Aix-Marseille Université
Institut de Mathématiques de Luminy

THÈSE

Pour obtenir le grade de

Docteur de l'Université Aix-Marseille II

Discipline : Mathématiques Appliquées

Spécialité : Statistique

Meïli BARAGATTI

Le 10 novembre 2011

Sélection bayésienne de variables et méthodes de type *Parallel Tempering* avec et sans vraisemblance

Thèse financée par la société IPSOGEN

Jury composé de :

M. Christophe ABRAHAM	Montpellier SupAgro	*Examinateur*
M. Avner BAR-HEN	Université Paris Descartes	*Examinateur*
Melle Sabrina CARPENTIER	Ipsogen	*Examinatrice*
M. Marc CHADEAU-HYAM	Imperial College London	*Examinateur*
M. Nicolas CHOPIN	ENSAE	*Rapporteur*
Mme Adeline LECLERCQ-SAMSON	Université Paris Descartes	*Examinatrice*
M. Jean-Michel MARIN	Université Montpellier II	*Rapporteur*
M. Denys POMMERET	Aix-Marseille Université	*Directeur de thèse*

Remerciements

Je tiens tout d'abord à remercier Denys Pommeret pour avoir dirigé ma thèse durant ces trois années. En effet, c'est lui qui m'a encouragée à me lancer dans la recherche lorsqu'au bout de deux ans de travail en tant qu'ingénieur je me suis rendue compte que ce que je faisais ne comblait pas totalement mes attentes. Merci de m'avoir toujours encouragée et d'avoir toujours été optimiste, même quand les résultats n'étaient pas au rendez-vous, surtout au début. Merci de m'avoir fait confiance et de m'avoir laissé écrire mon premier papier toute seule, d'avoir toujours accepté les directions que j'ai prises. Merci de m'avoir présentée à de nombreuses personnes, surtout lors des congrés, et de m'avoir supportée trois ans avec mes questions et sollicitations fréquentes ! Cette thèse m'a énormément appris et apporté, et je pense avoir trouvé ma voie !

Je suis reconnaissante à Nicolas Chopin et Jean-Michel Marin pour m'avoir fait l'honneur d'accepter de rapporter cette thèse, et pour leurs lectures attentives du manuscript. Je remercie également les autres personnes du jury, pour avoir accepter de juger mon travail. En particulier Christophe Abraham, Avner Bar-Hen, Marc Chadeau-Hyam et Adeline Leclercq-Samson, que je n'ai rencontrés que lors de congrés ou séminaires et qui ont malgré tout accepté volontiers de faire partie du jury.

Mes remerciements s'adressent ensuite à la société Ipsogen, pour avoir financé cette thèse, pour m'avoir fait confiance dés le début, et pour m'avoir si bien acceuillie. Je pense en particulier à Fabienne Hermitte et Sabrina Carpentier, grâce à qui j'ai pu effectuer ma thèse dans de très bonnes conditions. Je remercie également Virginie Fasolo, pour m'avoir beaucoup appris en R, LaTex et Sweave, pour son énergie communicative et pour les nombreuses et sympathiques discussions que nous avons eues ! De manière générale, je remercie l'ensemble du personnel d'Ipsogen, car tout le monde fut chaleureux et j'ai pu partager de bons moments avec chacun d'entre eux, notamment lors des repas de midi !

Je tiens également à exprimer mes remerciements à Agnès Grimaud, avec qui il est agréable de travailler, et qui a surtout su m'écouter et me conseiller durant cette thèse, à n'importe quel moment quand j'en avais besoin.

Merci à Badih Ghattas de m'avoir fait confiance pour les enseignements, en me donnant la chance de donner des cours et des TD à ses étudiants. De même, les membres de ma nouvelle équipe de travail à Aix-Marseille I (Evolution, Génome et Environnement) m'ont fait confiance dés le début en me donnant la responsabilité d'un module entier. Je pense que cette année d'ATER va beaucoup m'apprendre, étant donné ce que j'ai déjà appris en un mois et demi !

Je remercie également Daniel Birnbaum et François Bertucci de l'institut Paoli Calmette pour avoir accepté que j'utilise leurs données de puces à ADN. De même, je remercie Sébastien Briolant du Service de Santé des Armées et François Nosten directeur de l'unité de recherche sur la malaria de Shoklo, pour m'avoir permis d'utiliser leurs données sur le paludisme afin d'illustrer un des algorithmes développés dans cette thèse.

Je tiens à exprimer toute ma gratitude à tous les membres de ma famille, qui ont toujours cru en moi et qui ont toujours été très présents pour me soutenir. En particulier merci à mes tantes Véronique et Béatrice pour leurs conseils, ainsi qu'à ma grand-mère qui m'a en plus supportée lors de la rédaction de ce manuscript. Merci surtout à mes parents, qui ont fait de moi ce que je suis et qui m'ont toujours encouragée dans les directions que je prenais.

Enfin, je ne peux finir sans remercier Guillaume, pour son soutien quotidien indéfectible, dans les bons et les mauvais moments, et qui a notamment su montrer lors des circonstances difficiles des derniers mois que je peux entièrement compter sur lui. Sa présence et le bonheur qu'il m'apporte permettent mon épanouissement personnel et professionnel.

Table des matières

Présentation générale

Cette thèse se compose de deux parties pouvant être lues indépendamment. La première partie a été motivée par une problématique rencontrée par la société Ipsogen dans l'élaboration de signatures génomiques, qui nous a amené à développer une méthode bayésienne de sélection de variables dans un modèle probit mixte. Dans ce type de problème assez complexe et ayant de nombreux paramètres, le choix de la méthode d'échantillonnage peut avoir une grande influence. Cela nous a conduit, dans une seconde partie, à étudier plus en détails les différents échantillonneurs existants, et notamment ceux basés sur la théorie des Monte Carlo Markov Chains (MCMC) et utilisant des populations de chaînes. Nous proposons alors un nouvel algorithme MCMC, le Parallel Tempering with Equi-Energy Moves. Enfin, dans certains cas complexes l'inférence peut être difficile car le calcul de la vraisemblance des données peut être trop coûteux, ou bien encore impossible. De nombreuses méthodes sans vraisemblance ont été développées (aussi appelées méthodes Approximate Bayesian Computation) et nous en proposons une nouvelle, basée sur la théorie des MCMC et sur une population de chaînes.

Sélection bayésienne de variables dans un modèle probit mixte

La sélection de variables est un problème important que l'on rencontre dans de nombreux domaines scientifiques, en particulier en bioinformatique et en médecine. En effet, grâce aux nouvelles technologies comme les puces microarray ou le séquençage haut-débit, nous pouvons obtenir de nombreuses informations précises sur les patients, et il devient possible de sélectionner des gènes impliqués dans le développement de certaines maladies comme le cancer. Les gènes identifiés peuvent être utilisés pour développer des modèles permettant de faire des prédictions pour de nouveaux patients, ces modèles étant appelés des signatures génomiques. L'utilisation de telles signatures peut par exemple permettre de prédire le statut d'un patient comme malade/pas malade, sa survie à 5 ans, ou encore sa réponse à un traitement, et donc influencer le traitement administré.

Les données à notre disposition sont des données de puces microarray pour des patientes atteintes de cancer du sein. Une puce est effectuée pour chaque patiente, et chacune donne les mesures d'expression de plusieurs dizaines de milliers de probesets. Un probeset est un ensemble

de probes qui mesurent en quelque sorte l'expression de petits bouts d'un même gène. Nous pouvons considérer qu'un probeset est associé à un seul gène. Cependant, un gène peut être représenté par plusieurs probesets.

La problématique de la société Ipsogen était de sélectionner quelques probesets associés au statut des récepteurs aux oestrogènes (ER), les statuts possibles étant présence/absence, appelés par les médecins ER positif/ER négatif. L'objectif est de pouvoir prédire les statuts ER de patientes à l'aide de ces probesets sélectionnés. En effet, la sévérité d'un cancer du sein pour une patiente est directement liée à son statut ER, car les oestrogènes jouent un rôle important dans le développement et la croissance de ces cancers. De plus, si des récepteurs aux oestrogènes sont présents sur les cellules cancéreuses, ils peuvent être ciblés par une thérapie hormonale. Ce statut ER est traditionnellement mesuré par des dosages immunohistochimiques (méthode biologique), mais les résultats obtenus varient beaucoup d'un laboratoire à un autre. Prédire le statut ER à l'aide d'une signature génomique permettrait d'avoir une meilleure reproductibilité entre laboratoires et entre pays.

Le statut ER étant une variable binaire, nous sommes dans le cadre classique de la sélection de variables à l'aide de modèles probit ou logit, étudié par de nombreux auteurs. Cependant, nous devons également prendre en compte une particularité de nos données : la réalisation d'une puce microarray est relativement chère, donc les jeux de données réalisés sont souvent de tailles limitées, pas plus d'une centaine d'individus en général. Dans l'objectif d'avoir le plus de données possibles et de tirer parti des nombreux jeux de données en libre accès sur internet, nous souhaitons utiliser plusieurs jeux de données différents. Chacun de ces jeux donne les résultats de puces réalisées sur des patientes atteintes de cancer du sein, mais ils proviennent de différents pays, différents laboratoires, et ont été receuillis par différentes équipes et protocoles. En général, si la présence de ces différentes sources n'est pas prise en compte dans l'analyse statistique, la plupart des effets dus aux gènes peuvent être masqués par un effet « jeu ». C'est une problématique courante en bioinformatique (voir par exemple Cheng et al. [40]). D'autant plus que de nombreux jeux de données sont accessibles et exploitables gratuitement sur le site NCBI GEO (voir [62]). Notre objectif est de prendre en compte cet effet, en considérant la provenance d'une patiente comme un effet aléatoire. Pour cela, nous développons une méthode bayésienne de sélection de variables prenant en compte la structure des données à l'aide d'effets aléatoires dans un modèle mixte.

Deux types d'approches bayésiennes sont couramment utilisées dans des problèmes de sélection de variables ou de modèles : des approches utilisant des facteurs de Bayes, ou bien des approches « Stochastic Search Variable Selection » (SSVS). Dans le premier cas des modèles sont comparés entre eux, tandis que dans le deuxième nous nous intéressons plus spécifiquement aux variables et non plus aux modèles entiers. Or nous disposons d'environ 20 000 variables, ce qui correspond à environ 2^{20000} modèles possibles. Il n'est donc pas envisageable de tous les comparer deux à deux. Même si nous nous concentrons sur les modèles ayant les plus fortes probabilités

a posteriori comme dans Madigan et Raftery [146], le nombre de comparaisons à effectuer reste trop important. De plus, dans notre cas particulier nous sommes dans une optique exploratoire. Sélectionner un peu trop de variables expliquant le statut ER des patientes n'est pas forcément génant, car une variable ne pourra être considérée intéressante que si elle a un sens biologique pertinent et si elle n'est pas déjà brevetée. Dans ce contexte il est préférable pour nous de sélectionner des variables potentiellement intéressantes plutôt que de sélectionner directement un modèle exploitable. C'est pour cela que nous optons pour une méthode SSVS.

Cette première partie est introduite au chapitre 1, qui contient une brève revue des méthodes « Stochastic Search Variable Selection » existantes. Au chapitre 2 nous développons une méthode de sélection de variables dans un modèle probit mixte. Cette méthode étend la méthode de sélection de variables dans un modèle probit de Lee et al. [131] au cas d'un modèle probit mixte. La première étape consiste à spécifier le modèle, ainsi que les distributions a priori, avec notamment l'utilisation de l'a priori conventionnel de Zellner [238] pour le vecteur des coefficients associés aux effets fixes. Dans une seconde étape, nous utilisons un algorithme Metropolis-within-Gibbs couplé à la « grouping » technique de Liu [141] afin de surmonter certaines difficultés d'échantillonnage. Puis la méthode développée est appliquée à la recherche de probesets pertinents pour expliquer le statut ER de patientes ayant un cancer du sein. Une limite de cette méthode est que la matrice de covariance utilisée dans l'a priori de Zellner doit nécessairement être inversible. Pour pallier cet inconvénient, nous proposons dans le chapitre 3 une modification de l'a priori de Zellner en y introduisant un paramètre ridge, ainsi qu'une manière de choisir les hyper-paramètres associés. Enfin, le chapitre 4 discute les méthodes précédemment développées et donne une liste non exhaustive de perspectives.

Méthodes de type *Parallel Tempering* avec et sans vraisemblance

Le problème de sélection de variables précédemment introduit nous a amené à étudier plus en détails les différents échantillonneurs existants, et notamment les méthodes Monte Carlo Markov Chains (MCMC). Lors de l'utilisation de méthodes MCMC classiques, comme l'algorithme de Metropolis-Hastings [157, 94], l'échantillonneur de Gibbs [70], ou encore l'algorithme Metropolis-within-Gibbs [220], une seule chaîne de Markov est générée. Pour pouvoir inférer à partir de cette chaîne, celle-ci doit avoir convergé et avoir visité l'ensemble de l'espace des états. Cependant il est connu que dans le cas de distributions multimodales ou de modèles complexes ayant de nombreux paramètres, la chaîne génerée a tendance à évoluer lentement, à mal se mélanger, et à rester bloquée dans un maximum local de l'espace des états, qui n'est donc pas entièrement visité. Cela pose des problèmes car des modes peuvent ne pas avoir été visités ou bien être représentés dans des proportions incorrectes. C'est particulièrement le cas pour les modèles de mélange (voir Celeux et al. [36]).

Afin de résoudre ce problème, de nombreuses méthodes et améliorations des échantillonneurs classiques ont été proposées, parmi lesquelles les mouvements à rejet retardé [159, 85], le simulated tempering [151, 78], le Parallel Tempering [76], ou l'Equi-Energy Sampler [118]. En particulier, l'algorithme Equi-Energy Sampler (EES) introduit par Kou et al. [118] semble plus efficace que l'algorithme Parallel Tempering (PT) introduit par Geyer [76]. Cependant, il n'est pas adapté pour se combiner avec un échantillonneur de Gibbs, et il nécessite un gros stockage de données. Nous proposons un algorithme combinant le PT avec le principe d'échanges entre chaînes ayant des niveaux d'énergie similaires, dans le même esprit que l'EES. Cette adaptation appelée Parallel Tempering with Equi-Energy Moves (PTEEM) conserve l'idée originale qui fait la force de l'algorithme EES, tout en assurant de bonnes propriétés théoriques et une utilisation facile en combinaison avec un échantillonneur de Gibbs. Pour évaluer les performances de cet algorithme, nous l'appliquons à des exemples simples et bien connus : une simulation suivant un mélange de gaussiennes bivariées, et des exemples d'estimation de paramètres dans des modèles de mélanges (jeu simulé et jeu de données Galaxy [184]). Puis nous l'appliquons à un problème plus difficile sur la recherche de sites promoteurs de facteurs de transcription dans des séquences d'ADN. Enfin, nous illustrons cette méthode sur un jeu de données réelles de chimiosensibilité de *Plasmodium Falciparum* à la doxycycline.

Les méthodes précédentes nécessitent la connaissance et le calcul de la vraisemblance des données. Dans certains cas complexes l'inférence peut être difficile car le calcul de cette vraisemblance s'avère trop coûteux, voire impossible si celle-ci ne s'exprime pas sous forme explicite. Les méthodes sans vraisemblance offrent une solution lorsque des données peuvent être simulées suivant le modèle supposé, et que ces données peuvent être assez bien résumées par un nombre raisonnable de statistiques. Alors, la distribution a posteriori d'un paramètre d'intérêt peut être approximée sans avoir à calculer la vraisemblance. De nombreuses méthodes basées sur ce principe ont été développées, parmi lesquelles l'ABC-MCMC [153], l'ABC-PMC [18] ou l'ABC-SMC [55]. La méthode ABC-MCMC est longtemps restée la méthode de référence, mais il apparaît depuis quelques années que les méthodes sans vraisemblance séquentielles comme l'ABC-PMC ou l'ABC-SMC sont bien plus performantes. Nous proposons une nouvelle méthode sans vraisemblance basée sur la théorie MCMC, utilisant une population de chaînes et permettant des échanges entre elles, par analogie avec le Parallel Tempering. Cette méthode, que nous appelons ABC-Parallel Tempering, est comparée aux méthodes existantes à l'aide de l'exemple jouet standard et de ce même exemple modifié. Puis nous l'illustrons sur un jeu de données réelles sur la transmission de la tuberculose.

Cette seconde partie est constituée de deux chapitres dédiés aux méthodes nécessitant le calcul de la vraisemblance, et de deux chapitres dédiés aux méthodes sans vraisemblance. Le chapitre 5 présente une revue des méthodes MCMC qui ont été proposées pour améliorer l'exploration de l'espace des états. Puis dans le chapitre 6 l'algorithme PTEEM est développé et son utilisation est

illustrée. Nous discutons également cet algorithme et des perspectives envisageables. Le chapitre 7 propose une revue des méthodes sans vraisemblances. Dans le chapitre 8 l'algorithme ABC-Parallel Tempering est développé et illustré. Plusieurs perspectives concernant les méthodes sans vraisemblances sont également présentées.

Première partie

Sélection bayésienne de variables

Chapitre 1

Introduction

Dans le cadre des modèles de régression, plusieurs méthodes bayésiennes ont été développées pour sélectionner des sous-ensembles de variables pertinentes. Nous pouvons citer notamment le critère bien connu du Bayesian Information Criterion développé par Schwartz (BIC, [194]), ou encore des méthodes basées sur des facteurs de Bayes [112] ou des pseudos facteurs de Bayes [23, 170]. Cependant dans le cas où le nombre de variables disponibles est très grand, le nombre de modèles possibles explose (2^p modèles si p variables), et des méthodes plus automatiques, de type « Stochastic Search Variable Selection » (SSVS) ont été développées. Ces dernières méthodes incluent le modèle de régression dans un modèle bayésien hiérarchique plus large, et utilisent un vecteur de variables latentes permettant d'identifier des sous-ensembles de variables pertinentes dans une optique de prédiction. C'est la distribution a posteriori de ce vecteur qui est d'intérêt, et une méthode MCMC (souvent un échantillonneur de Gibbs) est utilisée pour identifier les éléments de ce vecteur ayant les plus fortes probabilités a posteriori.

Les premiers auteurs ayant parlé de méthodes SSVS sont George et McCulloch en 1993 [74]. Ils ont expliqué cette approche en détail en 1997 [75], et en 2001 dans un papier écrit avec Chipman [41]. D'autres auteurs ont ensuite proposé des méthodes dans le même esprit, comme Kuo et Mallick [121], Dellaportas et al. [56], ou Brown et al. [30]. Une revue de ces méthodes est proposée par O'Hara [171]. Cependant, toutes ces méthodes concernent le modèle linéaire classique.

Peu après, des méthodes SSVS ont été développées pour des modèles de régression logit et probit. En ce qui concerne les modèles probit, un article de référence est celui de Lee et al. [131] qui se base sur un article d'Albert et Chib [2] sur l'analyse bayésienne de données binaires et polytomiques. La méthode de Lee et al. a été étendue au cas de modèles probit multinomiaux et non linéaires par Zhou et Wang [242, 243], et à la même période Sha et al. [197] ont également propose une méthode pour ces mêmes modèles. En ce qui concerne les modèles logistiques, nous

pouvons citer Chen et Dey [38] et Zhou et al. [241] qui ont été les premiers à combiner des méthodes SSVS avec des modèles logistiques, mais également Tüchler [225], Holmes et Held [96], Fouskakis et al. [66] et Kinney et Dunson [115]. Notons que dans le cadre de modèles logistiques, il est en général difficile d'exprimer les densités a posteriori de manière explicite car les propriétés de conjugaison sont perdues, des approximations sont donc utilisées dans la plupart des cas : approximation numérique [38], approximation par des mélange de normales [225], approximation par une loi normale [241] ou de Student [115], ou encore approximation de Laplace [66]. Seuls Holmes et Held ont proposé une méthode exacte, mais au prix de l'introduction de nouvelles variables latentes. Des méthodes SSVS ont également été proposées dans le cadre de modèles linéaires généralisés, voir notamment Ntzoufras et al. [168] et Cai et Dunson [31]. Cependant ils utilisent également des approximations (utilisation de séries de Taylor). Concernant notre problématique sur le statut ER de patientes atteintes de cancer du sein, nous sommes dans le cadre de données binaires, et bien que les coefficients d'une régression logistique soient plus facilement interprétables que ceux d'une régression probit, nous préférons utiliser un modèle probit. En effet, notre objectif est d'avoir un outil efficace pour la sélection de quelques variables parmi des dizaines de milliers. Or les modèles probit sont plus avantageux computationnellement que les modèles logit, car plus faciles d'utilisation dans un cadre bayésien pour les raisons précédentes.

A propos des modèles mixtes, certains comme Chen et Dunson [39] et Frühwirth-Schnatter et Wagner [69] se sont intéressés à la sélection des effets aléatoires. En ce qui nous concerne, nous voulons sélectionner des effets fixes en présence d'effets aléatoires. Cela a été étudié par Tüchler [225] et Kinney et Dunson [115] dans le cadre de modèles logistiques mixtes, et par Cai et Dunson [31] dans le cadre de modèles linéaires généralisés mixtes. Mais ces auteurs utilisent des approximations et leurs méthodes ne sont appliquées qu'à des jeux ayant quelques dizaines de variables seulement.

Enfin, d'autres méthodes SSVS ont été développées, comme par exemple celle de Tadesse et al. [214] dans le cadre de mélanges de lois normales. Leur méthode a été étendue au cas de données structurées par Schwartz et al. [195]. Récemment Gosh et al. [81] se sont intéressés aux modèles à classes latentes, en les modélisant également par des mélanges de lois normales. Sha et al. [196] ont étudié la sélection de variables bayésienne dans le cas de données censurées. Li et Zhang [134] ont quant à eux supposé une structure de l'espace des covariables. Dans un contexte d'économie de la santé, Fouskakis et al. [66] se sont placés dans un cadre de limite de coût. Une bonne revue sur la sélection de variables bayésienne en incorporant de l'information biologique a récemment été effectuée par Stingo et Vannucci [212].

Dans le cadre des modèles probit mixtes, il n'y a pas à notre connaissance de méthode SSVS efficace pour traiter des jeux avec des dizaines de milliers de variables. Nous en proposons une

dans le chapitre 2. Cette méthode est une extension de la méthode de Lee et al. [131] qui prend en compte des effets aléatoires. La méthode est appliquée à la recherche de probesets pertinents pour expliquer le statut ER de patientes atteintes de cancer du sein. Une étude de stabilité et de sensibilité est effectuée, ainsi qu'une comparaison avec des méthodes déjà existantes. Cependant, cette méthode utilise l'a priori de Zellner (g-prior, [238]) pour le vecteur des coefficients des effets fixes, et la matrice de covariance utilisée dans cet a priori doit nécessairement être inversible. Or il y a clairement deux cas pour lesquels cette matrice est singulière :

1. Lorsque le nombre de variables sélectionnées dépasse le nombre d'observations.
2. Lorsque des variables sont combinaisons linéaires d'autres variables.

En ce qui concerne le premier cas, plusieurs auteurs ont cherché des alternatives. Maruyama et George [155] ont proposé une généralisation du g-prior qui conserve la structure de covariance des données, en travaillant avec la décomposition en valeurs sigulières de la matrice de design X. Cependant leur approche n'est valable que dans le contexte de modèles linéaires classiques. Yang et Song [231] ont proposé de remplacer la matrice de covariance $(X'X)^{-1}$ par une inverse généralisée de Moore-Penrose. Mais cela ne permet pas d'obtenir une forme explicite de la distribution jointe a posteriori d'intérêt (voir la partie 3.1 du chapitre 3). Une solution pour éviter ce premier cas serait de fixer le nombre de variables à sélectionner à chaque itération à une valeur inférieure à n. Cela apporte certains avantages (voir la partie 2.2.3 du chapitre 2), mais ne règle pas le problème du second cas. De même, les a priori de Maruyama et George, et de Yang et Song ne solutionnent pas ce second cas. Dans l'esprit d'une régression ridge (voir [154]), Gupta et Ibrahim [89] ont proposé une extension du g-prior, en introduisant un paramètre ridge. Leur a priori peut être utilisé dans les deux cas présentés ci-dessus, mais ils n'ont pas considéré le second cas où des variables sont des combinaisons linéaires d'autres. Récemment Kwon et al. [122] ont proposé une méthode SSVS prenant en compte des corrélations élevées entre prédicteurs. Cependant leur approche n'est plus valable lorsque des prédicteurs sont combinaisons linéaires d'autres prédicteurs.

Il n'y a pas, à notre connaissance, dans la littérature de méthode SSVS développée pour traiter ce cas de variables qui sont combinaisons linéaires d'autres variables. Dans le chapitre 3, nous nous plaçons donc dans ce cas et proposons une modification de l'a priori de Zellner en y introduisant un paramètre ridge. De plus, nous proposons une manière de choisir les hyper-paramètres associés qui permet de conserver la variance totale des données. L'a priori obtenu est légèrement différent de celui de Gupta et Ibrahim [89], et est un compromis entre le g-prior classique et l'a priori supposant l'indépendance des coefficients de régression. Comme dans le chapitre 2 nous utilisons un modèle probit mixte, mais l'approche est généralisable à d'autres modèles. Pour illustrer la méthode des simulations sont réalisées, et une partie des jeux de données sur le statut ER de patientes atteintes de cancer du sein est utilisée. Les résultats obtenus sont comparés à des résultats obtenus par une approche de Lasso bayésien (Park et Casella [175]). La sensibilité de la méthode SSVS développée est ensuite étudiée.

Enfin, il n'y a pas à notre connaissance de méthode bayésienne de sélection de variables dans le cadre de modèles de mélanges pour des données censurées. Nous avons alors développé une méthode de sélection de variables dans le cadre d'un mélange de modèles à temps de sortie accéléré mixtes (Mixed Effects Accelerated Failure Time Models). Cette méthode est adaptée au cas de données de survie que nous suspectons d'être issues d'un mélange, avec éventuellement des effets aléatoires. Nous ne présentons pas de résultats associés car les jeux de données dont nous disposions ne nous ont pas permis d'obtenir des résultats encourageants. Pour ne pas alourdir la partie I, nous présentons ce travail en annexe B.

Chapitre 2

Sélection bayésienne de variables dans un modèle probit mixte

Notre méthode étend le modèle utilisé par Lee et al. [131] en y ajoutant des effets aléatoires. Nous sommes confrontés à plusieurs difficultés. Tout d'abord toutes les distributions conditionnelles complètes ne peuvent pas être générées directement. Nous utilisons donc la « grouping » technique de Liu [141], et combinons un échantillonneur de Gibbs avec un algorithme de Metropolis-Hastings. L'algorithme complet est donc un Metropolis-within-Gibbs ([186]) combiné avec la grouping technique de Liu. Pour des raisons computationnelles, nous fixons le nombre de variables sélectionnées à chaque itération de l'algorithme. Il en résulte que la valeur choisie pour le « coefficient de sélection de variables » utilisé dans notre modèle a une influence réduite. C'est particulièrement avantageux, car le choix de cette valeur est souvent difficile et influent dans d'autres méthodes de sélection de variables, voir par exemple Bottolo et Richardson qui ont alors proposé de poser un a priori sur ce coefficient [28].

2.1 Le modèle bayésien hiérarchique

2.1.1 Le modèle probit mixte

Supposons que n évènements binaires sont observés, notés Y_i, $i = 1, \ldots, n$. L'ensemble des prédicteurs dont nous disposons est de taille p, avec $p \gg n$. Nous considérons le modèle probit mixte suivant :

$$P(Y_i = 1 \mid U, \beta) = \Phi(X_i'\beta + Z_i'U), \tag{2.1}$$

avec

- Φ la fonction de répartition de la loi normale centrée réduite.
- X_i et Z_i correspondent aux effets fixes et aléatoires associés à la $i^{\text{ème}}$ observation, et sont de tailles p et q respectivement.

- β le vecteur des coefficients des effets fixes, de taille p.
- U le vecteur des coefficients des effets aléatoires, de taille q. Nous supposons être en présence de K effets aléatoires, et notons $U = (U_1', \ldots, U_K')'$. Chaque U_l est de taille q_l, et $\sum_{l=1}^{K} q_l = q$.
- Nous notons X et Z les matrices de design associées aux effets fixes et aléatoires respectivement.

Suivant le principe utilisé par Albert et Chib [2], un vecteur de variables latentes L est introduit. Nous notons $L = (L_1, \ldots, L_n)$ et nous posons :

$$Y_i = \begin{cases} 1 & \text{si } L_i > 0 \\ 0 & \text{si } L_i < 0. \end{cases}$$

En supposant que (I_n étant la matrice identité d'ordre n) :

$$L \mid U, \beta \sim \mathcal{N}_n(X\beta + ZU, I_n), \tag{2.2}$$

nous avons alors :

$$P(Y_i = 1 \mid U, \beta) = P(L_i > 0 \mid U, \beta) = \Phi(X_i'\beta + Z_i'U).$$

Puis, un vecteur γ constitué de p variables indicatrices est introduit afin de pouvoir sélectionner des variables :

$$\gamma_j = \begin{cases} 1 & \text{si variable } j \text{ sélectionnée} \\ 0 & \text{si variable } j \text{ non sélectionnée.} \end{cases} \tag{2.3}$$

Sélectionner un sous-ensemble de prédicteurs pour Y revient à conserver tous les prédicteurs dont les coefficients associés β_j ne peuvent pas être estimés par 0. Nous supposons, comme Lee et al. [131], Zhou et al. [243] et Kwon et al. [122] parmi d'autres, que les coefficients des prédicteurs qui n'interviennent pas dans le modèle sont fixés à 0. Nous avons donc :

$$\gamma_j = \begin{cases} 1 & \text{si } \beta_j \neq 0, \quad \text{variable } j \text{ sélectionnée} \\ 0 & \text{si } \beta_j = 0, \quad \text{variable } j \text{ non sélectionnée.} \end{cases} \tag{2.4}$$

Sachant γ, β_γ est le vecteur des éléments non nuls de β, et X_γ correspond à la matrice X ne contenant que les colonnes associées aux éléments non nuls de γ.

Notons que George et McCulloch [74], Mitchell et Beauchamp [160] et Brown et al. [30] par exemple considèrent que la $j^{\text{ème}}$ variable ne doit pas être sélectionnée non pas quand $\beta_j = 0$, mais quand β_j est suffisament petit pour pouvoir être négligé et estimé par 0. Les différentes manières de modéliser le vecteur β ont été présentées par O'Hara et Sillanpää [171].

2.1.2 Les distributions a priori

Le modèle hiérarchique est complété par des distributions a priori pour $U \mid D$, $\beta_\gamma \mid \gamma$, γ et D, où D est une matrice de variance-covariance de dimension q.

Distribution a priori de β_γ

Seules interviennent dans les prédictions ou classifications les variables sélectionnées correspondant à un coefficient associé β_j non nul. Nous spécifions donc une distribution a priori du vecteur sans élément nul β_γ :

$$\beta_\gamma \mid \gamma \sim \mathcal{N}_{d_\gamma}(0, c(X'_\gamma X_\gamma)^{-1}), \qquad \text{avec} \qquad d_\gamma = \sum_{j=1}^{p} \gamma_j, \tag{2.5}$$

où c est un facteur d'échelle positif spécifié par l'utilisateur, appelé « coefficient de sélection de variables » par Bottolo et Richardson [28]. Cet a priori correspond à la distribution g-prior définie par Zellner [238]. C'est un a priori conventionnel dans la sélection bayésienne de variables. Il permet à l'utilisateur de spécifier un paramètre de localisation (ici 0) sans avoir à spécifier une structure de covariance, ce qui est le plus difficile. La structure de covariance est fixée par $c(X'_\gamma X_\gamma)^{-1}$ et permet en quelque sorte un ajustement automatique basé sur les données. En effet, cet a priori reflète la structure de variance-covariance (et donc de corrélation) des données, et est lié à la matrice d'information de Fisher dans le cas du modèle linéaire classique. De plus, il permet en général un calcul aisé de la vraisemblance marginale par conjugaison.

Le choix du coefficient de sélection de variables c peut avoir une influence sur le processus de sélection et a été considéré par de nombreux auteurs. Certains recommandent de choisir une valeur fixe, comme Gupta et Ibrahim [90] ou Smith et Kohn [202] qui préconisent de le choisir entre 10 et 100. D'autres proposent une estimation de ce coefficient, comme George et Foster [73] qui utilisent des méthodes bayésiennes empiriques. D'autres encore préconisent de mettre une distribution a priori dessus, comme Zellner et Siow [239] qui posent une distribution inverse-gamma $\mathcal{IG}(1/2, n/2)$. L'inconvénient de cet a priori de Zellner-Siow est que la vraisemblance marginale ne peut s'écrire sous forme explicite, et des approximations sont alors nécessaires. Cet a priori de Zellner-Siow peut être vu comme un mélange de g-priors. C'est à partir de ce constat que Liang et al. [137] ont proposé une nouvelle famille d'a priori sur c, appelée hyper-g prior, qui mène à des formules de vraisemblance marginales explicites, mais tout de même peu exploitables car par exemple dans le cas du modèle linéaire on a affaire à des fonctions hypergéométriques. De manière indépendante, Cui et George [52] ont proposé de poser une distribution inverse-gamma sur $(1 + c)$ (au lieu de c comme Zellner et Siow), ce qui définit une famille d'a priori sur c dont un cas particulier est la famille hyper-g prior. Bottolo et Richardson [28] ont également utilisé un a priori de cette forme. Dans le cadre de la régression linéaire, Celeux et al. [37] et Marin et Robert [150] ont proposé un a priori impropre discret sur ce paramètre c. Cependant cet a

priori introduit une somme infinie ce qui rend son utilisation difficile. Celeux et al. [35] ont alors proposé un a priori de Jeffrey continu sur c, et Guo et Speckman [88] ont montré la consistance des facteurs de Bayes associés.

Nous nous plaçons ici dans le cas simple d'un a priori conventionnel g-prior de Zellner avec la valeur de c considérée fixée, et nous verrons que le choix de cette valeur n'aura pas d'influence dans notre algorithme (parties 2.2.3 et 2.3.2).

Distribution a priori de γ

A priori les élements γ_j sont supposés suivre des distributions de Bernoulli indépendantes entre elles, avec :

$$P(\gamma_j = 1) = \pi_j, \qquad 0 \leq \pi_j \leq 1. \tag{2.6}$$

Dans le cas de notre application où nous ne savons pas a priori quels sont les probesets les plus susceptibles d'être pertinents, nous ne pouvons pas utiliser d'information a priori pour favoriser la sélection de certains probesets par rapport à d'autres. Nous posons alors $\pi_j = \pi = 1/p$, $\forall j = 1, \dots p$.

Distribution a priori de $U \mid D$ et D

De manière classique, nous supposons que le vecteur des coefficients associés aux effets aléatoires est gaussien et centré :

$$U \mid D \sim \mathcal{N}_q(0, D). \tag{2.7}$$

Dans le cas général, aucune structure n'est supposée pour la matrice de variance-covariance D, sa distribution a priori est une inverse-wishart $\mathcal{W}^{-1}(\Psi, m)$.
Si nous nous placons dans le cas d'une matrice D diagonale (cas de notre application), avec $D = diag(A_1, \dots, A_K)$, $A_l = \sigma_l^2 I_{q_l}$, $l = 1, \dots, K$ et I_{q_l} la matrice identité d'ordre q_l, alors les distributions a priori des σ_l^2 sont supposées être des inverse-gamma $\mathcal{IG}(a, b)$ (b correspond au facteur d'échelle).

2.2 L'échantillonneur bayésien

La distribution a posteriori de γ est celle qui nous intéresse, car elle contient de l'information sur la pertinence des différentes variables pour expliquer la variable d'intérêt Y. Le nombre de variables explicatives dont nous disposons est de l'ordre de plusieurs dizaines de milliers. Il en découle que le nombre de vecteurs γ possibles est extrêmement large (2^p). L'idée est alors d'utiliser un échantillonneur de Gibss classique pour explorer la distribution a posteriori de l'ensemble des paramètres du modèle $f(L, \beta_\gamma, U, \gamma, D \mid Y)$. Nous pourrons alors explorer la distribution a posteriori de γ et chercher les valeurs ayant une forte probabilité a posteriori.

La chaîne de Markov générée pour γ aura la propriété que les valeurs de γ ayant une forte probabilité a posteriori apparaîtront plus fréquemment que les valeurs ayant une faible probabilité a posteriori. Des variables pertinentes pour expliquer la variable d'intérêt Y seront alors rapidement et facilement identifiables alors que l'échantillonneur explorera la distribution a posteriori. Par conséquent la chaîne de Markov générée peut avoir une longueur bien inférieure à 2^p et tout de même servir à identifier certaines des variables pertinentes, qui auront une forte probabilité marginale a posteriori d'avoir $\gamma_j = 1$.

Dans le but de pouvoir utiliser un échantillonneur de Gibss classique, nous devons être capables de générer des observations suivant l'ensemble des distributions conditionnelles complètes.

2.2.1 Les distributions conditionnelles complètes

Distribution conditionnelle complète de L
Grâce à la structure hiérarchique du modèle, cette distribution ne dépend que de Y, β_γ, γ et U. En combinant (2.2) et (2.1.1), nous obtenons :

$$\begin{aligned} L_i \mid \beta_\gamma, \gamma, U, Y_i = 1 &\sim \mathcal{N}(X'_{\gamma i}\beta_\gamma + Z'_i U, 1) \quad \text{tronquée à gauche en 0} \\ L_i \mid \beta_\gamma, \gamma, U, Y_i = 0 &\sim \mathcal{N}(X'_{\gamma i}\beta_\gamma + Z'_i U, 1) \quad \text{tronquée à droite en 0.} \end{aligned} \tag{2.8}$$

Nous utilisons le fait que $X\beta = X_\gamma \beta_\gamma$. D'un point de vue pratique, il est plus facile d'utiliser la matrice X_γ associée au vecteur β_γ que la matrice X associée au vecteur β.

