résultat. Premièrement, nous avons

$$Z_\sigma = A_\sigma Z$$
$$Z_\sigma {Z_\sigma}^\dagger = A_\sigma Z {Z_\sigma}^\dagger$$
$$Z_\sigma {Z_\sigma}^\dagger Z'_\sigma = A_\sigma Z {Z_\sigma}^\dagger Z'_\sigma.$$

Comme $Z_\sigma {Z_\sigma}^\dagger Z'_\sigma = Z'_\sigma$ nous obtenons que

$$Z'_\sigma = B_\sigma Z {Z_\sigma}^\dagger Z'_\sigma.$$

Or, nous avons que $Z {Z_\sigma}^\dagger Z'_\sigma = Z'$.

Puis, nous avons simplement que,

$$\boldsymbol{\alpha}_\infty = Z \mathbf{1}_\varepsilon$$
$$\boldsymbol{\alpha}_\infty = Z' \mathbf{1}_\varepsilon,$$

car la première colonne de Z et Z' coïncide.

Enfin, nous avons

$$\boldsymbol{\alpha}_0^\top Z = \mathbf{1}_\varepsilon^\top H_{\mathcal{B}}$$
$$\boldsymbol{\alpha}_0^\top Z = \mathbf{1}_\varepsilon^\top Q Z$$
$$\boldsymbol{\alpha}_0^\top Z = \mathbf{1}_\varepsilon^\top Q' Z,$$

où Q' et Q sont les matrices telles que $H = Q'Z'$ et $H_{\mathcal{B}} = QZ$. La dernière égalité est établie par le fait que la première ligne de Q et Q' coïncident. Alors, nous pouvons écrire

$$\begin{aligned}\boldsymbol{\alpha}_0^\top Z Z^\dagger Z' &= \mathbf{1}_\varepsilon^\top Q' Z Z^\dagger Z' \\ \boldsymbol{\alpha}_0^\top Z' &= \mathbf{1}_\varepsilon^\top Q' Z' \\ \boldsymbol{\alpha}_0^\top Z' &= \mathbf{1}_\varepsilon^\top H,\end{aligned}$$

car $ZZ^\dagger Z' = Z'$.

Nous avons montré que $(\boldsymbol{\alpha}_0, A, \boldsymbol{\alpha}_\infty)$ satisfait les équations de Proposition 19, c'est donc un PRFA de dimension d qui réalise p. □

7.2.4 Algorithme CH-PRFA

Dans cette section, nous détaillons l'algorithme CH-PRFA en suivant le même raisonnement que l'algorithme CH-PNFA. À la différence des précédents algorithmes, CH-PRFA se base sur la caractérisation établie dans la section précédente liée à l'identification des arêtes de l'enveloppe conique des résiduels.

Ainsi, l'algorithme CH-PRFA commence par l'estimation des matrices $H_{\mathcal{B}}$, pour tout $\sigma \in \Sigma$, $H_{\mathcal{B}}^\sigma$ et du vecteur $\mathbf{h}_{\mathcal{P}}^{\Sigma^\star}$. Puis l'identification de l'enveloppe conique résiduelle est réalisée par un algorithme de NMF pour matrices d-séparables. Bien que l'identification puisse être réalisée uniquement à partir de $H_{\mathcal{B}}$, nous préférons, pour des raisons de stabilité, travailler sur une autre matrice. Nous posons, alors,

$$G = \begin{pmatrix} H_{\mathcal{B}} & H_{\mathcal{B}}^{\sigma_1} & \cdots & H_{\mathcal{B}}^{\sigma_{|\Sigma|}} \end{pmatrix}.$$

Cette concaténation horizontale des matrices permet de contraindre le cône généré à être approximativement stable. Une astuce similaire est utilisé au Chapitre 3. Selon l'algorithme de NMF pour matrices séparables que l'on utilise, nous serons amené à normaliser G différemment. Certains algorithmes sont capables de rechercher directement une enveloppe conique, cependant ils ne passent pas à l'échelle. D'autres algorithmes nécessitent de se placer dans le simplexe en normalisant les lignes de G. La section suivante détaille les propriétés de ces différents algorithmes. Les algorithmes que nous proposons d'utiliser supposent que les lignes forment une enveloppe convexe. C'est le cas pour D définie par,

$$D = \operatorname{diag}(\mathbf{h}_{\mathcal{P}}^{\Sigma^\star})^{-1} G.$$

Cette étape se traduit par le passage d'une probabilité jointe $(up(v))$ à une probabilité conditionnelle $(u^{-1}p(v))$ en divisant par le coefficient correspondant de $\mathbf{h}_{\mathcal{P}}^{\Sigma^\star}$ $(p(u\Sigma^\star))$. Ainsi, nous chercherons parmi les lignes de D, celles définissant, avec l'origine, l'enveloppe convexe contenant toutes les lignes de D. Ces lignes correspondent à des langages résiduels. De ces langages résiduels, nous extrayons les matrices Z et Z_σ de la Proposition 38. Finalement, les coefficients de la représentation linéaire sont retrouvés par moindres carrées sous contraintes, où les contraintes sont celles qui définissent un PNFA (Équation (1.1)). Ces étapes sont résumées dans l'Algorithme 4.

Algorithme 4 Algorithme CH-PRFA

Entrées Un alphabet Σ, un ensemble de mots d'apprentissage, un rang estimé d
Sortie Un PNFA $\langle \Sigma, \{1..d\}, \left\{\hat{A}_\sigma\right\}_{\sigma\in\Sigma}, \hat{\boldsymbol{\alpha}}_0, \hat{\boldsymbol{\alpha}}_\infty \rangle$

1: Choisir une base de préfixes $\mathcal{P} \subset \Sigma^\star$ et de suffixes $\mathcal{S} \subset \Sigma^\star$ contenant ε.
2: Estimer $\hat{H}_\mathcal{B}$, $\hat{\mathbf{h}}_\mathcal{P}^{\Sigma^\star}$ et $\forall\sigma \in \Sigma$, $\hat{H}_\mathcal{B}^\sigma$.
3: $\hat{G} \leftarrow \begin{pmatrix} \hat{H}_\mathcal{B} & \hat{H}_\mathcal{B}^{\sigma_1} & \cdots & \hat{H}_\mathcal{B}^{\sigma_{|\Sigma|}} \end{pmatrix}$
4: $\hat{D} \leftarrow \mathrm{diag}(\hat{\mathbf{h}}_\mathcal{P}^{\Sigma^\star})^{-1}\hat{G}$
5: Retrouver, à partir de $\hat{D}$, les d lignes de D qui définissent avec l'origine l'enveloppe convexe contenant toutes les lignes de D. Stocker les préfixes correspondant à ces lignes dans $\hat{\mathcal{R}}$.
6: Extraire $\hat{Z}$ et $\forall\sigma \in \Sigma$, $\hat{Z}_\sigma$ des lignes de $\hat{D}$ indexées par $\hat{\mathcal{R}}$.
7: $\hat{\boldsymbol{\alpha}}_\infty \leftarrow \hat{Z}\mathbf{1}_\varepsilon$
8: $\hat{\boldsymbol{\alpha}}_0 \leftarrow \mathrm{argmin}_{\boldsymbol{\alpha}_0\in\mathbb{R}^{+d}} \left\|\boldsymbol{\alpha}_0^\top \hat{Z} - \mathbf{1}_\varepsilon^\top \hat{H}_\mathcal{B}\right\|_2$ tel que $\boldsymbol{\alpha}_0^\top \mathbf{1} = 1$
9: $\left\{\hat{A}_\sigma\right\}_{\sigma\in\Sigma} \leftarrow \mathrm{argmin}_{\left\{A_\sigma\in\mathbb{R}^{+d\times d}\right\}_{\sigma\in\Sigma}} \sum_{\sigma\in\Sigma} \left\|A_\sigma\hat{Z} - \hat{Z}_\sigma\right\|_F$ tel que $\hat{\boldsymbol{\alpha}}_\infty + A_\Sigma\mathbf{1} = \mathbf{1}$

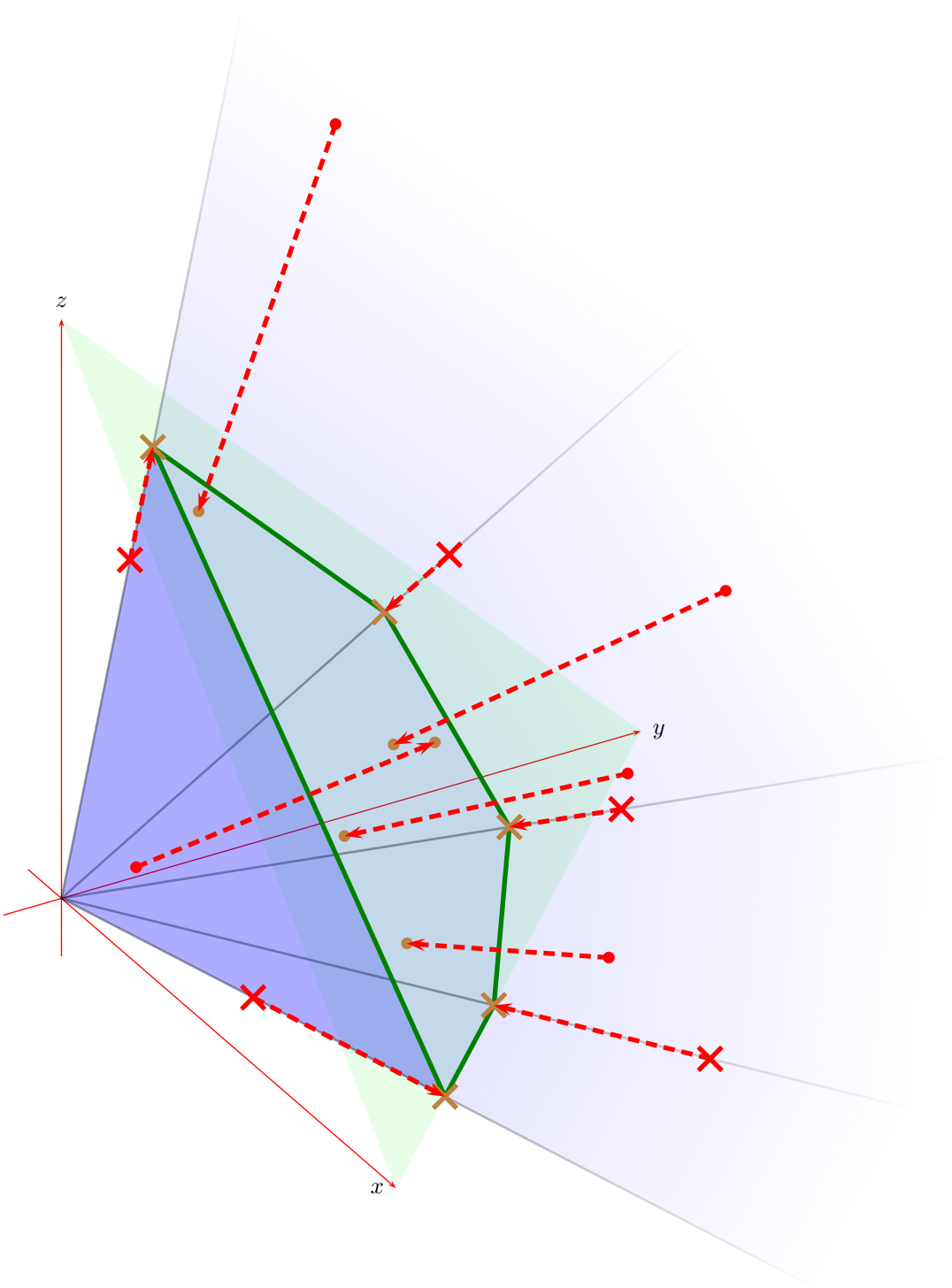


FIGURE 7.2 – Enveloppe conique d'un ensemble de points et projection sur le simplexe. Les séries $\{\dot{u}p | u \in \mathcal{P}\}$, formant les lignes de $H_{\mathcal{B}}$, sont représentées sous forme de vecteurs définis sur $\mathcal{S}$ en rouge. Les croix rouges sont les séries $\{\dot{u}p | u \in \mathcal{R}\}$. Elles supportent l'enveloppe conique contenant les autres séries, représentées par les points rouges. En marron, sont représentés les langages stochastiques résiduels $\{u^{-1}p | u \in \mathcal{P}\}$, sous forme de vecteurs définis sur $\mathcal{S}$. Comme ils représentent des distributions conditionnelles, ces vecteurs appartiennent au simplexe, représenté en vert. De même, les langages stochastiques résiduels qui supportent l'enveloppe convexe dans le plan du simplexe, sont indiqués par des croix marrons. L'enveloppe convexe résultante est dessinée par un trait gras vert. La projections des vecteurs rouges en vecteurs marrons est indiquée par des flèches rouges en pointillées.

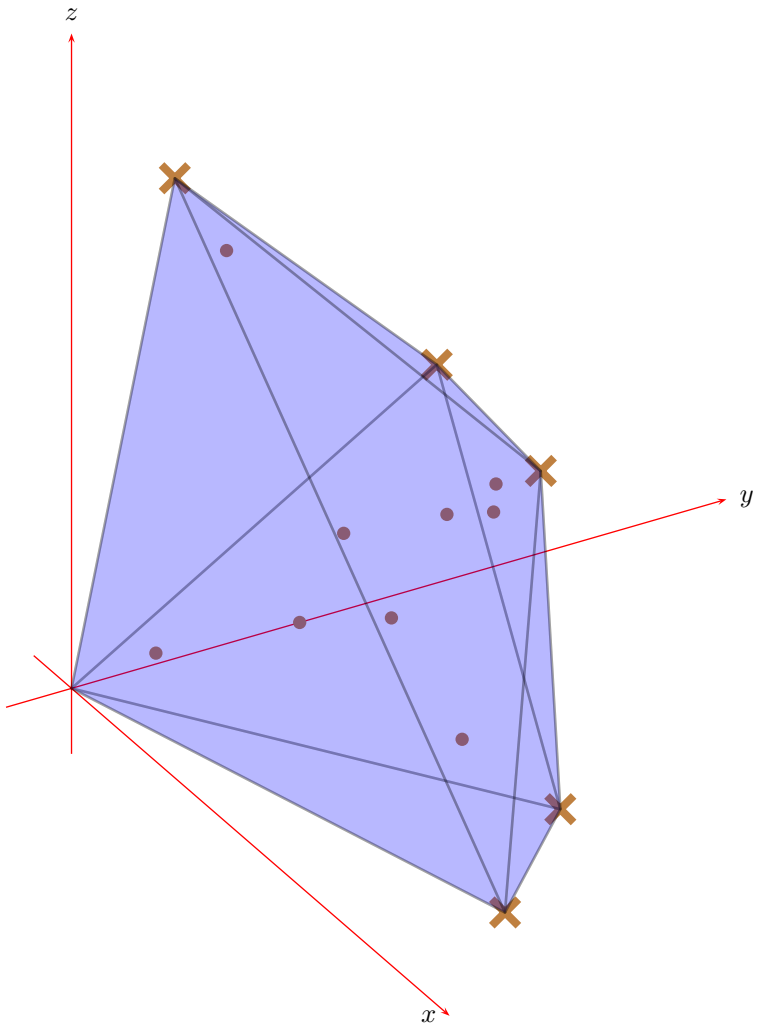


FIGURE 7.3 – Enveloppe convexe d'un ensemble de vecteurs et de l'origine. Les séries $\{\dot{u}p|u \in \mathcal{P}\}$, formant les lignes de $H_{\mathcal{B}}$, sont représentées sous forme de vecteurs définis sur $\mathcal{S}$ en marrons. Les croix marrons sont les séries $\{\dot{u}p|u \in \mathcal{R}\}$. Elles supportent l'enveloppe convexe contenant les autres séries (et l'origine), représentées par les points marrons.

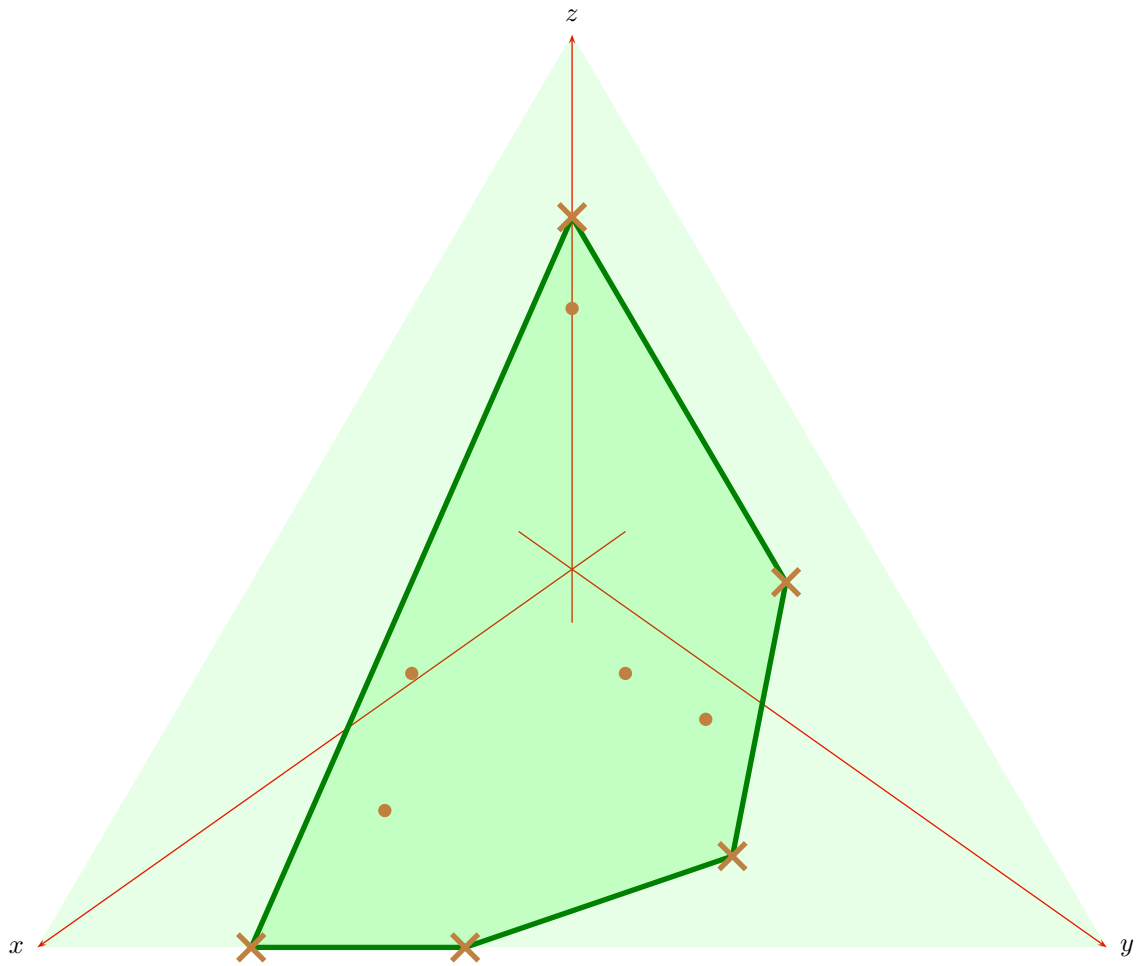


FIGURE 7.4 – Enveloppe convexe sur le simplexe contenant un ensemble de points. Les séries $\{u^{-1}p|u \in \mathcal{P}\}$, formant les lignes de $H_{\mathcal{B}}$, sont représentées sous forme de vecteurs définis sur $\mathcal{S}$ en marron. Les croix marrons sont les séries $\{u^{-1}p|u \in \mathcal{R}\}$. Elles supportent l'enveloppe convexe dans le simplexe contenant les autres séries, représentées par les points marrons. On voit bien sur cette figure que l'enveloppe convexe forme un polytope fini, ce ne serait pas le cas si les séries $\{u^{-1}p|u \in \mathcal{P}\}$ formaient un cercle. Dans ce cas, p ne serait pas réalisé par un PRFA.

Comme dans le chapitre précédent pour CH-PNFA, les minimisations des étapes 8 et 9 de l'Algorithme 4 sont effectuées par programmation quadratique. L'étape 5 fait l'objet de la section suivante.

7.2.5 Factorisation de matrices séparables

Dans cette section, nous nous intéressons aux conditions suffisantes sur la matrice de Hankel permettant à certains algorithmes de retrouver les lignes génératrices de l'enveloppe convexe à l'étape 5 de l'Algorithme 4. Nous donnons une liste non-exhaustives des algorithmes de factorisation de matrice séparables regroupés selon leur fonctionnement.

La séparabilité a été depuis plus d'une décennie identifiée comme une propriété clef de l'unicité de la décomposition en facteurs non-négatifs d'une matrice non-négative [Donoho et Stodden, 2003]. Cette condition a été identifiée indépendamment par plusieurs communautés. Elle apparaît pour la première fois dans le cadre du désentrelacement d'images hyper-spectrales. Dans [Boardman et collab., 1993], il est fait l'hypothèse que certains pixels ne contiennent la signature spectrale que d'une composante. Pour l'apprentissage de PRFA minimaux, Denis et Esposito [2008] établissent la Proposition 16 qui démontre l'unicité de l'espace d'état représenté comme une ensemble de préfixes. Les PRFA sont caractérisés par le fait que certaines séquences d'observations n'aboutissent que dans un seul état latent. La séparabilité a aussi été très utilisées pour la découverte de thèmes dans les documents depuis l'algorithme de Arora et collab. [2012b] inspiré de l'algorithme de Arora et collab. [2012a] qui traite de la NMF en général. Il est fait l'hypothèse que certains mots n'apparaissent que pour un seul thème. C'est souvent le cas des mots techniques. Elle intervient aussi dans des méthodes de séparation aveugle de sources [Chan et collab., 2008], où l'on fait l'hypothèse qu'à certains instants dans le temps une seule source est active. Cohen et Collins [2014] ont utilisé la séparabilité pour l'apprentissage de grammaires non-contextuelles probabilistes à variables latentes. Ce précédent travail est celui qui se rapproche le plus du nôtre. Enfin, la séparabilité intervient aussi pour l'apprentissage de classements [Ding et collab., 2015b]. Dans tous ces domaines, l'hypothèse de séparabilité est souvent vérifiée sur des données réelles. Pour la découverte de thèmes dans les documents, cela a été d'abord vérifié expérimentalement [Ding et collab., 2014] puis théoriquement [Ding et collab., 2015a].

En général, la séparabilité intervient comme une alternative à la décomposition en valeurs singulières. Cette dernière permet de trouver des axes formant un sous-espace de faible dimension décrivant comment les données varient en moyenne. Sous l'hypothèse de séparabilité, la recherche des arêtes d'une enveloppe conique permet de décrire les données en fonction d'exemples extrêmes. Il semble que, dans de nombreux domaines, les données sont mieux décrites par leur distance à des exemples extrêmes plutôt que par leurs variations par rapport à l'exemple moyen.

Cependant, bien que la séparabilité implique l'unicité et donc que le problème est bien posé, cette condition n'est pas suffisante pour permettre l'identification des arêtes de l'enveloppe conique par des algorithmes à partir d'une matrice bruitée. On s'intéresse ainsi à des algorithmes en fonction de leur complexité calculatoire, de leur robustesse au bruit et des hypothèses additionnelles faites.

Dans la suite, nous notons $M \in \mathbb{R}^{m \times n}$ une matrice d-séparable. D'après la

Définition 53, il existe Π, H et W telles que

$$M = \Pi \begin{pmatrix} I_r \\ H \end{pmatrix} W,$$

où $W \in \mathbb{R}^{+r \times n}$, $H \in \mathbb{R}^{+(m-r) \times r}$, Π est une matrice de permutation. On note aussi $\mathcal{K}$ l'ensemble d'indices tel que W correspond aux lignes de M indexées par $\mathcal{K}$. On peut alors écrire $M = YM$, où seules les colonnes de Y indexées par $\mathcal{K}$ sont non nulles.

On s'intéresse au cas où la matrice M est estimée. On note alors la matrice transmise à l'algorithme de factorisation $X = M + N$, où N est la matrice d'erreur. On note ϵ_i le maximum de la norme ℓ_i sur les lignes de N.

7.2.5.1 Approches par programmation linéaire

Les approches par programmation linéaire font les hypothèses les moins restrictives sur la matrice à factoriser mais sont en général les plus lents bien qu'ils possèdent une complexité polynomiale. La plupart des travaux cherchent Y telle que $M = YM$ en imposant à Y d'être positive et d'avoir peu de colonnes non nulles. Les approches par programmation linéaire sont principalement basées sur la recherche de Y telle que $M = YM$ où la séparabilité est imposée par des contraintes de parcimonie sur Y.

Les algorithmes présentés dans cette section supposent que M est simplicial. C'est-à-dire que les lignes de W sont à une distance (généralement ℓ_1) suffisante de l'enveloppe convexe générée par les autres lignes de W. Cette condition est à rapprocher de la distinguabilité dans les PDFA (voir Chapitre 2). Cette condition implique pour un PRFA A que le langage $p_{q,A}$ associé à chaque état q peut être facilement distinguer de tous les autres langages $\{p_{q',A}\}_{q' \in Q \setminus \{q\}}$. Contrairement aux PDFA, où la distinction entre états est faite deux-à-deux, ici la distinction doit être globale. Dans la suite, nous notons α la distance minimale entre les lignes de W et l'enveloppe convexe générée par les lignes complémentaires de W. Dans le cas où la matrice X est bruitée, Gillis [2013] montre que $\epsilon_1 < \frac{\alpha}{2}$ est une condition nécessaire pour être capable de distinguer les différentes lignes de X. On appelle α le paramètre simplicial. Dans certains travaux la définition de ce paramètre peut varier légèrement.

Arora et collab. [2012a] proposent une méthode qui requière la résolution de m programmes linéaires à m variables. Cette complexité le rend obsolète pour des problèmes où $m > 100$. L'algorithme travaille dans le simplexe et donc requière une normalisation des colonnes. Cette normalisation peut introduire des distorsions dans les données. Enfin, cet algorithme a besoin de connaître une estimation *a priori* de l'erreur ϵ_1 et de α. Les auteurs montrent que si $\epsilon_1 < \mathcal{O}(\alpha^2)$ alors l'erreur maximale commise sur les lignes extraites en norme ℓ_1 est inférieure à $\mathcal{O}\left(\frac{\epsilon_1}{\alpha}\right)$.

Les deux travaux qui suivent utilisent une hypothèse supplémentaire pour identifier les colonnes en ne résolvant qu'un seul problème de programmation linéaire. Cette résolution unique permet de traiter de problèmes de plus grande taille. L'hypothèse faite est que Y est à diagonale dominante (les termes diagonaux sont supérieurs aux coefficients dans leurs colonnes).

Recht et collab. [2012] proposent une formulation en un seul problème d'optimisation convexe à m^2 variables qui peut se résoudre exactement par programmation linéaire. Les auteurs proposent aussi une résolution par descente de gradient incrémentale pour les problèmes plus larges. Cette descente de gradient est facilement parallélisable et forme l'algorithme HOTTOPIXX. Cependant, leurs algorithmes ont besoin de connaitre la dimension d de la factorisation *a priori* et une estimation de ϵ_1. De même,

leurs algorithmes travaillent dans le simplexe et donc requièrent une normalisation des colonnes. En somme, la complexité calculatoire et les performances empiriques sont meilleures que pour l'algorithme de Arora et collab. [2012a]. La robustesse de HOTTOPIXX a été analysée par Gillis [2013] qui a introduit un nouveau paramètre γ_1 qui est égal à la distance minimale en norme ℓ_1 entre les lignes de W. Ainsi, on a toujours $\gamma_1 \geq \alpha$. L'auteur montre alors que si $\epsilon_1 < \mathcal{O}\left(\frac{\alpha\gamma_1}{d}\right)$ alors l'erreur maximale commise sur les lignes extraites en norme ℓ_1 est inférieure à $\mathcal{O}\left(d\frac{\epsilon_1}{\alpha}\right)$.