Distribution conditionnelle complète de β_γ
Toujours grâce à la structure hiérarchique du modèle, cette distribution ne dépend que de L, U et γ. Nous notons $V_\gamma = \frac{c}{1+c}(X'_\gamma X_\gamma)^{-1}$.

$$f(\beta_\gamma \mid L, U, \gamma) \propto f(L \mid \beta_\gamma, \gamma, U) f(\beta_\gamma \mid \gamma),$$

et en utilisant (2.2) et (2.5) nous obtenons :

$$\beta_\gamma \mid L, U, \gamma \sim \mathcal{N}_{d_\gamma}(V_\gamma X'_\gamma (L - ZU), V_\gamma) \qquad \text{avec} \qquad d_\gamma = \sum_{j=1}^{p} \gamma_j. \tag{2.9}$$

Distribution conditionnelle complète de U
Cette distribution ne dépend que de L, β_γ, γ et D. Nous notons $W = (Z'Z + D^{-1})^{-1}$.

$$f(U \mid L, \beta_\gamma, \gamma, D) \propto f(L \mid \beta_\gamma, \gamma, U) f(U \mid D),$$

et en utilisant (2.2) et (2.7) nous obtenons :

$$U \mid L, \beta_\gamma, \gamma, D \sim \mathcal{N}_q(WZ'(L - X_\gamma\beta_\gamma), W). \tag{2.10}$$

Distribution conditionnelle complète de D
Cette distribution ne dépend que de U.

$$f(D \mid U) \quad \propto \quad f(U \mid D)f(D).$$

Dans le cas général où a priori D suit une inverse-wishart $\mathcal{W}^{-1}(\Psi, m)$, en utilisant (2.7), nous obtenons :

$$D \mid U \quad \sim \quad \mathcal{W}^{-1}(UU' + \Psi, m + 1). \tag{2.11}$$

Dans le cas d'une matrice D diagonale, où les distributions a priori des σ_l^2 sont supposées être des inverse-gamma $\mathcal{IG}(a, b)$, nous obtenons :

$$\sigma_l^2 \mid U_l \quad \sim \quad \mathcal{IG}\Big(\frac{q_l}{2} + a, \frac{1}{2}U_l'U_l + b\Big). \tag{2.12}$$

Distribution conditionnelle complète de γ
Cette distribution ne dépend que de L, β_γ et U.

$$\begin{aligned} f(\gamma \mid L, \beta_\gamma, U) \quad &\propto \quad f(L \mid \gamma, \beta_\gamma, U)f(\gamma) \\ &\propto \quad f(L \mid \gamma, \beta_\gamma, U)f(\beta_\gamma \mid \gamma)f(\gamma). \end{aligned}$$

En utilisant (2.2), (2.5) et (2.6), la densité de la distribution conditionnelle complète de γ est donnée par :

$$\begin{aligned} f(\gamma \mid \beta_\gamma, L, U) \quad \propto \quad & (2\pi)^{-\frac{d_\gamma}{2}} \exp\Big[-\frac{1}{2}\Big(- L'X_\gamma\beta_\gamma - \beta_\gamma'X_\gamma'L + \beta_\gamma'X_\gamma'ZU + U'Z'X_\gamma\beta_\gamma + \beta_\gamma'V_\gamma^{-1}\beta_\gamma\Big)\Big] \\ & \times \mid c(X_\gamma'X_\gamma)^{-1} \mid^{-\frac{1}{2}} \prod_{j=1}^{p} \pi_j^{\gamma_j}(1 - \pi_j)^{1-\gamma_j}, \qquad \text{avec} \quad d_\gamma = \sum_{j=1}^{p} \gamma_j. \end{aligned} \tag{2.13}$$

2.2.2 Utilisation de la technique de « grouping »

La densité conditionnelle complète de γ (2.13) n'est pas la densité d'une distribution classique. Bien que Lee et al. [131] proposent de générer ce vecteur élément par élément, nous avons décidé de le générer suivant sa distribution conditionnelle complète à l'aide d'un algorithme de Metropolis-Hastings (justification dans la section 2.2.3). L'algorithme complet est alors un algorithme de Metropolis-within-Gibbs.
Cependant, dans l'optique de générer γ, le paramètre β_γ est un paramètre de nuisance. En effet,

à une itération donnée la valeur de γ est associée à la valeur de β_γ. Par conséquent, la probabilité d'acceptation pour un candidat proposé γ^* dans l'algorithme de Metropolis-Hastings dépendra également du β_{γ^*} associé. Le problème est que la valeur du β_{γ^*} est inconnue. Ainsi, les paramètres γ et β_γ doivent être considérés ensemble, puisqu'il y a une structure de dépendance entre les deux.

Pour considérer γ et β_γ ensemble, nous utilisons la technique de « grouping » de Liu [141] (également appelée technique de « blocking »). L'idée est de grouper les paramètres β_γ et γ. Au lieu de s'intéresser séparement aux distributions conditionnelles complètes de β_γ et γ, nous nous intéressons alors à la distribution conditionnelle complète de (β_γ, γ). cette technique de grouping conserve la distribution stationnaire de la chaîne de Markov d'intérêt, peut aider à sa mise en oeuvre (gain de temps), et facilite la convergence de la chaîne en réduisant les autocorrélations (voir Liu [141] et van Dyk et Park [227]). L'échantillonneur obtenu est un cas particulier des « partially collapsed Gibbs samplers », échantillonneurs introduits et étudiés par van Dyk et Park. L'annexe A explique brièvement la manière d'obtenir ces échantillonneurs. Nous observons que

$$f(\beta_\gamma, \gamma \mid L, U) \propto f(\gamma \mid L, U) f(\beta_\gamma \mid \gamma, L, U).$$

Ainsi, générer (β_γ, γ) suivant la distribution $(\beta_\gamma, \gamma) \mid L, U$ revient à générer γ suivant sa distribution conditionnelle complète intégrée en β_γ, puis β_γ d'après sa distribution conditionnelle complète (donc conditionnellement au γ qui vient d'être généré). A chaque itération de l'algorithme nous devons générer γ avant $\beta\gamma$, afin de conserver la structure de dépendance entre $\beta\gamma$ et γ (voir van Dyk et Park [227]). La distribution intégrée $\gamma \mid L, U$ ne dépend plus du paramètre de nuisance β_γ, et nous pouvons donc facilement générer γ suivant cette distribution avec un algorithme de Metropolis-Hastings.

Pour obtenir la distribution intégrée de $\gamma \mid L, U$, nous devons intégrer $f(\gamma \mid \beta_\gamma, L, U)$ en β_γ. Pour cela nous calculons $f(L, \beta_\gamma \mid \gamma, U)$ en utilisant (2.2) et (2.5) :

$$\begin{aligned} f(L, \beta_\gamma \mid \gamma, U) &= f(L|\beta_\gamma, \gamma, U) f(\beta_\gamma|\gamma) \\ &= (1+c)^{-\frac{d_\gamma}{2}} |V_\gamma|^{-\frac{1}{2}} \exp\Big\{ -\frac{1}{2}\Big[\big(\beta_\gamma - V_\gamma X_\gamma'(L - ZU)\big)' V_\gamma^{-1} \big(\beta_\gamma - V_\gamma X_\gamma'(L - ZU)\big) \\ &\quad + (L - ZU)'(L - ZU) - \frac{c}{1+c}(L - ZU)' X_\gamma (X_\gamma' X_\gamma)^{-1} X_\gamma'(L - ZU) \Big] \Big\} (2\pi)^{-\frac{n+d_\gamma}{2}}. \end{aligned}$$

En reconnaissant la densité d'une variable gaussienne, nous pouvons intégrer cette densité en $\beta\gamma$, et nous obtenons :

$$\begin{aligned} f(L|\gamma, U) &= (1+c)^{-\frac{d_\gamma}{2}} (2\pi)^{-\frac{n}{2}} \exp\Big\{ -\frac{1}{2}\Big[(L - ZU)'(L - ZU) \\ &\quad - \frac{c}{1+c}(L - ZU)' X_\gamma (X_\gamma' X_\gamma)^{-1} X_\gamma'(L - ZU) \Big] \Big\}. \end{aligned}$$

Enfin, nous obtenons $f(\gamma \mid L, U)$ par :

$$\begin{aligned} f(\gamma|L,U) &\propto f(L|\gamma,U)f(\gamma) \\ &\propto (1+c)^{-\frac{\sum\gamma_j}{2}} \exp\Big\{-\frac{1}{2}(L-ZU)'\Big[I_n - \frac{c}{1+c}X_\gamma(X'_\gamma X_\gamma)^{-1}X'_\gamma\Big](L-ZU)\Big\} \\ &\quad \times \prod_{j=1}^{p} \pi_j^{\gamma_j}(1-\pi_j)^{1-\gamma_j}. \end{aligned} \tag{2.14}$$

2.2.3 Algorithme de l'échantillonneur

Algorithme de Metropolis-Hastings pour générer γ
Nous voulons générer γ suivant la densité (2.14).
Notons $\gamma^{(i)}$ la valeur de γ à l'itération i de l'algorithme MH et $q(.|.)$ un noyau de transition. A chaque itération, l'algorithme effectue les étapes suivantes :

1. Générer une valeur $\gamma^* \sim q(\gamma^*|\gamma^{(i)})$
2. Prendre
$$\gamma^{(i+1)} = \begin{cases} \gamma^* & \text{avec probabilité} \quad \rho(\gamma^{(i)}, \gamma^*) \\ \gamma^{(i)} & \text{avec probabilité} \quad 1-\rho(\gamma^{(i)}, \gamma^*), \end{cases}$$
où
$$\rho(\gamma^{(i)}, \gamma^*) = \min\left\{1, \frac{f(\gamma^* \mid L,U)q(\gamma^{(i)}|\gamma^*)}{f(\gamma^{(i)} \mid L,U)q(\gamma^*|\gamma^{(i)})}\right\}.$$

Nous prenons un noyau de transition symétrique, afin que la probabilité d'acceptation d'un candidat γ^* à partir de $\gamma^{(i)}$ se simplifie. En effet, dans ce cas nous aurons :

$$\begin{aligned} \rho(\gamma^{(i)}, \gamma^*) &= \min\Bigg\{\exp\Big[\frac{c}{2(1+c)}(L-ZU)'\Big(X_{\gamma^*}(X'_{\gamma^*}X_{\gamma^*})^{-1}X'_{\gamma^*} - X_{\gamma^{(i)}}(X'_{\gamma^{(i)}}X_{\gamma^{(i)}})^{-1}X'_{\gamma^{(i)}}\Big) \\ &\quad (L-ZU)\Big] \times (1+c)^{\sum_1^p(\gamma_j^*-\gamma_j^{(i)})} \times \Big(\frac{\pi}{1-\pi}\Big)^{\sum_1^p(\gamma_j^*-\gamma_j^{(i)})}, 1\Bigg\} \quad \text{si } \forall j\ \pi_j = \pi. \end{aligned} \tag{2.15}$$

Le plus simple pour obtenir un noyau de transition symétrique est de proposer un γ^* correspondant à $\gamma^{(i)}$ pour lequel r éléments ont été choisis au hasard et modifiés.
En ce qui nous concerne, nous ne choisissons pas les r éléments à modifier totalement au hasard : r étant pair, $r/2$ éléments parmi les éléments de $\gamma^{(i)}$ valant 1 sont choisis au hasard et changés en 0, et $r/2$ éléments parmi les éléments de $\gamma^{(i)}$ valant 0 sont choisis au hasard et changés en 1. Cette astuce implique que le nombre d'éléments de $\gamma^{(i)}$ valant 1 est constant, et que le nombre de variables sélectionnées reste identique à chaque itération. Cela a trois avantages non négligeables :

1. Sans cette astuce, étant donné que p est très grand, au début de l'algorithme la probabilité de choisir un élément de γ valant 1 est beaucoup plus faible que la probabilité de choisir un élément de γ valant 0. Ainsi, l'échantillonneur essaie plus souvent d'ajouter des variables que d'en supprimer. Le nombre de variables sélectionnées augmente donc et peut éventuellement dépasser le nombre d'observations, ce qui entraîne une matrice $X'_\gamma X_\gamma$ non inversible (voir chapitre 3).

2. La formule de la probabilité d'acceptation de l'algorithme MH est simplifiée :

$$\rho(\gamma^{(i)}, \gamma^*) = \min\left\{ \exp\Big[\frac{c}{2(1+c)}(L-ZU)'\Big(X_{\gamma^*}(X'_{\gamma^*}X_{\gamma^*})^{-1}X'_{\gamma^*} - X_{\gamma^{(i)}}(X'_{\gamma^{(i)}}X_{\gamma^{(i)}})^{-1}X'_{\gamma^{(i)}}\Big)(L-ZU)\Big], 1\right\} \quad \text{si } \forall j\ \pi_j = \pi. \tag{2.16}$$

3. Le choix de la valeur à donner au facteur d'échelle c pour l'a priori sur β_γ (2.5) prend moins d'importance. En effet, dans l'algorithme nous devons générer β_γ et γ suivant les densités $f(\beta_\gamma \mid L, U, \gamma)$ et $f(\gamma \mid L, U)$. Ces densités dépendent de c à travers les facteurs $c/(1+c)$ et $(1+c)^{-\frac{\sum\gamma_i}{2}}$ (voir (2.14) et (2.9)). Or $c/(1+c)$ est relativement proche de 1 si c est assez grand, et quand le nombre d'élément de γ valant 1 reste constant, le facteur $(1+c)^{-\frac{\sum\gamma_i}{2}}$ ne joue pas de rôle dans la génération de γ car s'élimine dans le ratio de la probabilité d'acceptation (2.16) de l'algorithme de Metropolis-Hastings.

REMARQUE 2.2.1 *Certains auteurs, comme Lee et al. [131] et Zhou et al. [242] [242] par exemple, génèrent le vecteur γ élément par élément en utilisant un échantillonneur de Gibbs. En effet, chaque élément de γ suit une loi de Bernoulli. Cette méthode nécessite de calculer pour chaque élément de γ sa probabilité de valoir 1, tandis que générer γ à l'aide d'un algorithme de Metropolis-Hastings implique de calculer une probabilité d'acceptation à chaque itération de cet algorithme. Nous considérons plus avantageux de générer γ à l'aide d'un algorithme de Metropolis-Hastings plutôt qu'élément par élément :*

- *L'algorithme de Metropolis-Hastings tel que défini ci-dessus nous permet de pouvoir facilement générer un vecteur γ ayant un nombre invariant de composants valant 1, ce qui n'est pas le cas si chaque élément de γ est généré suivant une Bernoulli. Cela nous donne les trois avantages exposés ci-dessus.*
- *La probabilité de valoir 1 pour un élément de γ demande un peu plus de calculs que le taux d'acceptation pour une itération de l'algorithme MH (voir (2.16)). En effet :*

$$P(\gamma_j = 1 \mid \gamma_{\{-j\}}, L, U) = \frac{P(\gamma_j = 1 \mid \gamma_{\{-j\}}, L, U)}{P(\gamma_j = 1 \mid \gamma_{\{-j\}}, L, U) + P(\gamma_j = 0 \mid \gamma_{\{-j\}}, L, U)}.$$

Et d'après (2.14), en notant $\gamma^1 = (\gamma_1, \ldots, \gamma_{j-1}, 1, \gamma_{j+1}, \ldots, \gamma_p)$, $\gamma^0 = (\gamma_1, \ldots, \gamma_{j-1}, 0, \gamma_{j+1}, \ldots, \gamma_p)$,

$$P(\gamma_j = 1 \mid \gamma_{\{-j\}}, L, U) \propto (1+c)^{-\frac{\sum \gamma_j^1}{2}} \exp\Big[\frac{c}{2(1+c)}(L-ZU)' X_{\gamma^1}(X'_{\gamma^1}X_{\gamma^1})^{-1}X'_{\gamma^1}(L-ZU)\Big]\pi_j,$$
$$P(\gamma_j = 0 \mid \gamma_{\{-j\}}, L, U) \propto (1+c)^{-\frac{\sum \gamma_j^0}{2}} \exp\Big[\frac{c}{2(1+c)}(L-ZU)' X_{\gamma^0}(X'_{\gamma^0}X_{\gamma^0})^{-1}X'_{\gamma^0}(L-ZU)\Big](1-\pi_j).$$

Dans le cas de notre application, il est donc plus rapide de faire 500, 1000 ou 2000 itérations d'un algorithme de Metropolis-Hastings que de générer plus de 19000 éléments suivant une Bernoulli.

Algorithme complet

Le nombre d'itérations est noté $b+m$, où b correspond à la période de burn-in et m à la période post-burn-in. L'algorithme va générer une séquence :

$$\gamma^{(1)}, \beta_\gamma^{(1)}, D^{(1)}, L^{(1)}, U^{(1)}, \ldots\ldots, \gamma^{(b+m)}, \beta_\gamma^{(b+m)}, D^{(b+m)}, L^{(b+m)}, U^{(b+m)}.$$

Algorithme du Metropolis-within-Gibbs modifié par la grouping technique

---

Nous commençons l'algorithme à partir de valeurs initiales $L^{(0)}, U^{(0)}, \beta^{(0)}, D^{(0)}, \gamma^{(0)}$.

Etapes de l'échantillonneur de Gibbs :

A l'itération $t+1$ $(t \geq 0)$:

1. **Etapes du Metropolis-Hastings** :
   Générer $\gamma^{(t+1)}$ suivant $f(\gamma \mid L^{(t)}, U^{(t)})$ (voir 2.14). Avec $\gamma^{(t)}$ comme valeur initiale et sachant $L^{(t)}$ et $U^{(t)}$, à chaque itération $i+1$:
   (a) Générer un candidat γ^*, en échangeant les valeurs de $r/2$ éléments de $\gamma^{(i)}$ valant 1 et $r/2$ éléments de $\gamma^{(i)}$ valant 0.
   (b) Prendre

$$\gamma^{(i+1)} = \begin{cases} \gamma^* & \text{avec probabilité} \quad \rho(\gamma^{(i)}, \gamma^*) \quad \text{voir (2.16)} \\ \gamma^{(i)} & \text{avec probabilité} \quad 1-\rho(\gamma^{(i)}, \gamma^*). \end{cases}$$

   $\gamma^{(t+1)}$ sera le $\gamma^{(k)}$ obtenu à la $k^{\text{ème}}$ itération de l'algorithme MH (k arbitrairement fixé).
2. Générer $\beta_\gamma^{(t+1)}$ suivant $f(\beta_\gamma \mid L^{(t)}, U^{(t)}, \gamma^{(t+1)})$ (voir (2.9)).
3. Générer $L^{(t+1)}$ suivant $f(L \mid Y, \beta_\gamma^{(t+1)}, \gamma^{(t+1)}, U^{(t)})$ (voir (2.8)).
4. Générer $U^{(t+1)}$ suivant $f(U \mid L^{(t+1)}, \beta_\gamma^{(t+1)}, \gamma^{(t+1)}, D^{(t)})$ (voir (2.10)).
5. Générer $D^{(t+1)}$ suivant $f(D \mid U^{(t+1)})$ (voir (2.11) ou (2.12)).

2.2.4 Sélection et prédiction

Les variables les plus pertinentes sont celles correspondant aux éléments de γ ayant les plus fortes probabilités a posteriori de valoir 1. Or les fréquences empiriques des éléments de γ sont des estimateurs consistants de leurs probabilités a posteriori. Par conséquent, les variables les plus pertinentes sont celles correspondant aux éléments de γ valant le plus souvent 1, et peuvent donc être facilement identifiées une fois que nous disposons de la séquence $\{\gamma^{(t)}, t = b+1, \ldots, b+m\}$.

En pratique, pour décider quelles variables doivent être conservées dans la sélection finale d'un run de l'algorithme, nous suggérons d'utiliser un boxplot du nombre d'itérations durant lesquelles les variables ont été sélectionnées. En général, un groupe de variables se distingue des autres et peut être sélectionné en fixant un seuil : si une variable a été sélectionnée lors d'un nombre d'itérations supérieur à ce seuil, elle est conservée dans la sélection finale.

Lorsqu'un groupe de variables pertinentes a été sélectionné, nous pouvons nous en servir pour ajuster un modèle probit mixte, et obtenir une règle de classification permettant de classifier de futures observations. Cependant, si lors de l'étape de sélection bayésienne nous obtenons plus de variables que nécessaires pour ajuster un modèle probit mixte pertinent, une seconde sélection doit être effectuée en ne retenant que celles permettant d'ajuster un modèle probit mixte satisfaisant. Cette seconde sélection est effectuée sur le jeu d'apprentissage à l'aide d'outils classiques de sélection de modèles : AIC, BIC, facteurs de Bayes, ... Puis le modèle probit mixte final est testé sur le jeu de validation. Il est également possible d'utiliser l'ensemble des variables sélectionnées lors de l'étape de sélection bayésienne dans d'autres méthodes de classification, comme les « Support Vector Machines » par exemple [91] (mais ils ne prennent pas en compte les effets aléatoires).

2.3 Résultats expérimentaux

2.3.1 Probesets expliquant le statut ER de patientes ayant un cancer du sein

Description des jeux de données

Trois jeux de données différents sont utilisés : un jeu de données privé de l'institut Paoli Calmettes et d'Ipsogen constitué de 151 échantillons de tumeurs de cancers du sein, et deux jeux de données publics disponibles sur le site NCBI GEO [62], de numéros d'accession GSE2109 et GSE5460, consitués respectivement de 310 et 124 échantillons.

Chaque jeu est corrigé pour le bruit de fond et normalisé par rapport à un jeu de référence, à l'aide de la procédure RMA d'Irizarry et al. [104]. Les trois jeux sont chacun séparés en un jeu d'apprentissage et un jeu de validation, contenant les mêmes proportions de patientes ER positives et ER négatives. Puis les trois jeux d'apprentissage sont fusionnés ensemble (497 patientes), de même que les trois jeux de validation (88 patientes).

Pour chaque patiente, les mesures d'expression de 54000 probesets de sa tumeur ont été obtenues à l'aide de puces Affymetrix. Deux filtres sont appliqués sur ces probesets : seuls ceux suffisament exprimés, pouvant être différenciés du bruit, et ceux ne pouvant pas être considérés comme invariants sont conservés. Il reste alors 19382 probesets.

L'objectif est de sélectionner des probesets pertinents pour expliquer le statut ER des patientes, en prenant en compte les différentes conditions expérimentales liées aux différentes provenances des jeux de données. Les effets fixes correspondent aux mesures d'expression des 19384 probesets, et nous sommes en présence d'un seul effet aléatoire ($K = 1$), qui correspond au jeu de données.

Choix de valeurs des hyper-paramètres pour l'algorithme

Nous devons choisir des valeurs pour les hyper-paramètres. L'influence de ces derniers est étudiée dans la partie 2.3.2.

- Suivant les recommandations de Smith et Kohn [202], le coefficient de sélection de variables c utilisé dans la distribution a priori de β_γ (2.5) est choisi égal à 50.
- Trente probesets sont sélectionnés à chaque itération de l'échantillonneur de Gibbs, lorsque le vecteur γ est généré ; et $r = 10$ d'entre eux sont changés à chaque itération de l'algorithme de Metropolis-Hastings (cinq 0 et cinq 1).
- Les trois jeux de données sont considérés indépendants car obtenus dans différents pays, par différentes équipes, en utilisant des équipements différents et des patientes différentes. La matrice de variance-covariance de l'effet aléatoire D est donc une matrice diagonale de dimension 3×3, avec σ^2 sur la diagonale. Connaissant l'ordre de grandeur de nos données, nous prenons une $\mathcal{IG}(2, 3)$ comme distribution a priori de σ_1^2.
- Pour l'algorithme complet 2.2.3, 60000 itérations sont effectuées, dont 30000 de burn-in. Pour les étapes de Metropolis-Hastings à l'intérieur de cet algorithme, 500 itérations sont effectuées ($k = 500$).

Résultats et prédictions

Modèle obtenu

En utilisant le résultat de l'algorithme, une sélection post-processing de variables est effectuée en conservant les gènes les plus sélectionnés lors des itérations post-burn-in, voir la figure 2.1. Quarante probesets sont sélectionnés au moins une fois lors des 30 000 itérations post-burn-in. Parmi ces quarante probesets, vingt-trois sont sélectionnés lors des 30 000 itérations, et trente le sont dans plus de 20 000 itérations, voir la table 2.1 ci-dessous. Les probesets sélectionnés dans plus de 20 000 itérations se distinguent vraiment des autres, ce sont donc ces trente probesets qui sont conservés.

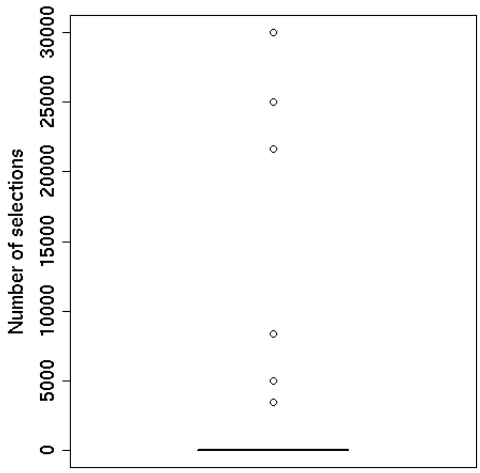

FIGURE 2.1 – Boxplot des nombres d'itérations lors desquelles les probesets sont sélectionnés, sur 30 000 itérations post-burn-in. Un point représente un probeset (ou plusieurs probesets ayant été sélectionnés un même nombre de fois). Par exemple, le point du haut représente 23 probsets qui ont été sélectionnés lors des 30 000 itérations.

Nombre de sélections	Nombres de probesets
30000	23
25043	2
21619	5
8381	2
4957	5
3424	3

TABLE 2.1 – Nombres d'itérations lors desquelles les probesets sont sélectionnés et nombres de probesets correspondants, pour les 30 000 itérations post-burn-in.

Une seconde sélection de probesets est effectuée afin d'ajuster un modèle probit mixte pertinent. Cette seconde sélection est effectuée à l'aide d'une sélection stepwise (AIC) sur le jeu d'apprentissage (497 patientes) et des classifications obtenues sur le jeu de validation (88 patientes). Cinq probesets sont finalement conservés, ayant les symboles Affymetrix `228241_at`, `205862_at`, `202376_at`, `216222_s_at` et `1568760_at`. Voir la table 2.2 pour les coefficients du modèle associés et les gènes auxquels appartiennent ces probesets.

Les effets aléatoires estimés pour ce modèle final sont : -0.284 pour le jeu de données provenant de l'Institut Paoli Calmettes, 0.199 pour le jeu GSE2109 et 0.087 pour le jeu GSE5460.

Prédictions sur le jeu de validation

A l'aide de ce modèle, nous effectuons des prédictions sur le statut ER des patientes du jeu de

Probeset	Gène	Coefficient	Pvalue
Intercept		-9.12074	1.92e-05
228241_at	AGR3	0.45046	1.12e-15
205862_at	GREB1	0.77639	4.18e-08
202376_at	SERPINA3	0.37965	0.000149
216222_s_at	MYO10	-0.63551	0.004967
1568760_at	MYH11	0.42742	0.050219

TABLE 2.2 – Probesets sélectionnés dans le modèle final et coefficients associés.

validation, de deux façons :

1. En utilisant la connaissance que nous avons du jeu de provenance de chaque patiente, soit en utilisant les effets aléatoires estimés du modèle.
2. Sans utiliser la connaissance que nous avons du jeu de provenance de chaque patiente, donc sans utiliser les effets aléatoires estimés du modèle. Cela correspond au cas réel d'une patiente provenant d'un centre inconnu, d'un jeu de données non utilisé pour contruire le modèle et dont on ne connait pas l'effet aléatoire associé.

Une patiente est prédite positive si sa probabilité d'être ER positive estimée par le modèle est supérieure à 0.5, et négative si celle-ci est inférieure à 0.5.
Sur le jeu de validation, les deux méthodes donnent exactement les mêmes prédictions. Ces prédictions sont bonnes, car nous obtenons une seule mauvaise classification pour 88 patientes, ce qui donne une spécificité de 1 et une sensibilité de 0.98, voir figure 2.2.

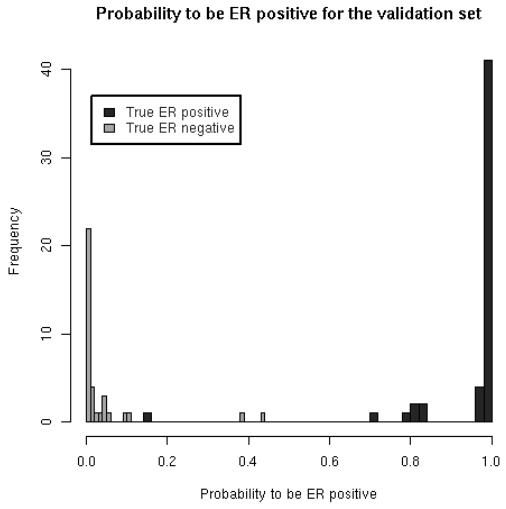

FIGURE 2.2 – Histogramme des probabilités d'être ER positive obtenue par le modèle final, pour les 88 patientes du jeu de validation.

En général, dans les études cliniques ou biomédicales, il est fréquent de définir une « zone grise » ou « zone d'incertitude » lorsqu'une classification binaire est utilisée. Aucune prédiction n'est alors fournise pour une patiente dont la probabilité d'être positive estimée par le modèle tombe dans cette zone. Ce type de zone doit son existence au principe de précaution. En effet, la classification d'un patient en tant que positif ou négatif est souvent d'une extrême importance car implique le choix d'un traitement.
En définissant une zone grise entre 10% et 90% de probabilité d'être positive, la prédiction erronée est éliminée, et dix patientes sont considérées comme indéterminées, voir la table 2.3.

	ER négative	ER positive
Prédiction négative	33	0
Prédiction positive	0	45
Indéterminée	3	7

TABLE 2.3 – Prédictions sur les 88 patientes du jeu de validation avec une zone grise.

Prédiction sur des jeux indépendants
Le modèle est ensuite testé sur deux jeux de données totalement indépendants des jeux utilisés pour estimer les paramètres du modèle. Nous utilisons deux jeux GEO : le jeu GSE6532 et le jeu GSE12763. Nous ne connaissons pas les effets aléatoires associés à ces jeux. Nous sommes donc dans le cas réel de patientes provenant de jeux inconnus.
La table 2.4 donne les prédictions obtenues pour ces deux jeux, avec ou sans zone grise. Les résultats sont ici aussi très bons, car sans zone grise nous obtenons 1 seule fausse prédiction parmi 29 pour le jeu GSE12763, et aucune parmi 86 pour le jeu GSE6532.

Sans zone grise	jeu GSE6532		jeu GSE12763	
	ER négative	ER positive	ER négative	ER positive
Prédiction négative	1	0	8	0
Prédiction positive	0	85	1	20
Avec zone grise	**GSE6532 dataset**		**GSE12763 dataset**	
	ER negative	ER positive	ER negative	ER positive
Prédiction négative	1	0	8	0
Prédiction positive	0	84	1	19
Indéterminée	0	1	0	1

TABLE 2.4 – Prédictions du modèle final pour deux jeux de données indépendants : le jeu GSE6532 et le jeu GSE12763.

2.3.2 Analyse de sensibilité et de stabilité

Nous étudions la sensibilité et la stabilité de cet algorithme de sélection de variables. Pour cela nous utilisons la mesure de « relative weighted consistency » de somol et Novovicova CW_{rel}

[203]. Cette mesure se base sur le nombre de variables en commun dans des sous-ensembles de variables sélectionnées sur différents runs d'un algorithme de sélection de variable. Elle prend des valeurs entre 0 et 1, où 0 signifie que les variables sont sélectionnées totalement aléatoirement, et où 1 indique qu'exactement les mêmes variables sont sélectionnées lors des différents runs, et donc que l'algorithme est très stable.

REMARQUE 2.3.1 *Il est aussi possible d'utiliser l'indice de stabilité de Kuncheva $\mathcal{S}$ [119]. Cependant cet index nécessite que les sous-ensembles de variables sélectionnées lors des différents runs soient de mêmes tailles, ce qui n'est pas toujours le cas. De plus, sur nos données $\mathcal{S}$ donne la même valeur que CW_{rel} au millième près dans le cas de sous-ensembles de variables sélectionnées de mêmes tailles.*

Pour des raisons de temps de calcul, nous utilisons dans cette analyse 4000 variables parmi les 19382 dont nous disposons. Le but ici n'étant pas de sélectionner les variables les plus pertinentes pour expliquer le statut ER, les 4000 variables retenues ne sont pas les plus susceptibles d'expliquer ce statut, elles ont été sélectionnées totalement au hasard.
Plusieurs runs de l'algorithme sont lancés, avec des modifications dans le jeu d'apprentissage ou les paramètres a priori, voir la table 2.5. Pour chaque run, un sous-ensemble de probesets de taille raisonnable est facilement identifiable. En effet, entre 2 et 10 probesets sont sélectionnés vraiment plus souvent que les autres, voir par exemple la figure 2.3 (dans la majorité des runs entre 2 et 4 probesets se détachent vraiment). Il n'est donc pas nécessaire d'effectuer une seconde sélection, comme dans la partie 2.3.1.

Stabilité

La stabilité est définie comme étant la sensibilité de l'algorithme à des modifications dans le jeu d'apprentissage.
Comme montré dans les trois premières lignes de la la table 2.5, l'algorithme a été lancé sur trois jeux d'apprentissage différents contenant chacun 497 puces parmi les 585 disponibles. Les mêmes valeurs d'hyper-paramètres ont été choisies pour les trois runs. Quatre probesets sont dans les sélections finales de deux runs parmi les trois, et la mesure de Somol et Novovicova vaut $CW_{rel} = 0.25$, ce qui semble raisonnable compte tenu du fait que le nombre de variables disponibles est très grand.

Sensibilité

Pour étudier la sensibilité de l'algorithme aux hyper-paramètres, il est lancé plusieurs fois sur le même jeu d'apprentissage mais en faisant varier les valeurs des hyper-paramètres :

- Différentes valeurs de c (lignes 4 à 7 de la table 2.5 : $c = 10, 50, 100, 1000$). Cinq probesets sont dans les sélections finales d'au moins deux runs parmi les quatres, et $CW_{rel} = 0.375$. Le probeset `215552_s_at` est conservé dans les quatre runs.

Run	Dataset	Valeur de c	A priori de σ^2	Nb probesets sélectionnés à chaque itération de l'algo	Nb probesets changés à chaque itération MH	Nb iter pour l'algo	burn-in pour l'algo	CW_{rel}
1	1							
2	2	50	$IG(2,3)$	15	6	12000	6000	
3	3							0.266
4		10						
5	1	50	$IG(2,3)$	15	6	12000	6000	
6		100						
7		1000						0.332
8			$IG(2,3)$					
9	1	50	$IG(1,1)$	15	6	12000	6000	
10			$IG(2,5)$					0.266
11				15	6			
12	1	50	$IG(2,3)$	5	2	12000	6000	
13				30	10			0.399
14	1	50	$IG(2,3)$	15	6	30000	15000	

TABLE 2.5 – Valeurs des hyper-paramètres utilisées dans les différents runs de l'analyse de stabilité et sensibilité, et indices de stabilité associés. Il y a toujours 500 itérations dans l'étape de Metropolis-Hastings.

- Différentes distributions a priori pour σ_1^2 (lignes 8 à 11 de la table 2.5) : une $\mathcal{IG}(2,3)$ paraissant raisonnable connaissant nos données, une $\mathcal{IG}(2,5)$ favorisant des valeurs un peu plus élevées, une $\mathcal{IG}(3,1)$ favorisant des valeurs un peu plus faibles, et une $\mathcal{IG}(1,1)$ qui est non-informative sans avoir pour autant des paramètres trop faibles, ce qui pourrait poser problème (voir Gelman [71]). Quatre probesets sont dans les sélections finales d'au moins deux runs parmi les quatre, et $CW_{rel} = 0.292$.

- Différents nombres de probesets devant être sélectionnés à chaque itération de l'algorithme et devant être changés à chaque itération de l'étape de Metropolis-Hastings (lignes 12 à 14 de la table 2.5) : 15 sélectionnés et 6 changés, 5 sélectionnés et 2 changés, ou 30 sélectionnés et 10 changés. Quatre probesets sont dans les sélections finales d'au moins deux runs parmi les trois, et $CW_{rel} = 0.5$. De plus nous notons que ces hyperparamètres ne paraissent pas modifier le nombre de probesets se détachant des autres. Le probeset `209603_at` est dans les sélections finales des trois runs.

- Enfin, l'algorithme est lancé pour 30 000 itérations dont 15 000 de burn-in, pour voir si les résultats obtenus avec 30 000 itérations sont différents de ceux obtenus précédemment avec 12 000. Les cinq probesets dans la sélection finale de ce run avec 30 000 itérations sont également dans les sélections finales d'au moins un des runs précédents avec 12000

itérations.