Enfin, Gillis et Luce [2014] s'attaquent aux défauts de HOTTOPIXX en proposant un algorithme qui ne nécessite pas de normalisation et ajuste automatiquement la dimension d de la factorisation. La formulation du problème permet d'incorporer différents modèles de bruit et de détecter les valeurs aberrantes. Enfin, le seul paramètre requit par l'algorithme est une estimation du bruit ϵ_1. Les auteurs fournissent aussi une analyse de la robustesse semblable à celle de HOTTOPIXX.

7.2.5.2 Approches récursives

Les approches récursives se caractérisent par l'identification successive des indices de $\mathcal{K}$. Ainsi la solution de dimension $d-1$ peut être extraite directement de la solution de dimension d. Cette caractéristique offre un avantage certain lorsque la dimension optimale est inconnue et doit, par exemple, être trouvée par validation croisée. Les algorithmes récursifs sont élaborés à partir du point de vue géométrique de la séparabilité. En pratique, ces algorithmes ne font intervenir que des opérations de projection et d'algèbre linéaire les rendant particulièrement rapides. Malheureusement, les hypothèses additionnelles faites par ces algorithmes sont légèrement plus restrictives que celles des approches basées sur la programmation linéaire. Néanmoins, même lorsque les hypothèses ne sont pas vérifiées ces algorithmes ont tendance à extraire des lignes qui maximisent le volume de l'enveloppe convexe. Ce comportement est souhaitable dans la grande majorité des applications, dont la nôtre.

Le premier algorithme présenté [Araújo et collab., 2001] existe dans de nombreuses variantes car, à cause de sa simplicité, il a été redécouvert dans plusieurs domaines très différents [Arora et collab., 2013; Chan et collab., 2011; Çivril et Magdon-Ismail, 2009; Ren et Chang, 2003; Winter, 1999]. L'algorithme SPA pour *Successive Projection Algorithm* extrait récursivement les lignes de W. Comme, il travaille dans l'enveloppe convexe, il peut être nécessaire de normaliser les lignes de X. À chaque pas, SPA extrait la ligne dont la norme ℓ_2 est maximale, puis projette toutes les lignes sur le complément orthogonal des lignes extraites. Cet algorithme très simple s'exécute en $2mnd + \mathcal{O}(md^2)$ opérations. De plus, si d est inconnu, il peut être estimé en utilisant la norme du résidu (les lignes après projection). L'algorithme SPA a été généralisé par Gillis et Vavasis [2014] à l'utilisation d'autres fonctions à la place de la norme ℓ_2, bien que la norme ℓ_2 est montrée théoriquement optimale. Les auteurs fournissent ainsi une analyse de la robustesse et montre que si $\epsilon_2 \leq \mathcal{O}\left(\frac{\sigma_d^3(W)}{\sqrt{d}}\right)$ alors l'erreur maximale commise sur les lignes extraites en norme ℓ_2 est inférieure à $\mathcal{O}\left(\frac{\epsilon_2}{\sigma_d^2(W)}\right)$, où $\sigma_d^2(W)$ est la plus faible valeur singulière de W. Malheureusement, comme l'indique la borne, l'algorithme est correct uniquement si X est de rang d, ou de façon équivalente, si W est de rang plein. On peut montrer [Gillis et Vavasis, 2014] que $\sigma_d(W) \leq \alpha$ et donc que la borne est moins bonne que celles des algorithmes basés sur la programmation linéaire. Néanmoins, sur les images hyper-spectrales, il a été noté que SPA obtient de meilleures performances empiriques que HOTTOPIXX.

Le schéma de l'algorithme précédent a été réutilisé pour proposer une amélioration par Gillis [2014a]. Le nouvel algorithme, nommé SNPA pour *Successive Nonnegative Projection Algorithm*, profite de la positivité des facteurs de la factorisation pour remplacer la projection sur le complément orthogonal par une projection sur l'enveloppe convexe de l'origine et des lignes déjà extraites. Cette formulation permet de s'affranchir de la contrainte de rang plein sur W. La contrainte est remplacée par une contrainte moins forte qui impose seulement que les lignes non-extraites soit non-nulles après projection sur l'enveloppe convexe. Gillis [2014a] conduit aussi une analyse de la robustesse. L'auteur identifie alors un nouveau paramètre quantifiant la robustesse au bruit de la matrice originale. Les bornes sur l'erreur et les techniques de preuve sont semblables à celle de SPA.

On présente un dernier algorithme qui a la particularité de travailler dans le cône et donc ne requière aucune normalisation. Proposé par Kumar et collab. [2013], l'algorithme XRAY effectue des projections sur le cône plutôt que sur l'enveloppe convexe. Sa robustesse au bruit n'a pas été établie.

7.2.5.3 Approches par projections aléatoires

Les approches aléatoires se caractérisent par l'utilisation d'une propriété géométrique simple. Après projection sur un espace de dimension inférieure, les arêtes de l'enveloppe conique des points dans ce sous-espace font partie des arêtes de l'enveloppe conique des points dans l'espace d'origine. Cette idée a inspiré deux lignes de travaux.

La première approche [Ding et collab., 2013a, 2015a, 2013b, 2014] effectue des projections aléatoires isotropiques indépendantes et identiquement distribuées sur des sous-espaces de dimension 1. L'identification d'une arête extrême est alors très simple, c'est le point maximal sur la demi-droite. L'algorithme compte le nombre de fois sur les p projections que chaque point est identifié comme extrême. Les d arêtes retournées sont celles qui ont été le plus fréquemment identifiées. La robustesse de l'algorithme a été analysée par les auteurs. Comme les projections font intervenir l'aléatoire, la borne est donnée avec forte probabilité. De plus, dans [Ding et collab., 2013a, 2015a] les auteurs établissent une hiérarchie entre les différentes hypothèses utilisées par les algorithmes (matrice de rang plein, à diagonale dominante ou encore simpliciale). Ils montrent que la condition simpliciale est nécessaire et suffisante. Dans le cadre des PRFA, ce résultat étend naturellement la condition nécessaire de distinguabilité pour les PDFA. Leur algorithme basé sur les projections aléatoires suppose uniquement que la matrice est simpliciale. De plus, le calcul peut efficacement être distribué, ce qui lui permet de traiter un grand volume de données.

La deuxième approche, développée par Zhou et collab. [2014], utilise les projections aléatoires pour construire un algorithme basé sur le principe « diviser pour régner ». Comparé à la première approche, où les projections se font sur des droites tirées aléatoirement, leur algorithme réduit la complexité par des projections successives sur des espaces de dimension de plus en plus petite. Le problème d'identification d'enveloppe conique est divisé en sous-problème sur des espaces de dimension deux fois moins élevée et tirés aléatoirement. Ces problèmes sont à nouveau séparés en plusieurs problèmes sur des espaces de dimension inférieures et ainsi de suite. Lorsque la dimension est faible n'importe quel algorithme peut être utilisé pour résoudre le sous-problème. Une possibilité est d'utiliser des algorithmes élémentaires en dimension 1, comme dans l'approche précédente, ou bien en dimension 2, en travaillant sur les angles. Les résultats sont ensuite agrégés successivement en remontant l'arbre. À chaque nœud, seules les arêtes identifiées le plus fréquemment sont gardées. Leur algorithme

permet de réduire la complexité et de distribuer le calcul de n'importe quel algorithme de NMF séparable. Aucune analyse de robustesse n'est donnée.

7.3 Analyse de la convergence

Dans cette section, nous analysons l'erreur en probabilité de l'Algorithme 4 qui utilise l'algorithme SPA (avec la norme 2) pour trouver l'enveloppe convexe. L'analyse se décompose en quatre parties et aboutit au Théorème 19 et aux Corollaires 1 et 2. Les résultats et l'interprétation des bornes sont donnés après la preuve.

D'abord, il faut borner en probabilité l'erreur sur les lignes de $\hat{D}$ comme dans [Hsu et collab., 2012]. Cette erreur est ensuite propagée par l'algorithme SPA dans $\hat{Z}$ et $\hat{Z}_\sigma$ en utilisant les résultats de [Gillis et Vavasis, 2014]. Après cette étape, les lignes de $\hat{Z}_\sigma$ correspondent aux lignes de Z à une permutation près. Puis, l'analyse de la robustesse de la solution d'un problème de moindres carrés non négatifs avec des contraintes linéaires [Lőtstedt, 1983] permet de borner l'erreur sur les paramètres de la représentation linéaire $(\hat{\boldsymbol{\alpha}}_0, \hat{A}, \hat{\boldsymbol{\alpha}}_\infty)$ à partir de l'erreur sur $\hat{Z}$ et $\hat{Z}_\sigma$. Enfin, les erreurs sur la représentation linéaire estimées sont multipliées astucieusement pour obtenir une borne en variation totale entre $\hat{p}$ et p grâce aux travaux de Balle [2013].

7.3.1 Erreur d'estimation

Nous supposons que N séquences indépendantes et identiquement distribuées sont utilisées pour estimer la matrice $\hat{D}$. Pour tout $u \in \mathcal{P}$, nous notons $\mathbf{d}_u$ (resp. $\hat{\mathbf{d}}_u$) le vecteur colonne correspondant à la ligne de D (resp. $\hat{D}$) indexée par u. La première étape consiste à borner avec forte probabilité l'erreur $\epsilon^{\text{est}} = \max_{u\in\mathcal{P}} \left\|\mathbf{d}_u - \hat{\mathbf{d}}_u\right\|_2$. Pour cela, nous utilisons d'abord le lemme suivant emprunté à [Hsu et collab., 2012, Proposition 19] et utilisant l'inégalité de McDiarmid [McDiarmid, 1989]. Pour tout $u \in \mathcal{P}$, soit z une variable aléatoire à valeurs dans $\Sigma^\star$, nous cherchons à estimer le vecteur $\mathbf{d}_u^\infty$ tel que $\forall v \in \Sigma^\star$, $\mathbf{d}_u^\infty[v] = \mathbb{P}\left(z = v|u\right) = \frac{p(uv)}{p(u\Sigma^\star)}$ à partir de n_u copies identiquement distribuées et indépendantes z_i de z. Nous notons $\hat{\mathbf{d}}_u^\infty$ le vecteur ainsi estimé.

Lemme 2.
En reprenant les notations ci-dessus, pour tout $\delta_u > 0$ on a que

$$\mathbb{P}\left(\left\|\mathbf{d}_u^\infty - \hat{\mathbf{d}}_u^\infty\right\|_2 \geq \frac{1}{\sqrt{n_u}}\left(1 + \sqrt{\log\left(\frac{1}{\delta_u}\right)}\right)\right) \leq \delta_u.$$

Pour la suite nous notons, $n = \min_{u\in\mathcal{P}} n_u$, en particulier $n_\varepsilon = N$.

Proposition 39 (Erreur d'estimation des distributions conditionnelles)**.**
En reprenant les notations ci-dessus, pour tout $\delta \in [0, 1]$ on a que

$$\mathbb{P}\left(\epsilon^{est} \leq \frac{1}{\sqrt{n}}\left(1 + \sqrt{\log\left(\frac{|\mathcal{P}|}{\delta}\right)}\right)\right) \geq 1 - \delta.$$

Étant donné que les $\mathbf{d}_u$ représentent des distributions conditionnelle, la borne sur ϵ^{est} proposée dans la Proposition 39 dépend nécessairement de n. La Proposition 39 fait aussi apparaître le logarithme de la taille de la base $\mathcal{P}$ de préfixes. Il semble néanmoins possible d'obtenir un résultat indépendant de $|\mathcal{P}|$ à partir des résultats de [Denis et collab., 2014].

Démonstration. D'après le Lemme 2, nous avons que,

$$\begin{aligned}
\mathbb{P}\left(\max_{u\in\mathcal{P}}\left\|\mathbf{d}_u^\infty-\hat{\mathbf{d}}_u^\infty\right\|_2\leq\frac{1}{\sqrt{n}}\left(1+\sqrt{\log\left(\frac{|\mathcal{P}|}{\delta}\right)}\right)\right)&\\
&=1-\mathbb{P}\left(\exists u\in\mathcal{P},\ \left\|\mathbf{d}_u^\infty-\hat{\mathbf{d}}_u^\infty\right\|_2\geq\frac{1}{\sqrt{n}}\left(1+\sqrt{\log\left(\frac{|\mathcal{P}|}{\delta}\right)}\right)\right)\\
&\geq 1-\sum_{u\in\mathcal{P}}\mathbb{P}\left(\left\|\mathbf{d}_u^\infty-\hat{\mathbf{d}}_u^\infty\right\|_2\geq\frac{1}{\sqrt{n}}\left(1+\sqrt{\log\left(\frac{|\mathcal{P}|}{\delta}\right)}\right)\right)\\
&\geq 1-\sum_{u\in\mathcal{P}}\mathbb{P}\left(\left\|\mathbf{d}_u^\infty-\hat{\mathbf{d}}_u^\infty\right\|_2\geq\frac{1}{\sqrt{n_u}}\left(1+\sqrt{\log\left(\frac{|\mathcal{P}|}{\delta}\right)}\right)\right)\\
&\geq 1-\sum_{u\in\mathcal{P}}\frac{\delta}{|\mathcal{P}|}=1-\delta.
\end{aligned}$$

Ensuite, d'après la définition de la norme 2 que $\left\|\mathbf{d}_u-\hat{\mathbf{d}}_u\right\|_2\leq 2\left\|\mathbf{d}_u^\infty-\hat{\mathbf{d}}_u^\infty\right\|_2$ car certains coefficients de $\mathbf{d}_u-\hat{\mathbf{d}}_u$ peuvent apparaître deux fois dans $\mathbf{d}_u^\infty-\hat{\mathbf{d}}_u^\infty$. En passant au maximum sur $u\in\mathcal{P}$, nous obtenons le résultat. □

Nous remarquons pour la suite que d'après la définition de la norme 2 que $\left\|\hat{\mathbf{d}}_u-\mathbf{d}_u\right\|_2\leq 2\left\|\hat{\mathbf{d}}_u^\infty-\mathbf{d}_u^\infty\right\|_2$ car certains coefficients de $\hat{\mathbf{d}}_u-\mathbf{d}_u$ peuvent apparaître deux fois dans $\hat{\mathbf{d}}_u^\infty-\mathbf{d}_u^\infty$. En passant au maximum sur $u\in\mathcal{P}$, nous obtenons que

$$\max_{u\in\mathcal{P}}\left\|\hat{\mathbf{d}}_u-\mathbf{d}_u\right\|_2\leq 2\epsilon^{\text{est}}.$$

7.3.2 Propagation de l'erreur dans l'enveloppe convexe

Dans [Gillis et Vavasis, 2014], les auteurs proposent une analyse de la robustesse de SPA. Cet algorithme fait l'hypothèse que $\text{rang}((\mathbf{d}_u)_{u\in\mathcal{R}})=d$. Comme la base est supposée complète, nous avons aussi que $\text{rang}(Z)=d$. L'algorithme SNPA permet de s'affranchir de cette contrainte. Une analyse de sa robustesse est proposé dans [Gillis, 2014a]. L'algorithme SNPA est aussi plus gourmand. Comme nous ne l'avons pas utilisé dans nos expériences nous limitons notre analyse à SPA en supposant que $\text{rang}((\mathbf{d}_u)_{u\in\mathcal{R}})=d$.

Après l'exécution de SPA si l'erreur entre les lignes de $\hat{D}$ et D prises deux à deux est suffisamment faible, les lignes indexées par $\hat{\mathcal{R}}$ de $\hat{D}$ sont proches des lignes de D indexée par $\mathcal{R}$ à une permutation près. Nous pouvons ignorer cette permutation dans l'analyse car *in fine* elle n'induit qu'une permutation sur les états de la représentation linéaire et donc pas de différence sur la série réalisée. Nous notons alors $\mathcal{R}(i)$ (resp. $\hat{\mathcal{R}}(i)$) le i^{e} mot de $\mathcal{R}$ (resp. $\hat{\mathcal{R}}$). Nous notons $\epsilon^{\text{conv}}=\max_{i\in[1,d]}\left\|\hat{\mathbf{d}}_{\hat{\mathcal{R}}(i)}-\mathbf{d}_{\mathcal{R}(i)}\right\|_2$ l'erreur analysée.

Dans la suite, nous notons σ_k la k^{e} plus grande valeur singulière de $(\mathbf{d}_u)_{u\in\mathcal{R}}$. Nous remarquons que σ_k est aussi la k^{e} plus grande valeur singulière de Z. Ainsi σ_d est la plus petite valeur singulière non nulle de $(\mathbf{d}_u)_{u\in\mathcal{R}}$ et de Z. Nous notons aussi, $K=\max_{u\in\mathcal{R}}\|\mathbf{d}_u\|_2$.

Proposition 40 (Erreur dans l'enveloppe convexe estimée)**.**
En reprenant les notations précédentes, si $\epsilon^{est} \leq \frac{\sigma_d^3}{648\sqrt{d}}$ *alors,*

$$\epsilon^{conv} = \max_{i\in[1,d]} \left\|\hat{\mathbf{d}}_{\hat{\mathcal{R}}(i)} - \mathbf{d}_{\mathcal{R}(i)}\right\|_2 \leq 162\frac{\epsilon^{est}}{\sigma_d^2}.$$

Démonstration. Il suffit d'appliquer le Théorème 3 de [Gillis et Vavasis, 2014], puis de simplifier. Ce Théorème permet de borner l'erreur entre les lignes indexées par $\hat{\mathcal{R}}$ de $\hat{D}$ et les lignes de D indexées par $\mathcal{R}$. Ainsi, si

$$\max_{u\in\mathcal{P}} \left\|\hat{\mathbf{d}}_u - \mathbf{d}_u\right\|_2 < \sigma_d \min\left(\frac{1}{2\sqrt{d-1}}, \frac{1}{4}\right)\left(1 + 80\frac{K^2}{\sigma_d^2}\right)^{-1},$$

alors

$$\max_{i\in[1,d]} \left\|\hat{\mathbf{d}}_{\hat{\mathcal{R}}(i)} - \mathbf{d}_{\mathcal{R}(i)}\right\|_2 < \max_{u\in\mathcal{P}} \left\|\hat{\mathbf{d}}_u - \mathbf{d}_u\right\|_2 \left(1 + 80\frac{K^2}{\sigma_d^2}\right).$$

Nous remarquons, d'abord que

$$\frac{1}{4\sqrt{d}} \leq \min\left(\frac{1}{2\sqrt{d-1}}, \frac{1}{4}\right).$$

Ensuite, comme $\|\mathbf{d}_u\|_2 \leq \|\mathbf{d}_u\|_1 \leq 1$ par hypothèse, nous avons que $K \leq 1$. D'après le Lemme 4 de [Gillis et Vavasis, 2014], nous avons aussi $\sigma_d \leq K$ et donc,

$$\left(1 + 80\frac{K^2}{\sigma_d^2}\right) \leq \frac{1}{\sigma_d^2}\left(\sigma_d^2 + 80K^2\right) \leq \frac{81}{\sigma_d^2}.$$

Ainsi, nous avons que

$$\sigma_d \min\left(\frac{1}{2\sqrt{d-1}}, \frac{1}{4}\right)\left(1 + 80\frac{K^2}{\sigma_d^2}\right)^{-1} \geq \sigma_d \frac{1}{4\sqrt{d}}\left(1 + 80\frac{K^2}{\sigma_d^2}\right)^{-1} \geq \frac{\sigma_d^3}{324\sqrt{d}},$$

et,

$$\max_{u\in\mathcal{P}} \left\|\hat{\mathbf{d}}_u - \mathbf{d}_u\right\|_2 \left(1 + 80\frac{K^2}{\sigma_d^2}\right) \leq 81\frac{\max_{u\in\mathcal{P}} \left\|\hat{\mathbf{d}}_u - \mathbf{d}_u\right\|_2}{\sigma_d^2}.$$

Nous concluons grâce à l'inégalité suivante, démontrée dans la section précédente,

$$\max_{u\in\mathcal{P}} \left\|\hat{\mathbf{d}}_u - \mathbf{d}_u\right\|_2 \leq 2\epsilon^{\text{est}}.$$

□

7.3.3 Propagation de l'erreur dans les moindres carrés sous contraintes

Dans l'algorithme CH-PRFA, comme pour CH-PNFA au chapitre précédent, la résolution des problèmes de NNLS sous contraintes linéaires se fait par programmation quadratique. Dans cette section, nous analysons la robustesse de ces minimisations réécrites sous la forme suivante,

$$\mathbf{x}^* = \underset{\mathbf{x}}{\operatorname{argmin}} \|Q\mathbf{x} + \mathbf{q}\|_2 \tag{7.1}$$

$$\text{tel que } \begin{cases} B\mathbf{x} + \mathbf{b} \geq 0, \\ C\mathbf{x} + \mathbf{c} = 0 \end{cases}. \tag{7.2}$$

Lors de l'apprentissage, les paramètres de la minimisation (7.1) sont perturbés. En désignant ces paramètres par un chapeau, nous nous intéressons à borner $\|\hat{\mathbf{x}}^* - \mathbf{x}^*\|_2$ en fonction de $\left\|\hat{Q} - Q\right\|_2$, $\|\hat{\mathbf{q}} - \mathbf{q}\|_2$, $\left\|\hat{B} - B\right\|_2$, $\left\|\hat{\mathbf{b}} - \mathbf{b}\right\|_2$, $\left\|\hat{C} - C\right\|_2$, $\|\hat{\mathbf{c}} - \mathbf{c}\|_2$.

En posant $\hat{\mathbf{x}} = \hat{\boldsymbol{\alpha}}_0$, l'étape 8 correspond à,

$$\begin{aligned} &\hat{Q} = \hat{Z}^\top, && \hat{B} = I, && \hat{C} = \mathbf{1}^\top, \\ &\hat{\mathbf{q}} = -\hat{H}_{\mathcal{B}}^\top \mathbf{1}_\varepsilon, && \hat{\mathbf{b}} = 0, && \hat{\mathbf{c}} = -1. \end{aligned}$$

L'étape 9 se reformule de la façon suivante. Nous observons que plutôt que de résoudre un gros problème de NNLS sous contraintes à $d^2\,|\Sigma|$ variables, comme au Chapitre 6, nous pouvons résoudre d problèmes à $d\,|\Sigma|$ variables. Cette formulation est plus efficace et simplifie l'analyse. Ainsi pour le i^{e} problème, nous notons $\hat{\mathbf{x}}_i$ le vecteur de taille $d\,|\Sigma|$ tel que $\hat{\mathbf{x}}_i[j + d(\sigma - 1)] = \hat{A}_\sigma[i, j]$. Autrement dit, $\hat{\mathbf{x}}_i^\top$ est la concaténation horizontale des i^{es} lignes de $\hat{A}_1, \hat{A}_2, \ldots, \hat{A}_{|\Sigma|}$ et donc,

$$\hat{\mathbf{x}}_i^\top = \underbrace{\begin{pmatrix} \hat{A}_1[i,:] & \ldots & \hat{A}_{|\Sigma|}[i,:] \end{pmatrix}}_{d|\Sigma|} = \hat{A}[i,:].$$

Les autres paramètres sont alors donnés par,

$$\hat{Q} = \underbrace{\begin{pmatrix} \hat{Z}^\top & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \hat{Z}^\top \end{pmatrix}}_{d|\Sigma|},$$

$$\hat{\mathbf{q}}_i^\top = -\underbrace{\begin{pmatrix} \hat{Z}_1[i,:] & \ldots & \hat{Z}_{|\Sigma|}[i,:] \end{pmatrix}}_{d|\Sigma|},$$

$$\begin{aligned} &\hat{B} = I, \\ &\hat{\mathbf{b}} = 0, \\ &\hat{C} = \mathbf{1}^\top, \\ &\hat{\mathbf{c}}_i = \frac{\hat{p}(\hat{\mathcal{R}}(i))}{\hat{p}(\hat{\mathcal{R}}(i)\Sigma^\star)} - 1. \end{aligned}$$

7.3.3.1 Existence et unicité

Dans cette partie, le noyau d'une matrice M est notée $\mathrm{Ker}(M)$ et l'image de M est notée $Im(M)$. Vérifions d'abord des conditions suffisantes à l'existence et l'unicité d'une solution $\mathbf{x}^*$ au problème (7.1) et $\hat{\mathbf{x}}^*$ au problème perturbé. Nous adaptons le même formalise que Lőtstedt [1983] et notons E (originellement D) la matrice et $\mathbf{e}$ (originellement $\mathbf{d}$) le vecteur tels que

$$E = \begin{pmatrix} B \\ C \end{pmatrix}, \qquad \mathbf{e} = \begin{pmatrix} \mathbf{b} \\ \mathbf{c} \end{pmatrix}.$$

D'abord nous vérifions qu'aux étapes 8 et 9, l'ensemble de contraintes est $\{\mathbf{x} | B\mathbf{x} + \mathbf{b} \geq 0,\, C\mathbf{x} + \mathbf{c} = 0\}$ non vide d'après la définition d'un PRFA. Il en est de même pour les

contraintes perturbées $\{\hat{\mathbf{x}}|\hat{B}\hat{\mathbf{x}}+\hat{\mathbf{b}}\geq 0,\ \hat{C}\hat{\mathbf{x}}+\hat{\mathbf{c}}=0\}$. Ainsi, d'après le Théorème 1 de [Lőtstedt, 1983], $\mathbf{x}^*$ et $\hat{\mathbf{x}}^*$ existent.

Ensuite, nous montrons que

$$\mathrm{Ker}(Q)=\mathrm{Ker}(\hat{Q})=\{0\}, \tag{7.3}$$

$$\mathrm{Ker}(E)=\mathrm{Ker}(\hat{E})=\{0\}. \tag{7.4}$$

À l'étape 8, comme à l'étape 9, nous avons $B=\hat{B}=I$, donc $\mathrm{Ker}(E)=\mathrm{Ker}(\hat{E})=\{0\}$. De plus, comme $\mathrm{rang}(Z)=d$ l'hypothèse $\mathrm{Ker}(Q)=\{0\}$ est vérifiée. Avec probabilité 1, nous avons aussi $\mathrm{Ker}(\hat{Q})=\{0\}$ par densité.

Toujours d'après le Théorème 1 de [Lőtstedt, 1983] cela implique que $\hat{\mathbf{x}}^*$ et $\mathbf{x}^*$ sont uniques. Si l'hypothèse de rang plein ($\mathrm{rang}(Z)=d$) n'était pas vérifiée, en définissant $\hat{\mathbf{x}}^*$ et $\mathbf{x}^*$ comme les solutions de norme minimale, nous retrouverions l'unicité. L'analyse qui suit en serait alors plus complexe mais aboutirait à des bornes similaires. Nous avons choisi de traiter uniquement le cas du rang plein pour lequel les bornes sont plus simples.