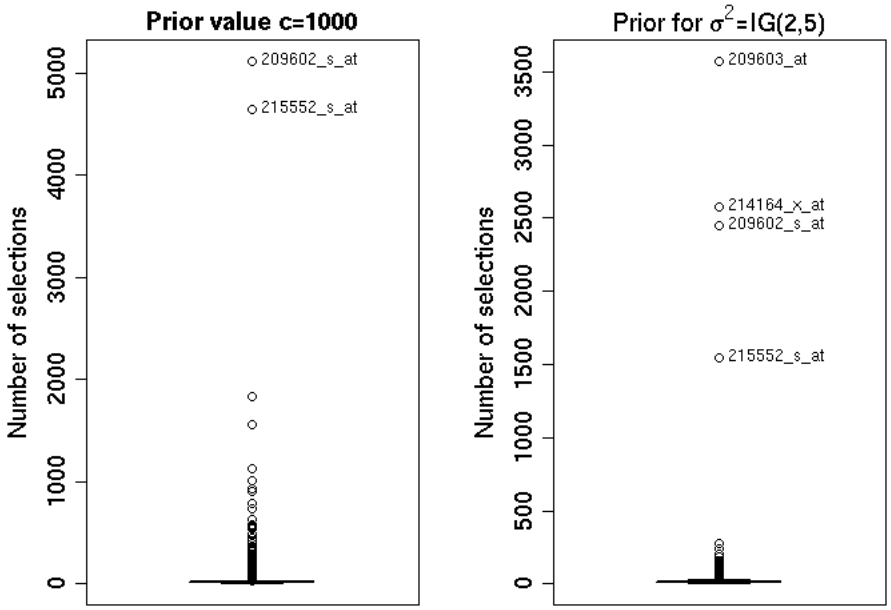

FIGURE 2.3 – Boxplots des nombres d'itérations lors desquelles les probesets sont sélectionnés, sur 6 000 itérations post-burn-in, pour deux runs de l'analyse de sensibilité. Le boxplot de gauche correspond au run où $c = 1000$: les deux probesets sélectionnés dans plus de 4000 itérations se détachent des autres, ce sont donc ces deux probesets qui sont sélectionnés. Le boxplot de droite correspond au run où $\sigma_1^2 \sim \mathcal{IG}(2,5)$: les quatre probesets sélectionnés dans plus de 1500 itérations se détachent des autres, ce sont donc ces quatre probesets qui sont sélectionnés.

Trois probesets en particulier sont dans les sélections finales de nombreux runs : les probesets `215552_s_at`, `209603_at` et `209602_s_at` sont dans les sélections finales de 12, 12 et 6 runs parmi 15 respectivement. En utilisant les sous-ensembles de variables sélectionnées lors des 15 runs, $CW_{rel} = 0.398$. Cette mesure est plutôt satisfaisante compte tenu du fait que la sélection s'effectue parmi 4000 probesets. Ces 4000 probesets ont été sélectionnés au hasard et ne sont donc pas forcément pertinents pour expliquer le statut ER. Nous nous attendions donc à une instabilité de l'algorithme dans l'hypothèse ou aucun des 4000 probesets ne permettrait d'expliquer la variable d'intérêt. De plus, des probesets différents peuvent représenter un même gène, et différents gènes peuvent être impliqués dans les mêmes processus biologiques. Par conséquent il est probable que les sous-ensemble de probesets conservés soient biologiquement plus similaires qu'il n'y paraît en ne comparant que les noms des probesets, et CW_{rel} est certainement sous-estimé. Par exemple, les probesets `209603_at` and `209602_s_at` mentionnés ci-dessus représentent tout deux le gène GATA3. Enfin, nous n'avons pas observé que le choix d'un hyper-paramètre en particulier induit plus de sensibilité que les autres hyper-paramètres.

2.3.3 Comparaison avec d'autres méthodes

Nous voulons comparer les performances de notre méthode avec celles d'autres méthodes ne prenant pas en compte les effets aléatoires : nous considérons la méthode de Lee et al. [131], ainsi que la méthode RFE-SVM de Guyon et al. [91] (Support-Vector-Machine with Recursive Feature-Elimination), dans le cas de noyaux linéaires et non-linéaires. Pour cela nous utilisons des données simulées. Nous générons 200 observations de 1000 variables suivant une uniforme $\mathcal{U}_{[-5,5]}$, et supposons que 5 variables et un effet aléatoire de taille 4 expliquent le vecteur de variables binaires Y par un modèle probit mixte :

$$\begin{aligned} p_i &= \Phi(X_i'\beta + Z_i'U), \qquad i = 1, \ldots, 200. \\ Y_i &\sim \mathcal{B}(p_i), \qquad i = 1, \ldots, 200. \end{aligned}$$

Nous prenons $\beta_\gamma = (-1, -1, 1, 1, 2)$ et supposons que 50 observations sont associées à chaque modalité de l'effet aléatoire. Différentes valeurs de U sont utilisées : $U1 = (0, 0, 0, 0)$, $U2 = (-3, -2, 2, 3)$, $U3 = (-5, -3, 3, 5)$, $U4 = (-10, -5, 5, 10)$ et $U5 = (-30, -10, 10, 30)$. Ce jeu de données de 200 observations est séparé en un jeu d'apprentissage et un jeu de validation, chacun d'eux étant de taille 100 et ayant 25 observations associées à chaque modalité de l'effet aléatoire. Pour notre méthode et la méthode de Lee et al. [131] nous prenons $c = 50$, 5 probesets sont sélectionnés à chaque itération de l'algorithme, $r = 2$ d'entre eux sont changés à chaque itération du Metropolis-Hastings (un 0 et un 1), D est une matrice diagonale 3×3 avec σ^2 sur sa diagonale, la distribution a priori de σ^2 est une $\mathcal{IG}(1, 1)$, 500 itérations sont effectuées lors de chaque étape de Metropolis-Hastings, et un total de 3000 ou 5000 itérations sont effectuées pour l'algorithme complet (1500 itérations de burn-in).
Pour notre méthode et celle de Lee et al. [131], les variables se détachant le plus des autres (utilisation de boxplots) sont conservées et utilisées pour effectuer des prédictions sur le jeu de validation. La méthode RFE-SVM [91] donne directement des ensembles de variables à conserver et les modèles associés, qui sont utilisés pour les prédictions. Les résultats obtenus sont dans la table 2.6.

Quand il n'y a pas d'effet aléatoire ou quand les effets aléatoires sont faibles, notre méthode est comparable à celle de Lee et al., et les résultats de ces deux méthodes sont comparables ou meilleurs que ceux obtenus par RFE-SVM. Mais quand les effets aléatoires sont élevés, en particulier dans les cas de $U4$ et $U5$, notre méthode obtient de meilleurs résultats que la méthode de Lee et al. et que RFE-SVM.

Méthode	Notre méthode		méthode de Lee et al.		RFE-SVM	
Effet aléatoire U	3000 it.	5000 it.	3000 it.	5000 it.	linéaire	non linéaire
$U1 = (0, 0, 0, 0)$	17	26	19	22	25	23
$U2 = (-3, -2, 2, 3)$	19	21	19	19	20	26
$U3 = (-5, -3, 3, 5)$	21	23	24	24	25	26
$U4 = (-10, -5, 5, 10)$	19	19	35	35	29	31
$U5 = (-30, -10, 10, 30)$	14	11	44	44	52	56

TABLE 2.6 – Nombres de mauvaises classifications sur le jeu de validation, pour différentes méthodes et différents effets aléatoires.

Chapitre 3

Cas de matrices de covariance singulières

3.1 Motivation

La méthode de sélection de variables pour un modèle probit mixte proposée dans le chapitre 2 a une limite : elle utilise l'a priori de Zellner [238] pour β_γ le vecteur des coefficients des effets fixes, et la matrice de covariance $X'_\gamma X_\gamma$ utilisée dans cet a priori doit nécessairement être inversible (cet inverse intervient dans l'expression de V_γ utilisée dans (2.9), ainsi que dans les formules (2.13) à (2.16)). Or il y a deux cas pour lesquels cette matrice est singulière :

1. Lorsque le nombre de variables sélectionnées dépasse le nombre d'observations.
2. Lorsque des variables sont combinaisons linéaires d'autres variables. Il y aura alors au moins un γ pour lequel $X'_\gamma X_\gamma$ sera singulière. En pratique, même si $X'_\gamma X_\gamma$ est théoriquement inversible, des variables peuvent être fortement corrélées entre elles et $X'_\gamma X_\gamma$ peut être computationnellement singulière.

Dans ces deux cas, l'a priori classique de Zellner [238] ne peut être utilisé. Comme nous l'avons indiqué dans l'introduction de cette partie (chapitre 1), différents auteurs se sont penchés sur le problème, notamment pour le premier cas.

En utilisant la méthode développée dans le chapitre 2, l'équipe biostatistique d'Ipsogen a rencontré le second cas en pratique, avec des variables qui donnaient une matrice $X'_\gamma X_\gamma$ computationnellement singulière. De plus, une solution pour le premier cas nous était demandée par un référé, qui nous renvoyait à Yang et Song [231]. Nous avons alors étudié l'article de Yang et Song, qui proposent de remplacer $(X'_\gamma X_\gamma)^{-1}$ dans la matrice de covariance de l'a priori de β_γ, par l'inverse généralisée de Moore-Penrose $(X'_\gamma X_\gamma)^+$. L'a priori obtenu est donc un a priori « generalized singular g-prior (gsg-prior) ». West [230] est le premier à avoir considéré cet a priori, et des expressions de la densité d'une gaussienne singulière ont été données par Srivastava [207],

ainsi que par Díaz-García et Ramos-Quiroga [58] parmi d'autres.

REMARQUE 3.1.1 *Ne pas confondre l'a priori « generalized singular g-prior (gsg-prior) » avec l'a priori « generalized g-prior » utilisé dans le cas de modèles linéaires généralisés (voir Sabanés-Bové et Held [192] par exemple).*

L'a priori gsg-prior est le suivant :

$$\beta_\gamma \mid \gamma \sim \mathcal{N}_{d_\gamma}(0, c(X_\gamma' X_\gamma)^+), \qquad \text{avec} \qquad d_\gamma = \sum_{j=1}^{p} \gamma_j.$$

La densité a priori de β_γ est alors proportionnelle à

$$(2\pi)^{-\frac{r_\gamma}{2}} \prod_{i=1}^{r_\gamma} \lambda_i^{-\frac{1}{2}} \exp\Big[-\frac{1}{2}\beta_\gamma'(c(X_\gamma' X_\gamma)^+)^+\beta_\gamma\Big] = (2\pi)^{-\frac{r_\gamma}{2}} \prod_{i=1}^{r_\gamma} \lambda_i^{-\frac{1}{2}} \exp\Big[-\frac{1}{2c}\beta_\gamma'(X_\gamma' X_\gamma)\beta_\gamma\Big],$$

par rapport à la mesure de Lebesgue sur l'hyperplan $\Lambda_2\beta_\gamma = 0$, où les λ_i sont les valeurs propres non nulles de $c(X_\gamma' X_\gamma)^+$, r_γ le rang de $(X_\gamma' X_\gamma)^+$, $\Lambda' = (\Lambda_1', \Lambda_2')$ avec Λ_1' une matrice $d_\gamma \times r_\gamma$ telle que $\Lambda_1\Lambda_1' = I_r$ et

$$c(X_\gamma' X_\gamma)^+ = \Lambda' \begin{pmatrix} D_\lambda & 0 \\ 0 & 0 \end{pmatrix} \Lambda' = \Lambda_1' D_\lambda \Lambda_1, \qquad D_\lambda = diag(\lambda_1, \ldots, \lambda_r).$$

Il est ensuite nécessaire d'intégrer la densité jointe a posteriori en β_γ, car β_γ est un paramètre de nuisance (voir la partie 2.2.2), et afin d'éviter d'avoir à générer β_γ suivant une loi normale dégénérée dans le cas où justement $X_\gamma' X_\gamma$ est singulière. Dans leur « supplementary material », Yang et Song [232] détaillent la manière dont cette intégration est effectuée. Ils utilisent notamment l'inverse de la matrice A suivante :

$$A = X_\gamma'\big[\big(I_n + h\mathbb{1}\mathbb{1}'\big)^{-1} + c^{-1}I_n\big]X_\gamma,$$

dans leur équation (A 7) donnée par :

$$-\frac{\beta_\gamma' A\beta_\gamma - 2\beta_\gamma' B}{2} - \frac{Z'\big(I_n + h\mathbb{1}\mathbb{1}'\big)^{-1}Z}{2} = -\frac{(\beta_\gamma - A^{-1}B)'A(\beta_\gamma - A^{-1}B)}{2} - \frac{Z'\big(I_n + h\mathbb{1}\mathbb{1}'\big)^{-1}Z - B'A^{-1}B}{2}.$$

Cependant, A n'est pas inversible si $X_\gamma' X_\gamma$ est singulière. Une idée pourrait être d'utiliser A^+ au lieu de A^{-1}, et de retrouver une équation similaire à (A 7). Mais même en utilisant A^+, nous ne pouvons pas retrouver une densité gaussienne pour β_γ puisque nous avons :

$$\beta_\gamma' A\beta_\gamma - 2\beta_\gamma' B \neq (\beta_\gamma - A^+B)'A(\beta_\gamma - A^+B) - B'A^+B.$$

Il semble donc difficile (voire impossible) d'intégrer la densité jointe a posteriori en β_γ, comme nous l'avons commenté dans [15].

REMARQUE 3.1.2 *Les formules précédentes sont légèrement différentes de ce que nous obtiendrons avec notre modèle, car nous n'introduisons pas d'intercept comme Yang et Song (discuté dans la partie 4.1). Leur matrice A correspond en fait à notre matrice V_γ^{-1} (voir 2.2.1), et nous aurions le même problème qu'eux si nous supposions un a priori gsg-prior pour β_γ.*

Notons que récemment Kwon et al. [122] ont utilisé la même approche, en prenant un a priori gsg-prior pour β_γ, et ils ont intégré la densité jointe a posteriori en β_γ. Cependant, en pratique ils n'avaient que des prédicteurs très corrélés, et non pas des prédicteurs qui sont combinaisons linéaires d'autres prédicteurs. Ils ne se sont donc jamais placés dans le cas de $X'_\gamma X_\gamma$ singulière. De même, Yang et Song [231] ne se sont jamais placés en pratique dans le cas où $n < d_\gamma$ et $X'_\gamma X_\gamma$ singulière. Or si $X'_\gamma X_\gamma$ est inversible, nous avons $(X'_\gamma X_\gamma)^+ = (X'_\gamma X_\gamma)^{-1}$.

La proposition de Yang et Song [231] n'étant pas satisfaisante dans le cas où la matrice $X'_\gamma X_\gamma$ est singulière, nous avons eu l'idée de modifier l'a priori classique de Zellner [238] en y introduisant un paramètre ridge. Nous développons cet a priori dans ce chapitre, ainsi qu'une manière de choisir les hyper-paramètres associés. Nous avons vu indépendamment qu'une version proche de notre a priori avait déjà été proposée par Gupta et Ibrahim [89]. Cependant, ils n'ont pas considéré le choix des hyperparamètres, et n'ont pas étudié le cas de variables combinaisons linéaires d'autres. Dans nos exemples, nous étudions donc principalement ce cas.

3.2 Introduction d'un paramètre ridge

D'une manière générale, notre approche peut s'appliquer à un modèle linéaire généralisé mixte, avec :

$$g(\mathbb{E}(Y \mid \beta, U)) = X\beta + ZU,$$

où g représente la fonction de lien du modèle. Cependant, pour ne pas ajouter de nouvelles notations, nous nous plaçons dans le même modèle qu'au chapitre 3 (voir la partie 2.1.1).

3.2.1 Distribution a priori avec un paramètre ridge

Nous utilisons les mêmes distribution a priori qu'au chapitre 3 (partie 2.1.2), excepté pour le paramètre β_γ, pour lequel nous supposons l'a priori suivant, avec λ un paramètre ridge strictement positif :

$$\beta_\gamma \mid \gamma \sim \mathcal{N}_{d_\gamma}\Big(0, (\frac{1}{c}X'_\gamma X_\gamma + \lambda I)^{-1}\Big), \qquad \text{avec} \qquad d_\gamma = \sum_{j=1}^{p} \gamma_j. \tag{3.1}$$

Ainsi, nous remplaçons la matrice de covariance $c(X'_\gamma X_\gamma)^{-1}$ utilisée dans l'a priori de Zellner [238] par $\Sigma_\gamma(\lambda) = (\frac{1}{c}X'_\gamma X_\gamma + \lambda I)^{-1}$ qui est toujours de rang plein. Nous obtenons donc une forme modifiée de la g-prior, qui est un compromis entre l'indépendance et l'instabilité. En effet, pour des valeurs élevées de λ ou c, c'est le terme λI qui prédomine et les composants de β_γ peuvent être considérés comme indépendants entre eux. Par contre, lorsque λ ou c tendent vers 0, c'est le terme $\frac{1}{c}X'_\gamma X_\gamma$ qui prédomine, et l'inverse de $\frac{1}{c}X'_\gamma X_\gamma + \lambda I$ sera instable si $X'_\gamma X_\gamma$ est singulière. Dans ce cas où λ ou c sont petits, la distribution (3.1) se rapproche de la distribution classique de Zellner. A titre de comparaison, l'a priori proposé par Gupta et Ibrahim [89] utilise $\Sigma_\gamma(\lambda) = c(X'_\gamma X_\gamma + \lambda I)^{-1}$ (avec un terme σ^2 en plus car ils sont dans un modèle linéaire).

3.2.2 Choix des hyper-paramètres c et λ

Notons $\Sigma_\gamma(0) = c_0(X'_\gamma X_\gamma)^{-1}$ la matrice de covariance utilisée dans l'a priori classique de Zellner [238], comme dans le chapitre 2. Une propriété intéressante de l'a priori de Zellner est qu'il reflète la structure de covariance (et donc de corrélation) des données. L'introduction du paramètre ridge nous empêche de préserver totalement cette structure. Cependant, il est possible de conserver la variance totale expliquée par les covariables. En effet, celle-ci correspond, à une constante de normalisation et une relocalisation près, à la trace de $\Sigma_\gamma(0)^{-1}$. Nous utilisons donc la contrainte suivante pour fixer λ et c :

$$tr\Big(\Sigma_\gamma(0)^{-1}\Big) = tr\Big(\Sigma_\gamma(\lambda)^{-1}\Big),$$

soit

$$\frac{1}{c_0}tr(X'_\gamma X_\gamma) = \frac{1}{c}tr(X'_\gamma X_\gamma) + \lambda p,$$

ce qui donne

$$c \;=\; c_0\Big[1 + \frac{\lambda p c_0}{tr(X'_\gamma X_\gamma) - \lambda p c_0}\Big], \qquad \text{avec} \quad tr(X'_\gamma X_\gamma) \neq \lambda p c_0.$$

De façon à simplifier l'expression de c et à prendre en compte le nombre total de prédicteurs p, nous choisissons $\lambda = 1/p$, et obtenons :

$$c = c_0\Big[1 + \frac{c_0}{tr(X'_\gamma X_\gamma) - c_0}\Big], \qquad \text{avec} \quad tr(X'_\gamma X_\gamma) \neq c_0.$$

Le vecteur γ étant susceptible de changer à chaque itération de l'algorithme, nous préférons utiliser la matrice de design complète $X'X$ au lieu de $X'_\gamma X_\gamma$, ce qui donne :

$$c = c_0\Big[1 + \frac{c_0}{tr(X'X) - c_0}\Big], \qquad \text{avec} \quad tr(X'_\gamma X_\gamma) \neq c_0. \tag{3.2}$$

En pratique, suivant cette méthodologie l'utilisateur n'a qu'à choisir le paramètre c_0 qu'il utiliserait dans l'a priori classique de Zellner [238]. Les paramètres λ et c sont ensuite obtenus par $1/p$ et (3.2). Nous étudierons l'influence des choix de ces hyper-paramètres dans la partie 3.5.3.

REMARQUE 3.2.1 *Le choix $\lambda = 1/p$ a l'avantage d'être automatique et de prendre en compte le nombre de variables. Mais si p est très grand, il peut mener à $\Sigma_\gamma(\lambda)$ quasi-singulière. Dans ce cas, nous pouvon utiliser un seuil ϵ et prendre $\lambda = \max(1/p, \epsilon)$.*

3.3 L'échantillonneur bayésien

Tout comme dans le chapitre 2, nous utilisons un algorithme de Metropolis-within-Gibbs couplé à la technique de « grouping ».

Les distributions conditionnelles des différents paramètres du modèle sont similaires aux distributions conditionnelles obtenues dans la partie 2.2.1, hormis celles de β_γ et γ. Pour le paramètre β_γ, nous obtenons la distribution conditionnelle complète suivante :

$$\beta_\gamma \mid L, U, \gamma \quad \sim \quad \mathcal{N}_{d_\gamma}(T_\gamma X_\gamma'(L - ZU), T_\gamma), \tag{3.3}$$

avec

$$T_\gamma = (\frac{1+c}{c} X_\gamma' X_\gamma + \lambda I)^{-1} \qquad \text{et} \qquad d_\gamma = \sum_{j=1}^{p} \gamma_j. \tag{3.4}$$

En ce qui concerne le paramètre γ, sa densité conditionnelle complète intégrée en β_γ est donnée par :

$$\begin{aligned} f(\gamma \mid L, U) \quad \propto \quad & \frac{\mid T_\gamma \mid^{\frac{1}{2}}}{\mid \Sigma_\gamma \mid^{\frac{1}{2}}} \exp\Big[-\frac{1}{2}\Big\{ (L - ZU)'(I - X_\gamma T_\gamma X_\gamma')(L - ZU) \Big] \\ & \times \prod_{j=1}^{p} \pi_j^{\gamma_j} (1 - \pi_j)^{1-\gamma_j}. \end{aligned} \tag{3.5}$$

REMARQUE 3.3.1 *Le paramètre ridge λ intervient à travers le ratio $R = \frac{|T_\gamma|}{|\Sigma_\gamma|}^{\frac{1}{2}}$. Nous remarquons que :*

$$\begin{cases} \text{si } c \to \infty, & R \to |\frac{1}{\lambda} X_\gamma' X_\gamma + \lambda I|^{-1/2}, \\ \text{si } c \to 0, & R \to 1. \end{cases}$$

$$\begin{cases} \text{si } \lambda \to \infty, & R \to 1, \\ \text{si } \lambda \to 0, & R \to \left(\frac{1}{1+c}\right)^{\frac{d_\gamma}{2}}. \end{cases}$$

Algorithme de Metropolis-Hastings pour générer γ

L'algorithme de Metropolis-Hastings que nous utilisons pour générer γ suivant (3.5) est identique à celui décrit dans la partie 2.2.3, sauf sur deux points :

1. La probabilité d'acceptation est la suivante (à comparer avec (2.15)) :

$$\rho(\gamma^{(i)},\gamma^*) = \min\left\{\left(\frac{|T_{\gamma^*}\Sigma_{\gamma^{(i)}}|}{|\Sigma_{\gamma^*}T_{\gamma^{(i)}}|}\right)^{\frac{1}{2}}\exp\Big[\frac{1}{2}(L-ZU)'\Big(X_{\gamma^*}T_{\gamma^*}X'_{\gamma^*}-X_{\gamma^{(i)}}T_{\gamma^{(i)}}X'_{\gamma^{(i)}}\Big)(L-ZU)\Big]\times\Big(\frac{\pi}{1-\pi}\Big)^{\sum_1^p(\gamma_j^*-\gamma_j^{(i)})},1\right\} \quad \text{si } \forall j\ \pi_j=\pi. \tag{3.6}$$

   Remarque 3.3.2 *L'influence du coefficient de sélection de variables c apparaît à travers le ratio $Q=\frac{|T_{\gamma^*}\Sigma_{\gamma^{(i)}}|}{|\Sigma_{\gamma^*}T_{\gamma^{(i)}}|}$ qui satisfait :*

$$\begin{cases}\text{si } c\to\infty, & Q\to|X'_{\gamma^*}X_{\gamma^*}+\lambda I|^{-1}\times|X'_{\gamma^{(i)}}X_{\gamma^{(i)}}+\lambda I|,\\ \text{si } c\to 0, & Q\to 1.\end{cases}$$

$$\begin{cases}\text{si } \lambda\to\infty, & Q\to 1,\\ \text{si } \lambda\to 0, & Q\to 1.\end{cases}$$

2. Contrairement au chapitre 2, nous ne fixons plus le nombre de variables à sélectionner à chaque itération. Nous relâchons cette hypothèse, et nous pouvons avoir un nombre de variables sélectionnées qui dépasse le nombre d'observations.
   A l'itération $i+1$, le vecteur candidat γ^* proposé à partir du vecteur $\gamma^{(i)}$ correspond simplement à $\gamma^{(i)}$ pour lequel r éléments ont été choisis au hasard et modifiés.

Algorithme complet

L'algorithme Metropolis-within-Gibbs modifié par la grouping technique utilisé est similaire à celui de la partie 2.2.3, hormis les deux modifications présentées ci-dessus pour l'algorithme de Metropolis-Hastings, et les distributions conditionnelles de β_γ et γ, pour lesquels nous utilisons (3.5) et (3.3).

3.4 Sur le Lasso bayésien

Une approche bayésienne concurrente de l'approche SSVS pour la sélection de variables est le Lasso bayésien (voir Park et Casella [175] et Hans [93]), qui est inspiré du Lasso fréquentiste (Tibshirani [218]). Dans l'optique de comparer ces deux approches, nous avons adapté le Lasso bayésien aux modèles probit mixtes. Le Lasso bayésien a déjà été utilisé pour des modèles probit

(Bae et Mallick [14]), et dans le cadre de modèles mixtes (Legarra et al. [132]). En combinant ces deux approches, nous considérons les distributions a priori suivantes :

- Chaque $\beta_j, j = 1, \ldots, p$ est supposé suivre un a priori de Laplace : $\mathcal{L}aplace(0, 1/\sqrt{\delta})$. Cette distribution de Laplace peut s'exprimer comme un mélange d'échelle de distributions normales avec des variances indépendantes et de distributions exponentielles, voir Andrews et Mallows [3]. Cet a priori est alors équivalent à $\beta_j \mid \lambda_j \sim \mathcal{N}(0, \lambda_j)$ et $\lambda_j \sim \mathcal{E}xpo(\delta/2)$. En écrivant $\Lambda = diag(\lambda_1, \ldots, \lambda_p)$, nous avons $\beta \mid \Lambda \sim \mathcal{N}_p(0, \Lambda)$.
- Concernant le vecteur des effets aléatoires U et la matrice de variance-covariance D, nous utilisons les mêmes a priori classiques que ceux de l'approche SSVS, voir la partie 2.1.2.
- Suivant Park et Casella [175], une distribution a priori est supposée pour l'hyper-paramètre δ : $\delta \sim \mathcal{G}amma(e, f)$, f étant le paramètre d'échelle. Dans les résultats expérimentaux de la partie suivante, nous avons constaté comme Yi et Xu [235] et Li et al. [135] que les résultats obtenus ne sont pas trop sensibles aux hyper-paramètres e et f, du moment qu'ils sont choisis assez petits pour que la distribution de δ soit suffisamment non-informative. En pratique nous avons pris $e = f = 1$, mais des résultats similaires ont été obtenus avec $e = f = 10$ par exemple.

Les estimateurs du Lasso bayésien sont ensuite obtenus en utilisant un échantillonneur de Gibbs avec les distributions a posteriori suivantes :

- Pour L, U, D et les $\sigma_l^2, l = 1, \ldots, K$, les distributions conditionnelles complètes sont les mêmes que celles obtenues dans l'approche SSVS, voir (2.8), (2.10), (2.11) et (2.12).
- Pour le vecteur des coefficients β, la distribution conditionnelle est :

$$\beta | L, U, \Lambda \sim \mathcal{N}_p(V_\Lambda \mathbf{X}^T (L - ZU), V_\Lambda) \quad \text{avec} \quad V_\Lambda = \left[\mathbf{X}^T\mathbf{X} + \Lambda^{-1}\right]^{-1}. \tag{3.7}$$

- Les distributions conditionnelles des $1/\lambda_j, j = 1, \ldots, p$ sont des inverses Gaussiennes :

$$1/\lambda_j \mid \beta \sim \mathcal{IG}auss\Big(\frac{\sqrt{\delta}}{\beta_j}, \delta\Big). \tag{3.8}$$

- Concernant le paramètre Lasso δ, on obtient une distribution Gamma :

$$\delta \mid \Lambda \sim \mathcal{G}amma\Big(p + e, \Big(\frac{\sum \lambda_j}{2} + \frac{1}{f}\Big)^{-1}\Big). \tag{3.9}$$

Intreprétation des résultats

Le Lasso bayésien nous permet d'obtenir des estimateurs a posteriori pour les β_js et les λ_js, et les variables peuvent être sélectionnées de trois façons différentes :

1. Nous pouvons sélectionner les variables associées à une valeur absolue $|\beta_j|$ plus grande qu'un certain seuil, comme Yi et Xu [235] ou Li et al. [135] par exemple.

2. Bae et Mallick [14] considèrent que les variables associées à des β_js avec de faibles variances a posteriori n'ont pas d'effet et doivent être exclues du modèle. Ils proposent alors de ne conserver que les variables associées à des λ_js élevés.

3. Enfin, les résultats du Lasso permettent d'obtenir des intervalles de crédibilité (IC) a posteriori pour les β_js. Nous pouvons donc choisir de sélectionner les variables associées à des β_js ayant un IC a posteriori ne comprenant pas 0, voir par exemple Kyung et al.[123].

3.5 Résultats expérimentaux

3.5.1 Jeu de données simulées

Un jeu de données de 200 observations et 300 variables est simulé. Sur les 300 variables, 280 sont générées d'après une uniforme sur $[-5, 5]$ et notées $V1, \ldots, V280$. Puis 10 variables sont construites colinéaires aux 10 premières variables avec un facteur 2 ($V281, \ldots, V290$), 1 variable est construite comme une combinaison linéaire des variables 1 et 2 ($V291 = V1 + V2$), 1 autre est une combinaison linéaire des variables 3 et 4 ($V292 = V3 - V4$), et 8 autres sont des combinaisons linéaires des variables 5 à 12 et 13 à 20 ($V293 = V5 + V13$ par exemple). Les observations binaires Y sont obtenues par un modèle probit mixte utilisant seulement les 5 premières variables et un effet aléatoire de taille 4, ainsi elles peuvent être supposées provenir de quatre jeux de données différents. Le vecteur des coefficients associés à ces cinq variables est $\beta = (1, -1, 2, -2, 3)$. Les 100 premières observations constituent notre jeu d'apprentissage, et les 100 dernières le jeu de validation. Dans les jeux d'apprentissage et de validation, 25 observations sont supposées associées à chaque composant de l'effet aléatoire, dont le vecteur des coefficients est $U = (-3, -2, 2, 3)$. Nous sommes dans le cas d'un seul effet aléatoire de taille 4, et les quatre composants sont supposés indépendants. Nous avons donc $D = \sigma^2 I_4$.

L'objectif est d'étudier le comportement de la méthode de sélection de variables avec paramètre ridge lorsque des variables sont combinaisons linéaires d'autres, et de le comparer au cas où aucune variable n'est combinaison linéaire d'autres. Par conséquent nous lançons dix fois l'algorithme en utilisant seulement les 280 premières variables de la matrice, puis dix fois en utilisant les 300 variables. Dans les deux cas et pour chacun des runs nous utilisons les 100 premières observations et les mêmes hyper-paramètres : 5 variables sont initialement sélectionnées, 1 élément de γ est proposé d'être changé à chaque itération de l'algorithme de Metropolis-Hastings, la distribution a priori de σ^2 est une $\mathcal{IG}(1, 1)$, $\pi_j = 5/280$ pour tout j quand les 280 premières variables sont utilisées, $\pi_j = 5/300$ pour tout j quand les 300 variables sont utilisées, 5000 itérations sont lancées dont 1000 de burn-in, et chaque étape de Metropolis-Hastings est constituée de 500 itérations.

Dans le cas où l'a priori classique de Zellner peut être utilisé, un choix standard pour le coefficient de sélection de variables est $c_0 = 50$ (voir Smith et Kohn [202] par exemple). Nous

prenons donc $c_0 = 50$, et fixons les paramètres c et λ comme expliqué dans la partie 3.2.2 (voir (3.2)) pour le calcul de c. Cela donne $\lambda = 1/280$ et $c = 50.01075$ dans le cas où X contient les 280 premières variables, et $\lambda = 1/300$ et $c = 50.00885$ dans le cas où X contient les 300 variables.

Variables sélectionnées

Pour chacun des vingt runs, une sélection des variables est effectuée, en utilisant des boxplots comme expliqué dans la partie 2.2.4. Les figures 3.1 et 3.2 montrent des boxplots associés à des runs avec 280 et 300 variables respectivement.

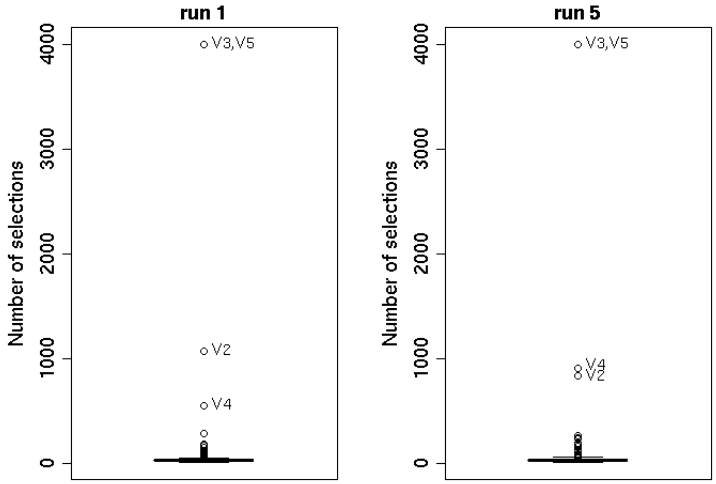

FIGURE 3.1 – Boxplots des nombres d'itérations lors desquelles les variables sont sélectionnées, exemples de deux runs avec 280 variables. Le boxplot de gauche correspond au run 1 et celui de droite au run 5. Pour les deux runs il y a un écart entre $V2, V3, V4$ et $V5$ sélectionnées plus de 500 fois et les autres, ce sont donc ces quatre variables qui sont sélectionnées.

La table 3.1 donne le nombre de fois qu'une variable ou qu'une combinaison linéaire de variables est sélectionnée sur les vingt runs. En ce qui concerne les runs avec 280 variables, 3 des 5 « vraies » variables sont sélectionnées par presque tous les runs, et la variable $V4$ est sélectionnée par la moitié des runs. Par contre, la variable $V1$ n'est sélectionnée par aucun des dix runs. En ce qui concerne les runs avec 300 variables, les variables $V1, V2, V3$ et $V5$ sont sélectionnées dans la plupart des runs, directement ou indirectement par l'intermédiaire de variables colinéaires. Contrairement aux runs avec 280 variables, les variables $V1$ et $V281$ sont sélectionnées par tous les runs, tandis que $V4$ et $V284$ ne le sont jamais. Cependant, la variable $V292$ est très souvent sélectionnée, et elle est une combinaison linéaire de $V3$ et $V4$, donc $V4$ est indirectement prise en compte. Finalement, les sélections obtenues dans le cas de 300 variables apparaissent aussi pertinentes que les sélections obtenues dans le cas de 280 variables, malgré le fait que des variables sont alors combinaisons linéaires d'autres variables.

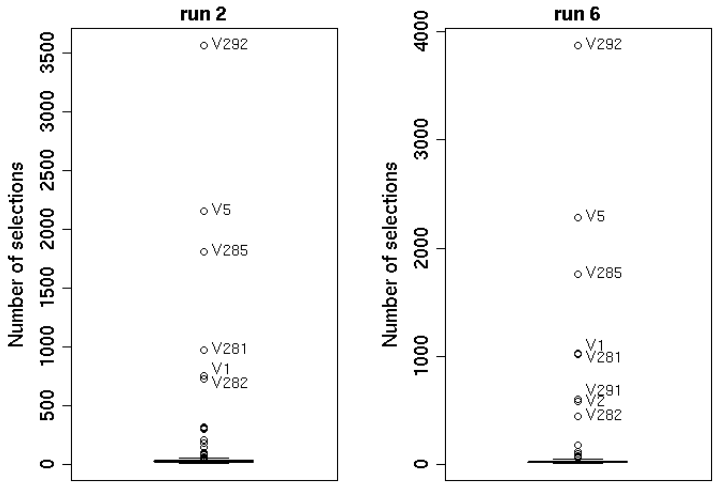

FIGURE 3.2 – Boxplots des nombres d'itérations lors desquelles les variables sont sélectionnées, exemples de deux runs avec 300 variables. Le boxplot de gauche correspond au run 2 : il y a un écart entre les variables sélectionnées plus de 700 fois et les autres, ce sont donc ces six variables qui sont sélectionnées. Le boxplot de droite correspond au run 6 : il y a un écart entre les variables sélectionnées dans plus de 400 itérations et les autres, ce sont donc ces huit variables qui sont sélectionnées.

Variables	Nb de sélections parmi les 10 runs avec 280 variables	Nb de sélections parmi les 10 runs avec 300 variables
$V1$	0	10
$V2$	9	8
$V3$	10	2
$V4$	5	0
$V5$	10	10
$V281 = 2 \times V1$	Non présentes	10
$V282 = 2 \times V2$		9
$V283 = 2 \times V3$		3
$V284 = 2 \times V4$		0
$V285 = 2 \times V5$		10
$V291 = V1 + V2$		7
$V292 = V3 - V4$		10
$V293 = V5 + V13$		0

TABLE 3.1 – Nombres de fois que les variables $V1, V2, V3, V4, V5$ ou des combinaisons linéaire de ces variables ont été conservées dans les sélection finales des 10 runs avec 280 variables, et des 10 runs avec 300 variables. Aucune autre variable n'a été sélectionnée au final sur les 20 runs.

REMARQUE 3.5.1 *Notons que nous avons fait tourner l'algorithme avec 1000 itérations seulement dont 500 de burn-in, et les résultats obtenus étaient assez similaires des résultats obtenus*

avec 5000 itérations, avec comme différence principale la variable $V4$ qui n'était pas sélectionnée.