7.3.3.2 Borne sur la perturbation

Remarquons que pour les problèmes correspondant aux étapes 8 et 9 de l'Algorithme 4, nous avons

$$B=\hat{B},\qquad \mathbf{b}=\hat{\mathbf{b}},\qquad C=\hat{C}.$$

Ce qui implique que $E=\hat{E}$. Ces propriétés vont permettre de simplifier l'analyse de la perturbation. Pour la suite, nous adaptons le même formalise que Lőtstedt [1983]. Nous définissons $G=EQ^\dagger$. Ainsi, nous avons $\hat{G}=E\hat{Q}^\dagger$. Nous définissons de plus, $\mathbf{v}^*=2(G^\dagger)^\top(Q\mathbf{x}^*+P_{\mathrm{Im}(Q)}\mathbf{q})$ où $P_{\mathrm{Im}(Q)}$ est le projecteur sur $\mathrm{Im}(Q)$. Comme $(\boldsymbol{\alpha}_0,A,\boldsymbol{\alpha}_\infty)$ définit bien un PRFA, nous déduisons que, pour le problème non perturbé, $Q\mathbf{x}^*+P_{\mathrm{Im}(Q)}\mathbf{q}=0$. Ainsi, nous avons que $\mathbf{v}^*=0$. Comme la base est complète, nous avons que $P_{\mathrm{Im}(Q)}\mathbf{q}=\mathbf{q}$. Ces observations nous permettent d'énoncer une version simplifiée du Théorème 3 de Lőtstedt [1983].

Théorème 18 (Borne simplifiée sur la perturbation d'un problème de moindres carrés sous contraintes d'inégalités linéaires)**.**
Sous les hypothèses définies Équation (7.3)*, soit* $\mathbf{x}^*$ *la solution du problème* (7.1) *et* $\hat{\mathbf{x}}^*$ *la solution du problème perturbé, on suppose que* $Q\mathbf{x}^*+P_{\mathrm{Im}(Q)}\mathbf{q}=0$*,* $P_{\mathrm{Im}(Q)}\mathbf{q}=\mathbf{q}$*,* $B=\hat{B}$*,* $\mathbf{b}=\hat{\mathbf{b}}$ *et* $C=\hat{C}$*, alors*

$$\begin{aligned}\|\hat{\mathbf{x}}^*-\mathbf{x}^*\|_2\leq 2\|\mathbf{q}\|_2\left(1+\left\|\hat{Q}\right\|_2\left\|\hat{Q}^\dagger\right\|_2\|E\|_2\left\|E^\dagger\right\|_2\right)\left\|\hat{Q}^\dagger-Q^\dagger\right\|_2\\ +\left\|\hat{Q}\right\|_2\left\|\hat{Q}^\dagger\right\|_2\left\|E^\dagger\right\|_2\|\hat{\mathbf{e}}-\mathbf{e}\|_2\\ +2\left\|\hat{Q}^\dagger\right\|_2\|\hat{\mathbf{q}}-\mathbf{q}\|_2.\end{aligned}$$

Démonstration. En partant de l'énoncé complet du Théorème 3 de Lőtstedt [1983], nous avons, avec les notations définies précédemment, que

$$\begin{aligned}\|\hat{\mathbf{x}}^*-\mathbf{x}^*\|_2\leq\left\|Q\mathbf{x}^*-P_{\mathrm{Ker}(Q^\top)}\mathbf{q}\right\|_2\left(\left\|\hat{Q}^\dagger-Q^\dagger\right\|_2+\left\|\hat{Q}^\dagger\right\|_2\left\|\hat{G}^\dagger\right\|_2\left\|\hat{G}-G\right\|_2\right)\\ +\left\|\hat{Q}^\dagger\right\|_2\left(2\left\|\frac{1}{2}\left(\hat{G}-G\right)^\top\mathbf{v}^*-(\hat{\mathbf{q}}-\mathbf{q})\right\|_2+\left\|\hat{G}^\dagger\right\|_2\|\hat{\mathbf{e}}-\mathbf{e}\|_2\right).\end{aligned}$$

Comme $P_{\mathrm{Ker}(Q^\top)} = P_{\mathrm{Im}(Q)}$ et que $Q\mathbf{x}^* + P_{\mathrm{Im}(Q)}\mathbf{q} = 0$ alors

$$\left\|Q\mathbf{x}^* - P_{\mathrm{Ker}(Q^\top)}\mathbf{q}\right\|_2 = 2\|\mathbf{q}\|_2.$$

De plus, $\mathbf{v}^* = 0$ et donc

$$\left\|\frac{1}{2}\left(\hat{G} - G\right)^\top \mathbf{v}^* - (\hat{\mathbf{q}} - \mathbf{q})\right\|_2 = \|\hat{\mathbf{q}} - \mathbf{q}\|_2.$$

Nous avons de plus que $G = EQ^\dagger$ et $E = \hat{E}$ donc,

$$\left\|\hat{G} - G\right\|_2 = \left\|E\left(\hat{Q}^\dagger - Q^\dagger\right)\right\|_2 \leq \|E\|_2\left\|\hat{Q}^\dagger - Q^\dagger\right\|_2,$$

de plus, comme $\mathrm{Ker}(E) = \{0\}$ et $\mathrm{Ker}(Q) = \{0\}$, nous avons $\hat{G}^\dagger = \left(E\hat{Q}^\dagger\right)^\dagger = \hat{Q}E^\dagger$ et donc,

$$\left\|\hat{G}^\dagger\right\|_2 \leq \left\|E^\dagger\right\|_2\left\|\hat{Q}\right\|_2.$$

Cela permet d'obtenir que

$$\begin{aligned}\|\hat{\mathbf{x}}^* - \mathbf{x}^*\|_2 \leq 2\|\mathbf{q}\|_2\left(1 + \left\|\hat{Q}\right\|_2\left\|\hat{Q}^\dagger\right\|_2\|E\|_2\left\|E^\dagger\right\|_2\right)\left\|\hat{Q}^\dagger - Q^\dagger\right\|_2 \\ + \left\|\hat{Q}\right\|_2\left\|\hat{Q}^\dagger\right\|_2\left\|E^\dagger\right\|_2\|\hat{\mathbf{e}} - \mathbf{e}\|_2 \\ + 2\left\|\hat{Q}^\dagger\right\|_2\|\hat{\mathbf{q}} - \mathbf{q}\|_2.\end{aligned}$$

□

Nous terminons par un lemme issu des inégalités de Wedin [1972] sur la perturbation des valeurs singulières.

Lemme 3 (Perturbation dans la pseudo-inverse)**.**
Si $\left\|\hat{Q} - Q\right\|_2\left\|Q^\dagger\right\|_2 \leq \kappa < 1$ *alors*

$$\left\|\hat{Q}^\dagger\right\|_2 \leq \frac{1}{1-\kappa}\left\|Q^\dagger\right\|_2,$$

et

$$\begin{aligned}\left\|\hat{Q}^\dagger - Q^\dagger\right\|_2 &\leq \sqrt{2}\left\|\hat{Q}^\dagger\right\|_2\left\|Q^\dagger\right\|_2\left\|\hat{Q} - Q\right\|_2 \\ &\leq \frac{\sqrt{2}}{1-\kappa}\left\|Q^\dagger\right\|_2^2\left\|\hat{Q} - Q\right\|_2.\end{aligned}$$

Ainsi, en appliquant le Lemme 3, nous obtenons que

$$\begin{aligned}\|\hat{\mathbf{x}}^* - \mathbf{x}^*\|_2 \leq \frac{2\sqrt{2}}{1-\kappa}\|\mathbf{q}\|_2\left(1 + \frac{1}{1-\kappa}\left\|\hat{Q}\right\|_2\left\|Q^\dagger\right\|_2\|E\|_2\left\|E^\dagger\right\|_2\right)\left\|Q^\dagger\right\|_2^2\left\|\hat{Q} - Q\right\|_2 \\ + \frac{1}{1-\kappa}\left\|\hat{Q}\right\|_2\left\|Q^\dagger\right\|_2\left\|E^\dagger\right\|_2\|\hat{\mathbf{e}} - \mathbf{e}\|_2 \\ + \frac{2}{1-\kappa}\left\|Q^\dagger\right\|_2\|\hat{\mathbf{q}} - \mathbf{q}\|_2.\end{aligned} \tag{7.5}$$

Dans la suite, nous nous intéressons aux instanciations du problème (7.1) aux étapes 8 et 9 de l'Algorithme 4.

7.3.4 Propagation de l'erreur dans la représentation linéaire

7.3.4.1 Borne sur la perturbation des poids initiaux

Dans cette section, nous étudions l'étape 8 de l'Algorithme 4, pour laquelle nous établissons la Proposition 41.

Proposition 41 (Borne sur les poids initiaux)**.**
Soit un réel positif $\kappa < 1$, si $\epsilon^{conv} \leq \frac{1}{\sqrt{d}}\kappa\sigma_d$, alors

$$\|\hat{\boldsymbol{\alpha}}_0 - \boldsymbol{\alpha}_0\|_1 \leq 2(\sqrt{2}+2)\frac{d^2}{\sigma_d^3\,(1-\kappa)^2}\epsilon^{conv} + 2\frac{\sqrt{d}}{\sigma_d\,(1-\kappa)}\epsilon^{est}.$$

Démonstration. D'abord, nous remarquons que $\hat{\mathbf{e}} = \mathbf{e} = \begin{pmatrix} \mathbf{b}^\top & \mathbf{c}^\top \end{pmatrix}^\top = \begin{pmatrix} 0 & \dots & 0 & -1 \end{pmatrix}^\top$. En remplaçant dans l'Équation (7.5), nous obtenons que

$$\|\hat{\mathbf{x}}^* - \mathbf{x}^*\|_2 \leq \frac{2\sqrt{2}}{1-\kappa}\|\mathbf{q}\|_2\left(1 + \frac{1}{1-\kappa}\left\|\hat{Q}\right\|_2\left\|Q^\dagger\right\|_2\|E\|_2\left\|E^\dagger\right\|_2\right)\left\|Q^\dagger\right\|_2^2\left\|\hat{Q} - Q\right\|_2 + \frac{2}{1-\kappa}\left\|Q^\dagger\right\|_2\|\hat{\mathbf{q}} - \mathbf{q}\|_2.$$

Ensuite, nous avons

$$E^\top E = B^\top B + C^\top C = I_d + U_d,$$

où I_d est la matrice identité de dimension d et U_d est la matrice unité (tous les coefficients sont égaux à 1) de dimension d. Or le spectre de U_d est composé de d (une fois) et 0 ($d-1$ fois). Nous notons $\lambda_{\max}(M)$ (resp. $\lambda_{\min}(M)$) la valeur propre maximale (resp. minimale) de M. Donc, nous obtenons que

$$\|E\|_2^2 = \lambda_{\max}(E^\top E) = \lambda_{\max}(I_d + U_d) = 1 + d,$$

et

$$\left\|E^\dagger\right\|_2^2 = \lambda_{\min}(E^\top E) = \lambda_{\min}(I_d + U_d) = 1.$$

Ces propriétés permettent de simplifier davantage l'Équation (7.5), pour obtenir

$$\|\hat{\mathbf{x}}^* - \mathbf{x}^*\|_2 \leq \frac{2\sqrt{2}}{1-\kappa}\|\mathbf{q}\|_2\left(1 + \frac{\sqrt{1+d}}{1-\kappa}\left\|\hat{Q}\right\|_2\left\|Q^\dagger\right\|_2\right)\left\|Q^\dagger\right\|_2^2\left\|\hat{Q} - Q\right\|_2 + \frac{2}{1-\kappa}\left\|Q^\dagger\right\|_2\|\hat{\mathbf{q}} - \mathbf{q}\|_2.$$

De plus, comme $Q = Z^\top$, nous avons que $\left\|Q^\dagger\right\|_2 = \frac{1}{\sigma_d}$, où σ_d est la plus petite valeur propre de Z.

Analysons maintenant $\|\mathbf{q}\|_2$ et $\left\|\hat{Q}\right\|_2$, où $\mathbf{q} = -H_{\mathcal{B}}^\top\mathbf{1}_\varepsilon$ et $\hat{Q} = \hat{Z}^\top$. Premièrement, nous avons que

$$\|\mathbf{q}\|_2 \leq \|\mathbf{q}\|_1 \leq \left\|H_{\mathcal{B}}^\top\mathbf{1}_\varepsilon\right\|_1 = \sum_{u\in\mathcal{S}} p(u) \leq 1.$$

Deuxièmement, l'inégalité de Hölder implique que

$$\left\|\hat{Q}\right\|_2 \leq \sqrt{\left\|\hat{Q}\right\|_1\left\|\hat{Q}\right\|_\infty}.$$

Or, nous avons d'une part que $\left\|\hat{Q}\right\|_1 = \left\|\hat{Z}^\top\right\|_1 = \max_{u\in\hat{\mathcal{R}}} \sum_{v\in\mathcal{S}} \frac{\hat{p}(uv)}{\hat{p}(u\Sigma)} \leq 1$. D'autre part, nous montrons que $\left\|\hat{Q}\right\|_\infty = \left\|\hat{Z}^\top\right\|_\infty = \max_{v\in\mathcal{S}} \sum_{u\in\hat{\mathcal{R}}} \frac{\hat{p}(uv)}{\hat{p}(u\Sigma)} \leq \left|\hat{\mathcal{R}}\right| = d$. Ainsi, nous concluons que

$$\left\|\hat{Q}\right\|_2 \leq \sqrt{d}.$$

Mises bout à bout ces propriétés permettent d'établir que

$$\|\hat{\boldsymbol{\alpha}}_0 - \boldsymbol{\alpha}_0\|_2 \leq \frac{2\sqrt{2}}{\sigma_d^2\,(1-\kappa)} \left(1 + \frac{\sqrt{d\,(1+d)}}{\sigma_d\,(1-\kappa)}\right) \left\|\hat{Z} - Z\right\|_2 + \frac{2}{\sigma_d\,(1-\kappa)} \|\hat{\mathbf{q}} - \mathbf{q}\|_2.$$

Nous avons vu lors de la propagation de l'erreur dans l'enveloppe convexe que $\sigma_d \leq 1$ et donc, comme $\kappa < 1$,

$$\begin{aligned} 1 + \frac{\sqrt{d\,(1+d)}}{\sigma_d\,(1-\kappa)} &\leq \frac{1 + \sqrt{d\,(1+d)}}{\sigma_d\,(1-\kappa)} \\ &\leq \frac{(1+\sqrt{2})d}{\sigma_d\,(1-\kappa)}, \end{aligned}$$

car pour $d \geq 1$, nous avons $1 + \sqrt{d\,(1+d)} \leq (1+\sqrt{2})d$. En combinant les inégalités, nous obtenons que

$$\|\hat{\boldsymbol{\alpha}}_0 - \boldsymbol{\alpha}_0\|_2 \leq \frac{2(\sqrt{2}+2)d}{\sigma_d^3\,(1-\kappa)^2} \left\|\hat{Z} - Z\right\|_2 + \frac{2}{\sigma_d\,(1-\kappa)} \|\hat{\mathbf{q}} - \mathbf{q}\|_2.$$

Il nous reste à insérer les bornes sur $\left\|\hat{Z} - Z\right\|_2$ et $\|\hat{\mathbf{q}} - \mathbf{q}\|_2$. Premièrement, nous avons que

$$\begin{aligned} \|\hat{\mathbf{q}} - \mathbf{q}\|_2 = \left\|\left(\hat{H}_{\mathcal{B}} - H_{\mathcal{B}}\right)^\top \mathbf{1}_\varepsilon\right\|_2 &= \sqrt{\sum_{v\in\mathcal{S}} \left(\hat{p}(v) - p(v)\right)^2} \\ &\leq \sqrt{\sum_{v\in\Sigma^\star} \left(\hat{p}(v) - p(v)\right)^2} = \left\|\hat{\mathbf{d}}_\varepsilon^\infty - \mathbf{d}_\varepsilon^\infty\right\|_2. \end{aligned}$$

Deuxièmement, nous utilisons les propriétés des normes 2 et de Frobenius pour montrer que

$$\begin{aligned} \left\|\hat{Z} - Z\right\|_2 \leq \left\|\hat{Z} - Z\right\|_F &= \sqrt{\sum_{i=1}^{d} \left(\left\|\hat{Z}[i,:] - Z[i,:]\right\|_2\right)^2} \\ &\leq \sqrt{\sum_{i=1}^{d} \left(\left\|\hat{\mathbf{d}}_{\hat{\mathcal{R}}(i)} - \mathbf{d}_{\mathcal{R}(i)}\right\|_2\right)^2} \\ &\leq \sqrt{d} \max_{i\in[1,d]} \left\|\hat{\mathbf{d}}_{\hat{\mathcal{R}}(i)} - \mathbf{d}_{\mathcal{R}(i)}\right\|_2. \end{aligned}$$

Ainsi, comme

$$\left\|\hat{Q} - Q\right\|_2 = \left\|\hat{Z} - Z\right\|_2 \leq \sqrt{d} \max_{i\in[1,d]} \left\|\hat{\mathbf{d}}_{\hat{\mathcal{R}}(i)} - \mathbf{d}_{\mathcal{R}(i)}\right\|_2 \kappa\sigma_d,$$

la condition $\left\|\hat{Q} - Q\right\|_2 \leq \kappa\sigma_d$ est satisfaite si, $\max_{i\in[1,d]} \left\|\hat{\mathbf{d}}_{\hat{\mathcal{R}}(i)} - \mathbf{d}_{\mathcal{R}(i)}\right\|_2 \leq \frac{1}{\sqrt{d}}\kappa\sigma_d$. Enfin, d'après les propriétés des normes, $\|\hat{\boldsymbol{\alpha}}_0 - \boldsymbol{\alpha}_0\|_1 \leq \sqrt{d}\|\hat{\boldsymbol{\alpha}}_0 - \boldsymbol{\alpha}_0\|_2$ et nous obtenons que

$$\|\hat{\boldsymbol{\alpha}}_0 - \boldsymbol{\alpha}_0\|_1 \leq 2(\sqrt{2}+2)\frac{d^2}{\sigma_d^3\,(1-\kappa)^2} \max_{i\in[1,d]} \left\|\hat{\mathbf{d}}_{\hat{\mathcal{R}}(i)} - \mathbf{d}_{\mathcal{R}(i)}\right\|_2 + 2\frac{\sqrt{d}}{\sigma_d\,(1-\kappa)} \left\|\hat{\mathbf{d}}_\varepsilon^\infty - \mathbf{d}_\varepsilon^\infty\right\|_2.$$

Il reste à remplacer $\max_{i\in[1,d]} \left\|\hat{\mathbf{d}}_{\hat{\mathcal{R}}(i)} - \mathbf{d}_{\mathcal{R}(i)}\right\|_2 = \epsilon^{\text{conv}}$ et $\left\|\hat{\mathbf{d}}^{\infty}_{\varepsilon} - \mathbf{d}^{\infty}_{\varepsilon}\right\|_2 \leq \max_{u\in\mathcal{P}} \left\|\hat{\mathbf{d}}^{\infty}_{u} - \mathbf{d}^{\infty}_{u}\right\|_2 = \epsilon^{\text{est}}$ pour conclure. □

7.3.4.2 Borne sur la perturbation des poids de transitions

Nous nous intéressons maintenant à l'étape 9 de l'Algorithme 4. La prochaine proposition suit les mêmes étapes de celle de la Proposition 41.

Proposition 42 (Borne sur les poids de transitions)**.**
Soit un réel positif $\kappa < 1$, si $\epsilon^{conv} \leq \frac{1}{\sqrt{d}}\kappa\sigma_d$, alors

$$\left\|\hat{A} - A\right\|_\infty \leq 2(\sqrt{2}+2)\frac{d^2\sqrt{|\Sigma|}}{\sigma_d^3(1-\kappa)^2}\epsilon^{conv} + \frac{d}{\sigma_d(1-\kappa)}\|\hat{\boldsymbol{\alpha}}_\infty - \boldsymbol{\alpha}_\infty\|_\infty + 2\frac{\sqrt{d}}{\sigma_d(1-\kappa)}\epsilon^{conv}.$$

Démonstration. D'abord, nous avons que

$$\begin{aligned} E^\top E &= B^\top B + C^\top C \\ &= I_{d|\Sigma|} + U_{d|\Sigma|}, \end{aligned}$$

Cela permet d'établir que,

$$\|E\|_2^2 = \lambda_{\max}\left(E^\top E\right) = \lambda_{\max}\left(I_{d|\Sigma|} + U_{d|\Sigma|}\right) = 1 + d\,|\Sigma|\,.$$

De plus, nous avons que

$$\left\|E^\dagger\right\|_2^2 = \lambda_{\min}\left(E^\top E\right) = \lambda_{\min}\left(I_{d|\Sigma|} + U_{d|\Sigma|}\right) = 1.$$

En remplaçant dans l'Équation (7.5), pour le problème i, nous obtenons la borne, en supposant que $\left\|\hat{Q} - Q\right\|_2 \leq \sigma_d\kappa$,

$$\begin{aligned} \|\hat{\mathbf{x}}_i^* - \mathbf{x}_i^*\|_2 \leq \frac{2\sqrt{2}}{1-\kappa}\|\mathbf{q}_i\|_2 \left(1 + \frac{\sqrt{1+d\,|\Sigma|}}{1-\kappa}\left\|\hat{Q}\right\|_2\left\|Q^\dagger\right\|_2\right)\left\|Q^\dagger\right\|_2^2\left\|\hat{Q} - Q\right\|_2 \\ + \frac{1}{1-\kappa}\left\|\hat{Q}\right\|_2\left\|Q^\dagger\right\|_2\|\hat{\mathbf{e}}_i - \mathbf{e}_i\|_2 \\ + \frac{2}{1-\kappa}\left\|Q^\dagger\right\|_2\|\hat{\mathbf{q}}_i - \mathbf{q}_i\|_2. \end{aligned}$$

De plus, comme Q est diagonale par bloc, Q a le même spectre que $Z^\top$ et donc $\left\|Q^\dagger\right\|_2 = \left\|Z^\top\right\|_2 = \frac{1}{\sigma_d}$.

Analysons maintenant $\|\mathbf{q}_i\|_2$ et $\left\|\hat{Q}\right\|_2$. Premièrement, nous avons que

$$\|\mathbf{q}_i\|_2 \leq \|\mathbf{q}_i\|_1 \leq \sum_{\sigma\in\Sigma}\sum_{v\in\mathcal{S}}\frac{p(\mathcal{R}(i)\sigma v)}{p(\mathcal{R}(i)\Sigma\Sigma^\star)} \leq \frac{p(\mathcal{R}(i)\Sigma^\star)}{p(\mathcal{R}(i)\Sigma^\star)} = 1.$$

Deuxièmement, l'inégalité de Hölder implique que

$$\left\|\hat{Q}\right\|_2 = \left\|\hat{Z}^\top\right\|_2 \leq \sqrt{\left\|\hat{Z}^\top\right\|_1\left\|\hat{Z}^\top\right\|_\infty}.$$

Or, nous avons d'une part que $\left\|\hat{Z}^\top\right\|_1 = \max_{u\in\hat{\mathcal{R}}}\sum_{v\in\mathcal{S}}\frac{\hat{p}(uv)}{\hat{p}(u\Sigma)} \leq 1$. D'autre part, nous montrons que $\left\|\hat{Z}^\top\right\|_\infty = \max_{v\in\mathcal{S}}\sum_{u\in\hat{\mathcal{R}}}\frac{\hat{p}(uv)}{\hat{p}(u\Sigma)} \leq \left|\hat{\mathcal{R}}\right| = d$. Ainsi, nous concluons que,

$$\left\|\hat{Q}\right\|_2 \leq \sqrt{d}.$$

Mises bout à bout ces propriétés permettent d'établir que,

$$\begin{aligned}\|\hat{\mathbf{x}}_i^* - \mathbf{x}_i^*\|_2 \leq \frac{2\sqrt{2}}{\sigma_d^2(1-\kappa)}\left(1 + \frac{\sqrt{d(1+d\,|\Sigma|)}}{\sigma_d(1-\kappa)}\right)\left\|\hat{Z} - Z\right\|_2 \\ + \frac{\sqrt{d}}{\sigma_d(1-\kappa)}\|\hat{\mathbf{e}}_i - \mathbf{e}_i\|_2 \\ + \frac{2}{\sigma_d(1-\kappa)}\|\hat{\mathbf{q}}_i - \mathbf{q}_i\|_2.\end{aligned}$$

Nous avons vu lors de la propagation de l'erreur dans l'enveloppe convexe que $\sigma_d \leq 1$ et donc, comme $\kappa < 1$,

$$\begin{aligned}1 + \frac{\sqrt{d\,(1+d\,|\Sigma|)}}{\sigma_d\,(1-\kappa)} &\leq \frac{1 + \sqrt{d\,(1+d\,|\Sigma|)}}{\sigma_d\,(1-\kappa)} \\ &\leq \frac{(1+\sqrt{2})d\sqrt{|\Sigma|}}{\sigma_d\,(1-\kappa)},\end{aligned}$$

car pour $d \geq 1$ et $|\Sigma| \geq 1$, nous avons $1 + \sqrt{d\,(1+d\,|\Sigma|)} \leq (1+\sqrt{2})d\sqrt{|\Sigma|}$. Cela permet la simplification suivante,

$$\|\hat{\mathbf{x}}_i^* - \mathbf{x}_i^*\|_2 \leq \frac{2(\sqrt{2}+2)d\sqrt{|\Sigma|}}{\sigma_d^3(1-\kappa)^2}\left\|\hat{Z} - Z\right\|_2 + \frac{\sqrt{d}}{\sigma_d(1-\kappa)}\|\hat{\mathbf{e}}_i - \mathbf{e}_i\|_2 + \frac{2}{\sigma_d(1-\kappa)}\|\hat{\mathbf{q}}_i - \mathbf{q}_i\|_2.$$

Comme la norme infinie est le maximum de la somme en valeur absolue des lignes, nous montrons que,

$$\begin{aligned}\left\|\hat{A} - A\right\|_\infty &= \max_{i\in[1,d]}\left\|\hat{A}[i,:]^\top - A[i,:]^\top\right\|_1 \\ &= \max_{i\in[1,d]}\|\hat{\mathbf{x}}_i - \mathbf{x}_i\|_1 \\ &\leq \sqrt{d}\max_{i\in[1,d]}\|\hat{\mathbf{x}}_i - \mathbf{x}_i\|_2.\end{aligned}$$

Il nous reste à insérer les bornes sur $\left\|\hat{Z} - Z\right\|_2$, $\max_{i\in[1,d]}\|\hat{\mathbf{q}}_i - \mathbf{q}_i\|_2$ et $\max_{i\in[1,d]}\|\hat{\mathbf{e}}_i - \mathbf{e}_i\|_2$. Premièrement, nous utilisons les propriétés des normes 2 et de Frobenius pour montrer que

$$\begin{aligned}\left\|\hat{Z} - Z\right\|_2 \leq \left\|\hat{Z} - Z\right\|_F &= \sqrt{\sum_{i=1}^{d}\left(\left\|\hat{Z}[i,:] - Z[i,:]\right\|_2\right)^2} \\ &\leq \sqrt{\sum_{i=1}^{d}\left(\left\|\hat{\mathbf{d}}_{\hat{\mathcal{R}}(i)} - \mathbf{d}_{\mathcal{R}(i)}\right\|_2\right)^2} \\ &\leq \sqrt{d}\max_{i\in[1,d]}\left\|\hat{\mathbf{d}}_{\hat{\mathcal{R}}(i)} - \mathbf{d}_{\mathcal{R}(i)}\right\|_2.\end{aligned}$$

Ainsi, comme

$$\left\|\hat{Q} - Q\right\|_2 = \left\|\hat{Z} - Z\right\|_2 \leq \sqrt{d}\max_{i\in[1,d]}\left\|\hat{\mathbf{d}}_{\hat{\mathcal{R}}(i)} - \mathbf{d}_{\mathcal{R}(i)}\right\|_2 \kappa\sigma_d,$$

la condition $\left\|\hat{Q} - Q\right\|_2 \leq \kappa\sigma_d$ est satisfaite si, $\max_{i\in[1,d]} \left\|\hat{\mathbf{d}}_{\hat{\mathcal{R}}(i)} - \mathbf{d}_{\mathcal{R}(i)}\right\|_2 \leq \frac{1}{\sqrt{d}}\kappa\sigma_d$. Deuxièmement, nous avons que

$$\begin{aligned}
\max_{i\in[1,d]} \|\hat{\mathbf{q}}_i - \mathbf{q}_i\|_2 &= \max_{i\in[1,d]} \left\|\begin{pmatrix} \hat{Z}_1[i,:] - Z_1[i,:] & \dots & \hat{Z}_{|\Sigma|}[i,:] - Z_{|\Sigma|}[i,:] \end{pmatrix}\right\|_2 \\
&\leq \max_{i\in[1,d]} \left\|\begin{pmatrix} \hat{Z}[i,:] - Z[i,:] & \hat{Z}_1[i,:] - Z_1[i,:] & \dots & \hat{Z}_{|\Sigma|}[i,:] - Z_{|\Sigma|}[i,:] \end{pmatrix}\right\|_2 \\
&\leq \max_{i\in[1,d]} \left\|\hat{\mathbf{d}}_{\hat{\mathcal{R}}(i)} - \mathbf{d}_{\mathcal{R}(i)}\right\|_2.
\end{aligned}$$

Troisièmement, comme $\hat{\mathbf{b}} = \mathbf{b}$, nous avons que

$$\max_{i\in[1,d]} \|\hat{\mathbf{e}}_i - \mathbf{e}_i\|_2 = \max_{i\in[1,d]} \|\hat{\mathbf{c}}_i - \mathbf{c}_i\|_2 = \max_{i\in[1,d]} \left|\frac{\hat{p}(\hat{\mathcal{R}}(i))}{\hat{p}(\hat{\mathcal{R}}(i)\Sigma^\star)} - \frac{p(\mathcal{R}(i))}{p(\mathcal{R}(i)\Sigma^\star)}\right| = \|\hat{\boldsymbol{\alpha}}_\infty - \boldsymbol{\alpha}_\infty\|_\infty.$$

Les trois dernières égalités et inégalités permettent de conclure la démonstration, le résultat final s'obtient en remplaçant $\max_{i\in[1,d]} \left\|\hat{\mathbf{d}}_{\hat{\mathcal{R}}(i)} - \mathbf{d}_{\mathcal{R}(i)}\right\|_2 = \epsilon^{\text{conv}}$. □

7.3.4.3 Borne sur la perturbation des poids finaux

La démonstration de Proposition 43 est plus simple que les précédentes car elle ne fait pas intervenir la résolution d'un problème de programmation quadratique.

Proposition 43 (Borne sur les poids de transitions)**.**

$$\|\hat{\boldsymbol{\alpha}}_\infty - \boldsymbol{\alpha}_\infty\|_\infty \leq \epsilon^{conv}.$$

Démonstration. Par définition $\|\hat{\boldsymbol{\alpha}}_\infty - \boldsymbol{\alpha}_\infty\|_\infty = \max_{i\in[1,d]} \left|\frac{\hat{p}(\hat{\mathcal{R}}(i))}{\hat{p}(\hat{\mathcal{R}}(i)\Sigma^\star)} - \frac{p(\mathcal{R}(i))}{p(\mathcal{R}(i)\Sigma^\star)}\right|$. Or, nous avons

$$\begin{aligned}
\max_{i\in[1,d]} \left|\frac{\hat{p}(u)}{\hat{p}(u\Sigma^\star)} - \frac{p(u)}{p(u\Sigma^\star)}\right| &= \max_{i\in[1,d]} \sqrt{\left(\frac{\hat{p}(\hat{\mathcal{R}}(i))}{\hat{p}(\hat{\mathcal{R}}(i)\Sigma^\star)} - \frac{p(\mathcal{R}(i))}{p(\mathcal{R}(i)\Sigma^\star)}\right)^2} \\
&\leq \max_{i\in[1,d]} \sqrt{\sum_{v\in\mathcal{S}} \left(\frac{\hat{p}(\hat{\mathcal{R}}(i)v)}{\hat{p}(\hat{\mathcal{R}}(i)\Sigma^\star)} - \frac{p(\mathcal{R}(i)v)}{p(\mathcal{R}(i)\Sigma^\star)}\right)^2} \\
&\leq \max_{i\in[1,d]} \left\|\hat{\mathbf{d}}_{\hat{\mathcal{R}}(i)} - \mathbf{d}_{\mathcal{R}(i)}\right\|_2 = \epsilon^{\text{conv}}.
\end{aligned}$$

□

7.3.5 Propagation de l'erreur dans la série

Il existe plusieurs façons dans la littérature pour propager l'erreur sur la représentation linéaire dans l'erreur sur la série qu'elle réalise. Nous avons déjà abordé le sujet au Chapitre 3 en différenciant les bornes ponctuelles de celles en variation totale. La clef pour obtenir une borne en variation totale avec une dépendance polynomiale est la convergence de la série estimée. L'Algorithme 4 étant contraint à retourner un PNFA assure la convergence de la série estimée. Dans ce chapitre, la propagation de l'erreur dans la série estimée est faite par la méthode de [Balle, 2013; Hsu et collab., 2012] qui permet d'obtenir une borne sur $\sum_{u\in\Sigma^t} |\hat{p}(u) - p(u)|$ avec une dépendance polynomiale

en t. En sommant sur t, nous pouvons obtenir une borne sur $\sum_{u\in\Sigma^{\leq t}} |\hat{p}(u) - p(u)|$. Enfin, comme la série estimée réalisée est un PNFA, nous pouvons utiliser la décroissance géométrique de $\sum_{u\in\Sigma^{>t}} \hat{p}(u)$ et $\sum_{u\in\Sigma^{>t}} p(u)$ pour obtenir une borne ℓ_1 entre $\hat{p}$ et p comme le fait [Bailly, 2011a]. Il existe d'autres façons d'obtenir des bornes non ponctuelles. Par exemple, dans [Foster et collab., 2012], les auteurs s'intéressent à l'erreur relative, ce qui leur permet d'obtenir des bornes sur la divergence KL ou la distance ℓ_1. Dans [Balle et collab., 2015], les auteurs établissent une borne ℓ_2 en passant par la construction d'un automate réalisant le carré de la différence entre la vraie série et l'estimée. Néanmoins, nous nous cantonnons à l'étude d'une borne en distance ℓ_1. Ce choix est justifié par l'étude des bornes pour les PDFA au Chapitre 2. Le reste de cette section est dédié à la preuve du Théorème 19 et de ces Corollaires 1 et 2.

Théorème 19 (Borne ℓ_1 en probabilité sur l'erreur de l'algorithme CH-PRFA)**.**
Soit p un langage stochastique de $S_{\mathbb{R}^+}^{[[Res]]}(\Sigma)$, une base $\mathcal{B} = (\mathcal{P}, \mathcal{S})$ complète et résiduelle avec $|P| \geq 2$, on note $R \subset \mathrm{Res}(p)$ l'unique ensemble fini minimal tel que $\mathrm{Res}(p) \subset [[R]]$. Soit d la taille de R, on note σ_d la d^e plus grande valeur singulière de la matrice $(r(v))_{r\in R, v\in\mathcal{S}}$. Soit $\mathcal{D}$ un ensemble de mots générés selon p, on note $n = \min_{u\in\mathcal{P}} |\{\exists v \in \Sigma^\star | uv \in \mathcal{D}\}|$. Il existe une constante K telle que, pour tout $0 < \delta < 1$, pour tout $t > 0$, $\epsilon > 0$, avec une probabilité $1 - \delta$, si

$$n \geq K \frac{t^4 d^4 |\Sigma|}{\epsilon^2 \sigma_d^{10}} \log\left(\frac{|\mathcal{P}|}{\delta}\right),$$

l'algorithme CH-PRFA *retourne un PNFA réalisant un langage stochastique $\hat{p}$ tel que*

$$\sum_{u\in\Sigma^t} |\hat{p}(u) - p(u)| \leq \epsilon.$$

Le Théorème 19 établit pour tout t une borne sur $\sum_{u\in\Sigma^t} |\hat{p}(u) - p(u)|$. Ce type de borne convient mieux aux processus stochastiques qui définissent pour chaque $t > 0$ une distribution sur Σ^t, comme nous le verrons dans la prochaine section. Donc, nous prouvons les deux corollaires suivants plus adaptés aux langages stochastiques.

Corollaire 1.
En reprenant les mêmes hypothèses que le Théorème 19, Pour tout $0 < \delta < 1$, il existe une constante K telle que, pour tout $t > 0$, $\epsilon > 0$, avec une probabilité $1 - \delta$, si

$$n \geq K \frac{t^4 d^4 |\Sigma|}{\epsilon^2 \sigma_d^{10}} \log\left(\frac{|\mathcal{P}|}{\delta}\right),$$

l'algorithme CH-PRFA *retourne un PNFA réalisant un langage stochastique $\hat{p}$ tel que*

$$\sum_{u\in\Sigma^{\leq t}} |\hat{p}(u) - p(u)| \leq \epsilon.$$

Le Corollaire 1 n'est qu'une simple application du Théorème 19. Il permet surtout d'établir le Corollaire 2 qui borne la distance en variation totale entre langage estimé et celui ayant généré les exemples. Pour ce faire, le Corollaire 2 utilise convergence des langages stochastiques et la décroissante exponentielle de la probabilité des longs mots. Le rayon spectral, défini ci-dessous, joue un rôle clef dans l'analyse.

Définition 55 (Rayon spectral d'une série)**.**
On note $\rho(r) = \inf\{\rho | \exists K \in \mathbb{R}+, \forall t \in \mathbb{N}, |r(\Sigma^t)| < K\rho^t\}$ le rayon spectral d'une série r.

Définition 56 (Rayon spectral d'une matrice)**.**
On note $\rho(M)$ le rayon spectral d'une matrice M égal au maximum des valeurs absolues des valeurs propres.

Par définition, pour toute matrice M, nous avons $\rho(M) \leq \|M\|_2$. Le prochain Lemme est extrait de [Bailly, 2011a, Lemme 8].

Lemme 4 (Propriétés des rayons spectraux)**.**
Une série rationnelle r est convergente ($r(\Sigma^\star) < \infty$) si et seulement $\rho(r) < 1$. Dans ce cas, pour tout $\mathbb{R}$-MA $(\boldsymbol{\alpha}_0, A, \boldsymbol{\alpha}_\infty)$, $\rho(r_M) \leq \rho(A_\Sigma)$, où $A_\Sigma = \sum_{\sigma\in\Sigma} A_\sigma$.

Une conséquence directe est que si r est convergente alors il existe K tel que $\forall x \in \mathbb{R}^+$, $r(\Sigma^{>t}) < K\rho^t$.

Corollaire 2.
En reprenant les mêmes hypothèses que le Théorème 19, Pour tout $0 < \delta < 1$, il existe des constantes K_1, K_2 et ρ telles que, $\max(\rho(\hat{p}), \rho(p)) \leq \rho < 1$ et pour tout $t > 0$, $\epsilon > 0$, avec une probabilité $1 - \delta$, si

$$n \geq K_1 \log\left(\frac{\epsilon}{2K_2}\right)^4 \log(\rho)^{-4} \frac{d^4\,|\Sigma|}{\epsilon^2\sigma_d^{10}} \log\left(\frac{|\mathcal{P}|}{\delta}\right),$$

alors,

$$\sum_{u\in\Sigma^\star} |\hat{p}(u) - p(u)| \leq \epsilon.$$

Avant de commencer la démonstration du Théorème, nous définissons quelques termes et établissons un Lemme utile dans la preuve principale. Nous notons

$$\begin{aligned}
\rho_0 &= \|\hat{\boldsymbol{\alpha}}_0 - \boldsymbol{\alpha}_0\|_1,\\
\rho_\infty &= \|\hat{\boldsymbol{\alpha}}_\infty - \boldsymbol{\alpha}_\infty\|_\infty,\\
\rho_\Sigma &= \sum_{\sigma\in\Sigma} \left\|\hat{A}_\sigma - A_\sigma\right\|_\infty = \left\|\hat{A} - A\right\|_\infty.
\end{aligned}$$

De plus, nous introduisons les variables :

$$\begin{aligned}
\gamma_k &= \sum_{u\in\Sigma^k} \left\|\boldsymbol{\alpha}_0^\top A_u\right\|_1,\\
\gamma_\infty &= \|\boldsymbol{\alpha}_\infty\|_\infty,\\
\gamma_\Sigma &= \|A\|_\infty.
\end{aligned}$$

Lemme 5 (Borne sur les paramètres d'un PNFA)**.**
Pour un PNFA $(\boldsymbol{\alpha}_0, A, \boldsymbol{\alpha}_\infty)$, on a

$$\gamma_k \leq 1, \qquad \gamma_\infty \leq 1, \qquad \gamma_\Sigma \leq 1.$$

Démonstration. Il suffit de remarquer que,

$$\begin{aligned}
\gamma_k &= \sum_{u\in\Sigma^k} p(u\Sigma^\star) = p(\Sigma^k\Sigma^\star) \leq 1,\\
\gamma_\infty &\leq \max_{i\in[1,d]} \frac{p(\mathcal{R}(i))}{p(\mathcal{R}(i)\Sigma^\star)} \leq 1,\\
\gamma_\Sigma &= \left\|\sum_{\sigma\in\Sigma} A_\sigma \mathbf{1}\right\|_\infty = \|\mathbf{1} - \boldsymbol{\alpha}_\infty\|_\infty \leq 1.
\end{aligned}$$

□

Démonstration du Théorème 19. Supposons que $\epsilon^{\text{est}} \leq \frac{\sigma_d^3}{648\sqrt{d}}$ alors,

$$\epsilon^{\text{conv}} = \max_{i\in[1,d]} \left\| \hat{\mathbf{d}}_{\hat{\mathcal{R}}(i)} - \mathbf{d}_{\mathcal{R}(i)} \right\|_2 \leq 162 \frac{\epsilon^{\text{est}}}{\sigma_d^2}.$$

Supposons que $\epsilon^{\text{conv}} \leq \frac{1}{\sqrt{d}}\kappa\sigma_d$, d'après les Propositions 41 à 43, en prenant $\kappa = \frac{1}{3}$ et c_0, c_∞ et c_Σ des constantes adaptées, en utilisant que $d \geq 1$ et $|\Sigma| \geq 1$, nous obtenons que

$$\begin{aligned}
\rho_0 &\leq \frac{9}{2}(\sqrt{2}+1)\frac{d^2}{\sigma_d^3}\epsilon^{\text{conv}} + 3\frac{\sqrt{d}}{\sigma_d}\epsilon^{\text{est}} \\
&\leq 9^3(\sqrt{2}+2)\frac{d^2}{\sigma_d^5}\epsilon^{\text{est}} + 3\frac{\sqrt{d}}{\sigma_d}\epsilon^{\text{est}} \\
&\leq \frac{c_0 d^2}{\sigma_d^5}\epsilon^{\text{est}}.
\end{aligned}$$

De même, si $\epsilon^{\text{conv}} \leq \frac{1}{\sqrt{d}}\kappa\sigma_d$ alors,

$$\rho_\infty \leq \epsilon^{\text{conv}} \leq \frac{162}{\sigma_d^2}\epsilon^{\text{est}} = \frac{c_\infty}{\sigma_d^2}\epsilon^{\text{est}}.$$

De même, si $\epsilon^{\text{conv}} \leq \frac{1}{\sqrt{d}}\kappa\sigma_d$ alors,

$$\begin{aligned}
\rho_\Sigma &\leq \frac{9}{2}(\sqrt{2}+2)\frac{d^2\sqrt{|\Sigma|}}{\sigma_d^3}\epsilon^{\text{conv}} + \frac{3}{2}\frac{d}{\sigma_d}\|\hat{\boldsymbol{\alpha}}_\infty - \boldsymbol{\alpha}_\infty\|_\infty + 3\frac{\sqrt{d}}{\sigma_d}\epsilon^{\text{conv}} \\
&\leq 9^3(\sqrt{2}+2)\frac{d^2\sqrt{|\Sigma|}}{\sigma_d^5}\epsilon^{\text{est}} + 3^5\frac{d}{\sigma_d^3}\epsilon^{\text{est}} + 2\cdot 3^5\frac{\sqrt{d}}{\sigma_d^3}\epsilon^{\text{est}} \\
&\leq \frac{c_\Sigma d^2\sqrt{|\Sigma|}}{\sigma_d^5}\epsilon^{\text{est}}.
\end{aligned}$$

Soit $c = \max(c_0, c_\infty, c_\Sigma)$ et

$$\rho = \frac{c d^2\sqrt{|\Sigma|}}{\sigma_d^5(1-\kappa)^2}\epsilon^{\text{est}},$$

alors, d'après les précédentes inégalités,

$$\max(\rho_0, \rho_\infty, \rho_\Sigma) \leq \rho.$$

Ensuite, nous appliquons le Lemme 5.4.4 de Balle [2013] qui établit que, pour tout entier $t \geq 0$,

$$\sum_{u\in\Sigma^t} |\hat{p}(u) - p(u)| \leq (\gamma_\infty + \rho_\infty)\left((\gamma_\Sigma + \rho_\Sigma)^t \rho_0 + \rho_\Sigma \sum_{i=0}^{t-1} (\gamma_\Sigma + \rho_\Sigma)^i \gamma_{t-i-1}\right) + \gamma_t\rho_\infty.$$

D'après le Lemme 5, nous pouvons écrire

$$\begin{aligned}
\sum_{u\in\Sigma^t} |\hat{p}(u) - p(u)| &\leq (1+\rho_\infty)\left((1+\rho_\Sigma)^t\rho_0 + \rho_\Sigma\sum_{i=0}^{t-1}(1+\rho_\Sigma)^i\right) + \rho_\infty \\
&= (1+\rho_\infty)\left(1 + (1+\rho_\Sigma)^t\rho_0 + \rho_\Sigma\sum_{i=0}^{t-1}(1+\rho_\Sigma)^i\right) - 1 \\
&= (1+\rho_\infty)\left(1 + (1+\rho_\Sigma)^t\rho_0 + (1+\rho_\Sigma)^t - 1\right) - 1 \\
&= (1+\rho_\infty)(1+\rho_\Sigma)^t(1+\rho_0) - 1.
\end{aligned}$$

En remplaçant ρ_0, ρ_Σ et ρ_∞ par ρ, nous obtenons alors que,

$$\sum_{u\in\Sigma^t} |\hat{p}(u) - p(u)| \leq (1+\rho)^{t+2} - 1.$$

Nous remarquons que si $p = \mathcal{O}(\frac{1}{t})$ alors en utilisant que $(1+\frac{x}{2t})^t \leq 1+x$ pour $x \leq 1$, nous pouvons obtenir une borne sur $\sum_{u\in\Sigma^t} |\hat{p}(u) - p(u)|$ sans dépendance exponentielle en t. Ainsi, si

$$\epsilon^{\text{est}} \leq \frac{\sigma_d^3}{648\sqrt{d}}, \qquad \epsilon^{\text{conv}} \leq 3\frac{\sigma_d}{\sqrt{d}}, \qquad \rho \leq \frac{1}{2(t+2)},$$

alors

$$\sum_{u\in\Sigma^t} |\hat{p}(u) - p(u)| \leq \left(1 + \frac{2(t+2)\rho}{2(t+2)}\right)^{t+2} - 1 \leq 2(t+2)\rho.$$

Comme nous avons que,

$$\epsilon^{\text{conv}} \leq 162\frac{\epsilon^{\text{est}}}{\sigma_d^2}, \qquad \epsilon^{\text{est}} = \frac{\sigma_d^5}{cd^2\sqrt{|\Sigma|}}\rho,$$

les conditions sont satisfaites si

$$\epsilon^{\text{est}} \leq \frac{c'\sigma_d^5}{d^2\sqrt{|\Sigma|}(t+2)} \leq \min\left(\frac{1}{648}\frac{\sigma_d^3}{\sqrt{d}}, \frac{1}{2\cdot 3^5}\frac{\sigma_d^3}{\sqrt{d}}, \frac{2}{81}\frac{\sigma_d^5}{cd^2\sqrt{|\Sigma|}(t+2)}\right), \tag{7.6}$$

où c' est une constante telle que la dernière inégalité est vérifiée.
Enfin, d'après la proposition 39, avec probabilité $1-\delta$, nous avons pour $|\mathcal{P}| \geq 2$ et tout $n \geq 1$ que

$$\epsilon^{\text{est}} \leq \frac{1}{\sqrt{n}}\left(1 + \sqrt{\log\left(\frac{|\mathcal{P}|}{\delta}\right)}\right) \leq \left(1+\sqrt{\frac{3}{2}}\right)\sqrt{\frac{1}{n}\log\left(\frac{|\mathcal{P}|}{\delta}\right)},$$

car comme $\log(2) > \frac{2}{3}$, nous avons $\sqrt{\frac{3}{2}}\sqrt{\log\left(\frac{|\mathcal{P}|}{\delta}\right)} \geq 1$.
Nous pouvons alors trouver constante K telle que pour tout ϵ si

$$n \geq K\frac{t^2d^4\,|\Sigma|}{\epsilon^2\sigma_d^{10}}\log\left(\frac{|\mathcal{P}|}{\delta}\right),$$

alors les conditions (7.6) sont vérifiées et

$$\sum_{u\in\Sigma^t} |\hat{p}(u) - p(u)| \leq \frac{9}{4}(t+2)(t+1)\frac{cd^2\sqrt{|\Sigma|}}{\sigma_d^5}\epsilon^{\text{est}} \leq \epsilon.$$

□

Démonstration du Corollaire 1. D'après le Théorème 19, en remplaçant ϵ par $\frac{\epsilon}{t}$, il existe une constante K telle que pour tout $t > 0$, $\epsilon > 0$, avec une probabilité $1-\delta$, si

$$n \geq K\frac{t^4d^4\,|\Sigma|}{\epsilon^2\sigma_d^{10}},$$

alors

$$\sum_{u\in\Sigma^{\leq t}} |\hat{p}(u) - p(u)| \leq \frac{\epsilon}{t}.$$

Nous continuons en sommant sur t pour obtenir que

$$\sum_{u\in\Sigma^{\leq t}} |\hat{p}(u) - p(u)| \leq \epsilon.$$

□

Démonstration du Corollaire 2. Nous décomposons l'erreur de façon à faire apparaître la décroissance exponentielle de la série,

$$\begin{aligned}\sum_{u\in\Sigma^\star} |\hat{p}(u) - p(u)| &= \sum_{u\in\Sigma^{\leq t}} |\hat{p}(u) - p(u)| + \sum_{u\in\Sigma^{>t}} |\hat{p}(u) - p(u)| \\ &\leq \sum_{u\in\Sigma^{\leq t}} |\hat{p}(u) - p(u)| + \sum_{u\in\Sigma^{>t}} |\hat{p}(u)| + \sum_{u\in\Sigma^{>t}} |p(u)| \\ &= \sum_{u\in\Sigma^{\leq t}} |\hat{p}(u) - p(u)| + \hat{p}(\Sigma^{>t}) + p(\Sigma^{>t})\end{aligned}$$

D'après le Corollaire 1, il existe une constante K_1 telle que pour tout $t > 0$, $\epsilon > 0$, si

$$n \geq K_1 \frac{t^4 d^4 |\Sigma|}{\epsilon^2 \sigma_d^{10}},$$

alors, avec probabilité $1 - \delta$.

$$\sum_{u\in\Sigma^{\leq t}} |\hat{p}(u) - p(u)| \leq \frac{\epsilon}{2}.$$

Comme $\hat{p}$ et p sont des langages stochastiques, les séries convergent. Or d'après le Lemme 4, il existe une constante $K_2 \geq 1$ telle que pour tout $t > 0$, $\hat{p}(\Sigma^{>t}) + p(\Sigma^{>t}) < K_2 \max(\rho(\hat{p}), \rho(p))^t$. En particulier pour

$$t = \log\left(\frac{\epsilon}{2K_2}\right) \log\left(\max(\rho(\hat{p}), \rho(p))\right)^{-1},$$

nous avons $\hat{p}(\Sigma^{>t}) + p(\Sigma^{>t}) \leq \frac{\epsilon}{2}$ et donc, si

$$n \geq K_1 \log\left(\frac{\epsilon}{2K_2}\right)^4 \log\left(\max(\rho(\hat{p}), \rho(p))\right)^{-4} \frac{d^4 |\Sigma|}{\epsilon^2 \sigma_d^{10}},$$

alors,

$$\sum_{u\in\Sigma^\star} |\hat{p}(u) - p(u)| \leq \epsilon.$$

□

7.3.6 Discussions

Dans cette section, nous critiquons l'analyse non-asymptotique de la convergence de la section précédente.

Nous discutons d'abord des hypothèses du Théorème 19. Premièrement, en ce qui concerne la généralisation des résultats aux autres types de systèmes séquentielles linéaires, l'algorithme CH-PRFA s'applique aussi bien à l'inférence de processus

stochastiques et de processus contrôlés. Il suffit de changer les contraintes sur les probabilités de transition lors des NNLS à l'étape 9 et de prendre $\hat{\boldsymbol{\alpha}}_\infty = \mathbf{1}$. De plus, la preuve du Théorème 19 peut s'adapter aux processus stochastiques pour établir un résultat équivalent. Le cas des processus contrôlés n'est pas abordé ici car des hypothèses supplémentaires doivent être faites sur la politique d'échantillonnage. Nous mentionnons néanmoins que si la politique d'échantillonnage est aléatoire uniforme alors l'analyse se réduit à celle d'un processus stochastique. En fait, l'analyse des processus stochastiques est même simplifiée car aucune perturbation n'apparaît dans $\hat{\boldsymbol{\alpha}}_\infty$ qui vaut dans ce cas $\mathbf{1}$. De même, les contraintes des problèmes de NNLS de l'étape 9 de l'Algorithme 4 ne sont plus perturbées car $\hat{\mathbf{c}} = \mathbf{c} = 1$. Cependant, ce sont les erreurs d'estimation de $\hat{Q}$ qui font intervenir des polynômes d'ordre élevé des variables du problème. Ainsi la vitesse de convergence est identique.

Deuxièmement, nous notons que nous faisons l'hypothèse que les mots servant à l'apprentissage ont été tirés indépendamment. Pour les processus stochastiques, cette hypothèse est peu vraisemblable car, en pratique, on a souvent recours à l'algorithme *suffix-history* qui introduit une dépendance entre les sous-séquences extraite d'une unique longue séquence. Nous pourrions quand même obtenir des bornes non-asymptotiques en faisant intervenir un *mixing time*.