Prédictions

Pour examiner la pertinence des sélections finales des runs, nous effectuons des prédictions sur le jeu de validation en utilisant des modèles ajustés sur le jeu d'apprentissage. En ce qui concerne les runs avec 300 variables, nous ne pouvons pas ajuster un modèle avec toutes les variables conservées dans les sélections finales, puisque certaines sont combinaisons linéaires d'autres. Nous avons donc ajusté un modèle avec les variables linéairement indépendantes $V281$, $V282$, $V283$, $V285$, et $V292$. Les sensibilités et spécificités associées sont présentées dans la table 3.2. En comparaison, en ajustant un modèle avec les 5 « vraies » variables, nous obtenons une sensibilité et une spécificité de 0.94 and 0.89 respectivement. C'est donc équivalent aux résultats obtenus en utilisant les variables $V281, V282, V283, V285$ et $V292$ sélectionnées par l'algorithme avec paramètre ridge.

Variables sélectionnées parmi 280			Variables sélectionnées parmi 300		
Variables	Sensibilité	Spécificité	Variables	Sensibilité	Spécificité
$V2, V3, V5$	0.87	0.89	$V281$, $V282$ $V283$, $V285$	0.94	0.89
$V2, V3, V4, V5$	0.93	0.96	et $V292$		

TABLE 3.2 – Sensibilités et spécificités obtenues sur le jeu de validation.

Nombre de variables sélectionnées au cours de l'algorithme

Le nombre d'éléments de γ valant 1 n'est plus fixé, contrairement au chapitre 2, et donc le nombre de variables sélectionnées varie au cours des itérations de l'algorithme. La figure 3.3 représente, pour les dix runs avec 300 variables, les nombres d'itérations de l'algorithme associées à un nombre de variables sélectionnées de 1 à 15. Par exemple, nous observons que dans 7140 itérations parmi les 40 000, 3 variables sont sélectionnées parmi les 300. Un graphique très similaire est obtenu pour les dix runs avec 280 variables. Notons que le nombre de variables sélectionnées à chaque itération se régule automatiquement, et n'a jamais dépassé 12 variables. Nous n'avons jamais été dans la situation $p > n$.

Comparaison avec le Lasso bayésien

Dix runs du Lasso bayésien ont été effectués, avec 20000 itérations dont 5000 de burn-in. A titre d'exemple, les résultats du run 7 sont illustrés dans la Figure 3.4. Pour ce run, en utilisant les valeurs absolues des β_js et un seuil de 0.85, les variables pertinentes $V281$, $V282$, $V283$, $V285$ et $V292$ sont sélectionnées ($V284$ est indirectemment sélectionnée par le biais de $V292$). Cependant, nous pouvons noter que les autres variables pertinentes ne sont pas sélectionnées, et en particulier les variables combinaisons linéaires des variables séléctionnées. De plus, la variable

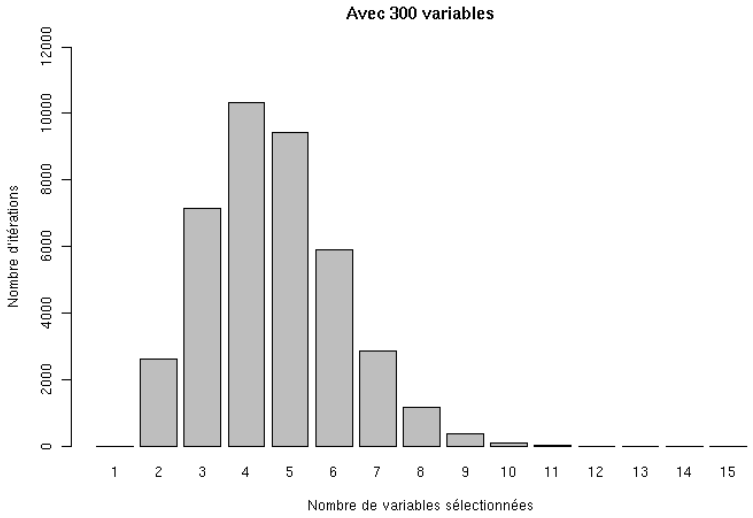

FIGURE 3.3 – Nombres d'itérations de l'algorithme associées à un nombre de variables sélectionnées de 1 à 15. Ce nombre d'itérations est calculé sur les dix runs avec 300 variables (40 000 itérations post-burn-in au total).

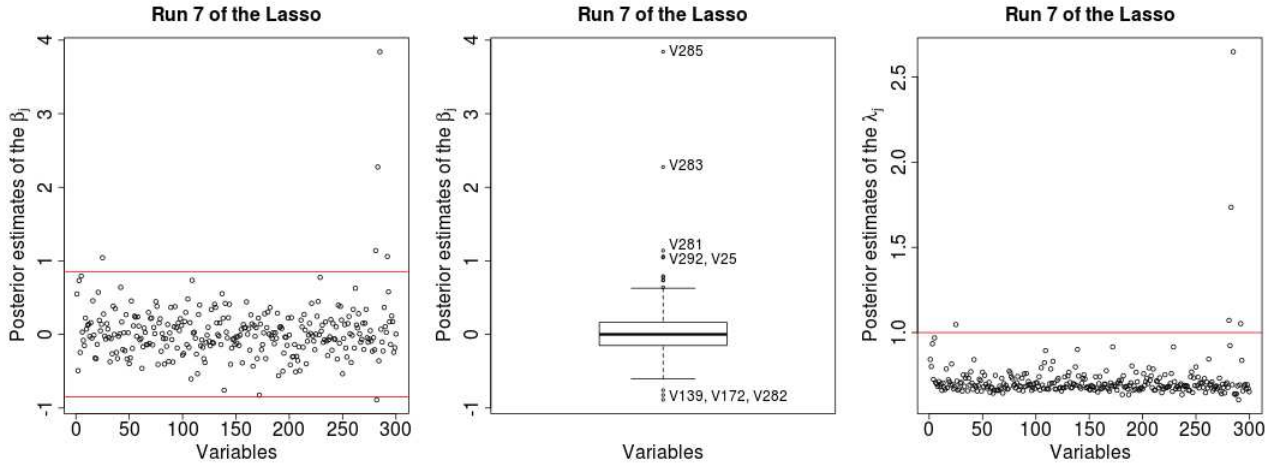

FIGURE 3.4 – Résultats du run 7 du Lasso bayésien. A gauche sont représentées les valeurs des β_js ainsi que le seuil utilisé pour la sélection de variables. Une autre manière de représenter ces valeurs est d'utiliser un un box-plot, représenté au milieu. A droite sont représentées les valeurs des λ_js et le seuil utilisé.

$V25$ qui ne fait pas partie des variables explicatives du modèle est sélectionnée. En utilisant les valeurs des λ_js et un seuil de 1, les variables pertinentes $V281$, $V283$, $V285$ et $V292$ sont sélectionnées, et la variable $V25$ est encore sélectionnée. Enfin, en utilisant les intervalles de crédibilité (IC) a posteriori, seules les variables $V283$ et $V285$ sont sélectionnées.

Les variables sélectionnées lors des dix runs du Lasso bayésien sont données dans le tableau 3.3.

Variables	Avec les $\|\beta_j\|$s	Avec les λ_js	Avec les IC
sélectionnées lors de 9 runs	$V285$	$V285$	$V285$
sélectionnées lors de 7 runs	$V283, V292$	$V283$	
sélectionnées lors de 6 runs		$V292$	$V283$
sélectionnées lors de 3 runs	$V282$		
sélectionnées lors de 2 runs	$V281$	$V281, V282$	
sélectionnées lors d'1 run	$V25, V208, V209, V250$ $V87, V127, V270$	$V25, V250, V87, V127$ $V270$	$V250, V87, V127$

TABLE 3.3 – Variables sélectionnées lors des 10 runs du Lasso bayésien, en utilisant les trois méthodes différentes présentées dans la partie 3.4.

De manière générale, il apparaît que l'utilisation des valeurs des β_js ou des λ_js permet de sélectionner plus de variables que l'utilisation des IC a posteriori, mais au prix de la sélection de quelques variables non pertinentes. De plus, nous pouvons noter que cette approche par le Lasso bayésien ne donne pas des résultats très reproductibles. En effet, certains runs permettent d'obtenir des sélections de variables très pertinentes, comme le run 1 par exemple à partir duquel les variables $V281, V282, V283, V285$ et $V292$ sont sélectionnées. A l'inverse, d'autres runs donnent des sélections pas vraiment pertinentes, comme le run 10 à partir duquel les variables $V127$ et $V270$ sont sélectionnées, ou le run 4 à partir duquel les variables $V285, V208, V209$ et $V250$ sont sélectionnées. Entre ces deux cas extrêmes, certains runs permettent de sélectionner quelques unes des variables pertinentes mais pas toutes, comme les runs 2, 5, 8 et 9 à partir desquels les variables $V283, V285$ et $V292$ sont sélectionnées. Il apparaît aussi que les sous ensembles de variables sélectionnées obtenus sont moins stables que ceux obtenus par l'approche SSVS développée utilisant l'a priori avec paramètre ridge. Enfin, plus de « bruit » est observé avec l'approche Lasso bayésien, puisque plusieurs variables non pertinentes sont sélectionnées dans un seul des dix runs (nous les considérons comme « bruit » car elles ne sont pas sélectionnées par la majorité des runs).

3.5.2 Données génomiques

Nous utilisons un jeu de données réduit du jeu utilisé dans le chapitre 2, voir la description des données dans la partie 2.3.1. L'objectif est toujours de sélectionner des probesets reliés au statut ER des patientes, en prenant en compte les différentes conditions expérimentales liées aux différentes provenances des jeux de données. Nous conservons pour chacune des patientes les mesures d'expression de 275 probesets, parmi lesquels plusieurs sont supposés être pertinents

pour expliquer le statut ER (notamment les probesets numérotés 148, 260, 263 et 273). Nous utilisons un jeu d'apprentissage constitué de 100 patientes et un jeu de validation constitué de 88 patientes. Afin de se mettre dans un cas où la matrice $X'_\gamma X_\gamma$ peut être singulière, nous ajoutons trois variables à la matrice des effets fixes X. Ces trois variables sont des combinaisons linéaires des 4 probesets jugés pertinents : $V276 = 2 \times V148$, $V277 = -V260$ et $V278 = V263 + V273$. Nous sommes dans le cas d'un seul effet aléatoire de taille 3, et les jeux de données de provenance différentes sont indépendants. Nous avons donc $D = \sigma^2 I_3$.

Nous lançons dix fois l'algorithme de sélection de variables avec paramètre ridge en utilisant seulement les 275 premiers probesets de la matrice, puis dix fois en utilisant les 278 probesets. Dans les deux cas et pour chacun des runs nous utilisons les mêmes 100 patientes et les mêmes hyper-paramètres : 5 variables sont initialement sélectionnées, 1 élément de γ est proposé d'être changé à chaque itération de l'algorithme de Metropolis-Hastings, la distribution a priori de σ^2 est une $\mathcal{IG}(1,1)$, $\pi_j = 5/275$ pour tout j quand les 275 premières variables sont utilisées, $\pi_j = 5/278$ pour tout j quand les 278 variables sont utilisées, 5000 itérations sont lancées dont 1000 de burn-in, et chaque étape de Metropolis-Hastings est constituée de 500 itérations.

Comme dans l'exemple précédent nous choisissons $c_0 = 50$, et fixons les paramètres c et λ comme précédemment. Cela donne $\lambda = 1/225$ et $c = 50.0009$ dans le cas où X contient 275 probesets, et $\lambda = 1/278$ et $c = 50.00088$ dans le cas où X contient les 278 probesets.

Variables sélectionnées

Pour chacun des vingt runs, une sélection des variable est effectuée, en utilisant des boxplots comme expliqué dans la partie 2.2.4. Par exemple la figure 3.5 donne deux boxplots associés respectivement à un run avec 275 probesets, et à un run avec 278 probesets.

Variables	Probeset associés	Nb de sélections parmi les 10 runs avec 275 variables	Nb de sélections parmi les 10 runs avec 278 variables
$V260$	228241_at	10	3
$V273$	205862_at	9	0
$V148$	209604_s_at	5	1
$V263$	228554_at	10	0
$V83$	203628_at	7	0
$V66$	202088_at	1	0
$V212$	215157_x_at	1	0
$V277 = -V260$	colinéaire	Non disponible	3
$V278 = V263 + V273$	combinaison linéaire		10

TABLE 3.4 – Nombres de sélections finales sur les vingt runs pour différentes variables. Aucune autre variable n'a été sélectionnée au final sur les vingt runs.

La table 3.4 donne les nombres de sélections des variables sur les vingt runs. Comme dans l'exemple précédent (3.5.1), les sélections obtenues sur les runs avec 278 probesets apparaissent

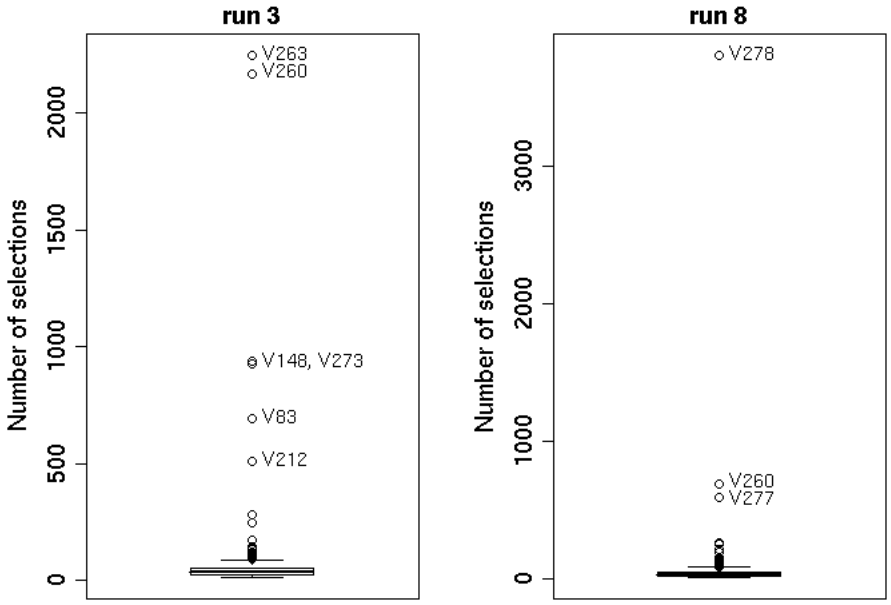

FIGURE 3.5 – Boxplots des nombres d'itérations lors desquelles les probesets sont sélectionnés, sur 4000 itérations post-burn-in. Le boxplot de gauche correspond à un run avec 275 probesets : il y a un écart entre les variables sélectionnée plus de 500 fois et les autres, ce sont donc ces six variables qui sont sélectionnées. Le boxplot de droite correspond à un run avec 278 probesets : il y a un écart entre $V278, V260$ et $V277$ sélectionnées dans plus de 500 itérations et les autres, ce sont donc ces trois variables qui sont sélectionnées.

aussi pertinentes que les sélections obtenues sur les runs avec 275 probesets, malgré la présence de variables combinaisons linéaires d'autres variables.

Prédictions

Pour examiner la pertinence des sélections finales des runs, nous avons effectué des prédictions sur le jeu de validation, en utilisant des modèles ajustés sur le jeu d'apprentissage. Concernant les runs avec 278 variables, nous ne pouvons pas utiliser les variables $V260$ et $V277$ simultanément, car elles sont colinéaires. Nous avons donc ajusté un modèle avec les variables $V277$ et $V278$. Les sensibilités et spécificités associées sont présentées dans la table 3.5. En comparaison, en ajustant un modèle avec les quatre probesets pertinents nous obtenons une sensibilité de 0.94 et une spécificité de 1. C'est équivalent aux résultats obtenus en utilisant seulement les deux variables $V278$ et $V277$.

Comparaison avec le Lasso bayésien

Dix runs du Lasso bayésien ont été effectués, avec 20000 itérations dont 5000 de burn-in. A titre d'exemple, les résultats du run 5 sont illustrés dans la Figure 3.6. Pour ce run, en utilisant

Variables sélectionnées parmi 275			Variables sélectionnées parmi 278		
Variables	Sensibilité	Spécificité	Variables	Sensibilité	Spécificité
$V260, V273, V263$	0.92	1	$V278$	0.87	0.97
			$V278$, $V277$	0.94	1

TABLE 3.5 – Sensibilité et spécificité sur le jeu de validation.

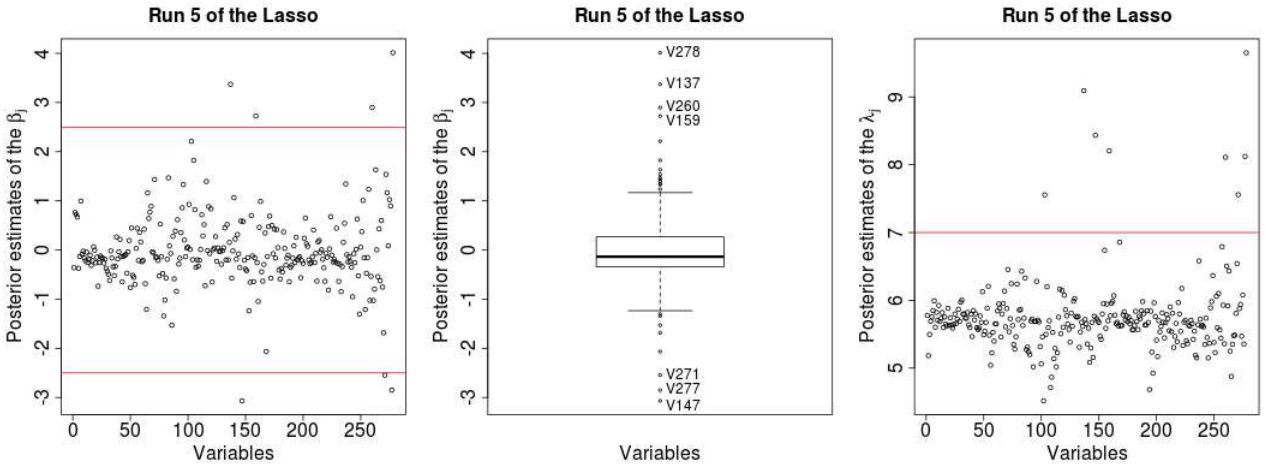

FIGURE 3.6 – Résultats du run 5 du Lasso bayésien. A gauche sont représentées les valeurs des β_js ainsi que le seuil utilisé pour la sélection de variables. Une autre manière de représenter ces valeurs est d'utiliser un un box-plot, représenté au milieu. A droite sont représentées les valeurs des λ_js et le seuil utilisé.

les valeurs absolues des β_js et un seuil de 2.5, les variables pertinentes $V260$, $V278$ et $V277$ sont sélectionnées. Cependant, les variables moins pertinentes $V137$, $V159$, $V147$, $V271$ sont également sélectionnées. En utilisant les valeurs des λ_js et un seuil de 7, les variables pertinentes $V260$, $V278$ et $V277$ sont sélectionnées, de même que les variables moins pertinentes $V103$, $V137$, $V159$, $V147$ et $V271$. Enfin, en utilisant les intervalles de crédibilité (IC) a posteriori, aucune variable n'est sélectionnée.
Les variables sélectionnées lors des dix runs du Lasso bayésien sont données dans le tableau 3.6.

Les mêmes remarques générales que celles faites pour l'exemple simulé s'appliquent.

Variables	Avec les $\|\beta_j\|$s	Avec les λ_js	Avec les IC
sélectionnées lors de 7 runs	$V278$	$V278$	
sélectionnées lors de 4 runs	$V277, V260$		
sélectionnées lors de 3 runs		$V260, V263, V277$	
sélectionnées lors de 2 runs	$V263, V114, V140, V272$ $V147, V271, V83$	$V140, V145, V83$	$V278, V145, V271, V266$
sélectionnées lors d'1 run	$V276, V80, V145, V71$ $V102, V137, V159, V59$ $V84, V105, V78, V95$ $V161, V266, V105, V2$ $V161, V266, V105, V2$	$V114, V80, V252, V71$ $V78, V215, V165, V102$ $V60, V137, V159, V147$ $V271, V103, V59, V84$ $V106, V2, V105, V266, V161$	$V114, V140, V272, V147$ $V84, V83, V95, V115$ $V161, V105, V272, V276$

TABLE 3.6 – Variables sélectionnées lors des 10 runs du Lasso bayésien, en utilisant les trois méthodes différentes présentées dans la partie 3.4.

3.5.3 Sensibilité de la méthode

Concernant le coefficient de sélection de variables c, la méthode de sélection de variables sans le paramètre ridge n'y est pas sensible (voir 2.3.2), mais cela est en partie dû au fait que le nombre de variables sélectionnées reste identique à chaque itération de cet algorithme. Comme ce n'est plus le cas pour l'algorithme avec paramètre ridge, il est nécessaire d'examiner sa sensibilité au coefficient c. De même, il est nécessaire d'examiner sa sensibilité au paramètre λ. Nous étudions donc l'influence de c et λ quand ils sont choisis comme proposé dans la partie 3.2.2, ou bien plus arbitrairement. Nous examinons également le comportement de l'algorithme lorsque la valeur des π_j, les paramètres de la distribution a priori de σ^2 ainsi que le nombre d'itérations varient.

Nous utilisons l'exemple des données simulées avec les 300 variables, et faisons varier les paramètres comme présenté dans la table 3.7. Dans cette table, le nombre de variables pertinentes dans les sélections finales des runs est donné, les treize variables pertinentes étant $V1, V2, V3, V4, V5, V281, V282, V283, V284, V285, V291, V292$ et $V293$.
Comme dans le chapitre 2, nous utilisons la mesure de « relative weighted consistency » de somol et Novovicova CW_{rel} [203].

De manière générale, l'algorithme s'avère robuste aux valeurs des différents paramètres, car la plupart des variables pertinentes sont en général présentes dans les sélections finales. Lorsque trois variables seulement sont conservées, ce sont trois variables permettant d'ajuster un modèle probit mixte avec de bonnes prédictions. Les boxplots obtenus sont en général similaires au boxplot de droite de la figure 3.2. En particulier, notons que l'algorithme s'avère peu sensible aux

Run	c_0	c	λ	Valeur pour π_j $\forall j$	A priori pour σ^2	Itérations (burn-in)	Nb de variables pertinentes	CW_{rel}
1	10	10.00035 (3.2)					3	
2	50	50.00885 (3.2)					8	
3	100	100.0354 (3.2)	$1/p = 1/300$	5/300	$\mathcal{IG}(1,1)$	5000 (1000)	8	0.857
4	1000	1003.553 (3.2)					8	
5	10000	10367.03 (3.2)					8	
6			$1/p = 1/300$				8	
7	(3.2) non		$100/p = 1/3$				8	
8		100	1				8	0.8
9	utilisée		10	5/300	$\mathcal{IG}(1,1)$	5000 (1000)	8	
10			100				3	
11		10	$1/p = 1/300$				5	0.348
12	(3.2) non	10	10				3	
13		1000	$1/p = 1/300$	5/300	$\mathcal{IG}(1,1)$	5000 (1000)	0	(0.639
14		1000	10				8	without
15	utilisée	100	$100/p = 1/3$				8	run 13)
16				5/300			8	
17	100	100.0354 (3.2)	1/p=1/300	50/300	$\mathcal{IG}(1,1)$	5000 (1000)	12	0.848
18				100/300			12	
19					$\mathcal{IG}(1,1)$		8	
20	100	100.0354 (3.2)	1/p=1/300	5/300	$\mathcal{IG}(2,5)$	5000 (1000)	8	1
21					$\mathcal{IG}(5,2)$		8	
22						1000 (500)	8	
23	100	100.0354 (3.2)	1/p=1/300	5/300	$\mathcal{IG}(1,1)$	5000 (1000)	8	1
24						50000 (10000)	8	

TABLE 3.7 – Valeurs des hyper-paramètres utilisées dans les différents runs de l'étude de sensibilité, et indices de stabilité associés. Pour chaque run, 5 variables sont initialement sélectionnées, 1 élément de γ est proposé d'être changé à chaque itération de l'algorithme de Metropolis-Hastings et chaque étape de Metropolis-Hastings est constituée de 500 itérations.

valeurs de c et λ. Un seul run (le $13^{\text{ème}}$) est remarquable car il ne donne pas de bons résultats : aucune variable ne se distingue vraiment des autres, et parmi les variables de tête aucune n'est une variable pertinente, voir la figure 3.7. Ce run correspond à un grand c et à un petit λ. Notons que le run 14 avec un grand c et un grand λ donne de bons résultats, bien que la covariance a priori de β_γ est alors proche d'une identité. Les runs 17 et 18 sont également remarquables, car presque toutes les variables pertinentes sont dans leurs sélections finales, voir la figure 3.7. Ils correspondent à des valeurs de π_j plus élevées, et le « prix à payer » est alors des temps de calcul beaucoup plus longs. Enfin, notons que les valeurs de c et π_j influencent le nombre de variables sélectionnées à chaque itération de l'algorithme : la valeur de c modifie la forme de la distribution de ce nombre (voir la figure 3.8), et ce nombre augmente avec la valeur de π_j (voir la figure 3.9). Néanmoins, même lorsque le nombre de variables sélectionnées à chaque itération est élevé, cela n'influe pas sur les sélections finales des runs, et sur le nombre de variables se distinguant des autres.

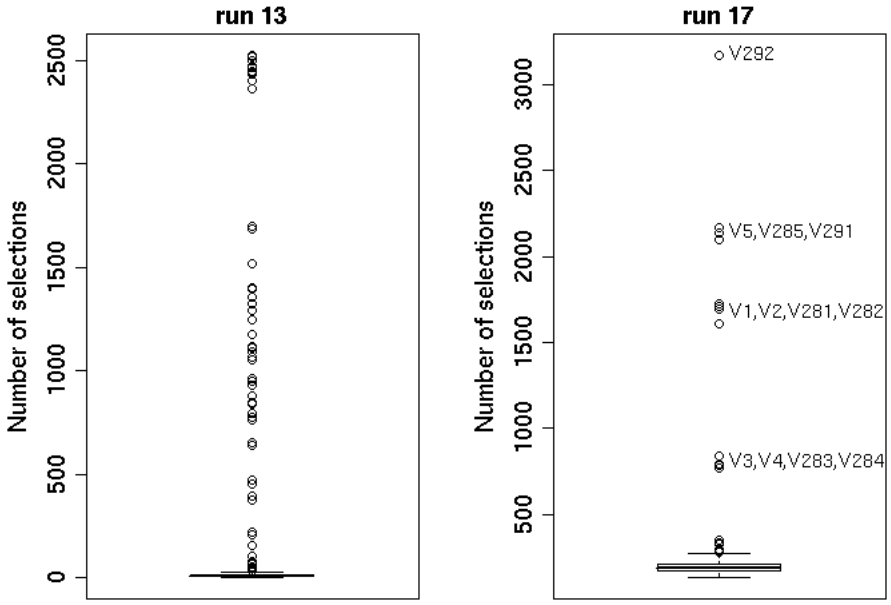

FIGURE 3.7 – Boxplots du nombre d'itérations lors desquelles une variable a été sélectionnée, exemples de deux runs. Le boxplot de gauche correspond au run 13 : aucune variable ne se détache, et les variables de tête ne sont pas les variables pertinentes. Le boxplot de droite correspond au run 17 : les 12 variables pertinentes sont sélectionnées dans plus de 500 itérations.

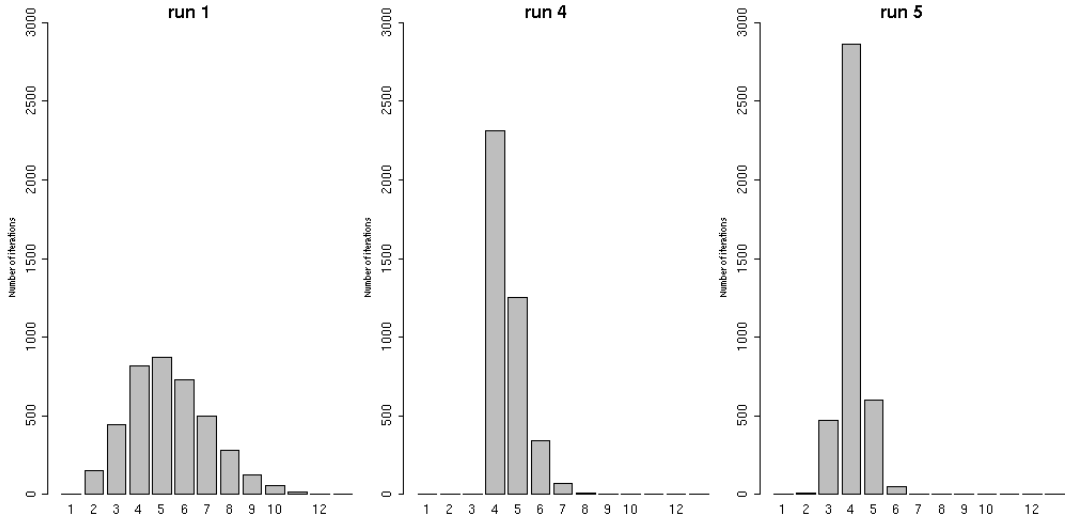

FIGURE 3.8 – Nombres d'itérations des runs 1, 4 et 5 associées à un nombre de variables sélectionnées de 1 à 14. Pour chaque run il y a 4000 itérations post burn-in.

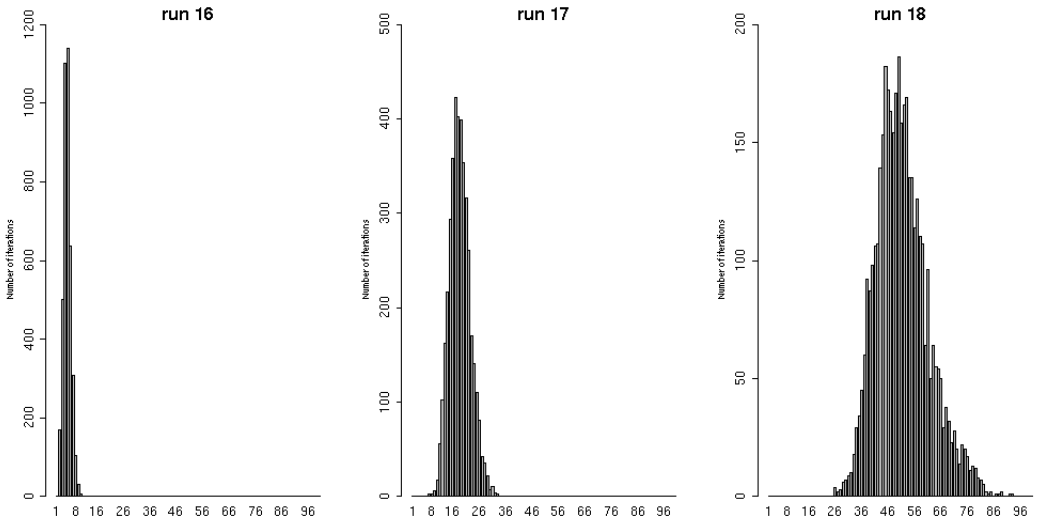

FIGURE 3.9 – Nombre d'itérations des runs 16, 17 et 18 associées à un nombre de variables sélectionnées de 1 à 100. Pour chaque run il y a 4000 itérations post burn-in.

Chapitre 4

Discussion et perspectives

4.1 Discussion du chapitre 2

Contribution et performances de la méthode

La méthode développée permet une sélection bayésienne de variables dans un cadre de modèle mixte, en prenant en compte le design des données. C'est particulièrement utile en génomique, car les jeux de données sont souvent de petites tailles. Notre méthode permet alors d'utiliser des jeux de données fusionnés dans le but d'avoir plus d'observations et une plus grande diversité des données afin d'éviter des biais dus à des jeux de données particuliers. De plus, les jeux fusionnés ayant une taille plus élevée que chaque jeu pris séparement, ils peuvent être séparés en des jeux d'apprentissage et de validation, ce qui évite d'avoir à utiliser une procédure de cross-validation pour évaluer la performance des modèles ajustés (comme par exemple Lee et al. [131], Yang et Song [231] et Sha et al. [197]). Une telle procédure de validation prend beaucoup de temps de calcul, et peut engendrer des biais si elle est mal menée. A l'inverse, lorsque des jeux de données sont fusionnés et qu'un jeu de validation indépendant peut être obtenu, la performance d'une signature génomique peut être directement obtenue sur ce jeu.

A l'aide de simulations nous avons pu montrer que notre méthode donne des résultats comparables à ceux de méthodes ne prenant pas en compte les effets aléatoires lorsque nous sommes en présence de faibles effets aléatoires, mais qu'elle est bien plus performante en présence d'effets aléatoires élevés. Elle peut donc s'avérer très utile en bioinformatique car de nombreux jeux de données sont en libre accés sur internet.

Les probesets sélectionnés par notre méthode pour caractériser le statut ER de patientes atteintes de cancer du sein nous ont permis d'ajuster des modèles permettant de bonnes prédictions. Trois gènes parmi les cinq conservés dans notre modèle ont déjà été sélectionnés par une méthode Support Vector Machine, et un autre groupe de trois gènes parmi ces cinq est connu pour être associé au cancer du sein et au statut ER : GREB1 [163, 224, 179], SERPINA3 [47] et MYH11 [198]. Les probesets sélectionnés par notre méthode apparaissent donc biologiquement pertinents.

Influence des hyper-paramètres

Dans le chapitre 2, le nombre de variables devant être sélectionnées à chaque itération de l'algorithme est fixé, et nous le considérons comme un hyper-paramètre. Cela n'est pas forcément un inconvénient, car dans l'optique d'avoir une sélection pertinente le nombre de variables à sélectionner doit être limité. De plus, l'analyse de sensibilité a montré que le nombre final de variables sélectionnées n'est pas trop influencé par cet hyper-paramètre. Le considérer comme fixé peut même être avantageux (voir la partie 2.2.3). En particulier, cela peut éviter le problème de singularité de la matrice de covariance utilisée dans l'a priori de Zellner [238]. Néanmoins, la méthode présentée dans le chapitre 3 propose une solution pour éviter de fixer ce nombre.

La valeur du coefficient de sélection de variable c de la distribution a priori de β_γ est également fixée. Cette valeur doit être assez grande pour que l'a priori de β_γ soit suffisament diffus. Le fait de fixer le nombre de variables devant être sélectionnées à chaque itération de l'algorithme diminue l'influence de cet hyper-paramètre, voir l'avantage 3 de la partie 2.2.3.

De manière générale, l'analyse de sensibilité a montré la robustesse de l'algorithme au choix de l'ensemble des hyper-paramètres.

Choix de l'échantillonneur

L'échantillonneur que nous utilisons est un Metropolis-within-Gibbs. Le fait d'utiliser un algorithme de Metropolis-Hastings pour générer le vecteur γ permet de facilement fixer le nombre de variables sélectionnées à chaque itération. De plus, dans le cas d'un très grand nombre de variables cela permet de gagner du temps de calcul par rapport à l'utilisation d'un échantillonneur de Gibbs (génération élément par élément).

Convergence

L'échantilloneur que nous utilisons a de bonnes propriétés théoriques. En effet, sous des hypothèses souvent vérifiées (vérifiées pour notre application) l'algorithme Metropolis-within-Gibbs génère une chaîne de Markov Harris-récurrente (voir théorème 12 de Roberts et Rosenthal [188]), et la technique de « grouping » améliore les propriétés de convergence (voir van Dyk et Park [227]).

En pratique, il n'y a pas à notre connaissance d'outils formels permettant de vérifier que l'algorithme a convergé vers sa distribution stationnaire, tels que le critère de Gelman et Rubin [72]. En effet, il est difficile d'étudier le comportement des vecteurs utilisés dans l'algorithme étant donné qu'ils ne sont pas forcément associés aux mêmes variables d'une itération à l'autre. Notre approche n'est donc pas formelle, mais nous avons essayé de conserver l'idée de Gelman et Rubin en comparant les résultats de chaînes ayant différentes valeurs initiales. C'est pourquoi nous avons réalisé l'analyse de sensibilité et de stabilité, afin de vérifier que des chaînes ayant

différentes valeurs initiales et différents paramètres a priori ont le même comportement et donnent des sous-ensembles de variables sélectionnées qui se recoupent. La même approche a été employée par Brown et al. [30], Lee et al. [131], Yang and Song [231], Fouskakis et al. [66] et Tadesse et al. [214] parmi d'autres. Nous ne pouvons pas affirmer que la convergence a bien été atteinte lors des exemples donnés. En effet, il est possible que des modes locaux de la distribution a posteriori de γ aient été atteints et non pas des modes globaux. Cependant nous considérons comme satisfaisants les résultats de l'analyse de stabilité et de sensibilité qui montrent que globalement la méthode est stable à des modifications dans le jeu d'apprentissage, et robuste au choix de l'ensemble des hyper-paramètres. D'après notre expérience, il est suffisant de lancer l'algorithme sur un nombre d'itérations égal à trois fois la taille de l'ensemble des prédicteurs, les résultats obtenus sur un plus grand nombre d'itérations restant la plupart du temps similaires.

Introduction d'un intercept dans le modèle

Nous n'avons pas introduit d'intercept dans notre modèle probit mixte, tout comme Albert et Chib [2], George et McCulloch [74], Chipman et al. [41], Lee et al. [131], et Zhou et al. [241, 242, 243] parmi d'autres. A l'inverse, Brown et al. [30] et Sha et al. [197, 196] par exemple ont utilisé un intercept.

Il est difficile de savoir s'il est préférable ou non de mettre un intercept dans le modèle. En effet, ajouter un intercept peut permettre d'adapter plus précisément le modèle à des données. Cependant, dans le cadre de la sélection de variables nous n'avons pas d'intérêt à estimer cet intercept, qui est susceptible d'être différent à chaque itération étant donné que ce ne sont pas forcément les mêmes variables qui sont sélectionnées. De plus, il est capturé par les effets aléatoires.

Si nous ajoutons un intercept dans le modèle, nous supposons a priori :

$$L \mid \alpha, U, \beta \sim \mathcal{N}_n(\alpha \mathbb{1} + X\beta + ZU, I_n) \qquad \text{et} \qquad \alpha \sim \mathcal{N}(0, h). \tag{4.1}$$

Nous obtenons alors pour la densité jointe a posteriori de l'ensemble des paramètres :

$$\begin{aligned} f(\alpha, \beta_\gamma, L, \gamma, U, D \mid Y) &\propto f(L \mid \alpha, \beta_\gamma, \gamma, U, Y) f(\beta_\gamma \mid \gamma) f(U \mid D) f(D) f(\alpha) f(\gamma) \\ &\propto exp\Big\{ -\frac{1}{2}(L - \alpha - X_\gamma\beta_\gamma - ZU)'(L - \alpha - X_\gamma\beta_\gamma - ZU)\Big\} \prod_{i=1}^{n} \mathbb{1}_{A_i} \\ &\quad \times (2\pi)^{-\frac{d_\gamma}{2}} |c(X_\gamma' X_\gamma)^{-1}|^{-\frac{1}{2}} \exp\Big\{ -\frac{\beta_\gamma'(X_\gamma' X_\gamma)\beta_\gamma}{2c}\Big\} |D|^{-\frac{1}{2}} exp\Big\{ -\frac{U' D^{-1} U}{2}\Big\} \\ &\quad \times |D|^{-\frac{m+q+1}{2}} exp\Big\{ -\frac{tr(\Psi D^{-1})}{2}\Big\} exp\Big\{ -\frac{\alpha^2}{2h}\Big\} \prod_{j=1}^{p} \pi_j^{\gamma_j} (1 - \pi_j)^{1-\gamma_j}, \end{aligned} \tag{4.2}$$

avec $A_i = \{L_i : L_i > 0\}$ si $Y_i = 1$, et $A_i = \{L_i : L_i \leq 0\}$ si $Y_i = 0$.

Comme nous n'avons pas d'intérêt à estimer cet intercept, nous intégrons cette densité par rapport à α (technique de « collapsing », voir annexe A). Nous obtenons alors, en notant

$E^{-1} = \big[I_n + h\mathbb{1}\mathbb{1}'\big]^{-1}$:

$$\begin{aligned} f(\beta_\gamma, L, \gamma, U, D \mid Y) \quad &\propto \quad exp\Big\{ -\frac{1}{2}(L - X_\gamma\beta_\gamma - ZU)'E^{-1}(L - X_\gamma\beta_\gamma - ZU)\Big\} \prod_{i=1}^{n} \mathbb{1}_{A_i} \\ &\times (2\pi)^{-\frac{d_\gamma}{2}} |c(X'_\gamma X_\gamma)^{-1}|^{-\frac{1}{2}} exp\Big\{ -\frac{\beta'_\gamma(X'_\gamma X_\gamma)\beta_\gamma}{2c}\Big\} |D|^{-\frac{1}{2}} exp\Big\{ -\frac{U'D^{-1}U}{2}\Big\} \\ &\times |D|^{-\frac{m+q+1}{2}} exp\Big\{ -\frac{tr(\Psi D^{-1})}{2}\Big\} \prod_{j=1}^{p} \pi_j^{\gamma_j}(1-\pi_j)^{1-\gamma_j}. \end{aligned} \tag{4.3}$$

Il apparaît que l'intercept α n'interviendra dans les calculs a posteriori que par la matrice E^{-1}, avec :

$$E^{-1} = \big[I_n + h\mathbb{1}\mathbb{1}'\big]^{-1} = \big[I_n - (n + h^{-1})^{-1}\mathbb{1}\mathbb{1}'\big]. \tag{4.4}$$

En général il est difficile d'avoir une idée a priori de l'intercept. Un a priori diffus est donc choisi, en spécifiant une constante h assez grande. Or pour un nombre d'observations n fixé, nous notons que la matrice E^{-1} tend vers $\big[I_n - n^{-1}\mathbb{1}\mathbb{1}'\big]$ lorsque $h \to \infty$. Soit, pour n et h assez grand, $E^{-1} \to I_n$.

Nous considérons donc qu'il n'est en général pas nécessaire d'introduire un intercept dans le modèle, car l'avantage est contrebalancé par l'inconvénient d'avoir des formules plus compliquées nécessitant plus de calculs (utilisation de la matrice E^{-1} au lieu de I_n).

Utilisation de la technique de « collapsing »

Liu [141] a montré que de manière générale la technique de « collapsing » a de meilleures propriétés de convergence que la technique de « grouping ». Nous nous sommes alors posé la question s'il n'est pas plus avantageux d'intégrer totalement le paramètre β_γ dans la distribution a posteriori jointe des paramètres, plutôt que de simplement le grouper avec le paramètre γ, comme expliqué dans la partie 2.2.2.
La densité jointe a posteriori des paramètres du modèle est la suivante :

$$\begin{aligned} f(\beta_\gamma, L, \gamma, U, D \mid Y) \quad &\propto \quad f(L \mid U, \beta_\gamma, \gamma, Y) f(\beta_\gamma \mid \gamma) f(U \mid D) f(D) f(\gamma) \\ &\propto \quad exp\Big\{ -\frac{1}{2}(L - X_\gamma\beta_\gamma - ZU)'(L - X_\gamma\beta_\gamma - ZU)\Big\} \prod_{i=1}^{n} \mathbb{1}_{A_i} \\ &\times (2\pi)^{-\frac{d_\gamma}{2}} |c(X'_\gamma X_\gamma)^{-1}|^{-\frac{1}{2}} \exp\Big\{ -\frac{\beta'_\gamma(X'_\gamma X_\gamma)\beta_\gamma}{2c}\Big\} |D|^{-\frac{1}{2}} exp\Big\{ -\frac{U'D^{-1}U}{2}\Big\} \\ &\times |D|^{-\frac{m+q+1}{2}} exp\Big\{ -\frac{tr(\Psi D^{-1})}{2}\Big\} \prod_{j=1}^{p} \pi_j^{\gamma_j}(1-\pi_j)^{1-\gamma_j}, \end{aligned} \tag{4.5}$$

avec $A_i = \{L_i : L_i > 0\}$ si $Y_i = 1$, et $A_i = \{L_i : L_i \leq 0\}$ si $Y_i = 0$.
Si nous intégrons cette densité a posteriori jointe en β_γ, nous obtenons, en remarquant que

$|c(X'_\gamma X_\gamma)^{-1}|^{-\frac{1}{2}} = (1+c)^{-\frac{d_\gamma}{2}}|V_\gamma|^{-\frac{1}{2}}$:

$$\begin{aligned} f(L,\gamma,U,D \mid Y) \quad \propto \quad & \exp\Big\{-\frac{1}{2}(L-ZU)'\Big[I_n - \frac{c}{1+c}X_\gamma(X'_\gamma X_\gamma)^{-1}X'_\gamma\Big](L-ZU)\Big\}\prod_{i=1}^{n}\mathbb{1}_{A_i} \\ & \times(1+c)^{-\frac{d_\gamma}{2}}|D|^{-\frac{1}{2}}exp\Big\{-\frac{U'D^{-1}U}{2}\Big\} \\ & \times|D|^{-\frac{m+q+1}{2}}exp\Big\{-\frac{tr(\Psi D^{-1})}{2}\Big\}\prod_{j=1}^{p}\pi_j^{\gamma_j}(1-\pi_j)^{1-\gamma_j}. \end{aligned} \tag{4.6}$$

Par rapport à la partie 2.2.1, les distributions conditionnelle des L_i et de U sont alors modifiées. Nous obtenons pour les L_i (à comparer avec (2.8)) :

$$L \mid \gamma,U,Y \sim \mathcal{N}\Big(ZU, \Big[I_n - \frac{c}{1+c}X_\gamma(X'_\gamma X_\gamma)^{-1}X'_\gamma\Big]^{-1}\Big)\prod_{i=1}^{n}\mathbb{1}_{A_i}, \tag{4.7}$$

et pour U (à comparer avec 2.10) :

$$U \mid L,\gamma,D \sim \mathcal{N}\Big(W_\gamma Z'\Big[I_n - \frac{c}{1+c}X_\gamma(X'_\gamma X_\gamma)^{-1}X'_\gamma\Big]L, W_\gamma\Big), \tag{4.8}$$

avec

$$W_\gamma \quad = \quad \Big(Z'\Big[I_n - \frac{c}{1+c}X_\gamma(X'_\gamma X_\gamma)^{-1}X'_\gamma\Big]Z + D^{-1}\Big)^{-1}.$$

Les distributions conditionnelles de γ et D restent identiques, voir les formules (2.14) et (2.11). A la vue de ces formules, il apparaît que l'échantillonnage de L suivant une loi normale multivariée tronquée est problématique dans le cas d'un grand nombre d'observations. Une méthode de rejet est difficilement envisageable quand nous en avons plus d'une centaine. Nous pouvons essayer d'utiliser un échantillonneur de Gibbs, qui générerait séparement les $L_i \mid L_{-i},\gamma,U,D,Y$ suivant des lois normales tronquées univariées. Mais bien qu'il soit aisé de simuler suivant des lois normales tronquées univariées (voir Devroye [57] ou plus récemment Chopin [44]), il est beaucoup plus fastidieux de calculer les moyennes et les variances des $L_i \mid L_{-i},\gamma,U,D,Y$. En effet, ces calculs impliquent des produits et des inversions de matrices qui sont de tailles conséquentes si le nombre d'observations est élevé, et différentes pour chacun des L_i.
La génération de L suivant une normale multivariée tronquée apparaît donc bien plus longue que la génération de β_γ et de chacun des L_i lorsque la technique de « grouping » est utilisée (car les $L_i \mid \beta_\gamma,\gamma,U,D,Y$ sont alors indépendants entre eux (voir (2.8)). En termes de temps de calcul, nous considérons donc qu'il n'est pas avantageux d'utiliser la technique de « collapsing » plutôt que la technique de « grouping ».

4.2 Discussion du chapitre 3

La méthode développée dans le chapitre 3 peut être en grande partie discutée de la même manière que la méthode du chapitre 2. Cependant elle apporte une contribution supplémentaire, puisqu'elle permet de sélectionner un nombre de variables supérieur au nombre d'observations, et fournit une solution lorsque certaines variables sont combinaisons linéaires d'autres. En particulier, ce dernier cas peut arriver lorsque plusieurs jeux de données avec des variables communes mais des labels différents sont fusionnés.

La distribution a priori proposée pour le paramètre β_γ est une alternative possible à l'a priori classique de Zellner [238], de même que l'a priori proposé par Gupta et Ibrahim [89]. Dans cet a priori les hyper-paramètres c et λ peuvent être choisis indépendamment, et dans ce cas le choix de c n'influence pas le coefficient de la matrice identité. Au contraire, dans l'a priori de Gupta et Ibrahim [89] le choix de c influe nécessairement sur le coefficient de la matrice identité.

Nous proposons une manière de choisir conjointement c et λ, et les résultats obtenus sur les exemples sont satisfaisants, que des variables soient combinaisons linéaires d'autres ou pas. Nous aurions pu proposer d'autres façons de choisir ces hyper-paramètres, mais même quand c et λ sont choisis indépendamment la méthode s'avère robuste au choix de ces hyper-paramètres (seul un run sur 24 dans l'analyse de sensibilité donne de mauvais résultats).

En pratique, même si une matrice $X'_\gamma X_\gamma$ est théoriquement inversible, des variables peuvent être fortement corrélées entre elles et cette matrice peut être computationnellement singulière. De plus, nous ne savons pas forcément avant de lancer un run si la matrice des données X est de rang plein, et si ce n'est pas le cas il peut être très long d'extraire une sous-matrice de rang plein lorsque p est grand. Pour éviter un problème computationnel nous suggérons d'utiliser l'a priori avec paramètre ridge et l'algorithme associé, même si finalement $X'_\gamma X_\gamma$ n'est jamais singulière. Dans tous les cas, les résultats donnés par l'algorithme sont satisfaisants.
Une fois qu'une sélection finale notée $\gamma+$ est obtenue d'après un run de l'algorithme, le rang de la matrice $X_{\gamma+}$ associée peut être calculé. Si cette matrice n'est pas de rang plein alors certaines des variables sélectionnées sont combinaisons linéaires d'autres. Nous suggérons alors de prendre une sous-matrice de $X_{\gamma+}$ de rang plein comme matrice des effets fixes.

Nous avons comparé les résultats de la méthode SSVS développée dans le chapitre 3 avec ceux obtenus par une approche de Lasso bayésien. Tout comme la méthode développée, cette approche peut être utilisée lorsque $X'_\gamma X_\gamma$ n'est pas inversible. De plus, son implémentation est très facile et ne nécessite pas d'étape de Metropolis-Hastings. En comparaison avec la méthode développée, une itération nécessite donc moins de calculs. Cependant, le Lasso bayésien apparaît moins pratique que l'approche SSVS. En effet, dans la formule (3.7) c'est X qui est utilisée et non une sous matrice X_γ comme dans la méthode SSVS développée. Le temps de calcul pour multiplier ou inverser les matrices est donc plus important, et peut être problématique si p est très grand. De plus, sur les simulations et l'exemple présentés, il apparaît que plus d'itérations

sont nécessaires pour le Lasso bayésien par rapport à l'approche SSVS (20000 vs 5000). En ce qui concerne les résultats, nous avons noté que ceux obtenus par le Lasso bayésien sont moins reproductibles que ceux obtenus par l'approche SSVS, et que plus de « bruit » est observé, avec des variables non pertinentes qui sont sélectionnées dans un seul des dix runs. Enfin, il est important de noter que nous avons adapté un simple Lasso bayésien aux modèles probit mixtes, mais que de nombreuses extensions du Lasso classique existent, voir par exemple le fused Lasso (Tibshirani et al. [219]), le group Lasso (Yuan et Lin [237]) ou l'Elastic Net (Zou et Hastie [244]).

4.3 Perspectives

A priori sur les hyper-paramètres

Le coefficient de sélection de variable c a tendance à influencer les probabilités a posteriori des modèles (voir par exemple Celeux et al. [37]), et de nombreux auteurs ont proposé de poser un a priori dessus (voir la partie 2.1.2).

Etant donné que notre méthode s'est montrée robuste aux choix de c et λ, nous avons décidé de ne pas poser d'a priori sur ces hyper-paramètres. D'autant plus qu'il serait alors difficile d'échantillonner ces hyper-paramètres suivant leurs distributions a posteriori. Une solution serait d'utiliser la technique de « collapsing » (voir annexe A) et de les intégrer, mais cela se fait souvent grâce à des approximations de Laplace ou à des approximations numériques, ce qui n'est pas totalement satisfaisant. Une autre possibilité est d'utiliser un algorithme adaptive Metropolis-within-Gibbs pour actualiser les valeurs de c et λ, comme l'ont proposé Bottolo et Richardson [28]. Poser des a prioris sur ces hyper-paramètres compliquerait donc l'algorithme. Il faudrait vérifier si cela améliorerait les performances.

Nous pourrions également étudier l'influence des distributions a priori sur la performance de la méthode, en essayant par exemple d'autres distributions a priori pour σ^2, comme des « half-Cauchy » ou des distributions « folded-noncentral-t » (voir Gelman [71]), ou bien encore une distribution a priori beta-binomiale pour γ (voir Bottolo et Richardson [28]).

Robustesse à une mauvaise spécification du modèle

Il faudrait étudier la robustesse de la méthode dans le cas où le modèle d'où sont issues les données n'est pas un modèle probit mixte comme supposé, mais un mélange de modèles probit par exemple.

Echantillonneur MCMC

La majorité des méthodes de sélection de variables utilisent des échantillonneurs de Gibbs. Cependant, certains auteurs utilisent des échantillonneurs à sauts réversibles (voir Green [83]), comme par exemple Dellaportas et al. [56], Fridley [67], Lamnisos et al. [125] ou encore Fouskakis et al. qui utilisent un algorithme « population-based Reversible Jump » [66]. Notons que l'uti-

lisation d'échantillonneurs à sauts réversibles est en général associée à l'utilisation de facteurs de Bayes, les auteurs sont plus dans une optique de comparaison et de sélection de modèles que dans une optique de sélection de variables.

Récemment, Bottolo et Richardson [28] ont utilisé un algorithme Evolutionary Monte Carlo [139]. Il serait intéressant d'utiliser un tel algorithme « population-based » pour la sélection de variables dans un modèle probit mixte, voire même un algorithme Parallel Tempering with Equi-Energy Moves, algorithme développé dans le chapitre 6.

Bottolo et Richardson [28] ont également développé un algorithme « Fast Scan Metropolis-Hastings » pour générer le vecteur γ élément par élément, dans le but de gagner du temps de calcul par rapport à l'utilisation d'un échantillonneur de Gibbs classique. Brièvement, leur méthode consiste pour chaque élément de γ, à proposer une nouvelle valeur (0 ou 1) suivant une Bernoulli de paramètre très simple à obtenir. Si cette valeur est différente de la valeur actuelle de l'élément, alors une probabilité d'acceptation est calculée pour accepter ou pas ce changement. Il serait intéressant d'examiner si cet algorithme pourrait être avantageux par rapport à l'algorithme de Metropolis-Hastings que nous utilisons pour générer γ.

Influence de la taille des groupes de variables liées

Nous examinons plus en détail les résultats de la partie 3.5.1, et notamment le tableau 3.1 :

- Le groupe des variables liées à $V1$ est $V1, V281, V291$. Toutes les variables de ce groupe sont presque tout le temps sélectionnées. Le comportement du groupe de variables liées à $V2$ est similaire.
- Le groupe des variables liées à $V4$ est $V4, V284, V292$. Dans ce groupe seule $V292$ est sélectionnée (tout le temps). Le comportement du groupe de variables liées à $V3$ est similaire.

Deux questions intéressantes se posent :

1. Comment se comporte l'algorithme dans le cas d'un grand groupe de variables liées ? Est-ce qu'une seule variable du groupe va prédominer ? Ou est-ce que chacune des variables sera sélectionnée moins de fois, mais qu'en sommant les nombres de sélections de ces variables nous obtiendrons un nombre équivalent à ce que nous obtiendrons en cas d'une seule variable ? Si c'est le cas, comment être sûrs de détecter au moins une des variables du groupe ?
2. Si nous sommes en présence d'un groupe de variables liées de grande taille, ainsi que d'un groupe de petite taille, quel est celui dont le nombre de sélections va prédominer sur l'autre ?

Critère automatique pour la sélection post-processing

Le résultat d'un run de l'algorithme est un vecteur donnant le nombre de sélections de chaque variable au cours des itérations de l'algorithme. Nous avons proposé de visualiser ce vecteur à l'aide d'un boxplot pour décider quelles variables doivent être sélectionnées au final, en conservant celles se distinguant des autres. Cependant, il serait intéressant d'avoir un critère automatique

(non supervisé) permettant de décider quelles variables doivent être sélectionnées à la fin d'un run.

Comparaison avec des méthodes fréquentistes

Nous pourrions comparer les performances de notre méthode de sélection de variables bayésienne avec des méthodes de sélection de variables fréquentistes, de type Lasso par exemple (Tibshirani, [218]). Celeux et al. [35] ont récemment montré, dans le cadre du modèle linéaire, la supériorité de plusieurs méthodes bayésiennes de sélection de variables par rapport à plusieurs méthodes fréquentistes. Les premières sont en particulier plus parcimonieuses. Il serait intéressant de mener une étude similaire dans le cadre de sélection bayésienne de variables dans un modèle probit mixte avec un très grand nombre de prédicteurs possibles.

Par ailleurs, il est intéressant de noter que le Lasso et les méthodes de sélection de variables bayésiennes SSVS sont liées : Yuan et Lin [236] ont montré que dans le cadre d'un modèle linéaire, la méthode de George et McCulloch [74] est liée à l'algorithme Lasso si les a priori sur les coefficients du modèle sont des a prioris de Laplace, et qu'en particulier les deux méthodes sélectionnent le même modèle. D'ailleurs, le Lasso bayésien (voir Park et Casella [175] et Hans [93]) est un Lasso pour lequel un a priori de Laplace est supposé pour les coefficients du modèle (et non pas un a priori de Zellner comme la majorité des méthodes SSVS).

Remarquons que Zou et Hastie [244] ont combiné le Lasso avec la régression ridge pour proposer la méthode Elastic Net, adaptée au modèle linéaire avec données corrélées ou lorsque $p \gg n$. Récemment, Li et Lin [136] ont étudié cette méthode dans un cadre bayésien, l'a priori sur les coefficients du modèle est alors un compromis entre une Laplace et une gaussienne. Cependant, Tutz et Ulbrich [226] ont noté qu'en présence de variables fortement corrélées l'Elastic Net ne se comporte pas très bien, ils ont donc proposé une nouvelle pénalité pour prendre en compte les corrélations. Leur approche ne semble pas avoir été reprise dans un cadre bayésien, mais cela ne saurait tarder car récemment Kyung et al. [123] ont montré comment adapter le fused Lasso (Tibshirani et al. [219]), le group Lasso (Yuan et Lin [237]) ou l'Elastic Net à des approches bayésiennes. Il pourrait être intéressant de comparer ces extensions du Lasso bayésien aux approches SSVS développées dans les chapitres 2 et 3.

Deuxième partie

Méthodes de type *Parallel Tempering* avec et sans vraisemblance

Cette seconde partie est constituée de deux chapitres dédiés aux méthodes d'échantillonnage nécessitant le calcul de la vraisemblance, et de deux chapitres dédiés aux méthodes d'échantillonnage sans vraisemblance.
Le chapitre 5 propose une revue des méthodes MCMC qui ont été proposées pour améliorer l'exploration de l'espace des états. Dans le chapitre 6 un algorithme Parallel Tempering with Equi-Energy Moves est développé et son utilisation est illustrée. Nous discutons également cet algorithme et les perspectives envisageables.
Le chapitre 7 propose une revue des méthodes sans vraisemblances. Enfin, dans le chapitre 8 un algorithme ABC-Parallel Tempering est développé et illustré. Plusieurs perspectives concernant les méthodes sans vraisemblances sont également présentées.

Chapitre 5

Méthodes MCMC et Equi-Energy Sampler

Dans ce chapitre nous montrons quelques limites des échantillonneurs classiques tels que le Metropolis-Hastings ou l'échantillonneur de Gibbs. De nombreuses méthodes ont été proposées afin de pallier ces limites, et nous en présentons quelques unes, notamment le Simulated Tempering, le Parallel Tempering et l'Equi-Energy Sampler.

5.1 Limites des échantillonneurs classiques

Les échantillonneurs classiques tels que le Metropolis-Hastings ou l'échantillonneur de Gibbs génèrent une seule chaîne de Markov. Pour pouvoir utiliser cette chaîne dans un but d'inférence, celle-ci doit avoir convergé et avoir visité l'ensemble de l'espace des états.
Cependant nous remarquons que dans le cas de distributions multimodales ou de modèles complexes ayant de nombreux paramètres, la chaîne génerée a souvent deux types de défauts :

1. Soit elle évolue lentement (se mélange mal), et la convergence est difficilement atteinte en un temps de calcul raisonnable.

2. Soit elle reste bloquée dans le voisinage d'un maximum local de l'espace des états, et ne visite donc pas entièrement cet espace.

Que ce soit dans le second cas, ou dans le premier cas si nous devons arrêter la chaîne avant qu'elle n'ait pu bien visiter l'ensemble de l'espace des états, des problèmes se posent pour l'inférence. En effet, des modes peuvent ne pas avoir été visités ou bien être représentés dans des proportions incorrectes.
Nous illustrons ce problème sur deux exemples.

5.1.1 Cas d'un Metropolis-Hastings

Nous souhaitons simuler suivant un mélange de vingt lois normales bi-dimensionnelles (exemple utilisé par Kou et al. [118]) :

$$f(x) = \sum_{i=1}^{20} \frac{w_i}{\sigma_i \sqrt{2\pi}} \exp\Big(- \frac{1}{2\sigma_i^2}(x-\mu_i)'(x-\mu_i)\Big),$$

avec $\sigma_1 = \ldots = \sigma_{20} = 0.1$, $w_1 = \ldots = w_{20} = 0.05$, et

$$(\mu_1, \ldots, \mu_{20}) = \begin{pmatrix} 2.18 & 8.67 & 4.24 & 8.41 & 3.93 & 3.25 & 1.70 & 4.59 & 6.91 & 6.87 \\ 5.76 & 9.59 & 8.48 & 1.68 & 8.82 & 3.47 & 0.50 & 5.60 & 5.81 & 5.40 \\ 5.41 & 2.70 & 4.98 & 1.14 & 8.33 & 4.93 & 1.83 & 2.26 & 5.54 & 1.69 \\ 2.65 & 7.88 & 3.70 & 2.39 & 9.50 & 1.50 & 0.09 & 0.31 & 6.86 & 8.11 \end{pmatrix}. \quad (5.1)$$

La plupart des modes sont à plus de quinze écart-types les uns des autres, il est donc assez difficile de simuler suivant ce mélange de lois. Nous lançons un algorithme de Metropolis-Hastings sur 10 000 itérations dont 2000 de burn-in. Comme dans Kou et al. [118], nous proposons à chaque itération de l'algorithme de modifier l'état actuel de la chaîne par un terme suivant une gaussienne $\mathcal{N}_2(0, \tau^2 I_2)$, avec $\tau = 0.25$. La figure 5.1 représente l'échantillon obtenu, censé représenter le mélange des vingt normales (à comparer avec la figure 6.2 obtenue par l'algorithme PTEEM développé dans le chapitre 6). Nous observons que seuls trois modes sur les vingt ont été visités. Ces trois modes étant proches les uns des autres, la chaîne a réussi à passer de l'un à l'autre, mais elle n'a pas réussi à sortir de leur voisinage.

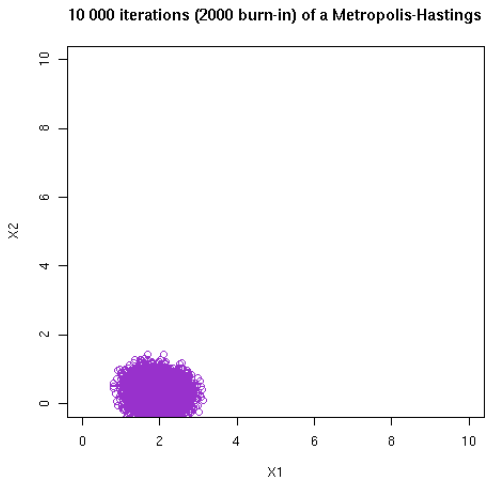

FIGURE 5.1 – Echantillon obtenu sur 8000 itérations d'un algorithme de Metropolis-Hastings, censé représenter le mélange de vingt normales dont les modes sont données par (5.1).

5.1.2 Cas d'un échantillonneur de Gibbs

Nous nous plaçons à nouveau dans un modèle de mélange de lois normales.

Modélisation

Soit n observations indépendantes y_i, $i \in \{1, \ldots, n\}$. Les observations sont supposées issues de k composants différents, k connu :

$$y_i \sim \sum_{p=1}^{k} w_p f(.|\mu_p, \sigma_p^2), \qquad i = 1, \ldots, n,$$

avec $f(.|\mu_p, \sigma_p^2)$ la densité d'une gaussienne $\mathcal{N}(\mu_p, \sigma_p^2)$. Les tailles des différents composants sont proportionnelles aux w_p, qui sont les poids des composants. Un vecteur d'étiquettes c est introduit, avec $c_i = p$ si l'observation y_i est issue du $p^{\text{ème}}$ composant.
A priori les c_i sont supposés indépendants et issus de distributions multinomiales, avec :

$$p(c_i = j) = w_p, \qquad p = 1, \ldots, k.$$

Les paramètres μ_p et σ_p^2 sont supposés issus indépendemment des a priori suivants ($p = 1, \ldots, k$) :

$$\mu_p \sim \mathcal{N}(\xi, \kappa^{-1}), \qquad \sigma_p^{-2} \sim \Gamma(\alpha, \beta) \qquad \text{et} \qquad \beta \sim \Gamma(g, h). \tag{5.2}$$

Et l'a priori sur w est une Dirichlet symétrique :

$$w \sim D(\delta, \delta, \ldots, \delta).$$

Les paramètres δ, ξ, κ, α, g et h sont supposés fixés.

Exemple simulé

Nous utilisons un exemple utilisé par Jasra et al. [106]. Un vecteur y de taille 100 est généré suivant un mélange de quatre lois normales :

$$\frac{1}{4}\Big(\mathcal{N}(-3, 0.55^2) + \mathcal{N}(0, 0.55^2) + \mathcal{N}(3, 0.55^2) + \mathcal{N}(6, 0.55^2)\Big). \tag{5.3}$$

La figure 5.2 donne la densité des données simulées.

Afin d'échantillonner suivant la densité a posteriori jointe des paramètres, nous utilisons un échantillonneur de Gibbs, lancé sur 5000 itérations dont 1000 de burn-in, avec les paramètres fixes suivants : $\alpha = 2$, $\xi = \bar{y}$, $\delta = 1$, $\kappa = 1/R^2$, $g = 0.2$ et $h = 10/R^2$, avec $R = \max(y) - \min(y)$. Nous nous intéressons au phénomène de « label-switching », qui nous permet de savoir si la

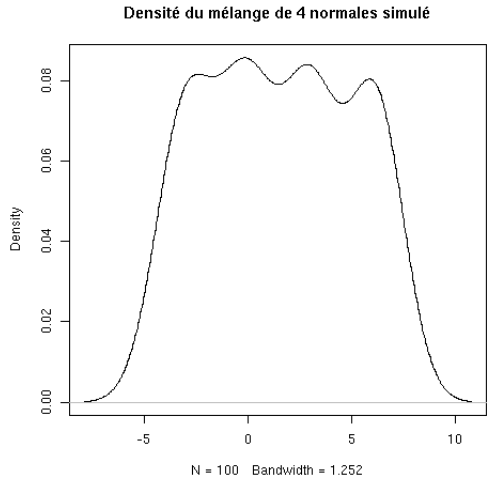

FIGURE 5.2 – Densité des données simulées suivant le mélange (5.3).

chaîne générée a bien visité l'ensemble de l'espace des paramètres (voir l'annexe C). Si la chaîne a correctement exploré cet espace, elle doit avoir visité les $4! = 24$ modes en proportions égales. La figure 5.3 montre les tracés des moyennes des quatres composants. Nous n'observons aucun « label-switching », ce qui montre que la chaîne générée est restée bloquée dans un seul des 24 modes.

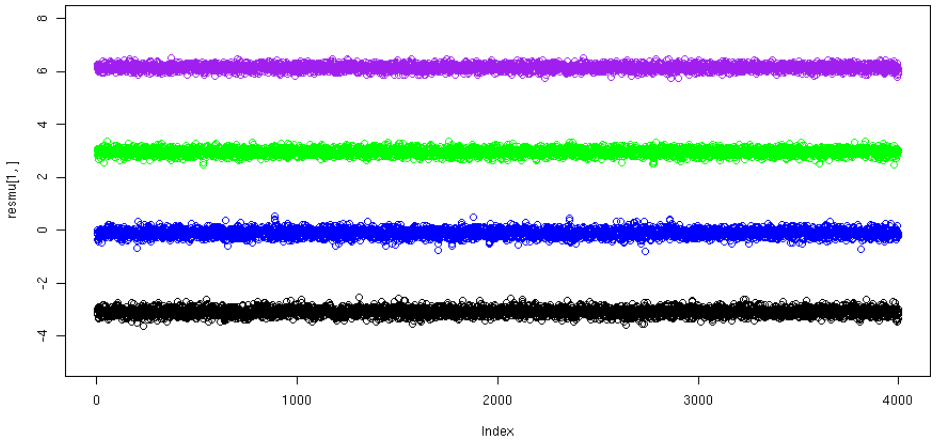

FIGURE 5.3 – Tracés des moyennes des quatres composants, sur 4000 itérations post-burn-in.

5.2 Méthodes basées sur une seule chaîne

Pour pallier les limites des échantillonneurs classiques, de nouveaux algorithmes ont été proposés.

5.2.1 Utilisation de distributions tempérées

Toutes les méthodes que nous décrivons plus loin utilisent des distributions tempérées. Dans cette partie, nous en expliquons l'intérêt.

Quand une distribution cible π est multimodale, les échantillonneurs classiques ont du mal à visiter tous les modes de cette distribution, comme illustré dans la partie 5.1. Plus les pics des modes sont pointus et éloignés entre eux et plus la chaîne a du mal à se déplacer. En effet, dans ce cas les modes sont séparés par des « vallées », des zones de très faible densité par rapport aux densités des états dans le voisinage du mode. Alors si la chaîne a une très faible probabilité de passer directement d'un mode à l'autre, comme elle a peu de chance de s'aventurer dans ces zones de faible densité pour finalement atteindre un autre mode, elle risque de rester bloquée dans le voisinage d'un seul mode.

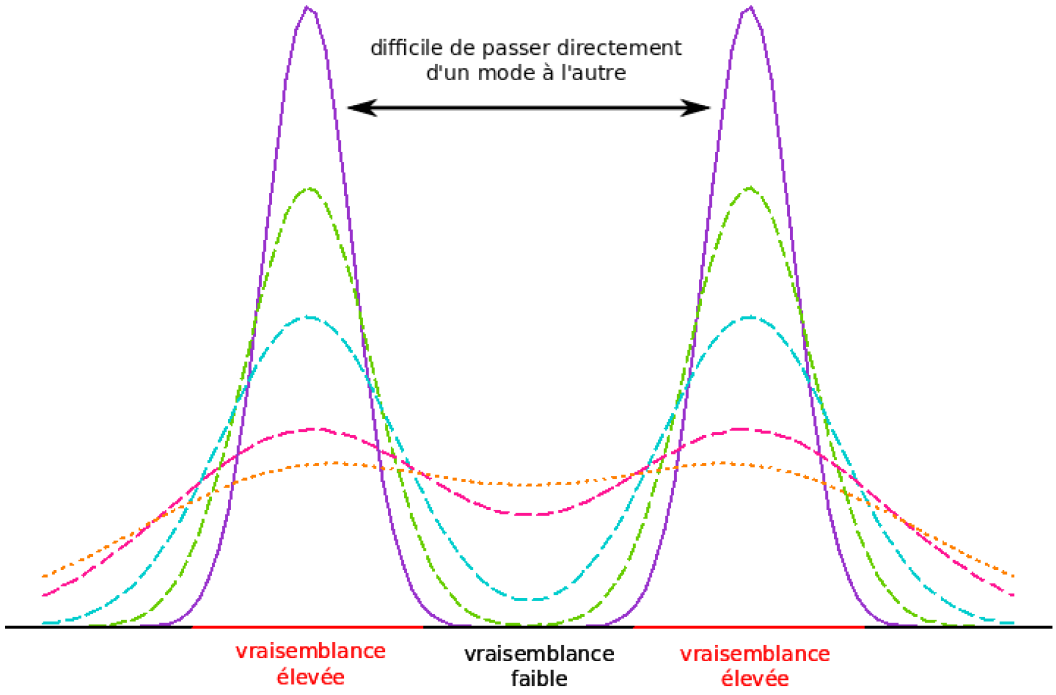

FIGURE 5.4 – Densité d'un mélange de deux lois normales en violet plein, et densités tempérées associées en vert, bleu, rose et orange pointillés par ordre de températures croissantes.

Tempérer la densité cible π par une température T revient à étudier $\pi^{1/T}$. Plus la température

T est élevée (> 1), et plus la distribution tempérée est aplatie, avec les pics des modes moins pointus, et les « vallées » comblées. Ce phénomène est illustré par la figure 5.4 sur un mélange de deux lois normales. Sur cette figure, la densité orange correspondant à la température la plus élevée n'a presque plus de « vallée » entre ses deux modes. Il est alors évident que plus la température est élevée et plus l'échantillonneur associé peut se déplacer facilement, car la chaîne peut alors s'échapper des modes aplatis et ne reste pas bloquée. Nous parlons alors de distributions « chaudes », par opposition aux distributions « froides » associées à de faibles températures.

5.2.2 Simulated Annealing (recuit simulé)

Cette méthode a été proposée par Kirkpatrick et al. en 1983 [116]. Une description en a été faite par Liu [142], sa convergence et son comportement en pratique ont été étudiés par Bertsimas et Tsitsiklis [24].
C'est la première méthode qui utilise le principe des distributions tempérées. Ses auteurs font un parallèle avec le processus physique d'« annealing », consistant à chauffer un solide à une température assez élevée pour qu'il fonde, puis à graduellement diminuer la température jusqu'à être proche de zéro. Ainsi les particules du systèmes sont tout d'abord mélangées, peuvent se déplacer facilement, puis vont peu à peu s'arranger jusqu'à la cristallisation, on obtient alors un solide dans un état de « faible énergie » (soit une vraisemblance élevée en statistique).
Kirkpatrick et al. [116] ont adapté ce processus physique à un algorithme de Metropolis-Hastings, dans le but de résoudre des problèmes d'optimisation. Cependant, cette méthode peut s'adapter à des problèmes d'échantillonnage.

Soit π une distribution d'intérêt que nous cherchons à maximiser. Soit une séquence de températures décroissantes $T_1 > T_2 > \ldots > T_i > \ldots > T_N$, et une série de distributions tempérées associées $\pi_i(x) \propto \pi(x)^{1/T_i}, i = 1, \ldots, N$. Pour i allant de 1 à N, nous supposons être capables de construire une chaîne de Markov ayant π_i comme distribution stationnaire.