Troisièmement, le semi-module résiduel du langage stochastique cible est supposé de même dimension que le sous espace vectoriel minimal contenant les résiduels. Autrement dît, les arêtes de l'enveloppe conique définissent une matrice de rang plein. D'une part, cette hypothèse n'est pas si restrictive qu'elle le laisse penser car la quantité de PRFA de rang non plein est faible. Nous pourrions utiliser des arguments sur la densité des matrices inversibles pour montrer qu'un PRFA peut être approché d'autant que l'on veut par un PRFA de rang plein. Cependant, des problèmes de conditionnement, représentés par la forte dépendance en σ_d peuvent survenir. D'autre part, cette hypothèse peut être remplacée par une autre moins restrictive en fonction de l'algorithme de NMF séparable utilisé. Par exemple SNPA donne, dans le cas du rang plein, la même borne que SPA, mais s'étend aux cas des matrices dont le rang est déficient. À ce moment, la dépendance en σ_d est remplacée par un autre paramètre non nul quantifiant la difficulté de la distribution à apprendre. Les travaux théoriques [Ding et collab., 2015a] sur la factorisation de matrices séparables donnent l'intuition qu'une borne dépendant du paramètre simplicial est la meilleure que l'on puisse espérer. De plus, ce paramètre simplicial correspond à généralisation directe de la distinguabilité pour les PDFA. La distinguabilité a été démontrée comme une propriété nécessaire pour l'apprentissage des PDFA. Comme pour le paramètre simplicial, elle quantifie la difficulté de l'inférence du langage cible. La condition de rang plein n'est pas nécessaire pour l'étape des NNLS sous contrainte En effet, bien que les minimums locaux du problème d'optimisation ne soient plus uniques, ils définissent des solutions conduisant au même coût. Ainsi, pour rendre l'analyse possible il est possible de s'assurer de l'unicité de la solution en imposant à celle-ci d'être de norme minimale. L'analyse de la robustesse peut alors être conduite comme dans [Lőtstedt, 1983]. Celle-ci est néanmoins bien plus compliquée et fait apparaître des constantes non-explicites.

Quatrièmement, nous comparons la vitesse de convergence par rapport aux bornes existantes pour l'algorithme SPECTRAL. Pour ce qui est de la dépendance en $\log(|\mathcal{P}|)$, celle-ci pourrait être supprimée par l'utilisation des résultats non-asymptotiques sur la convergence des matrices de Hankel estimées [Denis et collab., 2014]. Bien que la dépendance en $|\Sigma|$ est meilleure que celle en $|\Sigma|^2$ des bornes de [Balle, 2013; Hsu et collab., 2012], il semblerait qu'elle soit superflue comme le montre [Foster et collab.,

2012] pour les HMMs. La dépendance en ϵ^{-2} et $\log(\delta-1)$ provenant directement des bornes de concentration est similaire aux travaux antérieurs. Enfin, la dépendance en d^4 est moins bonne que dans les bornes précédentes. L'ordre élevé de la dimension dans la borne provient en grande partie du nombre de contraintes lors de la NNLS. Cela explique cette différence avec les bornes pour l'algorithme Spectral. La dépendance en σ_d^{-10}, plutôt que σ_d^{-6} est une conséquence directe de l'algorithme SPA.

La principale différence avec les bornes existantes réside dans la condition sur n, plutôt que N, où n est le plus petit nombre de fois qu'un préfixe de la base est observé dans l'ensemble d'apprentissage et N la taille de cet ensemble. En utilisant l'inégalité de Hoeffding, nous pourrions convertir la condition sur n en une condition sur N. Cette condition ferait intervenir les mêmes termes avec leurs exposants ainsi que la probabilité μ du préfixe le moins fréquent. Cette particularité provient de l'estimation de distributions conditionnelles (les langages résiduels de p) plutôt que de distributions jointes. Nous pouvons nous demander si l'on ne peut pas plutôt dériver un algorithme travaillant directement à partir de la matrice de Hankel (représentant la distribution jointe) tout en étant robuste. La principale limitation à un tel algorithme consiste en la réalisant de l'étape 5, la recherche de l'enveloppe conique. En fait, la plupart des algorithmes font l'hypothèse que les données sont représentées dans une enveloppe convexe ou, pire, sur le simplexe, quitte à normaliser. Cette faiblesse des algorithmes de NMF pour les matrices séparables a déjà été identifiée dans les problèmes d'extraction de thèmes dans un corpus de document. Comme le nombre de mots par document varie, les distributions empiriques des documents les plus courts sont les moins bien estimées. Revenons aux langages stochastiques. Alors que le bruit d'estimation des probabilités jointes de la matrice de Hankel est uniforme sur les lignes, le bruit d'estimation des probabilités conditionnelles est plus élevé dans les lignes correspondantes à des préfixes peu fréquents. Le problème est qu'une grande majorité des travaux supposent que le bruit est uniforme.

Par exemple, pour les approches récursives, l'algorithme SPA et SNPA sélectionnent dans l'ordre les vecteurs les plus extrêmes en norme ℓ_2 par rapport au sous espace vectoriel (dans le cas de SPA) ou à l'enveloppe convexe (dans le cas de SNPA) générés par les vecteurs déjà identifiés. Nous nous rendons bien compte qu'une telle procédure peut être fortement perturbée si le bruit est non-uniforme car un vecteur mal estimé dans un espace de très grande dimension peut très facilement être considéré comme étant extrême sans l'être réellement et masquer plusieurs vecteurs supportant le cône. Ces erreurs se répercutent ensuite dans l'étape de régression pour l'inférence des paramètres de la représentation linéaire. Finalement, c'est la perturbation du langage résiduel associé au préfixe le moins fréquent qui quantifie la robustesse de l'algorithme.

D'autres algorithmes récursifs, comme XRAY, travaillent directement dans le cône et donc avec les probabilités jointes. Cependant, aucune analyse de la robustesse n'est donnée et il reste donc possible que celle-ci dépende quand même du préfixe le moins fréquent. Nous verrons dans les expériences que, bien que XRAY, soit connu pour mieux fonctionner dans les problèmes d'extraction de thèmes, le problème de sensibilité aux préfixes peu fréquents reste présent.

Quelles sont alors les solutions possibles pour augmenter la robustesse de l'Algorithme 4 ? La première solution consiste à rechercher les vecteurs supports du cône parmi les langages résiduels associés aux préfixes les plus fréquents (c'est d'ailleurs l'approche utilisée pour construire la base dans les expériences). Cette solution contraint alors la base de préfixes à être de taille modeste pour ne pas inclure les préfixes les moins probables. Le nombre de suffixes, lui, n'a pas besoin d'être limité. Un grand nombre

de suffixes dans la base conduit d'ailleurs à une meilleure estimation de la représentation linéaire lors des problèmes de NNLS sous contraintes mais au prix d'un temps de calcul accru. Cependant, cette solution a un défaut assez évident. Pour la suite de la discussion, nous pouvons considérer que $\mathcal{S} = \Sigma^\star$ afin de simplifier le raisonnement, nous notons alors $\mathcal{B} = (\mathcal{P}, \Sigma^\star)$ la base utilisée. En réduisant le nombre de préfixes, nous risquons d'obtenir une base non résiduelle. Ainsi, certains langages résiduels de R (caractérisant le langage p, voir Proposition 16) ne pourront pas être identifiés parmi $\mathrm{Res}_{\mathcal{B}}(p)$. Nous notons $\overline{R}$ l'ensemble de ces langages ($\overline{R} = R \setminus \mathrm{Res}_{\mathcal{B}}(p)$). L'effet peut être plus ou moins dommageable selon la position des langages résiduels $\mathrm{Res}_{\mathcal{B}}(p)$. Soit $q \in \overline{R}$, s'il existe un langage $q' \in \mathrm{Res}_{\mathcal{B}}(p)$, tel que $q \approx q'$ alors q' sera probablement identifié à la place de q comme un des langages résiduels extrêmes, conduisant à une erreur faible. S'il n'existe pas de q' proche de q, alors l'erreur risque d'être bien plus élevée.

Dès lors, il est légitime d'adopter une stratégie qui, pour la sélection du prochain vecteur, va choisir un vecteur extrême mal estimé s'il n'existe pas de vecteur moins bruité dans son entourage, ou bien un vecteur moins extrême s'il est bien estimé. Ce genre de stratégie peut se révéler très bénéfique car les langages stochastiques résiduels de R sont souvent présents plusieurs fois dans $\mathrm{Res}_{\mathcal{B}}(p)$, à chaque fois associé à un préfixe différent. Prenons un langage $q \in R$, nous notons $\mathcal{U}$ l'ensemble des préfixes qui lui sont associés ($\mathcal{U} = u \in \mathcal{P} | u^{-1}p = q$). Soit $\hat{Q}$ l'ensemble des langages stochastiques résiduels estimés associé à $\mathcal{U}$ ($\hat{Q} = \{u^{-1}\hat{p} | u \in \mathcal{U}\}$). Nous remarquons que les langages de $\hat{Q}$ sont proches si bien qu'en l'absence d'erreur d'estimation, $\hat{Q}$ se résume à q. Or, parmi les langages de $\hat{Q}$ certains seront mieux estimés que d'autres. Ce sont ces langages que la stratégie doit identifier.

Une deuxième possibilité est de se tourner vers d'autres types d'algorithmes. En effet, certains algorithmes issus de l'approche par programmation linéaire autorisent une plus grande souplesse dans la définition du problème ce qui leur permet de prendre en compte différents modèles de bruit. Malheureusement, ces algorithmes sont très coûteux en temps de calcul. L'état de l'art de l'approche par projections aléatoires n'apporte pas de solution complète à notre problème de préfixes peu fréquents, mais peut constituer une bonne piste pour des travaux futurs pour deux raisons. La première est que les projections aléatoires permettent de réduire la complexité de calcul en divisant le problème en sous problèmes de moindre dimension (généralement 1 pour une droite ou 2 pour le plan). Les algorithmes utilisés actuellement pour résoudre les sous-problèmes ont été choisi pour leur simplicité et suppose que le bruit d'estimation est uniforme. À la place, nous pourrions mettre un algorithme issu de l'approche par programmation linéaire afin de combiner souplesse dans la définition du bruit et rapidité de calcul. La deuxième raison est que la formulation probabiliste des algorithmes par projections aléatoires pourrait servir de base à un algorithme utilisant des bornes non-asymptotiques sur l'erreur des langages résiduels et rendant explicite le compromis entre choisir des vecteurs extrêmes ou des vecteurs bien estimés.

7.4 Expériences

Dans les expériences, nous proposons d'étudier trois procédures différentes pour identifier l'enveloppe convexe. La première, dénommée SPA, utilisant l'algorithme du même nom, est celle utilisée pour l'analyse non-asymptotique de l'erreur d'apprentissage. Comme nous l'avons vu dans la section précédente, les performances de cette procédure peuvent se dégrader si la base choisie est trop grande. Les résultats des

expériences confirmeront notre intuition.

Pour palier ce problème, nous remarquons que les vecteurs supports de l'enveloppe conique de $\{\grave{u}p|u \in \mathcal{P} \cap \text{res}(p)\}$ sont ceux de l'enveloppe convexe de $\{u^{-1}p|u \in \mathcal{P} \cap \text{res}(p)\}$ une fois normalisé. Dans le premier ensemble, l'erreur d'estimation est la même pour tous les vecteurs et décroît en $N^{-1/2}$. Nous évitons donc le problème. Cependant, nous avons vu que SPA nécessite une renormalisation pour identifier une enveloppe conique afin de se ramener à l'identification d'une enveloppe convexe. Évidemment, cette procédure réintroduit le problème, c'est-à-dire une erreur élevée sur les vecteurs associés aux préfixes peu fréquents. Ainsi, la deuxième procédure consiste à remplacer l'algorithme SPA par XRAY qui est capable d'identifier une enveloppe conique sans normalisation. Nos résultats montrent que cette propriété permet d'atténuer le problème mais celui-ci reste tout de même présent.

Enfin, dans la troisième approche, appelée SPA+, nous proposons d'utiliser l'algorithme SPA pour identifier les vecteurs supports de l'enveloppe convexe (au lieu de conique) de $\{\grave{u}p|u \in \mathcal{P} \cap \text{res}(p)\}$. Nous notons $\mathcal{X}$ ces vecteurs supports une fois normalisés pour appartenir au simplexe. Soit $\mathcal{Y}$ les vecteurs supports de l'enveloppe convexe de $\{u^{-1}p|u \in \mathcal{P} \cap \text{res}(p)\}$, nous allons montrer que $\mathcal{Y} \subset \mathcal{X}$. Et donc, nous obtenons que $\text{conv}(\mathcal{X}) = \text{conv}(\mathcal{Y})$ car nous avons $\text{conv}(\mathcal{X}) \subset \text{conv}(\mathcal{Y})$ par définition de $\mathcal{Y}$. Ainsi, la procédure est bien fondée. En fait, celle-ci va juste identifier plus de vecteurs que nécessaire. De cette façon, l'automate appris sera équivalent mais ne sera plus forcément minimal. Pour montrer que $\mathcal{Y} \subset \mathcal{X}$, nous notons $\text{coni}(\mathcal{X})$ l'enveloppe conique de $\mathcal{X}$. Par définition, les vecteurs supports de $\text{coni}(\mathcal{X})$ font partie de $\mathcal{X}$ et sont les mêmes que $\{u^{-1}p|u \in \mathcal{P} \cap \text{res}(p)\} = \mathcal{Y}$. Donc, $\mathcal{Y}$ est contenu dans $\mathcal{X}$.

Pour que l'algorithme fonctionne, il est néanmoins nécessaire que l'enveloppe convexe de $\{\grave{u}p|u \in \mathcal{P} \cap \text{res}(p)\}$ soit générée par une famille finie. Expérimentalement cette troisième procédure produit de bons résultats quelle que soit la taille de $\mathcal{P}$.

7.5 Résultats

Les résultats reportés ici complètent ceux du Chapitre 5. Le détail des expériences et les résultats déjà exposés ne sont pas répétés dans ce chapitre. En conclusion de cette thèse, une comparaison de tous les algorithmes présentés est proposée.

7.5.1 Comparaison des algorithmes de NMF séparables

Dans cette section, nous comparons les trois procédures décrites dans la section précédente pour les trois ensembles de données selon les différents critères définis au Chapitre 5.

7.5.1.1 PAutomaC

Sur les Figures 7.5 et 7.6, les performances des trois procédures pour identifier l'enveloppe convexe sont comparées en fonction du nombre de suffixes et surtout de préfixes dans la base. Comme le prévoit l'analyse non-asymptotique, les résultats de SPA se dégradent quand le nombre de préfixes augmente. Pour XRAY, l'effet est légèrement atténué. En particulier pour la perplexité, si le nombre de préfixes dans la base est faible (200) les résultats des trois procédures sont comparables. Cela est dû au fait que les préfixes sélectionnés pour construire la base sont les plus fréquents. Ainsi la fréquence du préfixe le moins probable dans la base reste suffisamment élevé.

Pour le WER, il subsiste un léger écart de performance même avec 200 préfixes. En comparaison, les performances de SPA+ varient très peu en fonction de la taille de la base. Ainsi, SPA+ semble résoudre ce problème.

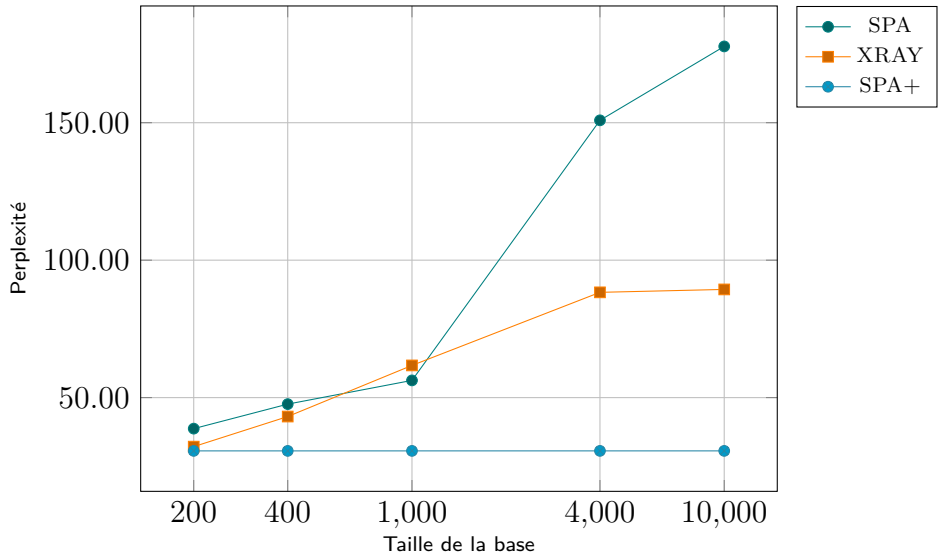


FIGURE 7.5 – Comparaison en termes de perplexité moyenne sur douze problèmes de PAutomaC des algorithmes de NMF pour CH-PRFA en fonction de la taille de la base.

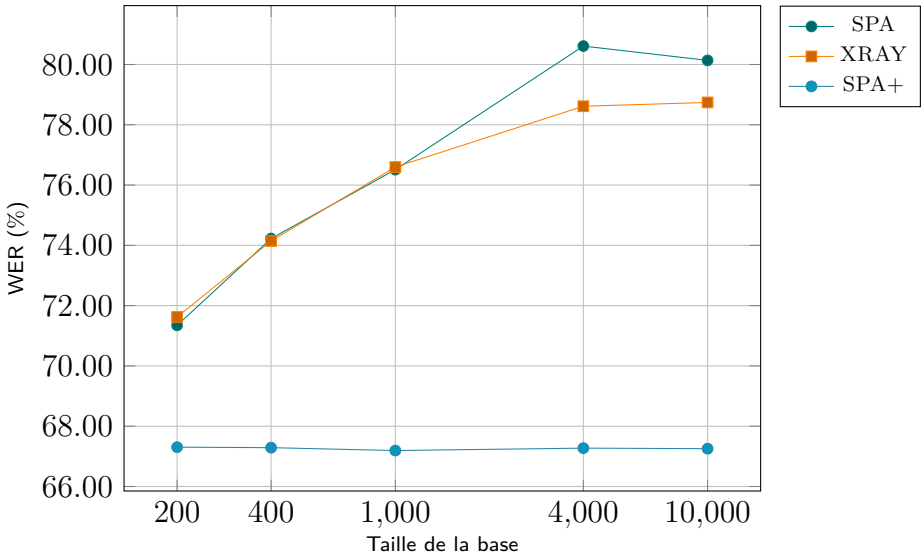


FIGURE 7.6 – Comparaison en termes de WER moyenne sur douze problèmes de PAutomaC des algorithmes de NMF pour CH-PRFA en fonction de la taille de la base.

7.5.1.2 Penn-Treebank

Sur les données de Penn-Treebank, les performances varient peu. Néanmoins, SPA+ est le plus performant.

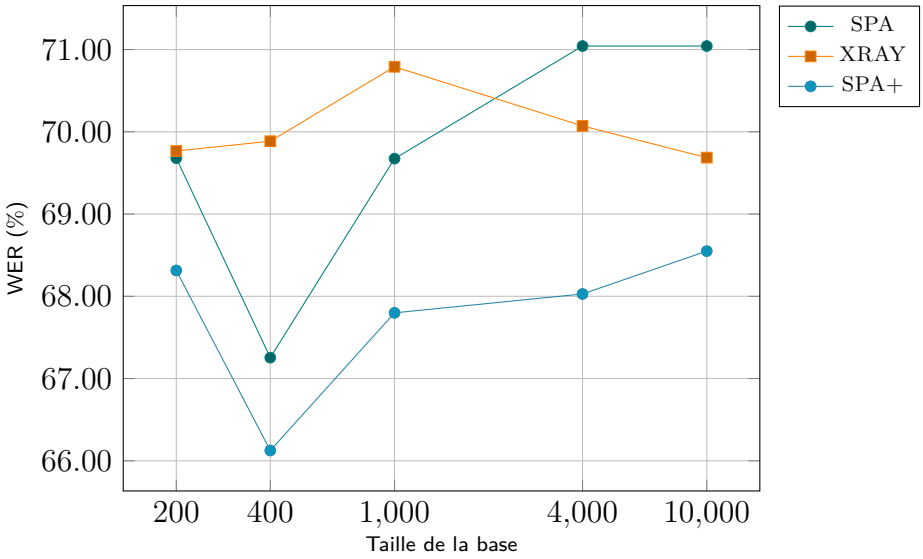


FIGURE 7.7 – Comparaison en termes de WER moyenne sur Penn-Treebank des algorithmes de NMF pour CH-PRFA en fonction de la taille de la base.

7.5.1.3 Wikipédia

Pour notre expérience consistant à apprendre le texte de Wikipédia, nous avons évalué les trois procédures selon deux critères en fonction de la dimension (de 10 à 100) et de la taille de la base (500 ou 10000) sur deux ensembles d'apprentissage (un petit et un moyen). Globalement, nous observons les mêmes écarts de performances que sur PAutomaC. Ainsi avec une grande base (10000 préfixes et suffixes), SPA+ est le meilleur, suivi de XRAY. Sur une petite base et uniquement pour la vraisemblance conditionnelle, les performances des trois méthodes sont similairement très bonnes. À partir d'une certaine dimension, nous observons que SPA et XRAY peuvent donner de meilleurs résultats que SPA+.

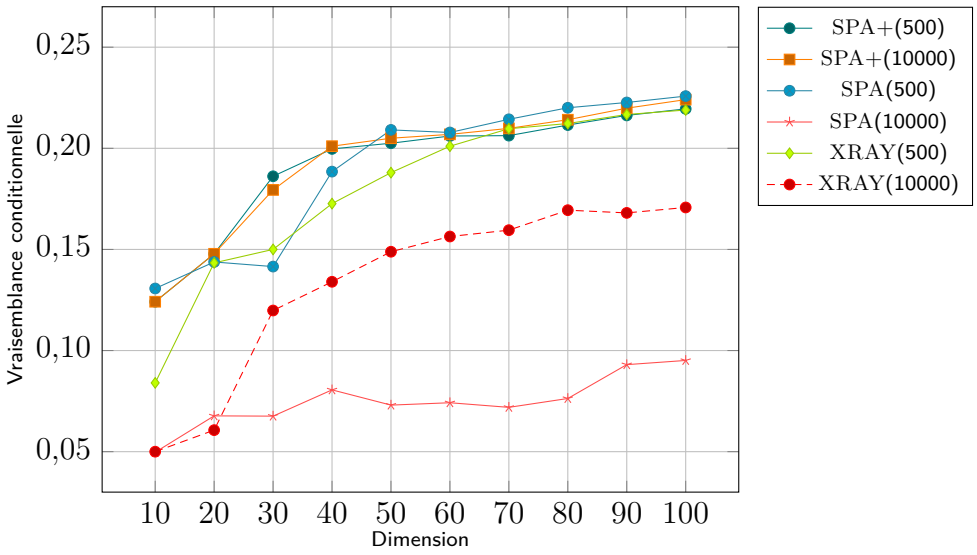


FIGURE 7.8 – Comparaison en termes de vraisemblance conditionnelle des algorithmes de NMF pour CH-PRFA en fonction de la dimension et de la taille de la base (indiquée entre parenthèses). 409 séquences de Wikipédia ont été utilisées pour l'apprentissage.

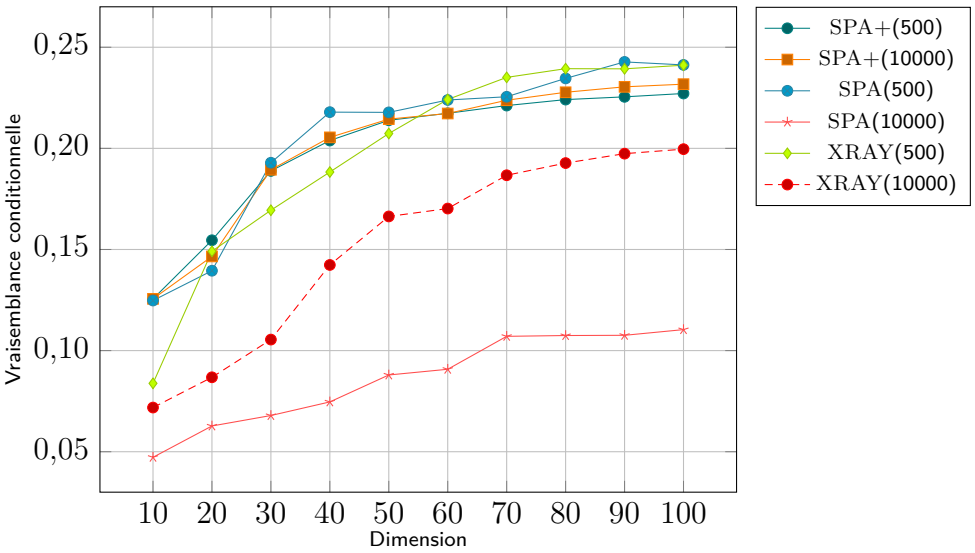


FIGURE 7.9 – Comparaison en termes de vraisemblance conditionnelle des algorithmes de NMF pour CH-PRFA en fonction de la dimension et de la taille de la base (indiquée entre parenthèses). 41943 séquences de Wikipédia ont été utilisées pour l'apprentissage.

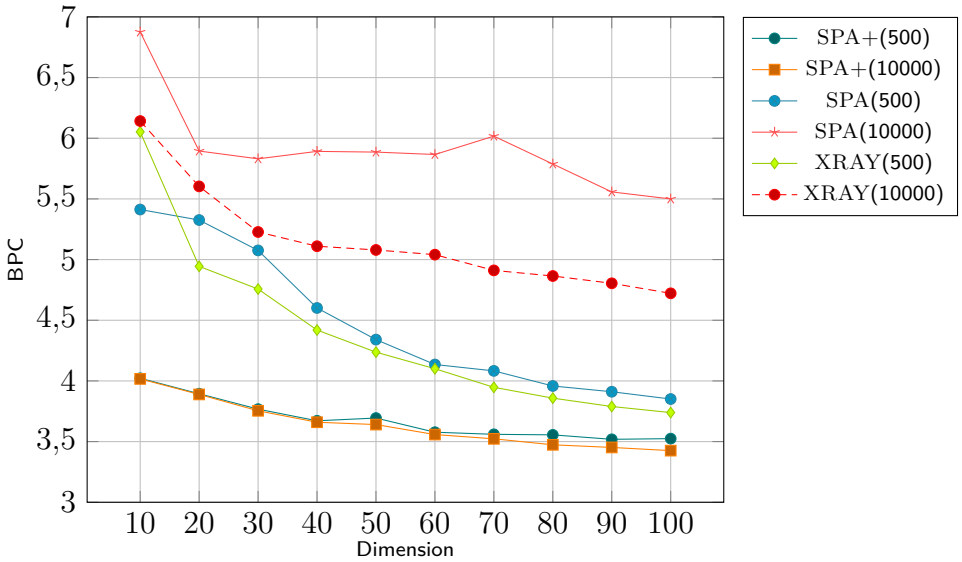


FIGURE 7.10 – Comparaison en termes de BPC des algorithmes de NMF pour CH-PRFA en fonction de la dimension et de la taille de la base (indiquée entre parenthèses). 409 séquences de Wikipédia ont été utilisées pour l'apprentissage.

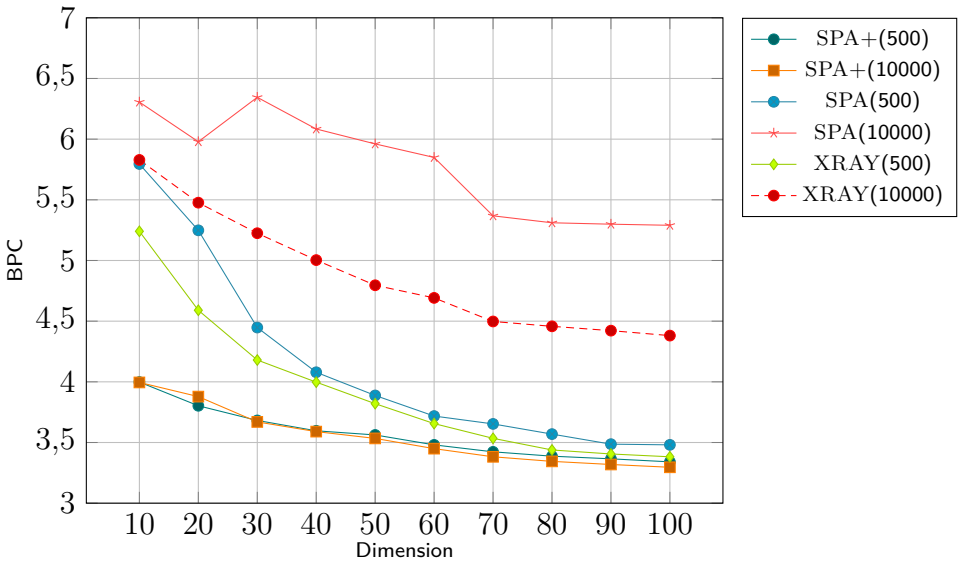


FIGURE 7.11 – Comparaison en termes de BPC des algorithmes de NMF pour CH-PRFA en fonction de la dimension et de la taille de la base (indiquée entre parenthèses). 41943 séquences de Wikipédia ont été utilisées pour l'apprentissage.

7.5.2 Initialisation d'algorithmes itératifs

Dans cette section, nous détaillons les performances pour CH-PRFA en utilisant SPA+ pour identifier l'enveloppe convexe. De plus, nous comparons celles-ci à CH-PRFA+BW et BW.

7.5.2.1 PAutomaC

Sur les données du challenge PAutomaC, pour le WER et la perplexité, nous remarquons que CH-PRFA montre des résultats compétitifs à l'algorithme BW. Combiné CH-PRFA+BW se classe premier pour les deux critères. Les performances de CH-PRFA sont meilleures que celles de BW pour la perplexité et moins bonnes pour le WER. Finalement, nous remarquons que les modèles appris par CH-PRFA sont de tailles raisonnables.

type	n°	BW	CH-PRFA	CH-PRFA+BW
HMM	1	56.325 (2)	**30.426 (42)**	**30.426 (43)**
	14	**116.847 (14)**	117.946 (18)	117.946 (19)
	33	**31.876 (26)**	32.391 (26)	32.333 (27)
	45	24.207 (2)	**24.101 (2)**	**24.101 (3)**
PDFA	6	**67.196 (26)**	67.517 (18)	67.517 (19)
	7	**51.250 (26)**	51.372 (14)	51.253 (15)
	27	69.462 (6)	**43.114 (46)**	**43.114 (47)**
	42	27.899 (6)	**16.031 (10)**	**16.031 (11)**
PNFA	29	24.971 (46)	24.787 (46)	**24.190 (47)**
	39	11.823 (6)	**10.033 (10)**	**10.033 (11)**
	43	37.930 (2)	**32.826 (10)**	**32.826 (11)**
	46	12.017 (46)	12.305 (42)	**12.010 (43)**
moyenne		35.886	30.603	**30.468**

TABLEAU 7.1 – Perplexité sur douze problèmes de PAutomaC de CH-PRFA, CH-PRFA+BW et l'algorithme de BW.

type	n°	BW	CH-PRFA	CH-PRFA+BW
HMM	1	77.138 (34)	76.085 (46)	**72.224 (47)**
	14	**68.414 (14)**	69.963 (38)	68.520 (39)
	33	74.228 (30)	74.318 (10)	**74.224 (11)**
	45	**78.095 (18)**	78.593 (2)	78.141 (3)
PDFA	6	59.653 (2)	48.520 (30)	**47.786 (31)**
	7	48.289 (34)	48.298 (42)	**48.260 (43)**
	27	**73.030 (30)**	77.776 (46)	73.117 (47)
	42	56.582 (30)	56.552 (14)	**56.522 (15)**
PNFA	29	**48.504 (42)**	50.681 (42)	48.788 (43)
	39	**59.174 (38)**	59.398 (10)	59.235 (11)
	43	**77.314 (26)**	78.097 (26)	77.711 (27)
	46	**77.357 (30)**	87.416 (42)	77.396 (43)
moyenne		66.482	67.141	**65.160**

TABLEAU 7.2 – WER en pourcentage sur douze problèmes de PAutomaC de CH-PRFA, CH-PRFA+BW et l'algorithme de BW.

7.5.2.2 Penn-Treebank

Les résultats sur Penn-Treebank montrent que CH-PRFA fournit une bonne initialisation pour l'algorithme de BW car CH-PRFA+BW se classe premier. L'écart

de performance entre BW et CH-PRFA est similaire à celui entre BW et les autres algorithmes issus de la MoM travaillant à partir de la série empirique d'origine (voir Chapitre 5). Sur cet exemple, nous constatons que CH-PRFA s'exécute 500 fois plus rapidement que l'algorithme de BW. La vitesse d'exécution de CH-PRFA est similaire sur les autres ensembles de données.

	BW	CH-PRFA	CH-PRFA+BW
WER (%)	60.774	66.125	**59.206**
dimension	30	46	46
taille de la base	*N/A*	400	10000
hankel (s)	0.00	0.95	1.14
semi-module (s)	0.00	0.09	6.94
paramètre (s)	1335.99	2.16	1710.59
total (s)	1335.99	3.21	1718.67

TABLEAU 7.3 – Résultats pour Penn-Treebank de CH-PRFA, CH-PRFA+BW et l'algorithme de BW.

7.5.2.3 Wikipédia

Les Figures 7.12 et 7.12 comparent les performances de CH-PRFA, CH-PRFA+BW et de l'algorithme de BW sur deux critères différents en utilisant uniquement 409 séquences. Même sur ce petit ensemble de données, l'algorithme BW ne termine pas en un temps raisonnable (inférieur à 12 heures) pour les modèles de dimension supérieure à 70. Pour la vraisemblance conditionnelle, utiliser le PNFA appris par CH-PRFA pour initialiser BW permet d'obtenir de meilleures performances qu'une initialisation aléatoire. Pour le BPC, l'écart de performances est beaucoup moins prononcé.

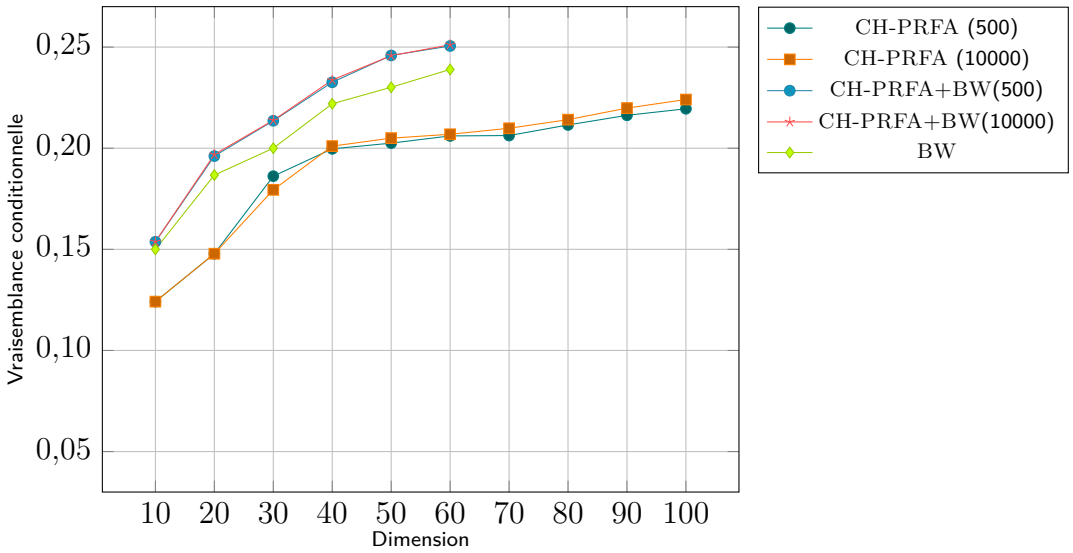


FIGURE 7.12 – Performances de CH-PRFA, CH-PRFA+BW et de l'algorithme de BW pour la vraisemblance conditionnelle en fonction de la dimension et de la taille de la base (indiquée entre parenthèses). 409 séquences de Wikipédia ont été utilisées pour l'apprentissage.

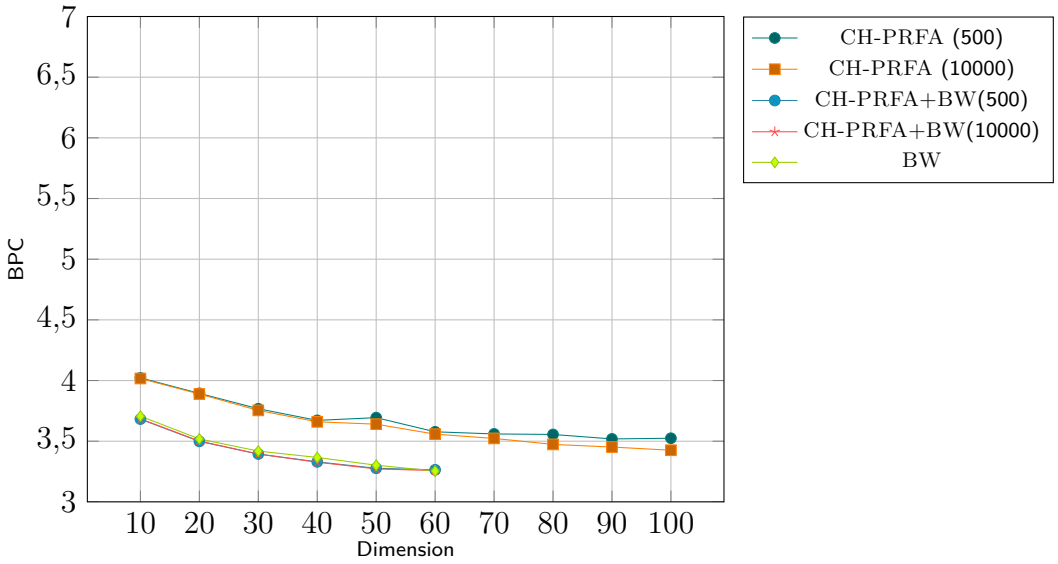


FIGURE 7.13 – Performances de CH-PRFA, CH-PRFA+BW et de l'algorithme de BW pour le BPC en fonction de la dimension et de la taille de la base (indiquée entre parenthèses). 409 séquences de Wikipédia ont été utilisées pour l'apprentissage.

7.6 Conclusions

Dans ce chapitre, pour apprendre des PRFA, nous avons proposé l'algorithme CH-PRFA, basé sur la caractérisation par une matrice de Hankel séparable. L'existence de plusieurs algorithmes de NMF pour les matrices séparables, à la fois rapides et robustes, nous ont permis de prouver que $S^{[[Res]]}_{\mathbb{R}^+}(\Sigma)$ est PAC-apprenable par un algorithme certes impropre mais qui retourne un PNFA réalisant donc une distribution. De plus, nous avons proposé trois procédures, SPA, XRAY et SPA+, pour identifier l'enveloppe convexe. L'analyse non-asymptotique et les expériences montrent que les résultats de SPA dépendent fortement du nombre de préfixes. Utiliser SPA+ corrige ce défaut. L'analyse non-asymptotique de SPA+ reste encore à établir.

Empiriquement, nous avons montré que CH-PRFA est performant, bien plus rapide que l'algorithme de BW qui ne passe pas à l'échelle. Comme le montrent les résultats présentés en Conclusion de cette thèse, les résultats de CH-PRFA sont presque aussi bons que NNSPECTRAL qui domine toutes les autres méthodes. À la différence de NNSPECTRAL, CH-PRFA contraint le modèle appris à réaliser une distribution et dispose de garanties PAC. Cette propriété fait de CH-PRFA une bonne initialisation pour l'algorithme BW.

Enfin, nous remarquons qu'à la différence de l'algorithme SPECTRAL ou NNSPECTRAL, l'étape de régression dans CH-PRFA, $\min_x \left\|\hat{Q}\hat{\mathbf{x}} + \hat{\mathbf{q}}\right\|_2$ avec $\hat{Q} = \text{diag}(\hat{Z}^\top, \ldots, \hat{Z}^\top)$, fait intervenir des probabilités conditionnelles. Comme pour l'étape de NMF, le bruit n'est pas uniforme selon les colonnes de Q et dépend de la fréquence des préfixes. Il serait intéressant d'étudier une variante de CH-PRFA, où la régression est pondérée par une matrice diagonale $\hat{W}$ contenant les fréquences empiriques des préfixes de façon à uniformiser le bruit. Cependant cela obligerait les contraintes définies Équation (1.1) à dépendre de la matrice de pondération $\hat{W}$. Du côté de l'analyse non-asymptotique, les perturbations entre W et $\hat{W}$ ferait apparaître de nouveaux termes.

Conclusion

Dans cette thèse, nous nous sommes intéressés à l'inférence de systèmes séquentiels linéaires, en particulier aux langages probabilistes. Nous avons montré au Chapitre 1 que le formalisme des automates à multiplicité permettait de traiter conjointement aussi d'autres types de systèmes séquentiels linéaires tels que les processus stochastiques et processus contrôlés. Nous y avons aussi décrit une hiérarchie de classes d'automates de richesses et de compacités différentes. Pour chacune de ces classes, nous avons donné plusieurs caractérisations, en particulier à partir de la matrice de Hankel. Dans le Chapitre 2 nous avons revu différents algorithmes d'apprentissage et leurs garanties théoriques. Associés aux résultats négatifs, ils permettent de définir une extension du modèle PAC adaptée à l'apprentissage de langages stochastiques. Parmi les algorithmes d'apprentissage nous avons présentés ceux basés sur la MoM. Ceux-ci fournissent une alternative consistante et efficace, aux algorithmes traditionnels, tels que EM, qui peuvent rester coincé dans un minimum local. De plus, la plupart de ces algorithmes possèdent des garanties non-asymptotiques et leur complexité est linéaire avec le nombre d'exemples. Cependant, ces algorithmes possèdent deux défauts majeurs, souvent évoqués dans la littérature.

Premièrement, les garanties théoriques de l'algorithme Spectral supposent que la dimension du modèle appris est égale à celle à celle du modèle cible. Ainsi, au Chapitre 3, nous avons étudié l'apprentissage de modèles compressés. Nous y avons montré que si le nombre d'exemples est faible il peut parfois être judicieux d'apprendre un modèle plus petit, quitte à ignorer une partie correspondante aux faibles valeurs singulières, afin d'éviter un éventuel sur-apprentissage. Le compromis entre sur-apprentissage et erreur de modélisation dépend de la décroissance du spectre de la matrice de Hankel. Ayant montré qu'adapter la taille du modèle aux données est primordiale, nous avons proposé un algorithme spectral régularisé à l'aide de la norme nucléaire. Afin d'éviter les instabilités liées au mauvais conditionnement de la matrice de Hankel, lors de l'étape de régression linéaire, nous avons aussi proposé une régularisation de Tikhonov. Les expériences montrent que sur des HMMs générés aléatoirement, notre algorithme produit de meilleurs résultats en repoussant le compromis biais-variance. Ce même algorithme a été utilisé pour une tâche de contrôle en guerre électronique au Chapitre 4. Ce chapitre plus appliqué aborde aussi les problématiques d'apprentissage en ligne et de compromis exploration-exploitation pour les méthodes spectrales.

Deuxièmement, nous avons vu que la classe la plus générale de langages stochastiques n'était pas PAC-apprenable par un algorithme propre. Étant donné que cette classe de langages n'est pas robuste, les algorithmes basés sur la MoM ne retournent pas en pratique des distributions. Ce problème a motivé la suite des chapitres de la thèse. D'abord, nous avons remarqué au Chapitre 5 que les erreurs dans les automates à paramètres réels, les $\mathbb{R}$-MA, pouvaient causer des changements de signe engendrant des erreurs relatives arbitrairement grandes, en particulier dans les probabilités condition-

nelles. Ainsi, nous avons proposé l'algorithme NNSpectral, contraint à apprendre des automates à paramètres positifs, les $\mathbb{R}^+$-MA. Étant basé sur une factorisation en matrices non-négatives, reconnue **NP**-dur, NNSpectral utilise des algorithmes de NMF convergeant vers des minimaux locaux. Ainsi NNSpectral est dépourvu de garanties non-asymptotiques. Cependant, expérimentalement, NNSpectral est bien moins sujet à converger vers des minimaux locaux mauvais que l'algorithme de BW. Ainsi, les résultats de NNSpectral sont bien meilleurs que les algorithmes basés sur la MoM et l'algorithme de BW. Malheureusement, le modèles appris ne réalise pas une distribution. Ainsi, seule une normalisation locale (sur un ensemble fini d'événements) est possible. C'est pourquoi aux Chapitre 6, nous avons proposé une modification de l'algorithme NNSpectral, appelé CH-PNFA, effectuant une normalisation globale pour apprendre des PNFA. Malheureusement, cette normalisation globale semble apporter des instabilités lors de l'apprentissage à partir d'un nombre fini d'exemples. Nous observons donc des performances inférieures à NNSpectral, bien que meilleures que les autres algorithmes basés sur la MoM. Nous avons aussi essayé d'utiliser le modèle retourné pour initialiser l'algorithme de BW. Les résultats mitigés montrent que l'absence de garanties non-asymptotiques ne permet pas d'obtenir une bonne initialisation, tout comme pour l'algorithme Tenseur. La quête d'un algorithme possédant des garanties PAC et retournant un automate réalisant une distribution de probabilité nous amène au Chapitre 7, où l'on a identifié une classe de modèle moins riches que les PNFA, appelés les PRFA. Afin d'apprendre des PRFA, nous avons proposé d'identifier l'enveloppe conique des lignes de la matrice de Hankel. Or, l'existence d'algorithmes récursifs, efficaces et robustes pour cette tâche, nous permet de proposer un algorithme efficace et consistant, CH-PRFA, ainsi qu'une analyse non-asymptotique de celui-ci donnant des garanties PAC. Malheureusement, l'identification des arêtes dans une matrice bruitée dépend du bruit relatif sur les lignes de la matrice de Hankel. Ainsi, la borne sur l'erreur ainsi que les performances empiriques dépendent de la plus faible probabilité des préfixes utilisés dans la base, ce qui limite la taille de celle-ci dans les expériences. Pour palier à ce problème, nous proposons plusieurs méthodes dont une permet d'obtenir des résultats compétitifs quelle que soit la taille de la base.

Résultats

En conclusion, nous récapitulons de façon succincte les performances de tous les algorithmes décrit dans cette thèse.

PAutomaC

La Tableau 8.1 classe les performances des algorithmes sur les douze problèmes de PAutomaC en fonction de deux critères. NNSpectral domine les autres algorithmes pour les deux critères. Sur la Section 7.6, nous avons comparé les temps d'exécutions. Comparé à la Section 5.5.1 au Chapitre 5, nous avons pris la moyenne du temps d'apprentissage à partir des séquences et des sous-séquences lorsque l'algorithme permet l'utilisation d'une série auxiliaire. Il apparaît que malgré ses excellentes performances NNSpectral soit assez lent. Finalement, CH-PRFA semble être un bon compromis entre performance et passage à l'échelle.

	Algorithme	WER moyen
1.	NNSpectral	64.618
2.	CH-PRFA+BW	65.160
3.	Spectral	65.529
4.	BW	66.482
5.	CH-PRFA	67.141
6.	CH-PNFA+BW	70.169
7.	CO	71.467
8.	CH-PNFA	72.789
9.	Tenseur+BW	73.544
10.	Tenseur	77.433

(a) WER moyens.

	Algorithme	Perplexité moyenne
1.	NNSpectral	30.364
2.	CH-PRFA+BW	30.468
3.	CH-PRFA	30.603
4.	CH-PNFA+BW	34.705
5.	CO	35.641
6.	BW	35.886
7.	Spectral	40.210
8.	CH-PNFA	41.507
9.	Tenseur+BW	47.655
10.	Tenseur	54.000

(b) Perplexité moyennes

Tableau 8.1 – Classement global des algorithmes sur PAutomaC.

Penn-Treebank

La Tableau 8.2 montre que l'utilisation des statistiques sur les sous-séquences permet aux algorithmes, issus de la MoM et qui peuvent travailler à partir de séries auxiliaires, d'obtenir de meilleures performances. Sur ce problème les algorithmes basés sur BW donnent les meilleurs résultats, suivis de Spectral et NNSpectral qui ont des performances comparables.

	Algorithme	WER
1.	CH-PNFA+BW	59.045
2.	CH-PRFA+BW	59.206
3.	BW	60.774
4.	Spectral	61.133
5.	NNSpectral	61.413
6.	Tenseur+BW	64.428
7.	CO	66.070
8.	CH-PRFA	66.125
9.	CH-PNFA	66.581
10.	Tenseur	70.790

Tableau 8.2 – Classement global des algorithmes sur Penn-Treebank.

Wikipédia

Pour Wikipédia, les performances de CO et Tenseur n'ont pas été reportées sur les Figures 8.2 et 8.3 car trop mauvaises. Les algorithmes issus de la MoM ont été entraînés sur le corpus de taille moyenne. Ceux basés sur BW, n'utilisent que le petit corpus car ces algorithmes ne passent pas à l'échelle. L'algorithme Spectral domine le classement pour la vraisemblance conditionnelle alors qu'il se retrouve dernier quand nous regardons le BPC. Quand les dimensions du problème ne sont pas trop importantes les approches utilisant l'algorithme BW obtiennent les meilleurs résultats si nous combinons les deux critères. Parmi les algorithmes issus de la MoM, CH-PRFA offre de bonnes performances à la fois pour la vraisemblance conditionnelle et pour le BPC, en plus d'être rapide.

8.6.1 Quel algorithme pour quel problème avec quels paramètres ?

Lorsque de vraies probabilités ne sont pas nécessaires, par exemple si l'on s'intéresse uniquement au maximum de vraisemblance, l'algorithme Spectral est à la fois

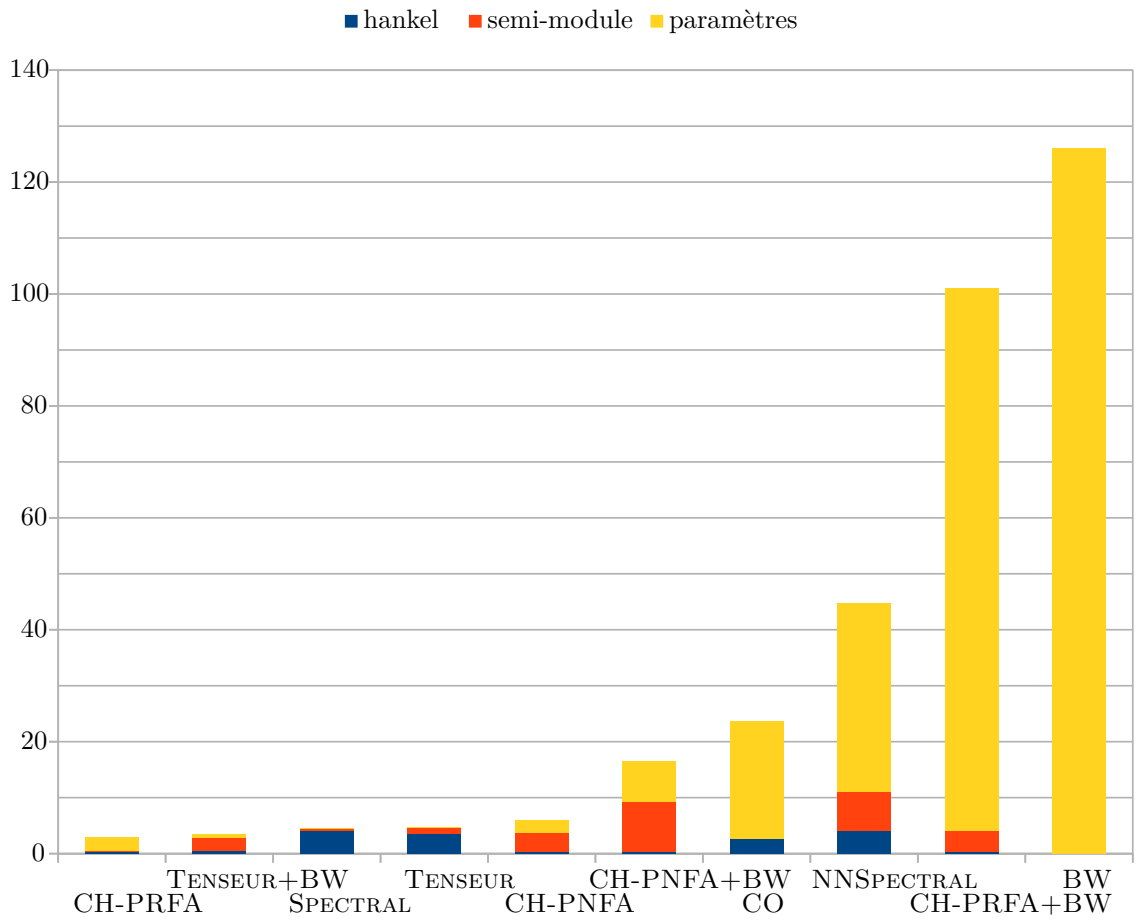


FIGURE 8.1 – Temps d'exécution moyen par algorithme sur douze problèmes de PAutomaC.

performant et rapide. Sa version régularisée, proposée au Chapitre 3, permet d'obtenir une plus grande robustesse et d'adapter la taille du modèle aux données. Cet avantage est intéressant lors de l'apprentissage en ligne.

Sur des problèmes de taille modeste, comme ceux de PAutomaC, l'algorithme NNSPECTRAL affiche les meilleures performances. Même si SPECTRAL et NNSPECTRAL estiment tout les deux très bien la probabilité des événements fréquents, NNSPECTRAL est meilleur pour estimer les événements faiblement probables. En somme, NNSPECTRAL est plus performant. Son principal inconvénient est son coût calculatoire lié à la résolution d'un problème de NMF par des heuristiques. Ce problème limite le nombre de préfixes et de suffixes dans la base. Par exemple, pour apprendre Wikipédia, il n'est pas capable de capturer correctement la dynamique du langage naturel car la base utilisée est trop petite. Limiter la taille de la base, limite le nombre d'observations et la longueur des suffixes et des préfixes. Toujours sur des problèmes de taille modeste, CH-PNFA est une variante de NNSPECTRAL permettant d'obtenir un automate réalisant un langage stochastique. Non seulement, c'est essentiel pour certaines applications, mais cela permet aussi d'initialiser d'autres algorithmes comme BW.