La méthode consiste à lancer successivement des échantillonneurs de distributions stationnaires π_i, pour i allant de 1 à N. A l'étape i un échantillonneur de distribution stationnaire π_i est lancé, et son état initial est le dernier état de la chaîne obtenue à l'étape $(i-1)$. L'algorithme est le suivant :

Simulated Annealing

---

1. `Déterminer` π `la distribution ou fonction d'intérêt,` N`, et la séquence de températures` $T_1 > T_2 > \ldots > T_i > \ldots > T_N$`, afin d'obtenir la série des distributions` $\pi_i, i = 1, \ldots, N$`.`
2. `Choisir un état initial` x_1`.`

3. `Faire` N_1 `itérations d'un échantillonneur MCMC ayant` π_1 `comme distribution stationnaire, à partir de l'état initial` x_1.
4. `Pour` i `allant de 2 à` N:
   - `Prendre comme état initial le dernier état de la chaîne précédente ayant` π_{i-1} `comme distribution stationnaire.`
   - `Faire` N_i `itérations d'un échantillonneur MCMC ayant` π_i `comme distribution stationnaire.`

L'algorithme part de distributions faciles à échantillonner pour se diriger petit à petit vers des distributions plus difficiles à échantillonner par des échantillonneurs classiques. Lorsque la température s'approche de zéro la distribution correspondante sera concentrée autour d'un maximum de π. Des problèmes d'optimisation peuvent donc être résolus en posant $\lim_{i\to\infty} T_i = 0$. Dans le cas ou nous voulons simplement échantillonner suivant π, il suffit de poser une température minimale de 1, soit $T_N = 1$. Nous aurons alors $\pi_N = \pi$. Seule la dernière partie de la chaîne associée à $T_N = 1$ pourra être considérée comme une chaîne de Markov de distribution stationnaire π.

Le principal intérêt de cette méthode est de pouvoir choisir des états initiaux pertinents à chaque étape de l'algorithme, dans l'espoir d'accélérer la convergence de la chaîne finale. Cependant, en théorie les températures doivent décroître plus doucement que ce qui est faisable en pratique pour atteindre le maximum global de π (Holley et al. [95]). De plus, cette méthode a été développée dans une optique d'optimisation, elle ne garantit pas de pouvoir visiter l'ensemble d'une distribution d'intérêt, surtout si elle est multimodale, ce qui est problématique dans une optique d'échantillonnage.

5.2.3 Simulated Tempering

Cette méthode a été initiée par Marinari et Parisi [151] puis légèrement modifiée par Geyer et Thompson [78]. Elle a été décrite par Liu [142] et comparée à l'algorithme Parallel Tempering par Zheng [240].
La principale différence avec le Simulated Annealing est qu'au lieu d'échantillonner suivant les distributions tempérées de façon systématique dans l'ordre des températures décroissantes, l'indice de la température utilisée lors d'une itération est choisi de façon aléatoire. D'un point de vue physique, le Simulated Annealing rompt l'équilibre du système à chaque fois que la température est changée et qu'une nouvelle chaîne est lancée, tandis que le Simulated Tempering change la température aléatoirement tout en conservant l'équilibre du système. D'un point de vue statistique, le Simulated Annealing peut être vu comme la génération de plusieurs chaînes de Markov les unes après les autres, tandis que le Simulated Tempering consiste à générer une seule chaîne, mais dans un espace augmenté par l'indice de température.

Soit π une distribution d'intérêt que nous cherchons à échantillonner, définie sur un espace noté $\mathcal{X}$. Soit une séquence de températures croissantes $1 = T_1 < T_2 < \ldots < T_i < \ldots < T_N$, et une série de distributions tempérées associées $\pi_i(x) \propto \pi(x)^{1/T_i}, i = 1, \ldots, N$. Pour i allant de 1 à N, nous supposons être capables de construire une chaîne de Markov ayant π_i comme distribution stationnaire. La distribution π_1 correspond à notre distribution d'intérêt car $T_1 = 1$.

L'indice de la température est considéré comme une variable aléatoire, et une nouvelle distribution cible est définie sur un plus grand espace des états :

$$\pi_{ST}(x, i) \propto c_i \times \pi_i(x) \qquad \text{définie sur} \qquad \mathcal{X} \times \{1, \ldots, N\}.$$

Les c_i sont des constantes déterminées à l'avance par l'utilisateur, pouvant être choisies de manière à ce que chacune des distributions tempérées ait la même chance d'être visitée que les autres. Elles correspondent en quelque sorte aux poids des différentes distributions. Idéalement, pour tout i il faut $c_i = (\int \pi_i dx)^{-1}$, soit l'inverse de la constante de normalisation de π_i. En pratique l'utilisateur doit ajuster ces constantes à l'aide d'études préliminaires. Une procédure pour effectuer ces ajustements appelée « reverse logistic regression » a été proposée par Geyer [77].

Le Simulated Tempering a comme distribution stationnaire cible π_{st}. L'algorithme est le suivant :

Simulated Tempering

Partir d'un état initial (x_0, i_0). Chaque itération actualise l'état (x, i) de la chaîne et se déroule de la manière suivante:

1. Actualiser x en utilisant un échantillonneur classique ayant π_i comme distribution stationnaire. La température est fixée.
2. Actualiser l'indice de la température par une marche aléatoire, x étant fixé.
   - Proposer un nouvel indice de température $l = i \pm 1$ suivant les transitions $q_{i,l}$ avec $q_{1,2} = q_{N,N-1} = 1$ et $q_{i,i-1} = q_{i,i+1} = \frac{1}{2}$.
   - Ce nouvel indice de température l est accepté pour remplacer i avec la probabilité d'acceptation suivante:

$$1 \wedge \frac{c_l \pi_l(x) q_{l,i}}{c_i \pi_i(x) q_{i,l}}.$$

L'objectif est que l'échantillonneur puisse s'échapper de modes locaux lorsque la température de la chaîne est élevée. Dans une optique d'échantillonnage, seules nous intéressent les réalisations issues de $\pi_1 = \pi$, et donc obtenues lorque la température vaut 1.

Il faut être attentif au choix du nombre de températures à utiliser ainsi qu'à la différence

entre ces températures. En effet, il faut pouvoir traverser l'ensemble des distributions, de la plus « chaude » à la plus « froide », en un nombre d'itérations raisonnable, ce qui nous amène à limiter le nombre de températures. Mais à l'inverse, si le nombre de températures est trop faible, deux distributions adjacentes π_i et π_{i+1} risquent d'être trop différentes, et les transitions de températures trop peu acceptées.

L'avantage du Simulated Tempering sur le Simulated Annealing est que les températures peuvent monter ou descendre au cours de l'algorithme, ce qui lui permet de pouvoir se déplacer facilement dans l'espace des paramètres $\mathcal{X}$. Un inconvénient est le besoin de déterminer au préalable de bonnes constantes c_i, pour que la chaîne puisse visiter chacune des distributions tempérées. Or la détermination de ces constantes peut être longue et difficile.

5.2.4 Tempered Transitions

Cette méthode a été proposée par Neal [166]. Son objectif était d'obtenir une méthode utilisant des distributions tempérées comme pour le simulated tempering, mais sans l'inconvénient d'avoir à estimer des constantes au préalable pour utiliser l'algorithme.

Nous utilisons les mêmes notations que pour le Simulated Tempering (5.2.3). A chaque itération, pour proposer un nouvel état x' à partir d'un état x, l'algorithme va traverser toute la série des distributions tempérées dans le sens des températures croissantes, puis dans le sens des températures décroissantes, pour revenir à la distribution d'intérêt π. Le but est de proposer un x' qui puisse être assez éloigné de x, éventuellement situé autour d'un autre mode.

Pour i allant de 2 à N, il est nécessaire de spécifier des paires de noyaux de transitions (P_i, P_i') ayant π_i comme distribution stationnaire et satisfaisant la condition suivante :

$$\pi_i(x)P_i(x,x') = \pi_i(x')P_i'(x',x) \qquad \forall x, x', i = 2, \ldots, N.$$

Si $P_i = P_i'$, ce noyau satisfait alors la « detailed balance condition ». A partir de l'état x, les noyaux de transition sont appliqués dans l'ordre $P_2P_3\cdots P_NP_N'\cdots P_3'P_2'$ afin d'obtenir des états intermédiaires x_2, x_3, $\cdots$, x_N, x_N', $\cdots$, x_3', x_2', jusqu'à finalement obtenir x'. Cet état x' est accepté suivant une probabilité d'acceptation prenant en compte les états intermédiaires, et assurant à l'échantillonneur d'avoir $\pi_1 = \pi$ comme distribution stationnaire.

L'algorithme est le suivant :

Tempered Transitions

`Initialiser` $x = x_0$. `Une itération de l'algorithme se déroule de la façon suivante:`

1. `Poser` $x_1 = x$, `valeur de l'état actuel.`
2. `Le candidat` x' `est généré à l'aide des noyaux de transitions:`

Générer x_2 d'après x_1 en utilisant P_2.
Générer x_3 d'après x_2 en utilisant P_3.
$\vdots$
Générer x_N d'après x_{N-1} en utilisant P_N.
Générer x'_{N-1} d'après x_N en utilisant P'_N.
$\vdots$
Générer x'_2 d'après x'_3 en utilisant P'_3.
Générer x'_1 d'après x'_2 en utilisant P'_2.

Le candidat x' correspond à x'_1.

3. Le candidat est accepté avec la probabilité suivante:

$$1 \wedge \Big[\prod_{i=1}^{N-1} \frac{\pi_{i+1}(x_i)}{\pi_i(x_i)}\Big] \times \Big[\prod_{i=1}^{i-1} \frac{\pi_i(x'_i)}{\pi_{i+1}(x'_i)}\Big].$$

Si le candidat est rejeté, l'ancien état x est conservé.

Notons que la deuxième partie de l'itération est similaire au Simulated Annealing, car nous parcourons les distributions auxiliaires dans le sens des températures décroissantes. Le fait de passer par des températures élevées va éventuellement permettre de sortir du voisinage du mode de l'état actuel x. Les transitions intermédiaires sont utilisées dans l'objectif d'avoir une probabilité d'acceptation raisonnablement élevée. Celeux et al. [36] ont proposé la modification suivante : entre deux itérations telles que décrites ci-dessus, effectuer un nombre arbitraire de simulations suivant la distribution cible π, en utilisant un algorithme de Metropolis-Hastings basé sur la diffusion de Langevin (Robert et Tweedie, [189]). L'objectif est d'améliorer l'exploration du voisinage du mode actuel.

L'avantage de cet algorithme Tempered Transitions par rapport au Simulated Tempering est qu'il n'est pas nécessaire de calculer les constantes de normalisation des π_i, car chaque π_i apparaît le même nombre de fois au numérateur et au dénominateur, et donc ces constantes s'annulent. Cependant cet avantage est contrebalancé par le fait que le nombre de températures (et donc d'états intermédiaires) nécessaires pour avoir des probabilités d'acceptation raisonnables est plus élevé que pour le Simulated Tempering (et même très élevé, plus de 50 en général). Ainsi, d'après Neal [166] cette méthode a une efficacité comparable à la méthode de simulated tempering pour des problèmes simples. Par contre, dans le cas de problèmes plus complexes, ces deux méthodes se comportent de façons différentes, en fonction des distributions auxiliaires utilisées. L'algorithme Tempered Transitions peut être la méthode la plus efficace dans certains cas, mais ce sera l'inverse dans d'autres. Behrens et al. [21] ont proposé une méthode pour calibrer les températures afin d'augmenter les probabilités d'acceptation, la température maximale T_N étant fixée.

5.3 Du Parallel Tempering à l'Evolutionary Monte Carlo

5.3.1 Le Parallel Tempering

Cet algorithme est connu sous différentes appellations. Il a été développé de façon indépendante par Geyer en 1991 sous le nom de « Metropolis Coupled Markov chain Monte Carlo $(MC)^3$ » [76], et par Hukushima et Nemoto en 1996 sous le nom de « Exchange Monte Carlo » [100]. Il est aussi connu sous le nom de « Swapping Algorithm » (voir Madras et Zheng par exemple [147]).

Notons π la distribution d'intérêt que nous cherchons à échantillonner, définie sur un espace noté $\mathcal{X}$. Nous devons définir une série de distributions auxiliaires liées entre elles et notées π_1, π_2, ..., π_N définies sur $\mathcal{X}$, avec $\pi_1 = \pi$. Pour i allant de 1 à N, nous supposons être capables de construire une chaîne de Markov ayant π_i comme distribution stationnaire. En général les π_i sont telles que plus i augmente et plus l'échantillonneur associé à π_i explore facilement l'espace des paramètres $\mathcal{X}$.

Au lieu de générer une seule chaîne de Markov comme les échantillonneurs précédemment présentés, plusieurs chaînes sont générées en parallèle. Comme les échantillonneurs associés aux indices i élevés sont censés mieux explorer $\mathcal{X}$ que les échantillonneurs associés aux indices i plus faibles, le principe est de permettre des échanges entre les chaînes. L'objectif est que les chaînes d'indices faibles, et notamment la première chaîne qui est la chaîne d'intérêt, puissent s'échapper de modes locaux dans lesquels elles pourraient rester bloquées si les échanges n'avaient pas lieu. Chaque chaîne n'est pas markovienne en elle-même, c'est l'ensemble des N chaînes qui constitue un processus markovien sur $\mathcal{X}^N$. Nous notons x_i l'état actuel de la $i^{\text{ème}}$ chaîne associée à π_i, et s l'ensemble des états actuels des N chaînes : $s = (x_1, x_2, \ldots, x_N)$.

L'algorithme est le suivant :

Parallel Tempering (ou $(MC)^3$, ou Exchange Monte Carlo, ou Swapping Algorithm)

Initialiser la population des chaînes $\{x_1, x_2, \ldots, x_N\}$.

A chaque itération de l'algorithme, chaque chaîne est actualisée localement, puis un mouvement d'échange entre deux chaînes est proposé, et accepté suivant une certaine probabilité d'acceptation:

1. Pour $i = 1, 2, \ldots, N$, actualiser la $i^{\text{ème}}$ chaîne avec un échantillonneur MCMC ayant π_i comme distribution stationnaire.
2. Choisir deux indices i_1 et i_2 $(i_1 < i_2)$ au hasard dans $\{1, 2, \ldots, N\}$, puis proposer d'échanger les états actuels des chaînes d'indices i_1 et i_2. L'état de l'ensemble des chaînes se note $s = (x_1, \ldots, x_{i_1}, \ldots, x_{i_2}, \ldots, x_N)$, et l'échange consiste à passer de s à $s' = (x_1, \ldots, x_{i_2}, \ldots, x_{i_1}, \ldots, x_N)$.

`Cet échange est accepté avec la probabilité suivante:`

$$\rho_1(s, s') = 1 \wedge \frac{\pi_{i_1}(x_{i_2})\pi_{i_2}(x_{i_1})}{\pi_{i_1}(x_{i_1})\pi_{i_2}(x_{i_2})}. \tag{5.4}$$

La probabilité (5.4) vérifie la « detailed balance condition » qui assure que la distribution stationnaire de l'ensemble des N chaînes est :

$$\pi^*(s) = \pi^*(x_1, x_2, \ldots, x_N) = \prod_{i=1}^{N} \pi_i(x_i) \qquad \text{sur} \quad \mathcal{X}^N, \tag{5.5}$$

(voir la preuve de la proposition 6.1.1 pour une justification théorique).
Les différentes chaînes sont asymptotiquement indépendantes et ont les π_i comme distributions stationnaires (voir Geyer [76]). Au final, seules nous intéresserons les réalisations issues de $\pi_1 = \pi$, la distribution d'intérêt.

Choix des distributions auxiliaires

Plusieurs manières de définir les distributions auxiliaires π_1, π_2, ..., π_N ont été proposées, et passées en revue par Jasra et al. [107]. Nous en présentons quelques unes.

1. Le choix le plus naïf est de poser $\pi_i = \pi$ pour tout i, comme Warnes [228]. Mais il n'y a pas grand intérêt à supposer toutes les distributions identiques, car si l'échantillonneur MCMC associé à π a des difficultés pour explorer l'espace des paramètres, alors toutes les chaînes auront des difficultés d'exploration, comme noté par robert et Casella [186].

2. Le choix le plus classique est d'utiliser des distributions tempérées de la distribution cible π (voir la partie 5.2.1), comme pour le Simulated Annealing, le Simulated Tempering ou l'algorithme Tempered Transitions : $1 = T_1 < T_2 < \ldots < T_i < \ldots < T_N$. C'est de ce choix que vient l'appelation « Parallel Tempering », et cela a été utilisé par Hukushima et Nemoto [100] et Goswami et Liu [82] parmi d'autres. Dans ce cas plus la température augmente et plus l'échantillonneur associé explore facilement l'espace des paramètres $\mathcal{X}$. L'idée est que les chaînes de températures élevées puissent facilement se déplacer dans l'espace des paramètres, et échangent ensuite de l'information avec les chaînes d'indices plus faibles pour leur permettre de s'échapper de modes locaux.

3. Comme la difficulté d'un échantillonneur peut être de passer d'une région modale à une autre, une idée est de contraindre certaines distributions à rester dans un sous-espace de l'espace des paramètres $\mathcal{X}$, et ainsi à en forcer l'exploration. Il faut pour cela partitionner $\mathcal{X}$ pour définir les sous-espaces où des distributions seront contraintes, et c'est la principale difficulté. Cependant, la partition n'est pas forcément difficile dans le cas où la dimension

de l'espace des paramètres peut varier, et où un échantillonneur trans-dimensionnel est utilisé (voir Jasra et al. [107] par exemple). La méthode du « multicanonical sampling » [22] et l'algorithme de Wang-Landau d'Atchadé et Liu [11] se basent sur une idée similaire de partitionnement de l'espace des paramètres $\mathcal{X}$.

4. Losrque la distribution d'intérêt est une distribution a posteriori $\pi(\theta \mid y_1, \ldots, y_{n_{obs}})$, Chopin [42] propose d'utiliser des distributions a posteriori partielles comme distributions auxiliaires : $\pi(\theta \mid y_1, \ldots, y_{n_1})$, $\pi(\theta \mid y_1, \ldots, y_{n_2})$, $\ldots$, $\pi(\theta \mid y_1, \ldots, y_{n_N})$ avec $n_1 < n_2 < \ldots < n_N = n_{obs}$. La motivation principale est de diminuer le fardeau computationnel lorsque les jeux de données sont très grands et qu'une itération utilisant l'ensemble des observations est longue. Cette méthode peut être très efficace lorsque les données sont naturellement ordonnées, comme dans les modèles de Markov cachés. Elle est principalement utilisée dans des méthodes de Monte Carlo Séquentielles (SMC) (voir Chopin [43] par exemple).

Calibration

Il faut être attentif au choix du nombre N de distributions auxiliaires à utiliser. Plus N est grand et plus une itération de l'algorithme présenté ci-dessus sera longue. Mais en même temps, il faut un N assez grand pour assurer une bonne diversité des mouvements d'échange, et donc un bon mélange de l'ensemble des N chaînes. Ce nombre N requis pour obtenir de bons résultats croît avec la complexité du problème.

Dans le cas de distributions auxiliaires qui sont des distributions tempérées de π, l'échantillonneur associé à la plus haute température doit pouvoir explorer facilement l'espace des paramètres $\mathcal{X}$. De plus, il ne faut pas que deux distributions adjacentes π_i et π_{i+1} soient trop différentes, auquel cas les échanges entre chaînes seraient trop peu acceptés. Le choix de N est donc guidé par ces deux conditions, afin d'assurer un bon mélange.
Sous certaines hypothèses, Atchadé et al. [12] ont montré que pour le Parallel Tempering et le Simulated Tempering, les températures optimales doivent être définies de manière à ce que la probabilité d'acceptation du mouvement d'échange de deux chaînes soit de 0.234. Les inverses des températures optimales n'ont donc pas forcément une progression géométrique, progression souvent utilisée pour choisir les températures (voir par exemple Nagata et Watanabe [164] ou Neal [166]). D'autres façons de calibrer les températures ont été proposées par Goswami et Liu [82].

Mouvement d'échange avec « delayed rejection »

Jasra et al. [108] ont développé un mouvement d'échange de deux chaînes avec « delayed rejection », en s'inspirant de Mira et Green [85] (voir supplément D.1.1). Ils se placent dans le cadre de distributions auxiliaires qui sont des distributions tempérées de π. Les mouvements d'échange proposés entre chaînes éloignées (de températures très différentes) sont alors plus ra-

rement acceptés que les mouvements d'échange proposés entre chaînes proches (de températures proches). En effet, les chaînes proches ont plus de chances d'avoir des états de vraisemblances proches, et donc une probabilité d'acceptation (5.4) élevée.

La proposition de Jasra et al. [108] est alors la suivante : deux chaînes sont choisies au hasard parmi les N chaînes, et un échange entre les deux est proposé. Si le mouvement est refusé, c'est peut être parce que les deux chaînes choisies sont trop éloignées l'une de l'autre. Un nouvel échange est alors proposé, mais entre deux chaînes adjacentes choisies au hasard parmi les $(N-1)$ paires possibles.

La probabilité d'acceptation de cette seconde proposition est définie dans le but de conserver la distribution stationnaire π^* (5.5) de l'ensemble des N chaînes.
Soit i_1 et i_2 $(i_1 < i_2)$ les indices des chaînes choisies pour le premier échange proposé. L'échange consiste à passer de $s = (x_1, \ldots, x_{i_1}, \ldots, x_{i_2}, \ldots, x_N)$ à $s' = (x_1, \ldots, x_{i_2}, \ldots, x_{i_1}, \ldots, x_N)$ (noyau de transition noté $q_1(s, s')$). Ce mouvement est accepté avec la probabilité $\rho_1(s, s')$ (5.4). Si le mouvement entre les chaînes i_1 et i_2 est refusé, deux chaînes adjacentes sont sélectionnées au hasard pour la seconde proposition. Notons i et $i+1$ les chaînes sélectionnées. Si cette seconde proposition est acceptée, les N chaînes passeront de $s = (x_1, \ldots, x_i, x_{i+1}, \ldots, x_N)$ à $s'' = (x_1, \ldots, x_{i+1}, x_i, \ldots, x_N)$. Le mouvement aller consiste alors à aller de s à s'' en passant par s' (noyau de transition noté $q_2(s, s'')$). Pour assurer la réversibilité, il faut imaginer un pseudo-mouvement qui est refusé. Ce mouvement propose de passer de s'' à s en passant par s^*, s^* correspondant à s'' pour lequel les chaînes d'indices i_1 et i_2 sont échangées. Notons que s^* n'est pas égal à s' (sauf si $i_1 = i$ et $i_2 = i+1$). Nous sommes donc dans le contexte de mouvements « delayed rejection » [85], qui généralisent les mouvements « splitting rejection » (voir Mira [159]).

La probabilité d'accepter la seconde proposition est :

$$\rho_2(s, s'') = 1 \wedge \frac{\pi_{i+1}(x_i)\pi_i(x_{i+1})\big(1 - \rho_1(s'', s^*)\big)}{\pi_i(x_i)\pi_{i+1}(x_{i+1})\big(1 - \rho_1(s, s')\big)}. \tag{5.6}$$

En effet, le noyau de transition pour passer de s à s'' s'écrit :

$$q_1(s, s'')\rho_1(s, s'') + \int q_1(s, s')\big(1 - \rho_1(s, s')\big)q_2(s, s'')\rho_2(s, s'')ds'.$$

A l'inverse, pour le retour, le noyau de transition pour passer de s'' à s s'écrit :

$$q_1(s'', s)\rho_1(s'', s) + \int q_1(s'', s^*)\big(1 - \rho_1(s'', s^*)\big)q_2(s'', s)\rho_2(s'', s^*, s)ds^*.$$

La condition de réversibilité par rapport à π^* est vérifiée si :

$$q_1(s, s')\big(1 - \rho_1(s, s')\big)q_2(s, s'')\rho_2(s, s'')\pi^*(s) = q_1(s'', s^*)\big(1 - \rho_1(s'', s^*)\big)q_2(s'', s)\rho_2(s'', s)\pi^*(s''),$$

et la plus petite probabilité $\rho_2(s, s'')$ vérifiant cette équation est

$$\rho_2(s, s'') = 1 \wedge \frac{\pi_{i+1}(x_i)\pi_i(x_{i+1})\big(1 - \rho_1(s'', s^*)\big)}{\pi_i(x_i)\pi_{i+1}(x_{i+1})\big(1 - \rho_1(s, s')\big)}.$$

En effet, $q_1(s, s') = q_1(s'', s^*)$ et $q_2(s'', s) = q_2(s, s'')$ car la probabilité de proposer s' à partir de s est la même que la probabilité de proposer s^* à partir de s'' (choix de deux indices parmi N), et la probabilité de proposer s'' à partir de s en ayant refusé s' est la même que la probabilité de proposer s à partir de s'' en ayant refusé s^* (choix d'une paire d'indices adjacents parmi $(N-1)$).

5.3.2 L'Evolutionary Monte Carlo

L'algorithme Parallel Tempering initialement développé et présenté dans la partie 5.3.1 ne contient qu'un seul type de mouvement global impliquant plus d'une chaîne : le mouvement d'échange. Liang et Wong ont proposé d'autres types de mouvements globaux [138, 139]. Ils se sont inspirés d'algorithmes génétiques (d'où l'appelation « Evolutionary Monte Carlo »), et ont défini ces mouvements de façon à ce qu'ils conservent la distribution stationnaire π^* de l'ensemble des chaînes (5.4). Dans cette partie, nous passons en revue ces différents mouvements.

Les distributions auxiliaires utilisées sont des distributions tempérées de la distribution cible π : $\pi_i(x) \propto \pi(x)^{1/T_i}, i = 1, \ldots, N$, avec $1 = T_1 < T_2 < \ldots < T_i < \ldots < T_N$. Nous utilisons les mêmes notations que dans la partie 5.3.1. Liang et Wong considèrent l'ensemble des états actuels des N chaînes $s = (x_1, x_2, \ldots, x_N)$ comme une population de N chromosomes.

Mutation et échange

Le mouvement de mutation est le mouvement d'actualisation local d'un seul chromosome avec un échantillonneur classique (ayant π_i comme distribution stationnaire si on actualise x_i), et le mouvement d'échange est tel que présenté dans la partie précédente 5.3.1.
Un algorithme avec seulement des mouvements de mutation et d'échange correspond alors à l'algorithme Parallel Tempering classique.
Liang et Wong [138, 139] ne suggèrent pas d'utiliser un mouvement d'échange avec « delayed rejection » (voir partie 5.3.1), mais par contre ils n'utilisent que des mouvements d'échange entre chaînes adjacentes.

Real crossover

Le terme « real » vient du fait que ce crossover est appliqué à des chromosomes prenant leurs valeurs dans $\mathbb{R}^d$. En effet, les chromosomes s'écrivent $x_i = (\beta_1^i, \ldots, \beta_d^i)$, les β_j^i étant réels et correspondant aux « bases ». Le mouvement se déroule de la façon suivante :

1. Deux chromosomes x_j et x_k sont sélectionnés :

(a) Le premier est choisi parmi l'ensemble des chromosomes munis de probabilités proportionnelles à leurs poids $w(x_i) \propto \pi(x_i)^{1/T_s}$, T_s étant une température de sélection proche ou égale à T_1.

(b) Le second est choisi de façon uniforme dans la population de chromosomes restants.

2. Deux nouveaux chromosomes y_j et y_k sont générés à partir de x_j et x_k, grâce à un crossover en un point, en plusieurs points, uniforme ou adaptatif.

(a) Pour un crossover en un point, ils sont obtenus de la façon suivante : un point de crossover c est choisi de manière uniforme sur $\{1, \ldots, d\}$, y_j et y_k sont alors obtenus en échangant leurs bases à droite de c.

$$\begin{array}{ll} x_j = (\beta_1^j, \ldots, \beta_c^j, \beta_{c+1}^j, \ldots, \beta_d^j) & y_j = (\beta_1^j, \ldots, \beta_c^j, \beta_{c+1}^k, \ldots, \beta_d^k) \\ & \Longrightarrow \\ x_k = (\beta_1^k, \ldots, \beta_c^k, \beta_{c+1}^k, \ldots, \beta_d^k) & y_k = (\beta_1^k, \ldots, \beta_c^k, \beta_{c+1}^j, \ldots, \beta_d^j) \end{array}$$

(b) Pour un crossover en plusieurs points, l'opération précédente est appliquée à chaque point de crossover.

(c) Pour un crossover uniforme, chaque base du chromosome y_j à générer est choisie aléatoirement entre les bases des chromosomes parents : la $l^{\text{ème}}$ base de y_j est choisie aléatoirement entre β_l^j et β_l^k, $l = 1, \ldots, d$. L'autre chromosome y_k est construit par complémentarité.

(d) Pour un crossover adaptatif, le processus est plus compliqué, voir Liang et Wong [138] ou Liu [142].

3. Ce mouvement de crossover actualise la population $s = \{x_1, \ldots, x_j, \ldots, x_k, \ldots, x_N\}$ par $s' = \{x_1, \ldots, y_j, \ldots, y_k, \ldots, x_N\}$. Notons $q(s, s')$ le noyau de transition pour passer de s à s'. C'est le produit de la probabilité de sélectionner x_j et x_k dans la population initiale et de la probabilité de générer y_j et y_k à partir de x_j et x_k, voir les formules dans Liang et Wong [138, 139]. Le mouvement est alors accepté avec la probabilité de Metropolis-Hastings suivante :

$$\rho(s, s') \quad = \quad 1 \wedge \frac{\pi_j(x_k)\pi_k(x_j)q(s', s)}{\pi_j(x_j)\pi_k(x_k)q(s, s')}.$$

Ce mouvement a plus de chances d'être accepté quand le point de crossover c est assez élevé (donc peu d'information échangée entre les chromosomes parents), et peut être vu comme un mouvement d'échange partiel. Cependant, il demande un coût computationnel plus élevé qu'un mouvement d'échange classique. Notamment dans un contexte de modèles de mélange, car les chromosomes parents doivent être labellisés avant de pouvoir s'échanger de l'information. En

effet, si on ne labellise pas avant le crossover, nous pouvons obtenir un individu avec une information en double sur un composant tout en ayant une information manquante sur un autre. Dans un algorithme de Parallel Tempering classique, la labellisation d'une chaîne peut se faire en « post-processing ».

Snooker crossover

Le mouvement snooker crossover est basé sur l'algorithme Snooker, qui est un cas particulier d'« Adaptive Direction Sampling » (voir Gilks et al. [80] et Roberts and Gilks [187]).
Un chromosome de la population est choisi aléatoirement, puis est actualisé en fonction d'une direction passant par un individu dit « d'ancrage ».

1. Un chromosome x_j est sélectionné uniformément dans la population des N chromosomes.
2. Un autre chromosome x_k est sélectionné dans la sous-population $s\backslash\{x_j\}$, avec une probabilité proportionnelle à $\pi(x_k)^{1/T_s}$, T_s étant une température de sélection proche ou égale à T_1. C'est le chromosome « d'ancrage ».
3. Une direction e est générée, $e = (x_j - x_k)/||x_j - x_k||$. Puis nous posons :

$$y_j = x_k + re,$$

avec r une variable aléatoire échantillonnée d'après la densité

$$f(r) \propto |r|^{d-1}\pi(x_k + re).$$

S'il est difficile d'échantillonner directement suivant f, Liang et Wong [139] suggèrent d'utiliser plusieurs mouvements de Metropolis-Hastings par exemple.

4. La nouvelle population actualisée est alors $s' = \{x_1, x_2, \ldots, y_j, \ldots, x_N\}$.

Ce mouvement tend à rapprocher les individus de la population. Il ne faut donc pas qu'il ait lieu trop souvent, sinon cela rapproche trop les chaînes les unes des autres et réduit la diversité de la population. Si toutes les chaînes sont proches au point d'être toutes dans le voisinage du même mode, nous perdons alors l'intérêt d'avoir plusieurs chaînes. Ici aussi dans un contexte de modèles de mélange les chromosomes parents doivent être labellisés avant d'effectuer le mouvement.

D'autres mouvements globaux ont été développés par Goswami et liu [82], et sont présentés dans le supplément D.1.2.

5.3.3 Cas trans-dimensionnel

Dans le cas de problèmes trans-dimensionnels (la dimension de l'espace des paramètres peut varier), les échantillonneurs classiquement utilisés sont à sauts réversibles (RJMCMC) (voir

Green [83] et Richardson et Green [184]). Cependant, il est connu qu'un tel échantillonneur a en général du mal à correctement explorer le support d'une distribution d'intérêt multimodale. Différentes améliorations ont donc été proposées (voir Brooks et al. [29], Al-Awadhi et al. [1] et Green [84] pour une revue de ces méthodes). Ces améliorations restent néanmoins dans le cadre d'une seule chaîne de Markov, et ne donnent pas toujours des résultats satisfaisants (voir Jasra et al. [108]). Ces derniers ont donc généralisé le principe du Parallel Tempering et de l'Evolutionary Monte Carlo au cas de problèmes trans-dimensionnels, en développant un algorithme combinant le principe des sauts réversibles et l'Evolutionary Monte Carlo.

Les mouvements locaux et globaux utilisés par Jasra et al. [108] sont des extensions des mouvements de l'Evolutionary Monte Carlo en dimension fixe (développés dans la partie 5.3.2). Nous les passons en revue dans cette partie.

La distribution d'intérêt π est définie sur un espace des états $\mathcal{X}$, qui est l'union de sous-espaces de dimensions différentes, soit

$$\mathcal{X} = \bigcup_{k \in \mathcal{K}} \big(\{k\} \times \mathcal{X}_k\big), \qquad \text{avec} \qquad \mathcal{K} \subseteq \mathbb{N} \qquad \text{et} \qquad \mathcal{X}_k \subseteq \mathbb{R}^{n_k}.$$

L'état de la $i^{\text{ème}}$ chaîne se note donc (k_i, x_i), avec x_i un chromosome de dimension n_{k_i}, appartenant à $\mathcal{X}_{k_i}$. Les distributions auxiliaires utilisées sont des distributions tempérées de la distribution cible π, avec $1 = T_1 < T_2 < \ldots < T_i < \ldots < T_N$. De plus, Jasra et al. [108] se placent dans un contexte de modèles de mélange. Les mouvements sont définis de façon à ce que la distribution stationnaire de l'ensemble des N chaînes soit :

$$\pi^*\Big((k_1, x_1), (k_2, x_2), \ldots, (k_N, x_N)\Big) = \prod_{i=1}^{N} \pi_i\Big((k_i, x_i)\Big) \qquad \text{sur} \qquad \mathcal{X}^N.$$

Mouvement de mutation

Le mouvement de mutation est le mouvement d'actualisation local d'un seul chromosome avec un échantillonneur à sauts réversibles (distribution stationnaire π_i pour le $i^{\text{ème}}$ chromosome).

Mouvement d'échange

C'est le même mouvement que celui présenté dans la partie 5.3.1, seules les notations s'alourdissent un peu : s'il est proposé d'échanger les chromosomes i_1 et i_2, l'échange est accepté avec la probabilité suivante :

$$1 \wedge \frac{\pi_{i_1}\Big((k_{i_2}, x_{i_2})\Big)\pi_{i_2}\Big((k_{i_1}, x_{i_1})\Big)}{\pi_{i_1}\Big((k_{i_1}, x_{i_1})\Big)\pi_{i_2}\Big((k_{i_2}, x_{i_2})\Big)}.$$

En cas de rejet de ce mouvement, Jasra et al. [108] préconisent d'en proposer un autre, et donc d'utiliser un mouvement Delayed Rejection (voir la section 5.3.1).

Mouvement crossover à dimension fixée

Ce mouvement ne peut se faire que si au moins deux chaînes ont des états de dimensions identiques. Si c'est le cas, deux chaînes de mêmes dimensions sont sélectionnées, avec des poids non uniformes, puis un mouvement real crossover est effectué (voir la partie 5.3.2).
Notons qu'il est nécessaire de labelliser les chromosomes avant le mouvement, et Jasra et al. [108] utilisent une contrainte d'identifiabilité. C'est la manière de labelliser la plus rapide, mais pas forcément la plus pertinente (voir annexe C.2).

Mouvement crossover à dimension variable

Ce mouvement ne peut se faire que si au moins deux chaînes ont des états de dimensions différentes. Si c'est le cas, deux chaînes de dimensions différentes sont sélectionnées (avec des poids non uniformes), puis labellisées en utilisant une contrainte d'identifiabilité sur les poids des composants. Les deux chaînes vont s'échanger leurs dimensions et les poids de leurs composants.
La chaîne de plus grande dimension va prendre la dimension la plus petite. Elle va donc « donner » des composants de faible poids à l'autre chaîne sélectionnée, en conservant à l'identique ses composants de poids les plus forts. La chaîne de plus petite dimension va prendre la dimension la plus grande, en « prenant » les composants de poids les plus faibles de l'autre chaîne, tout en gardant les siens.
L'inconvénient principal de ce mouvement est la nécessité de labelliser les chromosomes avant le mouvement, et Jasra et al. [108] considèrent que ses performances ne justifient pas son utilisation.

Mouvement snooker jump

Pour une chaîne donnée, le principe est de proposer un mouvement avec changement de dimension, en fonction d'une direction passant par une chaîne « d'ancrage » (comme dans le mouvement « snooker crossover » de la partie 5.3.2).
Nous devons choisir entre une création et une destruction de composant vide. Si nous avons choisi de créer un composant, nous choisissons une chaîne pour laquelle une telle création est possible (uniformément), ainsi qu'une chaîne « d'ancrage » (non uniformément). Nous créons ensuite un nouveau composant pour la première chaîne sélectionnée, à partir des paramètres des composants de la chaîne d'ancrage. Voir Jasra et al. [108] pour plus de détails ainsi que la formule de la probabilité d'acceptation. Le mouvement inverse de destruction de composant vide est défini de manière similaire.
Encore une fois Jasra et al. [108] considèrent que les performances de ce mouvement ne justifient pas son utilisation.

Echantillonage contraint

Les chaînes associées aux températures les plus basses restent souvent coincées dans des modes locaux, et n'explorent pas correctement l'espace des paramètres $\mathcal{X}$. Jasra et al. [108] proposent

de contraindre certaines chaînes de la population à rester dans un certain sous-espace de $\mathcal{X}$, et ainsi à en forcer l'exploration. Dans le cas de problèmes trans-dimensionnels, ils proposent de partitionner $\mathcal{X}$ en fonction des dimensions possibles. Ainsi une chaîne contrainte ne pourra prendre que certaines dimensions.
Ces chaînes contraintes ne peuvent intervenir que dans des mouvements crossover à dimension fixée, et dans des mouvements d'échange adaptés, puisqu'un échange entre une chaîne contrainte et une chaîne non contrainte ne peut être accepté que si la chaîne contrainte reste dans le sous-espace lui correspondant.

5.4 l'Equi-Energy Sampler

L'Equi-Energy Sampler (EES) a été développé par Kou et al. [118]. Tout comme le Parallel Tempering (5.3.1), il se base sur une population de chaînes associées à des températures. Cependant cet échantillonneur est très différent des échantillonneurs déjà présentés, car il utilise des informations du passé pour proposer des mouvements globaux, et ne génère pas un processus markovien.

Notons π la distribution d'intérêt que nous cherchons à échantillonner, pouvant s'écrire sous la forme $\pi(x) \propto \exp(-h(x))$ et définie sur un espace continu noté $\mathcal{X}$. $h(x)$ correspond à la fonction d'énergie. Une séquence de températures ordonnées est définie, $1 = T_1 < T_2 < \ldots < T_K$, ainsi qu'une séquence d'énergies $H_1 < H_2 < \ldots < H_K < H_{K+1} = \infty$, avec $H_1 \leq \inf_x h(x)$.
L'EES considère une série de K distributions auxiliaires notées $\tilde{\pi}_1, \tilde{\pi}_2, \ldots, \tilde{\pi}_K$. Elles sont indexées par une température et un niveau d'énergie, et définies par :

$$\tilde{\pi}_i(x) \propto \exp\Big(-\frac{h(x) \vee H_i}{T_i}\Big). \tag{5.7}$$

Pour i allant de 1 à K, nous supposons être capables de construire un algorithme de Metropolis-Hastings de distribution stationnaire $\tilde{\pi}_i$. Etant donné que $T_1 = 1$ et $H_1 \leq \inf_x h(x)$, $\tilde{\pi}_1$ correspond à la distribution d'intérêt π. Ces distributions auxiliaires sont des distributions tempérées modifiées. En effet, en comparaison les distributions tempérées s'écrivent (voir 5.2.1) :

$$\pi_i(x) \propto \exp\Big(-\frac{h(x)}{T_i}\Big).$$

Les distributions π_i et $\tilde{\pi}_i$ $(i = 1, \ldots, K)$ ont donc des caractéristiques similaires. En particulier, les échantillonneurs associés aux distributions auxiliaires de températures élevées peuvent se déplacer facilement dans l'espace des paramètres.

L'EES se base principalement sur un mouvement appelé « Equi-Energy jump » : l'état actuel

d'une chaîne est proposé d'être remplacé par un état d'énergie similaire appartenant au passé de la chaîne d'ordre supérieur. Pour cela l'espace des états $\mathcal{X}$ est partitionné en fonction des niveaux d'énergie :

$$\mathcal{X} = \bigcup_{j=1}^{K} D_j, \qquad \text{avec} \qquad D_j = \big\{x : h(x) \in [H_j, H_{j+1})\big\}, \quad 1 \leq j \leq K.$$

Les D_j regroupent donc des états d'énergies similaires. Pour tout $x \in \mathcal{X}$, l'indice $I(x)$ est défini tel que $I(x) = j$ si $x \in D_j$, soit si $h(x) \in [H_j, H_{j+1})$.

REMARQUE 5.4.1 *Un partitionnement de $\mathcal{X}$ a également été préconisé par Atchadé et Liu [11] et Mitsutake et al. [161]. Ils utilisent ce partitionnement en combinaison avec des arguments d'échantillonnage préférentiel (« importance sampling »).*

L'algorithme La figure 5.5 représente l'algorithme décrit ci-dessous.
Chaîne K : L'algorithme commence par générer une chaîne $X^{(K)}$, à l'aide d'un algorithme de Metropolis-Hastings (MH) ayant $\tilde{\pi}_K$ comme distribution stationnaire. Après *nburn* itérations, l'échantillonneur construit les anneaux d'énergie d'ordre K, les $\widehat{D}_j^{(K)}$, en regroupant les valeurs échantillonnées par niveaux d'énergie. $\widehat{D}_j^{(K)}$ regroupe les valeurs échantillonnées $X_n^{(K)}$ telles que $I(X_n^{(K)}) = j$.

Chaîne K − 1 : Après *nRings* itérations supplémentaires, l'échantillonneur commence à générer la chaîne $X^{(K-1)}$. Il continue pendant ce temps à générer la chaîne d'ordre supérieur $X^{(K)}$ et les anneaux d'énergie $\widehat{D}_j^{(K)}$. La chaîne $X^{(K-1)}$ est actualisée à chaque itération de deux façons possibles. Avec une probabilité $1 - p_{ee}$, l'état actuel $X_n^{(K-1)}$ est actualisé localement par un algorithme de MH ayant $\tilde{\pi}_{K-1}$ comme distribution stationnaire. Avec une probabilité p_{ee}, l'état actuel $X_n^{(K-1)}$ est actualisé par un saut Equi-Energy : un état y est choisi uniformément dans l'anneau de la chaîne d'ordre supérieur $\widehat{D}_j^{(K)}$, où $j = I(X_n^{(K-1)})$ correspond au niveau d'énergie de $X_n^{(K-1)}$. Cet état proposé y est accepté pour être le prochain état de la chaîne $X_{n+1}^{(K-1)}$ avec la probabilité $1 \wedge \frac{\tilde{\pi}_{K-1}(y)\tilde{\pi}_K(X_n^{(K-1)})}{\tilde{\pi}_{K-1}(X_n^{(K-1)})\tilde{\pi}_K(y)}$. Si y est non accepté, $X_{n+1}^{(K-1)}$ prend l'ancienne valeur $X_n^{(K-1)}$. Après *nburn* itérations, l'échantillonneur construit les anneaux d'énergie d'ordre $K-1$, les $\widehat{D}_j^{(K-1)}$, en regroupant les valeurs échantillonnées par niveaux d'énergie. $\widehat{D}_j^{(K-1)}$ regroupe les valeurs échantillonnées $X_n^{(K-1)}$ telles que $I(X_n^{(K-1)}) = j$.

Chaîne K − 2 : Après *nRings* itérations supplémentaires, l'échantillonneur commence à générer la chaîne $X^{(K-2)}$. Il continue pendant ce temps à générer les chaînes d'ordre supérieur $X^{(K)}$ et $X^{(K-1)}$ ainsi que les anneaux d'énergie $\widehat{D}_j^{(K)}$ et $\widehat{D}_j^{(K-1)}$. Tout comme la chaîne précédente, cette chaîne est actualisée localement par un algorithme de MH ayant $\tilde{\pi}_{K-2}$ comme distribution

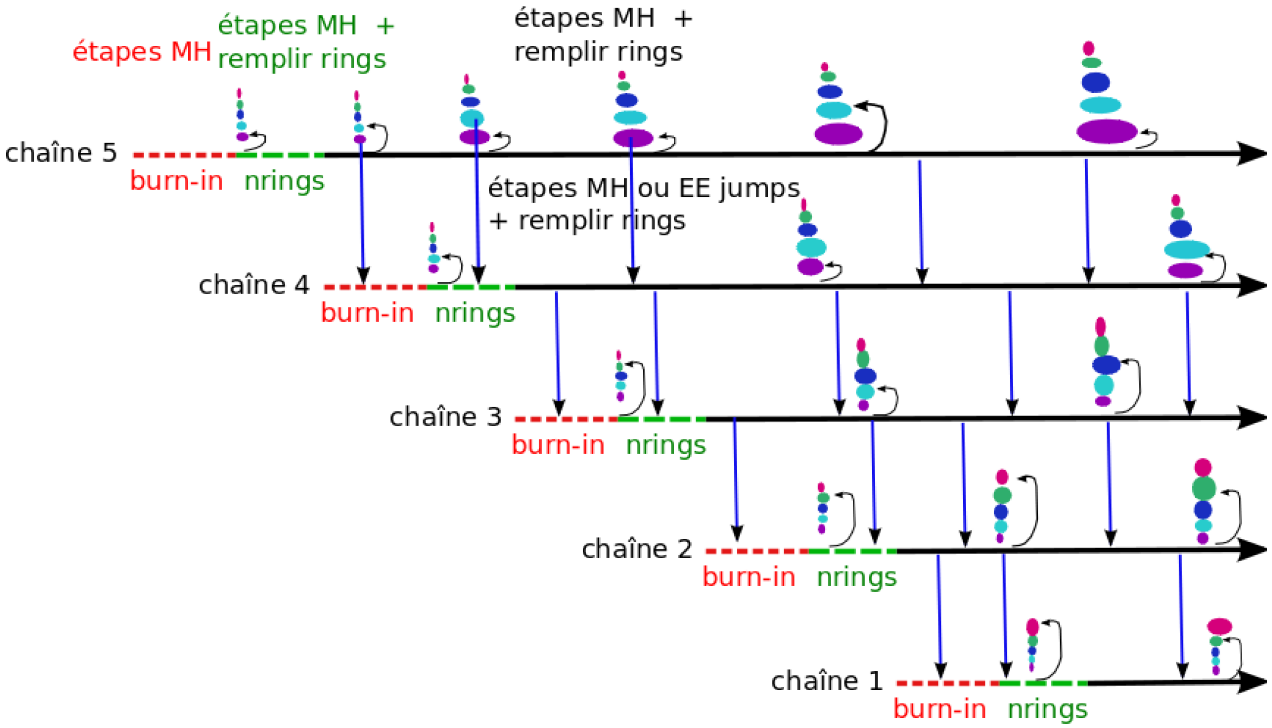

FIGURE 5.5 – Représentation de l'algorithme EES. Pour chaque chaîne sont représentées les itérations de burn-in, de remplissage des anneaux, et les suivantes. Les flèches bleues symbolisent un saut Equi-Energy, et les cercle ovales représentent les anneaux d'énergies, avec une couleur par anneau. Ils sont de plus en plus gros car ils se remplissent au fur et à mesure des itérations.

stationnaire et par des sauts Equi-Energy, avec des probabilités $1 - p_{ee}$ et p_{ee} respectivement. Concernant le saut Equi-Energy, un état y est choisi uniformément dans l'anneau de la chaîne d'ordre supérieur $\widehat{D}_j^{(K-1)}$, où $j = I(X_n^{(K-2)})$ correspond au niveau d'énergie de l'état actuel de la chaîne $X_n^{(K-2)}$. Cet état est accepté avec la probabilité $1 \wedge \frac{\tilde{\pi}_{K-2}(y)\tilde{\pi}_{K-1}(X_n^{(K-2)})}{\tilde{\pi}_{K-2}(X_n^{(K-2)})\tilde{\pi}_{K-1}(y)}$.

Autres chaînes : L'échantillonneur va lancer les K chaînes successivement. Chaque chaîne $X^{(i)}$, $1 \leq i < K$, est actualisée localement grâce à un algorithme de Metropolis-Hastings (avec une probabilité $1-p_{ee}$), ou grâce à un saut Equi-Energy (avec une probabilité p_{ee}). Lors de ce dernier, un état y choisi uniformément dans l'anneau de la chaîne d'ordre supérieur $\widehat{D}_{I(X_n^{(i)})}^{(i+1)}$ est proposé, et est accepté avec la probabilité $1 \wedge \frac{\tilde{\pi}_i(y)\tilde{\pi}_{i+1}(X_n^{(i)})}{\tilde{\pi}_i(X_n^{(i)})\tilde{\pi}_{i+1}(y)}$. De plus, après une période de burn-in, les anneaux d'énergie d'ordre i (les $\widehat{D}_j^{(i)}$) sont construits, et utilisés pour la prochaîne chaîne $X^{(i-1)}$. Ainsi de suite, jusqu'à atteindre la distribution d'intérêt $\tilde{\pi}_1$.

Equi-Energy Sampler

Soit i l'indice de la chaîne, j l'indice de l'anneau, $nburn$ la période de burn-in, et $nrings$ la période de remplissage des anneaux.
Pour tout i, j, initialiser $X_0^{(i)}$ et poser $\widehat{D}_j^{(i)} = \varnothing$.
Pour $n = 1, 2, \ldots$:
  Pour i allant de K à 1:
    Si $n > (K-i)(nburn + nrings)$:
      Si $i = K$ ou si $\widehat{D}^{(i+1)}_{I(X_{n-1}^{(i)})} = \varnothing$:
        partir de $X_{n-1}^{(i)}$ et faire une étape MH de distribution stationnaire $\tilde{\pi}_i$ pour obtenir $X_n^{(i)}$.
      Sinon:
        Avec proba $1-p_{ee}$: partir de $X_{n-1}^{(i)}$ et faire une étape MH de distribution stationnaire $\tilde{\pi}_i$ pour obtenir $X_n^{(i)}$.
        Avec proba p_{ee}: choisir y uniformément dans $\widehat{D}^{(i+1)}_{I(X_{n-1}^{(i)})}$, puis $X_n^{(i)}$ <- y avec proba $1 \wedge \frac{\tilde{\pi}_i(y)\tilde{\pi}_{i+1}(X_{n-1}^{(i)})}{\tilde{\pi}_i(X_{n-1}^{(i)})\tilde{\pi}_{i+1}(y)}$, $X_n^{(i)}$ <- $X_{n-1}^{(i)}$ sinon.
    Si $n > (K-i)(nburn + nrings) + nburn$:
      $\widehat{D}^{(i)}_{I(X_n^{(i)})}$ <- $\widehat{D}^{(i)}_{I(X_n^{(i)})} + \{X_n^{(i)}\}$

5.4.1 Le saut Equi-Energy

Lors de ce saut, l'état actuel d'une chaîne est proposé d'être remplacé par un état d'énergie similaire appartenant au passé de la chaîne d'ordre supérieur. En effet, les anneaux d'une chaîne regroupent des états du passé de cette chaîne d'énergies similaires. Ce mouvement n'est donc pas un mouvement d'échange, une seule chaîne est actualisée.

Le principe de ce saut est intéressant, car il permet un saut entre deux états d'énergies similaires mais pouvant être séparés par des « barrières » énergétiques (barrières correspondant aux vallées de faible densité dans la figure 5.4). De plus, comme les états ont des énergies similaires, le saut a en général plus de chances d'être accepté qu'un mouvement d'échange classique lors duquel les chaînes sélectionnées pour échanger leurs états sont choisies totalement au hasard.

5.4.2 Les distributions auxiliaires

Dans les distributions auxiliaires utilisées, des tronquations interviennent pour assurer aux différentes chaînes des niveaux d'énergie minimums (H_i/T_i pour la $i^{\text{ème}}$ chaîne) :

$$\tilde{\pi}_i(x) \quad \propto \quad \exp\Big(-\frac{h(x) \vee H_i}{T_i}\Big) \quad \propto \quad \pi(x)^{\frac{1}{T_i}} \wedge \exp\Big(-\frac{H_i}{T_i}\Big).$$

Prenons l'exemple simple d'une loi normale univariée, soit π la densité d'une $\mathcal{N}(m, \sigma^2)$, et $\pi^{\frac{1}{T_i}}$ la densité d'une $\mathcal{N}(m, \sigma^2 T_i)$. Les densités π, $\pi^{\frac{1}{T_i}}$ et $\tilde{\pi}_i$ correspondantes sont représentées dans la figure 5.4.2.

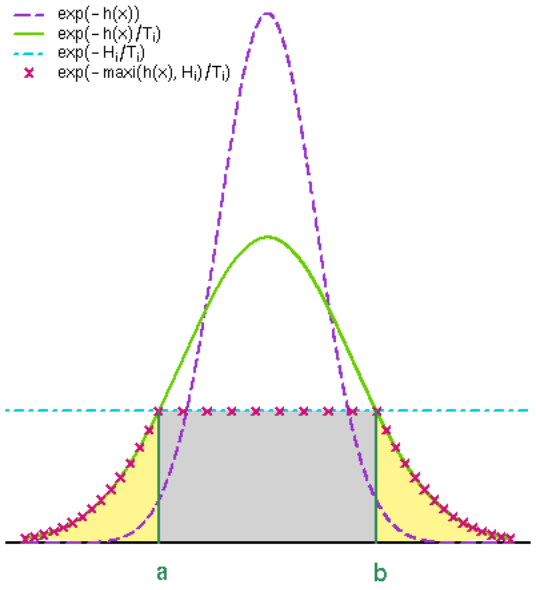

FIGURE 5.6 – Densités de π, $\pi^{\frac{1}{T_i}}$ et $\tilde{\pi}_i$ sur un exemple simple de loi normale univariée. la courbe violette en pointillés représente π, la courbe verte pleine représente $\pi^{\frac{1}{T_i}}$, la droite bleue en pointillés représente la constante de troncation $\exp(-H_i/T_i)$, et la courbe rose en croix représente $\tilde{\pi}_i$. Nous avons utilisé $\sigma^2 = 4$, $T_i = 3$ et $H_i = 9$.

La troncation « étête » la distribution tempérée $\pi^{\frac{1}{T_i}}$, pour obtenir $\tilde{\pi}_i$ qui n'a donc plus un mode mais un « plateau » modal. En utilisant de telles distributions « étêtées », l'exploration autour des modes de π est facilitée, car tous les états du « plateau » modal ont les mêmes densités.

5.4.3 Echantillonneur local

L'Equi-Energy Sampler (EES) tel que décrit par Kou et al. [118] n'utilise que des algorithmes de Metropolis-Hastings pour les actualisations locales des chaînes. Cependant, dans de nombreux problèmes l'utilisation d'un échantillonneur de Gibbs est préférable à celle d'un Metropolis-Hastings. Or il paraît difficile d'utiliser un échantillonneur de Gibbs pour les actualisations locales. En effet, pour la $i^{\text{ème}}$ chaîne, l'échantillonneur de Gibbs doit avoir $\tilde{\pi}_i$ comme distribution stationnaire. Or le fait que $\tilde{\pi}_i$ soit une distribution « étêtée » peut compliquer l'échantillonnage. Reprenons l'exemple simple d'une loi normale univariée du paragraphe précédent : en notant Φ la fonction de répartition de la loi normale centrée réduite, l'aire sous la courbe

représentant la densité $\tilde{\pi}_i(x)$ vaut (voir figure 5.4.2) :

$$\mathcal{A} = \Phi\Big(\frac{a-m}{\sigma\sqrt{T_i}}\Big) + (b-a)\exp\Big(-\frac{H_i}{T_i}\Big) + \Phi\Big(-\frac{b-m}{\sigma\sqrt{T_i}}\Big).$$

Ainsi, échantillonner suivant $\tilde{\pi}_i$ revient à échantillonner :

- Suivant une $\mathcal{N}(m, \sigma^2 T_i)\mathbb{1}_{]-\infty,a[}$ avec une probabilité $\Phi\Big(\frac{a-m}{\sigma\sqrt{T_i}}\Big)/\mathcal{A}$.
- Suivant une $\mathcal{U}_{[a,b]}$, avec une probabilité $(b-a)\exp\Big(-\frac{H_i}{T_i}\Big)/\mathcal{A}$.
- Suivant une $\mathcal{N}(m, \sigma^2 T_i)\mathbb{1}_{]b,+\infty]}$, avec une probabilité $\Phi\Big(-\frac{b-m}{\sigma\sqrt{T_i}}\Big)/\mathcal{A}$.

Cela est évidemment plus fastidieux et plus long que de devoir échantillonner tout simplement suivant une $\mathcal{N}(m, \sigma^2 T_i)$. Nous donnons ici un exemple simple, mais dans le cas général d'une distribution $\tilde{\pi}_i(x)$ multivariée et multimodale, il apparaît difficile d'échantillonner suivant $\tilde{\pi}_i(x)$ simplement et rapidement. Nous pourrions utiliser des algorithmes d'acceptation-rejet ou des méthodes sans vraisemblance (voir chapitre 7) par exemple, mais alors le temps de calcul s'allongerait et rendrait sans doute l'EES trop difficile d'utilisation.

5.4.4 Performances et stockage

Les performances de cet algorithme sont très satisfaisantes, bien meilleures en général que celles d'un Parallel Tempering classique (voir Kou et al. [118] ou les exemples du chapitre 6). Cependant, il faut noter que cet algorithme est assez long en pratique car pour lancer une nouvelle chaîne nous devons attendre que la chaîne d'ordre supérieur ait convergé et pré-rempli ses anneaux. De plus, l'EES nécessite un stockage important. En effet, chacune des chaînes stocke tous ses états passés dans ses anneaux d'énergie. Minary et Levitt [158] ont proposé de fixer une taille limite pour chaque anneau d'énergie. Cependant, il faut tout de même garder en mémoire un nombre conséquent d'états pour chaque anneau de chaque chaîne.

5.4.5 Convergence

D'un point de vue théorique, l'EES n'est pas un processus Markovien, puisque qu'une chaîne d'ordre $i < K$ utilise tout le passé de la chaîne d'ordre supérieur. Une notion de temps intervient, les chaînes générées ont un sens, et la réversibilité de l'ensemble des N chaînes n'est pas vérifiée (de même que la « detailed balance condition »). C'est la nature non markovienne de cet échantillonneur qui rend la preuve de sa convergence non triviale. En effet, cette preuve a une longue histoire. En 2006, Kou et al. [118] ont proposé un théorème visant à montrer que sous certaines hypothèses les différentes chaînes générées sont ergodiques de distributions stationnaires $\tilde{\pi}_i$, et qu'en particulier la première chaîne a π comme distribution stationnaire. Cependant, Atchadé et Liu [10] ont considéré que le théorème développé par Kou et al. était incomplet. Ils ont donc proposé une nouvelle preuve, avec des hypothèses plus fortes. Mais Andrieu et al. [4] ont montré

en 2007 que cette nouvelle preuve, bien que théoriquement correcte, ne correspond pas à l'algorithme de l'EES tel que décrit par Kou et al. [118]. Ils ont discuté les preuves précédentes ainsi que les difficultés rencontrées. Puis, dans un autre article [6], ils ont développé une nouvelle preuve de la convergence basée sur la théorie des MCMC non linéaires (voir Andrieu et al. [5]). Cependant cette preuve suppose des hypothèses assez fortes, et n'est valable que dans le cas $K = 2$. Ensuite, en 2010 Hua et Kou [99] ont proposé une preuve de la convergence dans le cas d'un espace des états $\mathcal{X}$ dénombrable. Ce n'est que récemment, en 2011, que des résultats de convergence généraux ont été établis par Atchadé et al. [9] et Fort et al. [65], pour des algorithmes adaptatifs de Monte Carlo par chaînes de Markov en interaction (dont fait partie l'EES). Notons que les hypothèses à vérifier pour obtenir ces résultats de convergence nous semblent assez lourdes. Remarquons qu'en 2010 Atchadé [8] a montré que les variances asymptotiques des algorithmes MCMC adaptatifs sont au moins aussi élevées que les variances asymptotiques des algorithmes MCMC correspondants, et que la différence entre les deux peut être substantielle. De ce point de vue, ils considèrent que l'EES est moins efficace qu'une chaîne de Markov qui aurait π^* comme distribution stationnaire (comme le Parallel Tempering par exemple).

5.4.6 Méthodes similaires

Atchadé [7] a récemment proposé deux méthodes qui comme l'EES ré-utilisent des informations du passé.

1. La première méthode contient une seule chaîne, qui lors de la plupart des itérations est actualisée par un échantillonneur MCMC classique, et lors d'itérations prédéterminées remplace son état actuel par un état de son passé (hors burn-in). La chaîne obtenue n'est pas markovienne, et contrairement à ce qui était attendu par Atchadé, le fait de ré-échantillonner à partir du passé ralentit la convergence.
2. La seconde méthode (Importance-Resampling MCMC) se rapproche plus de l'EES car plusieurs chaînes sont lancées en parallèle, et peuvent interagir avec leurs passés (la $i^{\text{ème}}$ chaîne utilise le passé de la $(i-1)^{\text{ème}}$). Par contre cet échantillonneur n'utilise pas de sauts Equi-Energy. De plus il est basé sur de l'échantillonnage d'importance alors que l'EES se base sur des probabilités de type Metropolis-Hastings. L'IR-MCMC donne en général de moins bons résultats que l'EES, tout en ayant comme ce dernier les inconvénients liés aux algorithmes adaptatifs de Monte Carlo par chaînes de Markov en interaction (stockage des états passés, variance asymptotique biaisée [8]). Cependant, l'IR-MCMC est plus facile d'utilisation que l'EES, car il n'est pas nécessaire de construire des anneaux d'énergie.

Chapitre 6

Méthode de type *Parallel Tempering* avec un Mouvement Equi-Energy

L'Equi-Energy Sampler développé par Kou et al. [118] donne de très bons résultats en pratique, bien meilleurs que ceux du Parallel Tempering. Ces performances sont notamment dues à la pertinence du saut Equi-Energy : l'état choisi pour remplacer un état actuel d'une chaîne a de grandes chances d'être accepté, puisque ces deux états ont des énergies similaires. De plus, les deux états peuvent être séparés par des « barrières » énergétiques, ce qui permet une bonne exploration de l'espace des paramètres. Cependant cet échantillonneur a plusieurs inconvénients. D'un point de vue théorique, sa convergence n'est pas facile à établir et sa variance asymptotique est au moins aussi élevée que celle d'un échantillonneur MCMC de même distribution cible (voir Atchadé [8]). D'un point de vue pratique, il nécessite un stockage important et est difficile d'utilisation en combinaison avec un échantillonneur de Gibbs (voir 5.4.3).

Notre objectif est donc de développer une méthode MCMC de type Parallel Tempering, ayant au moins d'aussi bons résultats que l'EES en pratique, mais sans les inconvénients. Cette méthode, appelée Parallel Tempering with Equi-Energy Moves (PTEEM), se base comme l'EES sur une population de chaînes, et un mouvement d'échange Equi-Energy est défini. Ce mouvement conserve l'idée originale de Kou et al. [118] qui est de partitionner l'espace des états en fonction de l'énergie.

6.1 Parallel Tempering with Equi-Energy Moves

6.1.1 Principe et algorithme

Notations

Soit π la distribution de probabilité d'intérêt que nous cherchons à échantillonner. Nous notons également π la densité associée par rapport à une mesure de probabilité λ. La densité π peut s'écrire sous la forme $\pi(x) \propto \exp(-h(x))$, et est définie sur un espace des états $\mathcal{X}$ à base dénombrable et muni de la tribu borélienne $\mathcal{B}(\mathcal{X})$. $\mathcal{X}$ est le support de la distribution de probabilité π, et $h(x)$ correspond à la fonction d'énergie.

Une séquence de températures ordonnées est définie, $1 = T_1 < T_2 < \ldots < T_N$. Nous considérons $\{\pi_1, \pi_2, \ldots, \pi_N\}$ une série de N distributions de probabilité auxiliaires, admettant des densités elles aussi notées $\pi_1, \pi_2, \ldots, \pi_N$, qui sont indexées par les températures : $\pi_i(x) \propto \pi(x)^{\frac{1}{T_i}}$, $i = 1, \ldots, N$. Etant donné que $T_1 = 1$, π_1 correspond à la distribution d'intérêt π.
Pour chacune des π_i nous supposons être capables de construire une chaîne de Markov ayant π_i comme distribution stationnaire, à l'aide d'un algorithme MCMC classique.

Nous notons x_i l'état actuel de la chaîne d'indice i. L'état de l'ensemble des chaînes peut se noter $s = (x_1, \ldots, x_N)$. La tribu produit sur $\mathcal{X}^N$ se note $\mathcal{B}(\mathcal{X})^N = \mathcal{B}(\mathcal{X}^N)$, et la mesure produit est notée λ_N. Soit A appartenant à $\mathcal{B}(\mathcal{X})^N$, il peut s'écrire $A = A_1 \times \ldots \times A_N$, les $A_i \in \mathcal{B}(\mathcal{X})$, et $\lambda_N(A) = \prod_{i=1}^N \lambda(A_i)$.

Une séquence d'énergies ordonnées est également définie, $H_1 < \ldots < H_d < H_{d+1} = \infty$, avec $H_1 \leq \inf_x h(x)$. Contrairement à Kou et al. [118], ces niveaux ne sont pas utilisés pour « étêter » les densités auxiliaires, afin de pouvoir facilement utiliser le PTEEM en combinaison avec des échantillonneurs de Gibbs ou des échantillonneurs à sauts réversibles.

Nous cherchons à construire un processus Markovien ayant comme distribution stationnaire la distribution de probabilité suivante :

$$\pi^*(dx_1, dx_2, \ldots, dx_N) = \prod_{i=1}^{N} \pi_i(x_i)\lambda(dx_i) \qquad \text{sur} \qquad (\mathcal{X}^N, \mathcal{B}(\mathcal{X})^N). \tag{6.1}$$

Principe du mouvement Equi-Energy

Le mouvement Equi-Energy est un mouvement global permettant à deux chaînes de s'échanger leurs états actuels. Nous n'utilisons pas de notion d'état passé, afin d'assurer la réversibilité. Ce mouvement est donc très similaire à un mouvement d'échange du Parallel Tempering, seule

la façon de choisir les deux chaînes intervenant dans le mouvement est modifiée.

Nous utilisons des ensembles d'énergies D_j que nous définissons comme ceux de Kou et al. [118], excepté le premier. L'espace des états $\mathcal{X}$ est partitionné en fonction des niveaux d'énergie, $\mathcal{X} = \bigcup_{j=1}^{d} D_j$, avec :

$$\begin{aligned} D_j &= \{x \in \mathcal{X} : h(x) \in [H_j, H_{j+1})\}, \quad j = 2, \ldots, d \\ D_1 &= \{x \in \mathcal{X} : h(x) \in (-\infty, H_2)\}. \end{aligned}$$

Nous définissons ensuite des anneaux d'énergie $\mathcal{R}_j$, qui contiennent les états actuels des N chaînes : la chaîne d'indice i appartient à l'anneau $\mathcal{R}_j$ si son état actuel x_i appartient à D_j, soit si $h(x_i) \in [H_j, H_{j+1})$. Il y a donc d anneaux partitionnant les N chaînes en fonction des énergies de leurs états actuels, et il est nécessaire d'avoir $N > d$. Contrairement à l'EES, les anneaux sont définis pour l'ensemble des chaînes, chaque chaîne n'a pas ses propres anneaux. De plus, ils ne contiennent pas d'états passés, un stockage de ces derniers n'est pas nécessaire.

Pour le mouvement d'échange Equi-Energy, un anneau d'énergie $\mathcal{R}_{j^*}$ contenant au moins deux chaînes est choisi aléatoirement ($j^* \in \{1, \ldots, d\}$). Deux chaînes d'indices i_1 et i_2 ($i_1 < i_2$) sont choisies uniformément parmi les chaînes de cet anneau, et un mouvement d'échange des états actuels de ces deux chaînes est ensuite proposé. L'échange consiste donc à passer de l'état $s = (x_1, \ldots, x_{i_1}, \ldots, x_{i_2}, \ldots, x_N)$ à l'état $s' = (x_1, \ldots, x_{i_2}, \ldots, x_{i_1}, \ldots, x_N)$.
Afin d'assurer la réversibilité de ce mouvement, ce dernier est accepté avec la probabilité d'acceptation suivante (voir partie 6.1.2) :

$$\rho(s, s') = 1 \wedge \frac{\pi^*(s')}{\pi^*(s)} = 1 \wedge \frac{\pi_{i_1}(x_{i_2})\pi_{i_2}(x_{i_1})}{\pi_{i_1}(x_{i_1})\pi_{i_2}(x_{i_2})}. \tag{6.2}$$

Remarquons que si le dénominateur est égal à 0, le numérateur l'est aussi (les $\pi_i \propto \pi^{\frac{1}{T_i}}$). Alors par convention $\rho(s, s') = 0$.

Les chaînes choisies pour effectuer un échange Equi-Energy sont dans un même anneau, et ont donc des états d'énergies similaires. Ce mouvement a donc plus de chance d'être accepté qu'un mouvement d'échange classique, où deux chaînes sont choisies totalement au hasard parmi les N pour échanger leurs états. De ce point de vue, le mouvement Equi-Energy est plus pertinent que le mouvement d'échange classique.
Contrairement au saut Equi-Energy de l'EES, le mouvement Equi-Energy est un échange entre deux chaînes, et la réversibilité du mouvement par rapport à la distribution π^* est assurée (voir la proposition 6.1.1).

Algorithme

Le PTEEM se base sur N chaînes lancées en parallèle et en même temps. Des mouvements locaux propres à chaque chaîne et des mouvements Equi-Energy entre chaînes sont utilisés pour actualiser le processus.

L'algorithme est le suivant :

Parallel Tempering with Equi-Energy Moves (PTEEM)

Initialiser la population des chaînes $\{x_1, x_2, \ldots, x_N\}$.
A chaque itération de l'algorithme, chaque chaîne est actualisée localement, puis un mouvement d'échange Equi-Energy entre deux chaînes est proposé, et accepté suivant une certaine probabilité d'acceptation:

1. Pour $i = 1, 2, \ldots, N$, actualiser la $i^{\text{ème}}$ chaîne avec un échantillonneur MCMC ayant π_i comme distribution stationnaire. Cet échantillonneur peut être aussi bien un Metropolis-Hastings qu'un échantillonneur de Gibbs ou qu'un échantillonneur à sauts réversibles.
2. Mouvement Equi-Energy: choisir uniformément un anneau d'énergie $j^* \in \{1, \ldots, d\}$ contenant au moins deux chaînes, puis choisir deux indices i_1 et i_2 $(i_1 < i_2)$ au hasard dans $\{1, 2, \ldots, N\}$. Proposer d'échanger les états actuels des chaînes d'indices i_1 et i_2.
   L'état de l'ensemble des chaînes se note $s = (x_1, \ldots, x_{i_1}, \ldots, x_{i_2}, \ldots, x_N)$, et l'échange consiste à passer de s à $s' = (x_1, \ldots, x_{i_2}, \ldots, x_{i_1}, \ldots, x_N)$.
   Cet échange est accepté avec la probabilité suivante:

$$\rho(s, s') = 1 \wedge \frac{\pi_{i_1}(x_{i_2})\pi_{i_2}(x_{i_1})}{\pi_{i_1}(x_{i_1})\pi_{i_2}(x_{i_2})}.$$

6.1.2 Propriétés théoriques

Notons S la chaîne de Markov sur $(\mathcal{X}^N, \mathcal{B}(\mathcal{X})^N)$ obtenue par l'algorithme PTEEM, un état de S est noté s. Le noyau de transition associé à une itération de PTEEM est noté P, et P^k est le noyau de transition associé à k itérations. Ils sont définis sur $(\mathcal{X}^N \times \mathcal{B}(\mathcal{X})^N)^2$. Le noyau de transition associé au mouvement local de la chaîne d'indice i est noté PL_i et est défini sur $(\mathcal{X} \times \mathcal{B}(\mathcal{X}))^2$. Le noyau associé à l'ensemble des mouvements locaux des N chaînes lors d'une itération de PTEEM est noté PL, défini sur $(\mathcal{X}^N \times \mathcal{B}(\mathcal{X})^N)^2$. Le noyau associé au mouvement global Equi-Energy est PE sur $(\mathcal{X}^N \times \mathcal{B}(\mathcal{X})^N)^2$. En notant

$$s = (x_1, \ldots, x_N), \qquad s' = (x'_1, \ldots, x'_N), \qquad \text{et} \qquad \tilde{s} = (\tilde{x_1}, \ldots, \tilde{x_N}),$$

Nous avons :

$$\begin{aligned} PL(s,s') &= \prod_{i=1}^{N} PL_i(x_i, x_i'), \\ P(s,s') &= (PE * PL)(s,s') = \int_{\mathcal{X}^N} PE(\tilde{s},s')PL(s,\tilde{s})d\tilde{s}. \end{aligned}$$

Nous notons $q(s,s')$ la distribution auxiliaire pour proposer s' à partir de s lors d'un mouvement d'échange Equi-Energy.
La norme en variation totale pour une mesure signée bornée μ sur $(\mathcal{X}^N, \mathcal{B}(\mathcal{X})^N)$ est définie par :

$$\|\mu\|_{TV} = \sup_{A\in\mathcal{B}(\mathcal{X})^N} \mu(A) - \inf_{A\in\mathcal{B}(\mathcal{X})^N} \mu(A) = \sup_{A\in\mathcal{B}(\mathcal{X})^N} |\mu(A)|$$

Proposition 6.1.1 *Si les noyaux de transition des mouvements locaux sont réversibles de distributions stationnaires π_i, apériodiques et fortement λ-irréductibles ($i = 1, \ldots, N$), alors S est réversible, fortement λ_N-irréductible et nous avons pour π^*-presque tout s :*

$$\lim_{n\to\infty} ||P^n(s,.) - \pi^*||_{TV} = 0 \qquad \pi^* - ps$$

La distribution stationnaire de S est π^*, et la chaîne associée à $T_1 = 1$ a des réalisations issues de $\pi_1 = \pi$ qui est la distribution d'intérêt.

Preuve :

1. La chaîne d'indice i est actualisée localement par un échantillonneur MCMC réversible de distribution stationnaire π_i. Il est clair qu'alors $PL = \prod_{i=1}^{N} PL_i$ est également réversible, et est donc stationnaire. Montrons qu'il est de distribution stationnaire π^*.
   Soit $A \in \mathcal{B}(\mathcal{X})^N$, il peut s'écrire sous la forme $A_1 \times A_2 \times \ldots \times A_N$ avec les $A_i \in \mathcal{B}(\mathcal{X})$. Nous pouvons alors écrire :

$$\begin{aligned} \pi^*(A) &= \prod_{i=1}^{N} \pi_i(A_i) \\ &= \prod_{i=1}^{N} \int_{\mathcal{X}} PL_i(x_i, A_i)\pi_i(dx_i) \\ &= \int_{\mathcal{X}^N} PL(s,A)\pi^*(ds). \end{aligned}$$

   Par conséquent PL est bien de distribution stationnaire π^*.