Sur des problèmes de grandes tailles, comme Wikipédia, l'algorithme CH-PRFA est efficace et performant. Malheureusement, l'expressivité limité des PRFA ne permet pas d'apprendre des modèles aussi complexe que le langage naturel. Ainsi, si le score mesuré dépend peu de la précision avec laquelle les probabilités faibles sont estimées (comme pour le WER), l'algorithme SPECTRAL est le plus performant. Dès lors, que

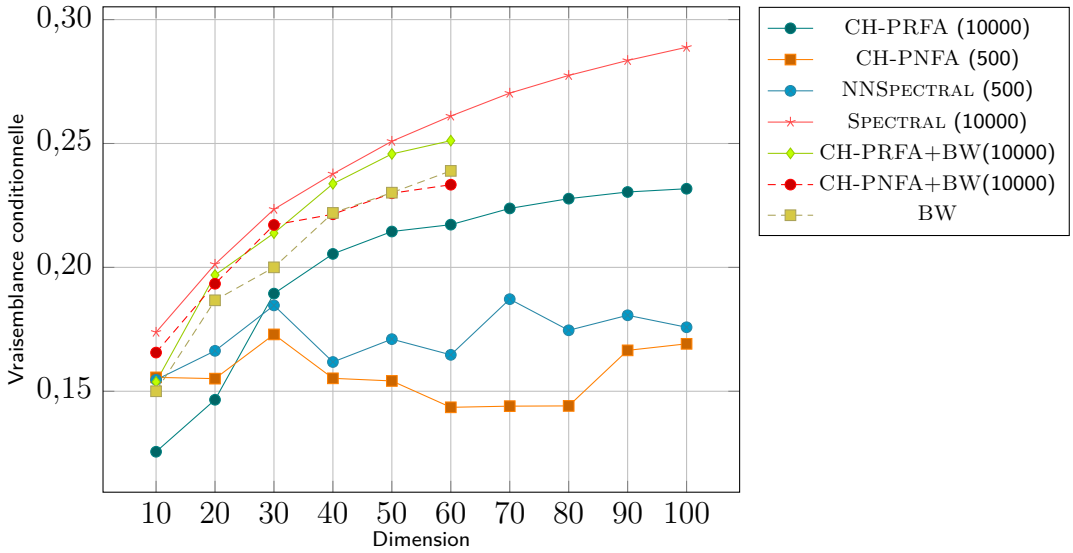


FIGURE 8.2 – Comparaison en termes de vraisemblance conditionnelle des algorithmes de NMF pour CH-PRFA en fonction de la dimension et de la taille de la base (indiquée entre parenthèses). 41943 séquences de Wikipédia ont été utilisées pour l'apprentissage, sauf pour les algorithmes utilisant BW, où seulement 409 séquences ont été utilisées.

l'on s'intéresse autant aux événements rares qu'aux fréquents, l'algorithme CH-PRFA affiche des performances bien meilleures. L'expressivité limitée des PRFA se constate aussi dans la différence entre les performances de CH-PRFA et de BW que ce soit pour le WER ou pour le BPC. Néanmoins, de nombreux problèmes non résolus peuvent être modélisés par des PRFA. Nous donnons un exemple dans la section suivante.

En règle générale, l'avantage des algorithmes basés sur la MoM réside aussi dans le faible nombre de paramètres à régler. Cependant, le choix de la base reste souvent délicat. Les résultats issus de concentration des sommes de matrices aléatoires, nous montrent que plus la base est grande, meilleures seront les performances. Le seul désavantage est la complexité calculatoire associées. Une heuristique qui marche bien en pratique est de prendre les sous-séquences les plus fréquentes ou bien les préfixes et les suffixes les plus fréquents.

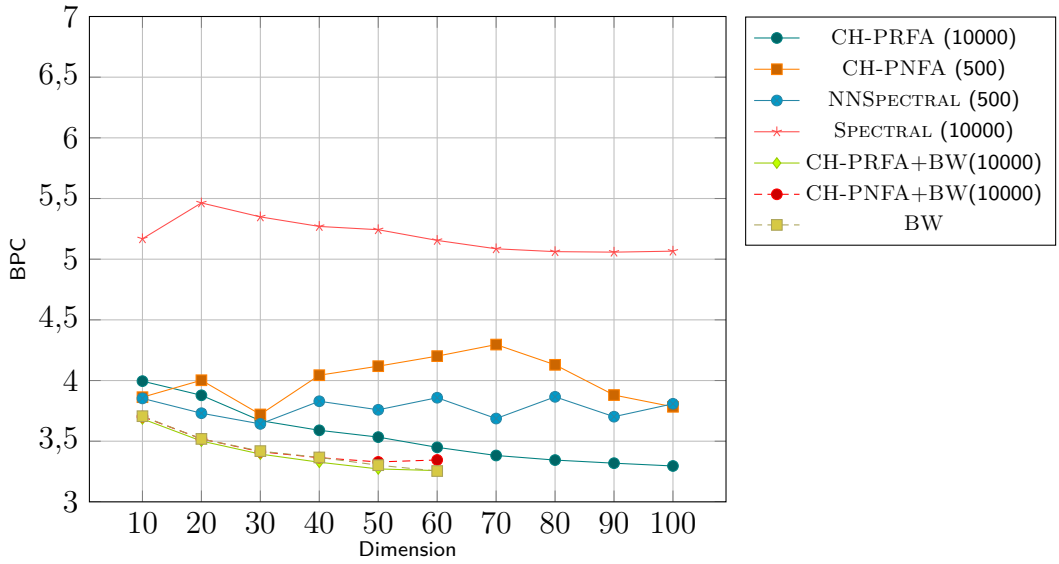


Figure 8.3 – Comparaison en termes de BPC des algorithmes de NMF pour CH-PRFA en fonction de la dimension et de la taille de la base (indiquée entre parenthèses). 41943 séquences de Wikipédia ont été utilisées pour l'apprentissage, sauf pour les algorithmes utilisant BW, où seulement 409 séquences ont été utilisées.

Travaux futurs

Les travaux dans cette thèse offrent de nombreux développements possibles, aussi bien applicatifs que théoriques.

À court terme, la prochaine étape consisterait à trouver un nouvel algorithme basé sur la NMF séparable et à dériver son analyse non-asymptotique mais ne faisant pas apparaître la plus faible probabilité des préfixes dans la base. Plusieurs pistes sont possibles. Par exemple, nous pourrions changer l'algorithme d'identification de l'enveloppe convexe pour prendre en compte différents modèles de bruits. D'autre part, changer l'algorithme de NMF (SPA) dans CH-PRFA permettrait d'alléger les hypothèses faites. Par exemple, avec l'algorithme SNPA, l'hypothèse de rang plein ne serait plus requise. L'analyse des perturbations dans la solution du problème de programmation quadratique serait alors bien plus complexe mais toujours envisageable. La borne finale ferait intervenir d'autres termes pour remplacer la plus petite valeur singulière. Le paramètre simplicial correspondant pour les PDFA à la distinguabilité semble être un bon candidat. D'ailleurs, il serait intéressant de montrer que ce paramètre est nécessaire et suffisant.

Toujours dans l'idée d'améliorer l'identification de l'enveloppe convexe, certaines contributions récentes suggèrent de possible développements. La récente contribution de Ge et Zou [2015], étend le concept de séparabilité et élargit l'ensemble des matrices pour lesquelles nous pouvons retrouver une enveloppe conique efficacement. D'une part, cela pourrait permettre d'élargir la classe des langages stochastiques PAC-apprenables. D'autre part, cela permettrait de relâcher l'hypothèse de séparabilité sur la base, c'est-à-dire la présence dans la base des préfixes associés aux arêtes du cône généré par la matrice de Hankel. En effet, si des préfixes permettent d'identifier les faces du cône, l'algorithme de Ge et Zou [2015] permet de retrouver les arêtes du cône en intersectant les faces.

À moyen terme, il serait intéressant d'élargir les garanties de CH-PRFA aux modèles compressés. Là encore, le paramètre simplicial pourrait jouer un rôle important.

Du côté applicatif, plusieurs développements sont envisageables. Le premier serait d'étendre les algorithmes des Chapitres 5 à 7 aux observations continues. Une possibilité serait d'utiliser des noyaux, comme dans [Boots et collab., 2013]. Une autre possibilité serait d'utiliser la récente contribution de Castro et collab. [2015]. Deuxièmement, il serait intéressant d'explorer si CH-PRFA peut apprendre des modèles de très grande taille. Nous avons vu que l'algorithme SPA était facilement distribuable. Ensuite, il faut résoudre

— d problèmes de programmation quadratique avec 1 contrainte linéaire à $d\,|\Sigma|$ variables contraintes et

— 1 problème avec 1 contrainte linéaire à d variables contraintes.

Des méthodes distribuées pourraient être développées en utilisant, par exemple, l'algorithme des directions alternées qui une modification de l'algorithme de lagrangien augmenté.

Toujours à moyen terme, les algorithmes présentés dans cette thèse, en particulier CH-PRFA, peuvent être utilisés pour apprendre des processus contrôlés. Il convient d'étudier comment, couplés à des algorithmes de planification, ils peuvent former des solutions complètes en apprentissage par renforcement dans les environnements partiellement observables. L'objectif serait d'obtenir un algorithme consistant avec des garanties non-asymptotiques dans les environnements partiellement observable. CH-PRFA est un bon candidat pour y arriver car il évite le problème des probabilités négatives qui était jusqu'à présent un obstacle à l'obtention de garanties non-asymptotiques. De plus, pour de nombreux problèmes en apprentissage par renforcement, l'expressivité des CH-PRFA semble suffisante. Prenons par exemple le cas des systèmes de dialogues. Oublions pour le moment l'incertitude liée à la reconnaissance de la parole et supposons que la compréhension du langage naturel est parfaite. Le choix de la prochaine action (question, confirmation,...) ne dépend pas que de la dernière observation (la phrase prononcée par l'utilisation) mais de toutes les informations collectées depuis le début du dialogue. En d'autre terme, l'environnement n'est pas Markovien. Le problème ne peut pas être modélisé par un MDP. Cependant, en supposant que les dialogues sont de taille finie et pas trop grande, un CP-DFA conviendrait pour modéliser le problème. Cependant, celui-ci n'est pas capable de prendre en compte les incertitudes liées au traitement de la parole et à la compréhension du langage. Ici, un CP-RFA semble parfaitement adapté.

Enfin, à long terme, il serait intéressant de continuer à explorer l'apprentissage par renforcement, en particulier les problématiques d'apprentissage en ligne. Un algorithme d'inférence consistant ouvre des portes pour établir des stratégies d'apprentissage en ligne réalisant un compromis entre l'exploration et l'exploitation et bénéficiant de bornes sur le regret. Une autre piste à explorer consiste à appliquer la MoM basée sur l'identification d'une enveloppe conique à d'autres modèles comme les modèles à mélanges ou les grammaires non contextuelles probabilistes à variables latentes.

Bibliographie

Abe, N. et M. K. Warmuth. 1990, «On the computational complexity of approximating distributions by probabilistic automata», dans *Proceedings of the Third Annual Workshop on Computational Learning Theory, COLT 1990, University of Rochester, Rochester, NY, USA, August 6-8, 1990.*, p. 52–66.

Anandkumar, A., R. Ge, D. Hsu, S. M. Kakade et M. Telgarsky. 2012a, «Tensor decompositions for learning latent variable models», *CoRR*, vol. abs/1210.7559.

Anandkumar, A., R. Ge, D. Hsu, S. M. Kakade et M. Telgarsky. 2014, «Tensor decompositions for learning latent variable models», *Journal of Machine Learning Research*, vol. 15, nº 1, p. 2773–2832.

Anandkumar, A., D. Hsu et S. M. Kakade. 2012b, «A method of moments for mixture models and hidden Markov models», dans *Proceedings of the 25th Annual Conference on Learning Theory, COLT 2012, 25-27 June 2012, Edinburgh, Scotland*, p. 33.1–33.34.

Anandkumar, A., N. Michael, A. K. Tang et A. Swami. 2011, «Distributed algorithms for learning and cognitive medium access with logarithmic regret», *IEEE Journal on Selected Areas in Communications*, vol. 29, nº 4, p. 731–745.

Angluin, D. 1987, «Queries and concept learning», *Machine Learning*, vol. 2, nº 4, p. 319–342.

Angluin, D. 1988, «Identifying languages from stochastic examples», .

Araújo, M. C. U., T. C. B. Saldanha, R. K. H. Galvão, T. Yoneyama, H. C. Chame et V. Visani. 2001, «The successive projections algorithm for variable selection in spectroscopic multicomponent analysis», *Chemometrics and Intelligent Laboratory Systems*, vol. 57, nº 2, p. 65–73.

Arora, S., R. Ge, Y. Halpern, D. M. Mimno, A. Moitra, D. Sontag, Y. Wu et M. Zhu. 2013, «A practical algorithm for topic modeling with provable guarantees», dans *Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013*, p. 280–288.

Arora, S., R. Ge, R. Kannan et A. Moitra. 2012a, «Computing a nonnegative matrix factorization - provably», dans *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012, New York, NY, USA, May 19 - 22, 2012*, p. 145–162.

Arora, S., R. Ge et A. Moitra. 2012b, «Learning topic models – going beyond SVD», dans *Foundations of Computer Science (FOCS), 2012 IEEE 53rd Annual Symposium on*, ISSN 0272-5428, p. 1–10.

Auer, P., N. Cesa-Bianchi, Y. Freund et R. E. Schapire. 2002, «The nonstochastic multiarmed bandit problem», *SIAM J. Comput.*, vol. 32, n° 1, p. 48–77.

Bailly, R. 2011a, *Méthodes spectrales pour l'inférence grammaticale probabiliste de langages stochastiques rationnels*, thèse de doctorat, Université Aix Marseille 1.

Bailly, R. 2011b, «Quadratic weighted automata : Spectral algorithm and likelihood maximization», *Journal of Machine Learning Research*, vol. 20, p. 147–162.

Bailly, R., F. Denis et L. Ralaivola. 2009, «Grammatical inference as a principal component analysis problem», dans *Proceedings of the 26th Annual International Conference on Machine Learning, ICML 2009, Montreal, Quebec, Canada, 14-18 June 2009*, p. 33–40.

Balle, B. 2013, *Learning finite-state machines : statistical and algorithmic aspects*, thèse de doctorat, Universitat Politècnica de Catalunya.

Balle, B., X. Carreras, F. M. Luque et A. Quattoni. 2014a, «Spectral learning of weighted automata - A forward-backward perspective», *Machine Learning*, vol. 96, n° 1-2, p. 33–63.

Balle, B., J. Castro et R. Gavaldà. 2012a, «Bootstrapping and learning PDFA in data streams», dans *Proceedings of the Eleventh International Conference on Grammatical Inference, ICGI 2012, University of Maryland, College Park, USA, September 5-8, 2012*, p. 34–48.

Balle, B., J. Castro et R. Gavaldà. 2013, «Learning probabilistic automata : A study in state distinguishability», *Theor. Comput. Sci.*, vol. 473, p. 46–60.

Balle, B., W. L. Hamilton et J. Pineau. 2014b, «Methods of moments for learning stochastic languages : Unified presentation and empirical comparison», dans *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, p. 1386–1394.

Balle, B. et M. Mohri. 2012, «Spectral learning of general weighted automata via constrained matrix completion», dans *Advances in Neural Information Processing Systems 25 : 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States.*, p. 2168–2176.

Balle, B. et M. Mohri. 2015a, «Learning weighted automata», dans *Algebraic Informatics - 6th International Conference, CAI 2015, Stuttgart, Germany, September 1-4, 2015. Proceedings*, p. 1–21.

Balle, B. et M. Mohri. 2015b, «On the rademacher complexity of weighted automata», dans *Algorithmic Learning Theory - 26th International Conference, ALT 2015, Banff, AB, Canada, October 4-6, 2015, Proceedings*, p. 179–193.

Balle, B., P. Panangaden et D. Precup. 2015, «A canonical form for weighted automata and applications to approximate minimization», dans *30th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2015, Kyoto, Japan, July 6-10, 2015*, p. 701–712.

Balle, B., A. Quattoni et X. Carreras. 2011, «A spectral learning algorithm for finite state transducers», dans *Machine Learning and Knowledge Discovery in Databases*, Springer, p. 156–171.

Balle, B., A. Quattoni et X. Carreras. 2012b, «Local loss optimization in operator models : A new insight into spectral learning», dans *Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012*.

Beal, M. J., Z. Ghahramani et C. E. Rasmussen. 2001, «The infinite hidden Markov model», dans *Advances in Neural Information Processing Systems 14 [Neural Information Processing Systems : Natural and Synthetic, NIPS 2001, December 3-8, 2001, Vancouver, British Columbia, Canada]*, p. 577–584.

Beimel, A., F. Bergadano, N. H. Bshouty, E. Kushilevitz et S. Varricchio. 2000, «Learning functions represented as multiplicity automata», *J. ACM*, vol. 47, nº 3, p. 506–530.

Bergadano, F. et S. Varricchio. 1994, «Learning behaviors of automata from multiplicity and equivalence queries», dans *Algorithms and Complexity, Second Italian Conference, CIAC '94, Rome, Italy, February 23-25, 1994, Proceedings*, p. 54–62.

Blunsom, P. et T. Cohn. 2011, «A hierarchical pitman-yor process HMM for unsupervised part of speech induction», dans *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics : Human Language Technologies-Volume 1*, Association for Computational Linguistics, p. 865–874.

Boardman, J. W. et collab.. 1993, «Automating spectral unmixing of aviris data using convex geometry concepts», dans *Summaries 4th Annu. JPL Airborne Geoscience Workshop*, vol. 1, JPL Publication 93–26, p. 11–14.

Boots, B. et G. J. Gordon. 2011, «An online spectral learning algorithm for partially observable nonlinear dynamical systems», dans *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2011, San Francisco, California, USA, August 7-11, 2011*.

Boots, B., G. J. Gordon et A. Gretton. 2013, «Hilbert space embeddings of predictive state representations», dans *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence, UAI 2013, Bellevue, WA, USA, August 11-15, 2013*.

Boots, B., S. M. Siddiqi et G. J. Gordon. 2010a, «Closing the learning-planning loop with predictive state representations», dans *9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2010), Toronto, Canada, May 10-14, 2010, Volume 1-3*, p. 1369–1370.

Boots, B., S. M. Siddiqi et G. J. Gordon. 2010b, «Closing the learning-planning loop with predictive state representations», dans *9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2010), Toronto, Canada, May 10-14, 2010, Volume 1-3*, p. 1369–1370.

Boutsidis, C. et E. Gallopoulos. 2008, «SVD based initialization : A head start for nonnegative matrix factorization», *Pattern Recognition*, vol. 41, nº 4, p. 1350–1362.

Bowling, M. H., P. McCracken, M. James, J. Neufeld et D. F. Wilkinson. 2006, «Learning predictive state representations using non-blind policies», dans *Machine Learning, Proceedings of the Twenty-Third International Conference (ICML 2006), Pittsburgh, Pennsylvania, USA, June 25-29, 2006*, p. 129–136.

Cai, J.-F., E. J. Candès et Z. Shen. 2010, «A singular value thresholding algorithm for matrix completion», *SIAM Journal on Optimization*, vol. 20, n° 4, p. 1956–1982.

Carrasco, R. C. et J. Oncina. 1994, «Learning stochastic regular grammars by means of a state merging method», dans *Grammatical Inference and Applications, Second International Colloquium, ICGI-94, Alicante, Spain, September 21-23, 1994, Proceedings*, p. 139–152.

Carrasco, R. C. et J. Oncina. 1999, «Learning deterministic regular grammars from stochastic samples in polynomial time», *ITA*, vol. 33, n° 1, p. 1–20.

Castro, J. et R. Gavaldà. 2008, «Towards feasible pac-learning of probabilistic deterministic finite automata», dans *Grammatical Inference : Algorithms and Applications, 9th International Colloquium, ICGI 2008, Saint-Malo, France, September 22-24, 2008, Proceedings*, p. 163–174.

Castro, Y. D., E. Gassiat et S. L. Corff. 2015, «Consistent estimation of the filtering and marginal smoothing distributions in nonparametric hidden markov models», .

Chan, T.-H., W.-K. Ma, A. Ambikapathi et C.-Y. Chi. 2011, «A simplex volume maximization framework for hyperspectral endmember extraction», *IEEE Transactions on Geoscience and Remote Sensing*, vol. 49, n° 11, p. 4177–4193.

Chan, T.-H., W.-K. Ma, C.-Y. Chi et Y. Wang. 2008, «A convex analysis framework for blind separation of non-negative sources», *IEEE Transactions on Signal Processing*, vol. 56, n° 10, p. 5120–5134, ISSN 1053-587X.

Çivril, A. et M. Magdon-Ismail. 2009, «On selecting a maximum volume sub-matrix of a matrix and related problems», *Theor. Comput. Sci.*, vol. 410, n° 47-49, p. 4801–4811.

Clark, A. et F. Thollard. 2004, «Pac-learnability of probabilistic deterministic finite state automata», *Journal of Machine Learning Research*, vol. 5, p. 473–497.

Clarkson, I. 2005, «Optimal periodic sensor scheduling in Electronic Support», *Proc. Defence Appl. Signal Process.*

Clarkson, I., J. Perkins et I. Mareels. 1996, «Number/theoretic solutions to intercept time problems», *IEEE Transactions on Information Theory*, vol. 42, n° 3, p. 959 –971, ISSN 0018-9448.

Clarkson, I. et A. Pollington. 2007, «Performance limits of sensor-scheduling strategies in electronic support», *IEEE Transactions on Aerospace and Electronic Systems*, vol. 43, n° 2, p. 645–650, ISSN 0018-9251.

Clarkson, I. V. L. 2003, «The arithmetic of receiver scheduling for Electronic Support», dans *Aerospace Conference, 2003. Proceedings. 2003 IEEE*, vol. 5, ISSN 1095-323X, p. 2049–2064.

Cohen, S. B. et M. Collins. 2014, «A provably correct learning algorithm for latent-variable PCfgs», dans *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 1 : Long Papers*, p. 1052–1061.

Cohen, S. B., K. Stratos, M. Collins, D. P. Foster et L. H. Ungar. 2013, «Experiments with spectral learning of latent-variable PCfgs», dans *Human Language Technologies : Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 9-14, 2013, Westin Peachtree Plaza Hotel, Atlanta, Georgia, USA*, p. 148–157.

Colombo, N. et N. Vlassis. 2015, «Stable spectral learning based on schur decomposition», dans *Proceedings of the 31st Conference on Uncertainty in Artificial Intelligence, UAI 2015, Amsterdam, NL, July 12-16, 2015*.

Csiszár, I. et G. Tusnády. 1984, «Information geometry and alternating minimization procedures», *Statistics & Decisions, Supplement Issue*, , nº 1, p. 205–237.

Cybenko, G. et V. Crespi. 2011, «Learning hidden Markov models using nonnegative matrix factorization», *IEEE Transactions on Information Theory*, vol. 57, nº 6, p. 3963–3970, ISSN 0018-9448.

Dempster, A. P., N. M. Laird et D. B. Rubin. 1977, «Maximum likelihood from incomplete data via the em algorithm», *Journal of the royal statistical society. Series B (methodological)*, p. 1–38.

Denis, F. et Y. Esposito. 2004a, «Identification in the limit of probabilistic non deterministic automata and undecidable problem for multiplicity automata», dans *Proceedings of The 17th Conference on Learning Theory, COLT 2004, Banff, Canada, 1-4 July 2004*, vol. 2004.

Denis, F. et Y. Esposito. 2004b, «Learning classes of probabilistic automata», dans *Learning Theory*, Springer, p. 124–139.

Denis, F. et Y. Esposito. 2008, «On rational stochastic languages», *Fundam. Inform.*, vol. 86, nº 1-2, p. 41–77.

Denis, F., Y. Esposito et A. Habrard. 2006, «Learning rational stochastic languages», dans *Learning Theory*, Springer, p. 274–288.

Denis, F., M. Gybels et A. Habrard. 2014, «Dimension-free concentration bounds on Hankel matrices for spectral learning», dans *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, p. 449–457.

Denis, F., A. Lemay et A. Terlutte. 2002, «Residual finite state automata», *Fundam. Inform.*, vol. 51, nº 4, p. 339–368.

Ding, W., P. Ishwar, M. H. Rohban et V. Saligrama. 2013a, «Necessary and sufficient conditions for novel word detection in separable topic models», *CoRR*, vol. abs/1310.7994.

Ding, W., P. Ishwar et V. Saligrama. 2015a, «Necessary and sufficient conditions and a provably efficient algorithm for separable topic discovery», *CoRR*, vol. abs/1508.05565.

Ding, W., P. Ishwar et V. Saligrama. 2015b, «A topic modeling approach to ranking», dans *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2015, San Diego, California, USA, May 9-12, 2015*.

Ding, W., M. H. Rohban, P. Ishwar et V. Saligrama. 2013b, «Topic discovery through data dependent and random projections», dans *Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013*, p. 1202–1210.

Ding, W., M. H. Rohban, P. Ishwar et V. Saligrama. 2014, «Efficient distributed topic modeling with provable guarantees», dans *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics, AISTATS 2014, Reykjavik, Iceland, April 22-25, 2014*, p. 167–175.

Donoho, D. L. et V. Stodden. 2003, «When does non-negative matrix factorization give a correct decomposition into parts ?», dans *Advances in Neural Information Processing Systems 16 [Neural Information Processing Systems, NIPS 2003, December 8-13, 2003, Vancouver and Whistler, British Columbia, Canada]*, p. 1141–1148.

Doshi-Velez, F., D. Pfau, F. Wood et N. Roy. 2015, «Bayesian nonparametric methods for partially-observable reinforcement learning», *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 37, nº 2, p. 394–407.

Dupont, P., F. Denis et Y. Esposito. 2005, «Links between probabilistic automata and hidden Markov models : probability distributions, learning models and induction algorithms», *Pattern Recognition*, vol. 38, nº 9, p. 1349–1371.

Dutertre, B. 2002, «Dynamic scan scheduling», dans *Proceedings of the 23rd IEEE Real-Time Systems Symposium (RTSS'02), Austin, Texas, USA, 3-5 December 2002*, p. 327–336.

El-Mahassni, E. D., S. D. Howard et I. V. L. Clarkson. 2004, «A Markov-chain model for sensor scheduling in electronic support», dans *Signals, Systems and Computers, 2004. Conference Record of the Thirty-Eighth Asilomar Conference on*, vol. 2, p. 2183–2187 Vol.2.

Ephraim, Y. et N. Merhav. 2002, «Hidden Markov processes», *IEEE Transactions on Information Theory*, vol. 48, nº 6, p. 1518–1569, ISSN 0018-9448.

Esposito, Y. 2004, *Contribution à l'inférence d'automates probabilistes*, thèse de doctorat, Université Aix-Marseille.

Esposito, Y., A. Lemay, F. Denis et P. Dupont. 2002, «Learning probabilistic residual finite state automata», dans *Grammatical Inference : Algorithms and Applications, 6th International Colloquium : ICGI 2002, Amsterdam, The Netherlands, September 23-25, 2002, Proceedings*, p. 77–91.

Finesso, L., A. Grassi et P. Spreij. 2010, «Approximation of stationary processes by hidden Markov models», *MCSS*, vol. 22, nº 1, doi: 10.1007/s00498-010-0050-7, p. 1–22.

Fliess, M. 1974, «Matrices de hankel», *J. Math. Pures Appl*, vol. 53, n° 9, p. 197–222.

Foster, D. P., J. Rodu et L. H. Ungar. 2012, «Spectral dimensionality reduction for HMMs», *CoRR*, vol. abs/1203.6130.

Friedman, H. D. 1954, «Coincidence of pulse trains», *Journal of Applied Physics*, vol. 25, n° 8, p. 1001–1005.

Gavaldà, R., P. W. Keller, J. Pineau et D. Precup. 2006, «Pac-learning of Markov models with hidden state», dans *Machine Learning : ECML 2006, 17th European Conference on Machine Learning, Berlin, Germany, September 18-22, 2006, Proceedings*, p. 150–161.

Ge, R. et J. Zou. 2015, «Intersecting faces : Non-negative matrix factorization with new guarantees», dans *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, p. 2295–2303.

Gillis, N. 2013, «Robustness analysis of hottopixx, a linear programming model for factoring nonnegative matrices», *SIAM J. Matrix Analysis Applications*, vol. 34, n° 3, p. 1189–1212.

Gillis, N. 2014a, «Successive nonnegative projection algorithm for robust nonnegative blind source separation», *SIAM J. Imaging Sciences*, vol. 7, n° 2, p. 1420–1450.

Gillis, N. 2014b, «The why and how of nonnegative matrix factorization», *CoRR*, vol. abs/1401.5226.

Gillis, N. et R. Luce. 2014, «Robust near-separable nonnegative matrix factorization using linear optimization», *Journal of Machine Learning Research*, vol. 15, n° 1, p. 1249–1280.

Gillis, N. et S. A. Vavasis. 2014, «Fast and robust recursive algorithmsfor separable nonnegative matrix factorization», *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, n° 4, p. 698–714, ISSN 0162-8828.

Gittens, A. et J. A. Tropp. 2011, «Tail bounds for all eigenvalues of a sum of random matrices», *CoRR*, vol. abs/1104.4513.

Glaude, H. et H. Boucard. 2015, «Method and system for determining a reception configuration and a duration of a time interval», WO Patent App. PCT/EP2014/069,804.

Glaude, H., C. Enderli, J. Grandin et O. Pietquin. 2015a, «Learning of scanning strategies for electronic support using predictive state representations», dans *25th IEEE International Workshop on Machine Learning for Signal Processing, MLSP 2015, Boston, MA, USA, September 17-20, 2015*, p. 1–6.

Glaude, H., C. Enderli et O. Pietquin. 2015b, «Apprentissage spectral non négatif de systèmes séquentiels linéaires», dans *Actes de CAP*.

Glaude, H., C. Enderli et O. Pietquin. 2015c, «Non-negative spectral learning for linear sequential systems», dans *Neural Information Processing - 22nd International Conference, ICONIP 2015, Istanbul, Turkey, November 9-12, 2015, Proceedings, Part II*, p. 143–151.

Glaude, H., C. Enderli et O. Pietquin. 2015d, «Spectral learning with proper probabilities for finite state automaton», dans *Automatic Speech Recognition and Understanding Workshop (ASRU 2015)*, IEEE.

Glaude, H. et O. Pietquin. 2016, «Pac learning of probabilistic automaton based on the method of moments», dans *Proceedings of the 33rd International Conference on Machine Learning, ICML 2016, New-York, NY, USA, 19-24 June 2016*, p. 820–829.

Glaude, H., O. Pietquin et C. Enderli. 2014, «Subspace identification for predictive state representation by nuclear norm minimization», dans *2014 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning, ADPRL 2014, Orlando, FL, USA, December 9-12, 2014*, p. 1–8.

Gold, E. M. 1967, «Language identification in the limit», *Information and Control*, vol. 10, n° 5, p. 447–474.

Gopalakrishnan, S. 2012, «Sharp utilization thresholds for some realtime scheduling problems», *SIGMETRICS Performance Evaluation Review*, vol. 39, n° 4, p. 12–22.

Gopalakrishnan, S., M. Caccamo et L. Sha. 2008, «Sharp thresholds for scheduling recurring tasks with distance constraints», *IEEE Transactions on Computers*, vol. 57, n° 3, p. 344–358, ISSN 0018-9340.

Gounalis, A. J. 2005, «System and method for non-maximum dwell duration selection for use in detecting emitter signals», G01S007/41 ; 342/13 ; 342/195 ; 342/88 ; 342/89 ; 342/90.

Gounalis, A. J. 2006a, «System and method for detecting and jamming emitter signals», G01S 7/38 ; 342 13 ; 342 14 ; 342 15 ; 342 89 ; 342 90 ; 342 98 ; 342 99 ; 342195.

Gounalis, A. J. 2006b, «System and method for receiver resource allocation and verification», G01S 13/00 ; G01S 7/40 ; 342 13 ; 342 20 ; 342 89 ; 342165 ; 342173 ; 342175 ; 342195.

Gounalis, A. J. 2007, «System and method for detection of emitter signals using multiple intercept rules», G01S 7/40 ; 342 13 ; 342 89 ; 342 90 ; 342 98 ; 342 99 ; 342195.

Gounalis, A. J. 2008, «System and method for detecting emitters signals having multi-valued illumination times», G01S 13/00 ; G01S 7/02 ; G01S 7/40 ; 342 13 ; 342 20 ; 342 89 ; 342175 ; 342195.

Gounalis, A. J. 2010, «Determining scan strategy for digital card», G01S 13/00 ; G01S 7/285 ; G01S 7/35 ; G01S 7/40 ; 342 13 ; 342 20 ; 342 89 ; 342175 ; 342195.

Guttman, O. 2006, *Probabilistic automata and distributions over sequences*, thèse de doctorat, Australian National University.

Guttman, O., S. V. N. Vishwanathan et R. C. Williamson. 2005, «Learnability of probabilistic automata via oracles», dans *Algorithmic Learning Theory, 16th International Conference, ALT 2005, Singapore, October 8-11, 2005, Proceedings*, p. 171–182.

Gybels, M., F. Denis et A. Habrard. 2014, «Some improvements of the spectral learning approach for probabilistic grammatical inference», dans *Proceedings of the 12th International Conference on Grammatical Inference, ICGI 2014, Kyoto, Japan, 17-19 September 2014*, p. 64–78.

Habrard, A., F. Denis et Y. Esposito. 2006, «Using pseudo-stochastic rational languages in probabilistic grammatical inference», dans *Grammatical Inference : Algorithms and Applications, 8th International Colloquium, ICGI 2006, Tokyo, Japan, September 20-22, 2006, Proceedings*, p. 112–124.

Hamilton, W. L., M. M. Fard et J. Pineau. 2013, «Modelling sparse dynamical systems with compressed predictive state representations», dans *Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013*, p. 178–186.

Hawkes, R. 1983, «The analysis of interception», *Unpublished Defense Science and Technology Organisation Res. Rep.*

Hefny, A., C. Downey et G. J. Gordon. 2015, «A new view of predictive state methods for dynamical system learning», *CoRR*, vol. abs/1505.05310.

de la Higuera, C. et F. Thollard. 2000, «Identification in the limit with probability one of stochastic deterministic finite automata», dans *Grammatical Inference : Algorithms and Applications, 5th International Colloquium, ICGI 2000, Lisbon, Portugal, September 11-13, 2000, Proceedings*, p. 141–156.

Hsu, D., S. M. Kakade et T. Zhang. 2009, «A spectral algorithm for learning hidden markov models», dans *Proceedings of the 22sn Annual Conference on Learning Theory, COLT 2009, 18-21 June 2009, Montreal, Quebec, Canada.*

Hsu, D., S. M. Kakade et T. Zhang. 2012, «A spectral algorithm for learning hidden Markov models», *Journal of Computer and System Sciences*, vol. 78, nº 5, p. 1460–1480.

Jacob, G. 1975, «Sur un théorème de shamir», *Information and Control*, vol. 27, nº 3, p. 218–261.

Jaeger, H. 1998, *Discrete Time, Discrete Valued Observable Operator Models : A Tutorial*, GMD-Forschungszentrum Informationstechnik.

Jaeger, H., M. Zhao, K. Kretzschmar, T. Oberstein, D. Popovici et A. Kolling. 2005, «Learning observable operator models via the es algorithm», *New directions in statistical signal processing : From systems to brains.*

Kearns, M. J., Y. Mansour, D. Ron, R. Rubinfeld, R. E. Schapire et L. Sellie. 1994, «On the learnability of discrete distributions», dans *Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing, 23-25 May 1994, Montréal, Québec, Canada*, p. 273–282.

Koksal, E. 2010, *Periodic Search Strategies For Electronic Countermeasure Receivers With Desired Probability Of Intercept For Each Frequency Band*, thèse de doctorat, Middle East Technical University.

Kulesza, A., N. Jiang et S. Singh. 2015a, «Low-rank spectral learning with weighted loss functions», dans *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2015, San Diego, California, USA, May 9-12, 2015*.

Kulesza, A., N. Jiang et S. Singh. 2015b, «Spectral learning of predictive state representations with insufficient statistics», dans *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA.*, p. 2715–2721.

Kulesza, A., N. R. Rao et S. Singh. 2014, «Low-rank spectral learning», dans *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics, AISTATS 2014, Reykjavik, Iceland, April 22-25, 2014*, p. 522–530.

Kullback, S. et R. A. Leibler. 1951, «On information and sufficiency», *The Annals of Mathematical Statistics*, p. 79–86.

Kumar, A., V. Sindhwani et P. Kambadur. 2013, «Fast conical hull algorithms for near-separable non-negative matrix factorization», dans *Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013*, p. 231–239.

Kunz, S., J. L. Twoey, B. L. Snedaker, R. Scheuerman, P. T. Coyne et D. Scheel. 2010, «Methods and apparatus for creating a scan strategy», G01S 13/00 ; G01S 5/02 ; G01S 7/42 ; 342 13 ; 342 89 ; 342158 ; 342422.

Lakshminarayanan, B. et R. Raich. 2010, «Non-negative matrix factorization for parameter estimation in hidden Markov models», dans *Machine Learning for Signal Processing (MLSP), 2010 IEEE International Workshop on*, IEEE, ISSN 1551-2541, p. 89–94.

Lee, D. D. et H. S. Seung. 1999, «Learning the parts of objects by non-negative matrix factorization», *Nature*, vol. 401, n^{o} 6755, p. 788–791.

Lewis, W. L. et J. A. Kardatzke. 2000, «Electronic support measures (ESM) duty dithering scheme for improved probability of intercept at low ESM utilization», G01S 736 ; 342/13 ; 342/162 ; 342/163 ; 342/17.

Lőtstedt, P. 1983, «Perturbation bounds for the linear least squares problem subject to linear inequality constraints», *BIT Numerical Mathematics*, vol. 23, n^{o} 4, doi: 10.1007/BF01933623, p. 500–519, ISSN 0006-3835.

Lin, C. 2007, «Projected gradient methods for nonnegative matrix factorization», *Neural Computation*, vol. 19, n^{o} 10, p. 2756–2779.

Littman, M. L., R. S. Sutton et S. P. Singh. 2001, «Predictive representations of state», dans *Advances in Neural Information Processing Systems 14 [Neural Information Processing Systems : Natural and Synthetic, NIPS 2001, December 3-8, 2001, Vancouver, British Columbia, Canada]*, p. 1555–1561.

Liu, K. et Q. Zhao. 2010, «Distributed learning in multi-armed bandit with multiple players», *IEEE Transactions on Signal Processing*, vol. 58, n^{o} 11, p. 5667–5681, ISSN 1053-587X.

MacKay, D. J. 1997, «Ensemble learning for hidden Markov models», cahier de recherche, University of Cambridge.

MacKay, D. J. et L. C. B. Peto. 1995, «A hierarchical dirichlet language model», *Natural language engineering*, vol. 1, n° 03, p. 289–308.

Marcus, M. P., M. A. Marcinkiewicz et B. Santorini. 1993, «Building a large annotated corpus of english : The penn treebank», *Computational linguistics*, vol. 19, n° 2, p. 313–330.

McCracken, P. et M. H. Bowling. 2005, «Online discovery and learning of predictive state representations», dans *Advances in Neural Information Processing Systems 18 [Neural Information Processing Systems, NIPS 2005, December 5-8, 2005, Vancouver, British Columbia, Canada]*, p. 875–882.

McDiarmid, C. 1989, «On the method of bounded differences», *Surveys in Combinatorics*, vol. 141, n° 1, p. 148–188.

Miller, K. et R. Schwarz. 1953, «On the interference of pulse trains», *Journal of Applied Physics*, vol. 24, n° 8, p. 1032–1036.

MOSEK. 2015, *The MOSEK Python optimizer API manual Version 7.1.* URL `http://docs.mosek.com/7.1/pythonapi/index.html`.

Mossel, E. et S. Roch. 2005, «Learning nonsingular phylogenies and hidden Markov models», dans *Proceedings of the 37th Annual ACM Symposium on Theory of Computing, Baltimore, MD, USA, May 22-24, 2005*, p. 366–375.

Oncina, J. et P. García. 1992, «Identifying regular languages in polynomial time», *Advances in Structural and Syntactic Pattern Recognition*, vol. 5, p. 99–108.

Ortner, R., D. Ryabko, P. Auer et R. Munos. 2014, «Regret bounds for restless Markov bandits», *Theor. Comput. Sci.*, vol. 558, p. 62–76.

Palmer, N. et P. W. Goldberg. 2005, «Pac-learnability of probabilistic deterministic finite state automata in terms of variation distance», dans *Algorithmic Learning Theory, 16th International Conference, ALT 2005, Singapore, October 8-11, 2005, Proceedings*, p. 157–170.

Pfau, D., N. Bartlett et F. Wood. 2010, «Probabilistic deterministic infinite automata», dans *Advances in Neural Information Processing Systems 23 : 24th Annual Conference on Neural Information Processing Systems 2010. Proceedings of a meeting held 6-9 December 2010, Vancouver, British Columbia, Canada.*, p. 1930–1938.

Quattoni, A., B. Balle, X. Carreras et A. Globerson. 2014, «Spectral regularization for max-margin sequence tagging», dans *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, p. 1710–1718.

Recht, B., C. Re, J. A. Tropp et V. Bittorf. 2012, «Factoring nonnegative matrices with linear programs», dans *Advances in Neural Information Processing Systems 25 : 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States.*, p. 1223–1231.

Ren, H. et C.-I. Chang. 2003, «Automatic spectral target recognition in hyperspectral imagery», *IEEE Transactions on Aerospace and Electronic Systems*, vol. 39, n° 4, p. 1232–1249, ISSN 0018-9251.

Richards, P. I. 1948, «Probability of coincidence for two periodically recurring events», *The Annals of Mathematical Statistics*, p. 16–29.

Ron, D., Y. Singer et N. Tishby. 1995, «On the learnability and usage of acyclic probabilistic finite automata», dans *Proceedings of the Eigth Annual Conference on Computational Learning Theory, COLT 1995, Santa Cruz, California, USA, July 5-8, 1995*, p. 31–40.

Rosencrantz, M., G. J. Gordon et S. Thrun. 2004, «Learning low dimensional predictive representations», dans *Machine Learning, Proceedings of the Twenty-first International Conference (ICML 2004), Banff, Alberta, Canada, July 4-8, 2004*.

Rydén, T. et collab.. 2008, «Em versus Markov chain Monte Carlo for estimation of hidden Markov models : A computational perspective», *Bayesian Analysis*, vol. 3, n° 4, p. 659–688.

Self, A. G. et B. G. Smith. 1985, «Intercept time and its prediction», *IEE Proceedings F Communications, Radar and Signal Processing*, vol. 132, n° 4, p. 215–220, ISSN 0143-7070.

Shaban, A., M. Farajtabar, B. Xie, L. Song et B. Boots. 2015, «Learning latent variable models by improving spectral solutions with exterior point methods», dans *Proceedings of the Thirty-First Conference on Uncertainty in Artificial Intelligence, UAI 2015, Amsterdam, NL, July 12-16, 2015*.

Shibata, C. et R. Yoshinaka. 2014, «A comparison of collapsed Bayesian methods for probabilistic finite automata», *Machine learning*, vol. 96, n° 1-2, p. 155–188.

Siddiqi, S. M., B. Boots et G. J. Gordon. 2010, «Reduced-rank hidden Markov models», dans *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2010, Chia Laguna Resort, Sardinia, Italy, May 13-15, 2010*, p. 741–748.

Siffre, J. 2003, *La guerre électronique : maître des ondes, maître du monde*, Renseignement & guerre secr'ete, Lavauzelle, ISBN 9782702510551.

Song, J. et K. C. Chen. 2015, «Spectacle : fast chromatin state annotation using spectral learning», *Genome biology*, vol. 16, n° 1, p. 33.

Song, L., B. Boots, S. M. Siddiqi, G. J. Gordon et A. J. Smola. 2010, «Hilbert space embeddings of hidden Markov models», dans *Proceedings of the 27th International Conference on Machine Learning (ICML-10), Haifa, Israel, 21-24 June 2010*, p. 991–998.

Stewart, G. et J. guang Sun. 1990, *Matrix perturbation theory*, Computer science and scientific computing, Academic Press, ISBN 9780126702309.

Sutskever, I., J. Martens et G. E. Hinton. 2011, «Generating text with recurrent neural networks», dans *Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, Washington, USA, June 28 - July 2, 2011*, p. 1017–1024.

Teh, Y. W. 2006, «A hierarchical Bayesian language model based on pitman-yor processes», dans *ACL 2006, 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, Sydney, Australia, 17-21 July 2006*.

Tekin, C. et M. Liu. 2011, «Adaptive learning of uncontrolled restless bandits with logarithmic regret», dans *2011 49th Annual Allerton Conference on Communication, Control, and Computing, Allerton Park & Retreat Center, Monticello, IL, USA, 28-30 September, 2011*, p. 983–990.

Tekin, C. et M. Liu. 2012, «Online learning of rested and restless bandits», *IEEE Transactions on Information Theory*, vol. 58, nº 8, p. 5588–5611, ISSN 0018-9448.

Tewari, A. et M. J. Giering. 2014, «Learning hidden Markov models using probabilistic matrix factorization», dans *Data Mining for Service*, Springer, p. 27–39.

Thon, M. et H. Jaeger. 2015, «Links between multiplicity automata, observable operator models and predictive state representations–a unified learning framework», *Journal of Machine Learning Research*, vol. 16, p. 103–147.

Valiant, L. G. 1984, «A theory of the learnable», dans *Proceedings of the 16th Annual ACM Symposium on Theory of Computing, April 30 - May 2, 1984, Washington, DC, USA*, p. 436–445.

Vanluyten, B., J. C. Willems et B. De Moor. 2008, «Structured nonnegative matrix factorization with applications to hidden Markov realization and clustering», *Linear Algebra and its applications*, vol. 429, nº 7, p. 1409–1424.

Vavasis, S. A. 2009, «On the complexity of nonnegative matrix factorization», *SIAM Journal on Optimization*, vol. 20, nº 3, p. 1364–1377.

Verwer, S., R. Eyraud et C. de la Higuera. 2012, «Results of the pautomac probabilistic automaton learning competition», dans *Proceedings of the Eleventh International Conference on Grammatical Inference, ICGI 2012, University of Maryland, College Park, USA, September 5-8, 2012*, p. 243–248.

Vidal, E., F. Thollard, C. de la Higuera, F. Casacuberta et R. C. Carrasco. 2005, «Probabilistic finite-state machines - part ii», *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, nº 7, p. 1013–1025, ISSN 0162-8828.

Wedin, P.-Å. 1972, «Perturbation bounds in connection with singular value decomposition», *BIT Numerical Mathematics*, vol. 12, nº 1, p. 99–111.

Wiewiora, E. 2005, «Learning predictive representations from a history», dans *Machine Learning, Proceedings of the Twenty-Second International Conference (ICML 2005), Bonn, Germany, August 7-11, 2005*, p. 964–971.

Wiley, R. G. 2006, *ELINT : The Interception and Analysis of Radar Signals*, Artech House Publishers.

Winter, M. E. 1999, «N-findr : an algorithm for fast autonomous spectral end-member determination in hyperspectral data», dans *SPIE's International Symposium on Optical Science, Engineering, and Instrumentation*, International Society for Optics and Photonics, p. 266–275.

Wolfe, B., M. R. James et S. P. Singh. 2005, «Learning predictive state representations in dynamical systems without reset», dans *Machine Learning, Proceedings of the Twenty-Second International Conference (ICML 2005), Bonn, Germany, August 7-11, 2005*, p. 980–987.

Xu, W., X. Liu et Y. Gong. 2003, «Document clustering based on non-negative matrix factorization», dans *SIGIR 2003 : Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, July 28 - August 1, 2003, Toronto, Canada*, p. 267–273.

Xun, Y., M. M. Kokar et K. Baclawski. 2004, «Control based sensor management for a multiple radar monitoring scenario», *Information Fusion*, vol. 5, n^o 1, p. 49–63.

Yang, Z. et E. Oja. 2012, «Quadratic nonnegative matrix factorization», *Pattern Recognition*, vol. 45, n^o 4, p. 1500–1510.

Zhang, H., Z. Yang et E. Oja. 2014, «Adaptive multiplicative updates for quadratic nonnegative matrix factorization», *Neurocomputing*, vol. 134, p. 206–213.

Zhao, M. et H. Jaeger. 2010, «Norm-observable operator models», *Neural Computation*, vol. 22, n^o 7, p. 1927–1959.

Zhao, M.-J., H. Jaeger et M. Thon. 2009, «A bound on modeling error in observable operator models and an associated learning algorithm», *Neural Computation*, vol. 21, n^o 9, p. 2687–2712.

Zhou, T., J. A. Bilmes et C. Guestrin. 2014, «Divide-and-conquer learning by anchoring a conical hull», dans *Advances in Neural Information Processing Systems 27 : Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, p. 1242–1250.

Annexe A

Liste des acronymes

BPC Bits Par Caractères, ou *Bits Per Character*.

BV Baum-Viterbi.

BW Baum-Welch.

CP-DFA automate fini déterministe réalisant des processus contrôlés, ou *Deterministic Finite Automata realizing Control Processes*.

CP-NFA automate fini non déterministe réalisant des processus contrôlés, ou *Nondeterministic Finite Automata realizing Control Processes*.

CP-RFA automate fini résiduel réalisant des processus contrôlés, ou *Residual Finite Automata realizing Control Processes*.

DFA automate fini déterministe, ou *Deterministic Finite Automata*.

DOA Direction D'Arrivée ou *Direction Of Arrival*.

ELINT Renseignements d'Origine Électro-magnétique ou *ELectronic INTelligence*.

EM Espérance-Maximisation, ou *Expectation-Maximization*.

ESM *Electronic Support Measures*.

FFT Transformation Rapide de Fourier ou *Fast Fourier Transform*.

FMA automate à multiplicité factorisé ou *Factorized Multiplicity Automata*.

FO Forme d'Onde.

HMM chaîne de Markov cachée, ou *Hidden Markov Model*.

IO-OOM modèle à opérateur observable à entrée-sortie, ou *Input-Output Observable Operator Model*.

KL Kullback-Leibler.

K-MA automate à multiplicité dans K ou *K-Multiplicity Automata*.

MAT *Minimal Adequate Teacher*.

MC chaîne de Markov, ou *Markov Chain*.

MCMC méthodes de Monte-Carlo par chaînes de Markov, ou *Monte Carlo Markov Chain*.

MDP processus décisionnel de Markov, ou *Markov Decision Process*.

MoM Méthode des Moments ou, *Method of Moments*.

NFA automate fini non déterministe, ou *Non-deterministic Finite Automata*.

NMF Factorisation de Matrice non-Négative, ou *Non-negative Matrix Factorization*.

NNLS Moindres Carrés Non-Negatifs, ou *Non-Negative Least Squares*.

OL Oscillateur Local.

OOM modèle à opérateur observable, ou *Observable Operator Model*.

PAC *Probably Approximately Correct*.

PDFA automate fini probabiliste déterministe, ou *Probabilistic Deterministic Finite Automata*.

PDW Ensemble des Descripteurs d'Impulsions ou *Pulse Description Word*.

PNFA automate fini probabiliste non déterministe, ou *Probabilistic Non-deterministic Finite Automata*.

POMDP processus décisionnel de Markov partiellement observable, ou *Partially Observable Markov Decision Process*.

PRFA automate fini probabiliste résiduel, ou *Probabilistic Residuel Finite Automata*.

PSR représentation à état prédictif, ou *Predictive State Representation*.

RF Radio Fréquence.

ROC Fonction d'Efficacité du Récepteur ou *Receiver Operating Characteristic*.

RWR Autoprotection ou *Radar Warning Receiver*.

S/N Signal sur Bruit ou *Signal over Noise*.

SDM Matrice de la Dynamique du Système, ou *System Dynamic Matrix*.

SH Super-Hétérodyne.

SP-DFA automate fini déterministe réalisant des processus stochastiques, ou *Deterministic Finite Automata realizing Stochastic Processes*.

SP-NFA automate fini non déterministe réalisant des processus stochastiques, ou *Non-deterministic Finite Automata realizing Stochastic Processes*.

SP-RFA automate fini résiduel réalisant des processus stochastiques, ou *Residual Finite Automata realizing Stochastic Processes*.

SVD Décomposition en Valeurs Singulières, ou *Singular Value Decomposition*.

SVT Seuillage des Valeurs Singulières, ou *Singular Value Thresholding*.

TOA Date D'Arrivée ou *Time Of Arrival*.

TSA Thalès Systèmes Aéroportés.

WER Taux d'Erreur de Mots, ou *Word Error Rate*.

Annexe B

Glossaire

alphabet ensemble de symboles, voir Section 1.2.1.

langage stochastique type de série formelle, voir Définition 20 page 24.

langage stochatique résiduel type de série formelle, voir Définition 25 page 28.

monoïde structure algébrique consistant en un ensemble muni d'une loi de composition interne associative et d'un élément neutre, voir Définition 1 page 14.

mot suite de symboles, voir Section 1.2.1.

préfixe symboles composant le début d'un mot.

processus contrôlé type de série formelle, voir Définition 38 page 35.

processus stochastique type de série formelle, voir Définition 30 page 32.

semi-anneau structure algébrique muni de deux lois, voir Définition 2 page 14.

semi-module structure algébrique définie sur les semi-anneau, voir Définition 3 page 14.

semi-module résiduel semi-module généré par les langages stochastiques résiduels.

série formelle extension des polynômes, voir Définition 4 page 15.

suffixe symboles composant la fin d'un mot.

Annexe C

Liste des symboles

ε mot vide, mot de longueur nulle.

$K\langle\Sigma\rangle$ ensemble des polynômes à coefficients dans K.

$K^{rat}\langle\langle\Sigma\rangle\rangle$ ensemble des séries formelles rationnelles à coefficients dans K, voir la Définition 5 page 15.

$K^{rec}\langle\langle\Sigma\rangle\rangle$ ensemble des séries formelles reconnaissable à coefficients dans K, voir la Définition 6 page 16.

$K\langle\langle\Sigma\rangle\rangle$ ensemble des séries formelles à coefficients dans K.

$S(\Sigma)$ ensemble des langages stochastiques.

$S_K^{rat}(\Sigma)$ ensemble des langages stochastiques rationnels dans K.

$S_{\mathbb{R}^+}^{Res}(\Sigma)$ ensemble des langages stochastiques rationnels dans $\mathbb{R}^+$ possèdant un nombre fini de langages stochastiques résiduels.

$S_{\mathbb{R}^+}^{[[Res]]}(\Sigma)$ ensemble des langages stochastiques rationnels dans $\mathbb{R}^+$ dont le semi-module résiduel est généré par une famille finie.

$S_K^{[[Res]]}(\Sigma)$ ensemble des langages stochastiques rationnels dans K dont le semi-module résiduel est généré par une famille finie.

$S_K^{Res}(\Sigma)$ ensemble des langages stochastiques rationnels dans K possèdant un nombre fini de langages stochastiques résiduels.

$CP(\Sigma)$ ensemble des processus contrôlés.

$CP_{\mathbb{R}^+}^{Res}(\Sigma)$ ensemble des processus contrôlés rationnels dans $\mathbb{R}^+$ qui possèdent un nombre fini de processus contrôlés résiduels à coefficient dans $\mathbb{R}^+$.

$CP_{\mathbb{R}^+}^{[[Res]]}(\Sigma)$ ensemble des processus contrôlés rationnels dans $\mathbb{R}^+$ dont le semi-module résiduel est généré par une famille finie.

$SP(\Sigma)$ ensemble des processus stochastiques.

$SP_{\mathbb{R}^+}^{Res}(\Sigma)$ ensemble des processus stochastiques rationnels dans $\mathbb{R}^+$ qui possèdent un nombre fini de processus stochastiques résiduels à coefficient dans $\mathbb{R}^+$.

$SP_{\mathbb{R}^+}^{[[Res]]}(\Sigma)$ ensemble des processus stochastiques rationnels dans $\mathbb{R}^+$ dont le semi-module résiduel est généré par une famille finie.

$SP_K^{rat}(\Sigma)$ ensemble des processus stochastiques rationnels dans K.