2. Montrons que PE le noyau de transition du mouvement d'échange Equi-Energy est réver-

sible de distribution stationnaire π^*. Le noyau de transition s'écrit de la façon suivante :

$$PE(s,s') = q(s,s')\rho(s,s') + \int_{\mathcal{X}} q(s,s'')(1-\rho(s,ds''))\mathbb{1}_{\{s'\}}(s). \tag{6.3}$$

La « detailed balance condition » s'écrit :

$$\forall A,B \in \mathcal{X}^{N^2} \quad \int_A \int_B PE(s,ds')\pi^*(ds) = \int_B \int_A PE(s',ds)\pi^*(ds'). \tag{6.4}$$

En utilisant (6.3) nous avons :

$$\forall B \in \mathcal{X}^N \quad PE(s,B) = \int_B q(s,ds')\rho(s,s') + \int_{\mathcal{X}} q(s,ds'')(1-\rho(s,s''))\mathbb{1}_B(s).$$

Alors (6.4) devient :

$$\int_A \pi^*(ds) \int_B q(s,ds')\rho(s,s') + \int_{A\cap B} \pi^*(ds) \int_{\mathcal{X}} q(s,ds'')(1-\rho(s,s'')) = \int_B \pi^*(ds') \int_A q(s',ds)\rho(s',s) + \int_{B\cap A} \pi^*(ds') \int_{\mathcal{X}} q(s',ds'')(1-\rho(s',s'')),$$

et vérifier (6.4) revient à avoir :

$$\int_A \int_B \pi^*(ds)q(s,ds')\rho(s,s') = \int_B \int_A \pi^*(ds')q(s',ds)\rho(s',s).$$

Cela est vérifié si les intégrandes sont égaux, soit si :

$$q(s,ds')\rho(s,s')\pi^*(ds) = q(s',ds)\rho(s',s)\pi^*(ds'). \tag{6.5}$$

Dans l'algorithme PTEEM, les deux chaînes candidates pour échanger leurs états actuels sont choisies uniformément parmi les chaînes d'un même anneau d'énergie. Nous avons donc $q(s,s') = q(s',s)$. De plus, la probabilité d'acceptation du mouvement est définie par (6.2). Il en résulte que (6.5) est vérifiée, et donc (6.4) est vérifiée. Le noyau de transition PE est donc réversible, et de distribution stationnaire π^*.

3. Les noyaux PE et PL sont réversibles, il est clair que P l'est également, et S est réversible donc stationnaire. Montrons qu'elle est de distribution stationnaire π^*, soit que $\forall A \in \mathcal{B}(\mathcal{X})^N$ nous avons $\pi^*(A) = \int_{\mathcal{X}^N} P(s,A)\pi^*(ds)$. Nous savons que PE et PL sont de distributions stationnaires π^*, soit que :

$$\begin{aligned} \pi^*(A) &= \int_{\mathcal{X}^N} PL(s,A)\pi^*(ds), \\ \pi^*(A) &= \int_{\mathcal{X}^N} PE(s,A)\pi^*(ds). \end{aligned}$$

Nous pouvons écrire :

$$\begin{aligned}\int_{\mathcal{X}^N} P(s,A)\pi^*(ds) &= \int_{\mathcal{X}^N}\int_{\mathcal{X}^N} PE(\tilde{s},A)PL(s,\tilde{s})d\tilde{s}\pi^*(ds)\\ &= \int_{\mathcal{X}^N} PE(\tilde{s},A)\Big[\int_{\mathcal{X}^N} PL(s,\tilde{s})\pi^*(ds)\Big]d\tilde{s}\\ &= \int_{\mathcal{X}^N} PE(\tilde{s},A)\pi^*(d\tilde{s})\\ &= \pi^*(A).\end{aligned}$$

Nous avons donc bien P de distribution stationnaire π^*.

4. Chaque PL_i est supposé fortement λ-irréductible et apériodique, donc PL est apériodique et fortement λ_N-irréductible. Comme PE est un noyau d'échange entre les états actuels de deux chaînes, $P = PE * PL$ est également fortement λ_N-irréductible et apériodique.
   En effet, montrons que $\forall s, s' \in \mathcal{X}^N \times \mathcal{X}^N$, nous avons $P(s,s') > 0$.
   - Soit $\pi^*(s') > 0$. Nous pouvons trouver s'' correspondant à s' pour lequel les états actuels de deux chaînes d'énergies similaires sont intervertis. Il est toujours possible de trouver un tel s'' dés lors que $N > d$. Comme les PL_i sont fortement λ-irréductibles, $PL(s,s'') = \prod_{i=1}^N PL_i(x_i, x_i'') > 0$. L'état s'' peut être atteint depuis s grâce aux mouvements locaux seulement. Puis il y a une probabilité non nulle de proposer s' à partir de s'' lors du mouvement Equi-Energy, tout comme il y a une probabilité non nulle que cet échange Equi-Energy soit accepté, car $\pi^*(s') > 0$. Nous avons donc $P(s,s') > 0$.
   - Soit $\pi^*(s') = 0$. Grâce aux hypothèses nous avons $PL(s,s') > 0$, donc s' peut être atteint depuis s grâce aux mouvements locaux seulement. Il y a une probabilité non nulle de proposer un autre état à partir de s' lors du mouvement Equi-Energy, mais il y a alors une probabilité nulle que cet échange Equi-Energy soit accepté, car $\pi^*(s') = 0$. Nous avons donc encore $P(s,s') > 0$.

   Au final, dans tous les cas nous avons $P(s,s') > 0$, il y a une probabilité non nulle de passer de s à s' en une itération de PTEEM. Nous avons donc S fortement λ_N-irréductible.

5. P est λ_N-irréductible, apériodique et de distribution stationnaire π^*. D'après le théorème 1 de Tierney [220], S converge vers sa distribution stationnaire π^* au sens de la norme en variation totale, et pour presque tout $s \in \mathcal{X}^N$.

6. Etant donné que la densité marginale de la première chaîne est $\pi_1 = \pi$, ses réalisations sont issues de $\pi_1 = \pi$ qui est la distribution d'intérêt.

■

Les hypothèses de cette proposition sont discutées à la fin de cette partie. Cette proposition montre que sous des hypothèses simples d'apériodicité et de ϕ-irréductibilité, la chaîne S converge en loi vers sa distribution stationnaire π^*, pour π^*-presque-tout point de départ. L'algorithme

PTEEM proposé est donc valide, mais il est toujours possible d'avoir un ensemble d'états de mesure nulle à partir duquel la convergence n'a pas lieu. Cela nous amène au lemme suivant.

Lemme 6.1.2 *Supposons que les noyaux de transition des mouvements locaux PL_i sont réversibles de distributions stationnaires π_i ($i = 1, \ldots, N$), apériodiques et fortement λ-irréductibles. Supposons de plus la stricte positivité de la densité π^* sur $\mathcal{X}^N$ ($\forall x \in \mathcal{X}^N, \pi^*(x) > 0$). Alors S est réversible, fortement λ_N-irréductible, positive et Harris-récurrente.*

Preuve :

1. D'après la proposition 6.1.1, S est réversible de distribution stationnaire π^* et fortement λ_N-irréductible. Alors S est positive.
2. Montrons qu'un état s' atteint depuis n'importe quel état de départ s après une itération de PTEEM ne peut pas appartenir à un ensemble $A \in \mathcal{X}^N$ tel que $\pi^*(A) = 0$ (preuve inspirée du théorème 8 de Roberts et Rosenthal [188]).
   En effet, à chaque itération, soit la chaîne atteint un nouvel état, soit elle n'a pas bougé. Le noyau P peut donc s'écrire, $\forall A \in \mathcal{X}^N$:

$$P(s, A) = r(s)M(s, A) + (1 - r(s))\mathbb{1}_s(A), \tag{6.6}$$

   avec $r(s)$ la probabilité de partir de s lors d'une itération. M est le noyau conditionnellement au fait de bouger. Nous avons pour tout $s \in \mathcal{X}^N$ la mesure de probabilité $M(s, .)$ qui est absolument continue par rapport à $\lambda_N(.)$. Soit un ensemble A tel que $\pi^*(A) = 1$ et $\pi^*(A^C) = 0$. D'après l'hypothèse de stricte positivité de π^*, nous avons $\pi^*(s) > 0 \; \forall s \in \mathcal{X}^N$, et alors $\lambda_N(A^C) = 0$. Par absolue continuité, nous avons $M(s, A^C) = 0$ et $M(s, A) = 1$. Donc si la chaîne bouge lors d'une itération, elle va forcément dans A tel que $\pi^*(A) = 1$. Alors si s' est un état atteint après une itération, il ne peut pas appartenir à un ensemble $A \in \mathcal{X}^N$ tel que $\pi^*(A) = 0$.
3. Pour montrer que la chaîne est Harris-récurrente, nous utilisons le Théorème 2 de Tierney [220] caractérisant les chaînes Harris-récurrentes : une chaîne de Markov est Harris-récurrente si et seulement si les seules fonctions harmoniques bornées de la chaîne sont les fonctions constantes. Une fonction harmonique h pour S vérifie, avec S_n la variable donnant la valeur de la chaîne à la n-ième itération :

$$\mathbb{E}(h(S_n)|s_0) = \mathbb{E}(h(S_1)|s_0) = h(s_0) \qquad \forall n \in \mathbb{N}. \tag{6.7}$$

   Nous allons utiliser le théorème 6.80 de Robert et Casella [186] (inspiré de Athreya et al. [13]) :
   Si le noyau de transition P satisfait : $\exists B \in \mathcal{B}(\mathcal{X})^N$ tel que :
   (i) $\forall s_0, \sum_{n=1}^{\infty} \int_B P^n(s_0, s) d\mu(s) > 0$, avec μ la distribution initale de la chaîne, $S_0 \sim \mu$.

(ii) $\inf_{s,s' \in B} P(s,s') > 0$.

Alors, pour π^*-presque tout s_0,

$$\lim_{n\to\infty} \sup_{A \in \mathcal{B}(\mathcal{X})^N} \Big| \int_A P^n(s_0,s)ds - \int_A \pi^*(s)ds \Big| = 0.$$

Ces deux hypothèses (i) et (ii) impliquent que la chaîne est π^*-irréductible et apériodique.

Pour utiliser ce théorème, nous vérifions (i) et (ii) pour $B = \mathcal{X}^N$.
Nous avons montré la forte irréductibilité de S (point 1.), donc nous avons $\forall (s,s') \in \mathcal{X}^N \times \mathcal{X}^N$, $P(s,s') > 0$. Alors (ii) est vérifiée sur $\mathcal{X}^N$. De même, il est clair que $\mathcal{X}^N$ est accessible depuis n'importe quel état initial s_0, et donc (i) est vérifiée.
Nous avons alors pour π^*-presque tout s_0 :

$$\lim_{n\to\infty} \sup_{A \in \mathcal{B}(\mathcal{X})^N} \Big| \int_A P^n(s_0,s)ds - \int_A \pi^*(s)ds \Big| = 0,$$

et en utilisant

$$\|\mu\|_{TV} = \sup_{A \in \mathcal{B}(\mathcal{X})^N} |\mu(A)| = \frac{1}{2} \sup_{|h|<1} | \int h(x)\mu(dx)|,$$

Nous pouvons écrire :

$$\begin{aligned} \sup_{A \in \mathcal{B}(\mathcal{X})^N} \Big| \int_A P^n(s_0,s)ds - \int_A \pi^*(s)ds \Big| &= \frac{1}{2} \sup_{|h|<1} \Big| \int h(s)\big(P^n(s_0,s)ds - \pi^*(s)\big)ds \Big| \\ &= \frac{1}{2} \sup_{|h|<1} \Big| E[h(S_n)|s_0] - E_{\pi^*}[h(s)] \Big|. \end{aligned}$$

Le résultat du théorème de Robert et Casella peut donc se réécrire de la façon suivante :

$$\lim_{n\to\infty} \sup_{|h|<1} \Big| \mathbb{E}[h(S_n)|s_0] - \mathbb{E}_{\pi^*}[h(s)] \Big| = 0.$$

Par extension, pour toute fonction bornée h,

$$\lim_{n\to\infty} \sup \Big| \mathbb{E}[h(S_n)|s_0] - \mathbb{E}_{\pi^*}[h(s)] \Big| = 0.$$

De plus, si h bornée vérifie (6.7), alors $\mathbb{E}[h(S_n)|s_0] = h(s_0)$.
Nous avons donc $h(s_0) = \mathbb{E}_{\pi^*}[h(s)]$ pour π^*-presque tout s_0. Soit h est π^*-presque partout constante et égale à $\mathbb{E}_{\pi^*}(h(S))$. Il reste à montrer que h est partout constante et égale à $\mathbb{E}_{\pi^*}(h(S))$.

Soit n'importe quel $s_0 \in \mathcal{X}^N$. Nous avons, en reprenant l'écriture du noyau dans l'équation

(6.6),

$$\begin{aligned}\mathbb{E}(h(S_1)|s_0) &= \int_{\mathcal{X}^N} P(s_0,s_1)h(s_1)ds_1 \\ &= \int_{\mathcal{X}^N} r(s_0)M(s_0,s_1)h(s_1)ds_1 + (1-r(s_0))h(s_0).\end{aligned}$$

La première intégrale correspond au fait de partir de s_0 et de bouger vers $s_1 \neq s_0$. Alors comme vu dans le point 2., s_1 atteint après une itération ne peut pas appartenir à un ensemble $A \in \mathcal{X}^N$ tel que $\pi^*(A) = 0$. Donc comme h est π^*-presque partout constante et égale à $\mathbb{E}_{\pi^*}(h(S))$, nous avons $h(s_1) = \mathbb{E}_{\pi^*}(h(S))$. Nous pouvons donc écrire :

$$\begin{aligned}\mathbb{E}(h(S_1)|s_0) &= \int_{\mathcal{X}^N} r(s_0)M(s_0,s_1)\mathbb{E}_{\pi^*}(h(S))ds_1 + (1-r(s_0))h(s_0) \\ &= \mathbb{E}_{\pi^*}(h(S))r(s_0)\int_{\mathcal{X}^N} M(s_0,s_1)ds_1 + (1-r(s_0))h(s_0) \\ &= \mathbb{E}_{\pi^*}(h(S))r(s_0) + (1-r(s_0))h(s_0).\end{aligned}$$

Or $\mathbb{E}(h(S_1)|s_0) = h(s_0)$ car h harmonique, donc :

$$h(s_0) = \mathbb{E}_{\pi^*}(h(S))r(s_0) + (1-r(s_0))h(s_0).$$

Soit

$$\Big(h(s_0) - \mathbb{E}_{\pi^*}(h(S))\Big)r(s_0) = 0.$$

Or, comme S est λ_N-irréductible, nous avons $\forall s_0, r(s_0) > 0$. Alors $\forall s_0, h(s_0) = \mathbb{E}_{\pi^*}(h(S))$, et h est partout constante et égale à $\mathbb{E}_{\pi^*}(h(S))$. Nous en déduisons la Harris-récurrence.

■

Nous pouvons alors énoncer la proposition suivante :

Proposition 6.1.3 *Supposons que les noyaux de transition des mouvements locaux PL_i sont réversibles de distributions stationnaires π_i ($i = 1,\ldots,N$), apériodiques et fortement λ-irréductibles. Supposons de plus la positivité de la densité π^* sur $\mathcal{X}^N$ ($\forall x \in \mathcal{X}^N, \pi^*(x) > 0$). Alors pour tout $s \in \mathcal{X}^N$:*

$$\lim_{n\to\infty} ||P^n(s,.) - \pi^*||_{TV} = 0 \qquad \forall s \in \mathcal{X}^N$$

Preuve :
En utilisant le lemme 6.1.2 ainsi que la proposition 6.1.1, S est une chaîne de Markov π^*-irréductible, apériodique, de loi stationnaire π^* et Harris-récurrente. Le résultat s'obtient en appliquant le théorème 1 de Tierney [220].

■

Les proposition 6.1.1 et 6.1.3 ont des hypothèses sur les échantillonneurs locaux qui ne sont généralement pas difficiles à vérifier (voir Robert et Casella [186]). Par exemple, dans le cas d'un

échantillonneur de Gibbs, l'hypothèse de forte irréductibilité est vérifiée sous la condition de positivité de la densité cible sur $\mathcal{X}$, ou sous la condition d'absolue continuité du noyau de transition par rapport à la mesure dominante. Dans le cas d'un algorithme de Metropolis-Hastings, cette hypothèse revient à avoir la positivité de la distribution auxiliaire servant à proposer un nouvel état, $\forall (x, x') \in \mathcal{X} \times \mathcal{X}$, $q_i(x, x') > 0$.

Nous pourrions affaiblir l'hypothèse de réversibilité de chacun des noyaux locaux (nous nous en servons dans le point 3 de la preuve de la proposition 6.1.1). Si nous enlevons cette hypothèse, alors nous perdons seulement la réversibilité de la chaîne S, les propriétés de convergence et de Harris-récurrence demeurent. Notons que la réversibilité peut être intéressante car permet de pouvoir appliquer des théorèmes limites (voir partie 6.7 de Robert et Casella [186]).

Concernant l'apériodicité de tous les noyaux locaux, cette hypothèse peut être affaiblie car il suffit qu'un seul des N noyaux de transition des mouvements locaux soit apériodique pour que P le soit aussi.

6.1.3 En pratique

En pratique, avant de faire tourner un algorithme PTEEM, nous devons choisir le nombre de chaînes et le nombre d'anneaux d'énergie à utiliser. Il faut assez d'anneaux pour que les états des chaînes d'un même anneau puissent être considérés d'énergies similaires. Ensuite, il faut un nombre de chaînes N plus grand que le nombre d'anneaux d, car il faut au moins deux chaînes dans un anneau pour effectuer un échange Equi-Energy. D'après notre expérience, il suffit d'avoir $N = c \times d$, avec c une constante entre 3 et 5.

Ensuite, il faut calibrer les températures et les niveaux d'énergie. En particulier, comme pour l'EES, il faut faire attention à bien choisir les niveaux d'énergie. En effet, si ces niveaux ne sont pas adéquats les chaînes se répartiront mal dans les anneaux, et nous perdrions l'avantage apporté par le mouvement Equi-Energy.

Niveaux d'énergie

Les niveaux $H_1, H_2, \ldots, H_d$ sont associés à d anneaux d'énergies, le premier incluant les états ayant un niveau d'énergie compris entre H_1 et H_2 ainsi que quelques états ayant un niveau d'énergie plus petit que H_1, et le dernier incluant les états ayant un niveau d'énergie plus grand que H_d. Une fois que les valeurs H_1 et H_d ont été déterminées, les autres niveaux d'énergie peuvent être choisis de manière à ce que les $ln(H_i)$ ou les $ln(H_{i+1} - H_i)$ soient régulièrement espacés sur une échelle logarithmique.

Pour choisir H_1 et H_d nous pouvons utiliser un ou quelques runs d'un algorithme MCMC classique ayant $\pi_1 = \pi$ comme distribution stationnaire. Concernant H_d nous prenons le niveau d'énergie associé à un état ayant une énergie élevée par rapport aux autres, mais finie. Concernant H_1, nous prenons le niveau d'énergie associé à un mode observé. En pratique, nous pouvons prendre

pour H_d l'énergie associée à un état après quelques itérations seulement de l'algorithme, et pour H_1 l'énergie associée à un état obtenu après la période de burn-in.

REMARQUE 6.1.1 *Concernant H_1, si les modes de la distribution d'intérêt sont connus, il suffit de prendre H_1 légèrement inférieur à l'énergie du mode le plus élevé.*

Températures

La distribution associée à la température la plus élevée doit être suffisament aplatie afin que la chaîne associée puisse se déplacer facilement d'un mode à un autre. Après avoir choisi une valeur pour T_N il est donc nécessaire de vérifier que la chaîne associée se déplace bien. T_1 est égale à 1, et est associée avec la chaîne d'intérêt. Une fois que T_1 et T_N ont été déterminées, les autres températures peuvent être choisies de manière à être régulièrement espacées sur une échelle logarithmique, ou bien encore de façon à ce que leurs inverses soient régulièrement espacés ou aient une progression géométrique (voir par exemple Kou et al. [118], Nagata et Watanabe [164] ou Neal [166]). D'autres façons de calibrer les températures ont été proposées par Goswami et Liu [82].

REMARQUE 6.1.2 *Sous certaines hypothèses, Atchadé et al. [12] ont montré que pour le Parallel Tempering et le Simulated Tempering, les températures optimales doivent être définies de manière à ce que la probabilité d'acceptation du mouvement d'échange de deux chaînes soit de 0.234.*

Vérification des calibrations

Il est nécessaire de vérifier sur un run de PTEEM que les choix de températures et de niveaux d'énergie sont pertinents. La première chaîne doit avoir presque tous ses états dans l'anneau 1, la dernière doit avoir presque tous ses états dans le dernier anneau, et entre les deux les états des différentes chaînes doivent être assez bien répartis dans les anneaux. Les chaînes de températures les plus faibles ont normalement plus d'états dans les premiers anneaux, et les chaînes de températures plus élevées plus d'états dans les derniers anneaux. La répartition dans les anneaux peut être considérée correcte lorsqu'il n'y a pas de « saut d'énergie » entre chaînes adjacentes, et que chaque chaîne effectue des mouvements Equi-Energy avec plusieurs autres chaînes. Si peu d'échanges Equi-Energy sont observés entre les chaînes, il est nécessaire d'ajuster les températures ou les niveaux d'énergie, en ajoutant de nouvelles températures par exemple ou bien en utilisant une nouvelle calibration. Un tel problème est illustré dans la table 6.1.3, ainsi que dans l'exemple simulé de la partie 6.2.2.

Nombres de mouvements nécessaires

Il est intéressant de comparer le nombre total de mouvements locaux et globaux nécessaires pour l'algorithme PTEEM et pour l'EES. Soit B le nombre d'itérations burn-in, R le nombre

	Mauvaise répartition					Bonne répartition				
anneau d'énergie	1	2	3	4	5	1	2	3	4	5
chaîne $i-2$	990	10	0	0	0	990	10	0	0	0
chaîne $i-1$	950	50	0	0	0	701	202	97	0	0
chaîne i	900	100	0	0	0	387	408	205	0	0
chaîne $i+1$	0	2	237	511	250	45	312	355	288	0
chaîne $i+2$	0	0	105	610	285	0	64	517	353	66

TABLE 6.1 – La partie gauche donne un exemple de mauvaise répartition dans les anneaux, il y a un « saut d'énergie » entre les chaînes i et $i+1$. Aucun mouvement Equi-Energy n'est possible entre une chaîne d'indice inférieur ou égal à i et une chaîne d'indice supérieur ou égal à $i+1$. La partie droite donne un exemple de bonne répartition dans les anneaux, il n'y a pas de « saut d'énergie ».

d'itérations nécessaires au remplissage initial des anneaux dans l'EES, et M le nombre d'itérations voulu pour la chaîne d'intérêt après burn-in.

- Pour l'EES, le nombre total de mouvements locaux effectués est :

$$\Big[\frac{(K-1)K}{2}(B+R)+(K-1)(M-R)\Big](1-p_{ee})+K(B+R)+(M-R).$$

  Et le nombre total de mouvements globaux effectués est :

$$\Big[\frac{(K-1)K}{2}(B+R)+(K-1)(M-R)\Big]p_{ee}.$$

- Pour PTEEM, le nombre total de mouvements locaux effectués est :

$$NB+NM.$$

  Et le nombre total de mouvements globaux effectués est :

$$B+M.$$

En termes de mouvements nécessaires, PTEEM nécessite plus de mouvements quand M est grand car en général N est plus grand que K.

En termes de temps de calcul, nous devons prendre en compte que dans certains problèmes les échantillonneurs locaux utilisés par l'EES et le PTEEM peuvent être différents (voir l'illustration 6.2.4 par exemple), et peuvent donc avoir des temps de calcul différents.

En termes de stockage, pour obtenir la $(i+1)^{\text{ème}}$ itération de la chaîne d'intérêt, l'EES a $KR+i+(1-p_{ee})\left(\frac{(K-1)KR}{2}+(K-1)i\right)$ états en mémoire, tandis que le PTEEM n'en a que N.

6.2 Illustrations

6.2.1 En combinaison avec un algorithme de Metropolis-Hastings

Nous reprenons l'exemple du mélange de vingt lois normales bi-dimensionnelles utilisé par Kou et al. [118], et présenté dans la partie 5.1.1. Nous souhaitons comparer les performances du Parallel Tempering (PT), du PTEEM et de l'EES. Chacun des trois algorithmes est lancé 100 fois. Pour les algorithmes PT et PTEEM, 5000 itérations sont lancées, dont 2500 de burn-in. Pour l'EES la période de burn-in est aussi de 2500 itérations, la période de construction des anneaux est de 500 itérations, et 2500 itérations sont simulées après la période de burn-in et la période de construction des anneaux pour la première chaîne (5500 itérations au total pour cette chaîne). Les chaînes sont actualisées localement par des algorithmes de Metropolis-Hastings, et comme dans Kou et al. [118] la distribution des modifications proposées à chaque itération est une $\mathcal{N}_2(0, \tau_i^2 I_2)$, avec $\tau_i = 0.25\sqrt{T_i}$.
Pour l'EES nous prenons le même nombre de chaînes, les mêmes niveaux d'énergie, les mêmes températures et la même probabilité de saut Equi-Energy que Kou et al. [118] ($K = 5$, $p_{ee} = 0.1$, $T = (1, 2.8, 7.7, 21.6, 60)$, $H = (0.2, 2, 6.3, 20, 63.2)$). Pour les algorithmes PT et PTEEM, nous prenons $N = 20$ chaînes, ayant des températures entre 1 et 60 également réparties en échelle logarithmique. Pour le PTEEM nous prenons 5 groupes d'énergies identiques à ceux de l'EES.
Pour le Parallel Tempering, nous ne proposons que des mouvements d'échanges entre chaînes adjacentes, pour qu'ils aient plus de chance d'être acceptés (on sait qu'il est plus efficace de proposer des mouvements entre chaînes adjacentes).
Pour chaque run de chaque algorithme, les états initiaux des chaînes sont générés d'après une uniforme sur $[0, 1]^2$, ce qui rend l'échantillonnage plus difficile car c'est une région assez éloignée des modes du mélange.

Afin d'évaluer la capacité d'exploration des algorithmes, nous avons examiné pour chacun des runs le nombre de modes du mélange visités par la première chaîne (chaîne d'intérêt) et à quelles fréquences, ainsi que les estimations des premiers et seconds moments $\big(\mathbb{E}(X_1), \mathbb{E}(X_2)\big)$ et $\big(\mathbb{E}(X_1)^2, \mathbb{E}(X_2)^2\big)$ de cette chaîne.

Estimations des premiers et second moments
Les estimations de l'EES et du PTEEM sont bien plus précises que celles du PT, avec des écart-types beaucoup plus faibles. De plus, il apparaît sur ces 100 runs que les estimations du PTEEM sont plus justes que celles de l'EES, voir le tableau 6.2 (A).

		$E(X_1)$	$E(X_2)$	$E(X_1)^2$	$E(X_2)^2$
True value		4.478	4.905	25.605	33.920
(A)	EES	4.448 (0.301)	4.953 (0.458)	25.229 (3.112)	34.226 (4.507)
	PT	3.971 (0.809)	4.137 (1.114)	21.510 (7.741)	27.510 (10.407)
	PTEEM	4.483 (0.324)	4.912 (0.454)	25.556 (3.366)	33.889 (4.406)
(B)	EES	5.088 (0.373)	6.001 (0.515)	32.005 (4.086)	45.306 (5.638)
	PTEEM	4.745 (0.365)	5.468 (0.491)	28.908 (3.774)	40.617 (5.019)

TABLE 6.2 – Estimations des premiers et seconds moments de la chaîne 1, obtenues sur 100 runs pour l'EES et les algorithmes PT et PTEEM. Les nombres entre parenthèses sont les écart-types obtenus sur ces différents runs. (A) correspond au cas avec variances égales, et (B) au cas avec variances non égales.

Nombre de modes visités

L'algorithme PT n'a visité que 14.31 modes sur les 20 en moyenne sur les cent runs. De meilleurs résultats sont obtenus pour l'algorithme PTEEM et pour l'EES : respectivement 19.98 et 19.92 modes visités en moyenne sur les cent runs, voir le tableau 6.3.

PT	EES	PTEEM
De 10 à 18.	De 18 à 20.	De 19 à 20.
14.31 en moyenne.	19.92 en moyenne.	19.98 en moyenne.

TABLE 6.3 – Nombre de modes visités lors des 100 runs par l'EES et les algorithmes PT et PTEEM.

Fréquence de visite des modes

Pour chaque run de chaque algorithme nous calculons l'erreur absolue de fréquence pour chaque mode $err_m = |\widehat{f_m} - 0.05|$, où $\widehat{f_m}$ est la fréquence empirique du nombre de visite du mode m. Puis nous calculons la médiane et le maximum de err_m sur les cent runs, ceci pour chaque algorithme. Ces valeurs sont données dans le tableau 6.4, qui donne également les ratios de ces valeurs entre les trois algorithmes.

Comme Kou et al. [118], nous retrouvons le fait que l'EES améliore sensiblement les simulations par rapport à l'algorithme PT (moyenne des ratios *médiane PT/médiane EES* sur les vingt modes de 2.42, et moyenne des ratios *maximum PT/maximum EES* sur les vingt modes de 2.92). Comme attendu, nous trouvons que l'algorithme PTEEM améliore lui aussi les simulations par rapport à l'algorithme PT (moyenne des ratios *médiane PT/médiane PTEEM* sur les vingt modes de 2.52, et moyenne des ratios *maximum PT/maximum PTEEM* sur les vingt modes de 3.07). De plus, nous voyons que l'algorithme PTEEM donne en moyenne des résultats légèrement meilleurs que ceux de l'EES (moyenne des ratios *médiane EES/médiane PTEEM* sur les vingt modes de 1.05, et moyenne des ratios *maximum EES/maximum PTEEM* sur les vingt modes de 1.13).

En illustration, deux figures montrent les 2500 simulations après le burn-in pour les chaînes

	μ_1	μ_2	μ_3	μ_4	μ_5	μ_6	μ_7	μ_8	μ_9	μ_{10}
$med(err_i)$ PT	0.05	0.05	0.04	0.05	0.03	0.05	0.04	0.05	0.04	0.04
$max(err_i)$ PT	0.42	0.19	0.16	0.23	0.15	0.25	0.21	0.20	0.20	0.24
$med(err_i)$ EES	0.02	0.02	0.01	0.02	0.02	0.02	0.01	0.02	0.02	0.01
$max(err_i)$ EES	0.12	0.07	0.06	0.11	0.10	0.10	0.08	0.13	0.07	0.05
$med(err_i)$ PTEEM	0.02	0.01	0.01	0.02	0.02	0.02	0.01	0.02	0.02	0.01
$max(err_i)$ PTEEM	0.12	0.11	0.07	0.10	0.06	0.08	0.04	0.07	0.07	0.05
R_{med} PT/EES	2.16	2.80	2.92	2.22	1.98	2.21	3.10	2.07	2.07	2.69
R_{max} PT/EES	3.59	2.61	2.81	2.10	1.55	2.43	2.54	1.53	2.93	4.50
R_{med} PT/PTEEM	2.60	3.72	2.63	2.19	1.79	2.97	2.77	2.55	2.32	2.64
R_{max} PT/PTEEM	3.44	1.76	2.27	2.30	2.44	3.06	5.23	2.92	2.83	5.23
R_{med} EES/PTEEM	1.21	1.33	0.90	0.99	0.91	1.35	0.89	1.23	1.12	0.98
R_{max} EES/PTEEM	0.96	0.67	0.81	1.09	1.58	1.26	2.06	1.91	0.96	1.16
	μ_{11}	μ_{12}	μ_{13}	μ_{14}	μ_{15}	μ_{16}	μ_{17}	μ_{18}	μ_{19}	μ_{20}
$med(err_i)$ PT	0.04	0.05	0.04	0.05	0.05	0.04	0.04	0.04	0.05	0.05
$max(err_i)$ PT	0.22	0.14	0.22	0.33	0.23	0.19	0.16	0.22	0.38	0.13
$med(err_i)$ EES	0.01	0.02	0.02	0.02	0.02	0.02	0.01	0.02	0.02	0.02
$max(err_i)$ EES	0.05	0.09	0.07	0.07	0.07	0.09	0.06	0.05	0.11	0.10
$med(err_i)$ PTEEM	0.02	0.02	0.03	0.02	0.02	0.02	0.01	0.01	0.02	0.02
$max(err_i)$ PTEEM	0.07	0.07	0.09	0.08	0.07	0.10	0.04	0.06	0.11	0.06
R_{med} PT/EES	2.51	2.46	2.77	2.63	2.39	1.76	3.06	2.22	2.10	2.37
R_{max} PT/EES	4.58	1.60	3.23	4.61	3.26	2.10	2.83	4.77	3.50	1.36
R_{med} PT/PTEEM	2.14	1.98	1.79	2.84	2.75	2.18	2.72	2.78	2.43	2.60
R_{max} PT/PTEEM	3.05	2.02	2.35	4.16	3.44	1.79	3.72	3.78	3.50	2.16
R_{med} EES/PTEEM	0.85	0.81	0.65	1.08	1.15	1.24	0.89	1.25	1.16	1.10
R_{max} EES/PTEEM	0.67	1.26	0.73	0.90	1.06	0.85	1.32	0.79	1.00	1.58

TABLE 6.4 – **Pour chaque mode** : médiane et maximum des erreurs de fréquence de visite obtenues sur 20 runs pour l'EES, PT et PTEEM, ratio de médiane (maximum) obtenue par PT sur médiane (maximum) obtenue par l'EES, ratio de médiane (maximum) obtenue par PT sur médiane (maximum) obtenue par PTEEM, et ratio de médiane (maximum) obtenue par l'EES sur médiane (maximum) obtenue par PTEEM.

1,7,14 et 20, dans le cas d'un run de PT et d'un run de PTEEM (figures 6.1 et 6.2), et une figure montre les simulations après burn-in des cinq chaînes dans le cas d'un run de l'EES (figure 6.3).

Mouvements d'échange et Equi-Energy

Les taux d'acceptation moyens des mouvements locaux et des mouvements d'échange sur les cents runs sont donnés dans le tableau 6.5.

	Mouvements Locaux	Mouvements d'échange
EES	0.387	0.799
PT	0.337	0.905
PTEEM	0.333	0.822

TABLE 6.5 – Taux d'acceptation des mouvements locaux (MH) et des mouvements d'échange sur cent runs, pour l'EES et les algorithmes PT et PTEEM.

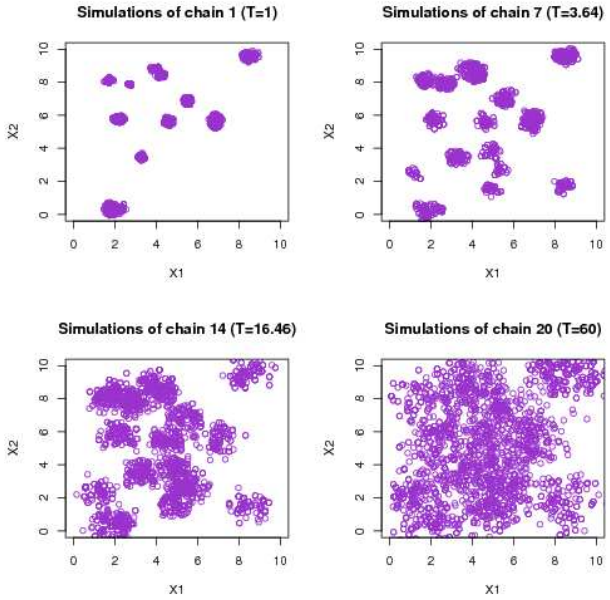

FIGURE 6.1 – Simulations générées par les chaînes 1, 7, 14 et 20 d'un run de l'algorithme PT. La première chaîne n'est pas parvenu à visiter tous les modes, des modes n'ont pas été propagés des chaînes de températures élevées jusqu'à la première chaîne.

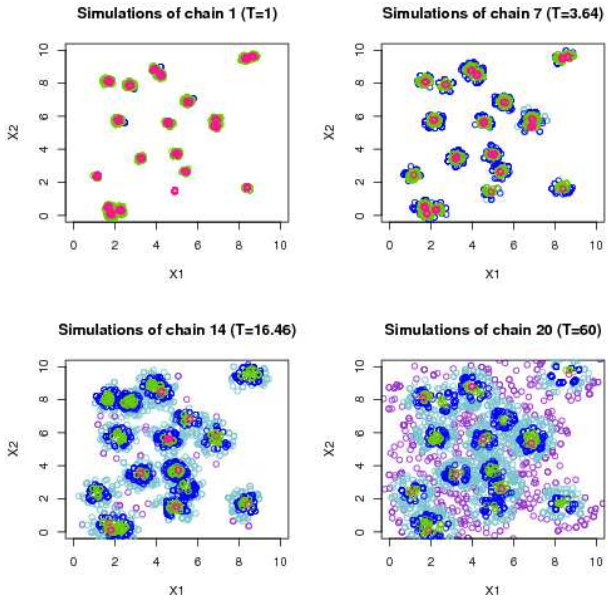

FIGURE 6.2 – Simulations générées par les chaînes 1, 7, 14 et 20 d'un run de l'algorithme PTEEM. Les couleurs correspondent aux cinq niveaux d'énergie possibles. La première chaîne a visité tous les modes.

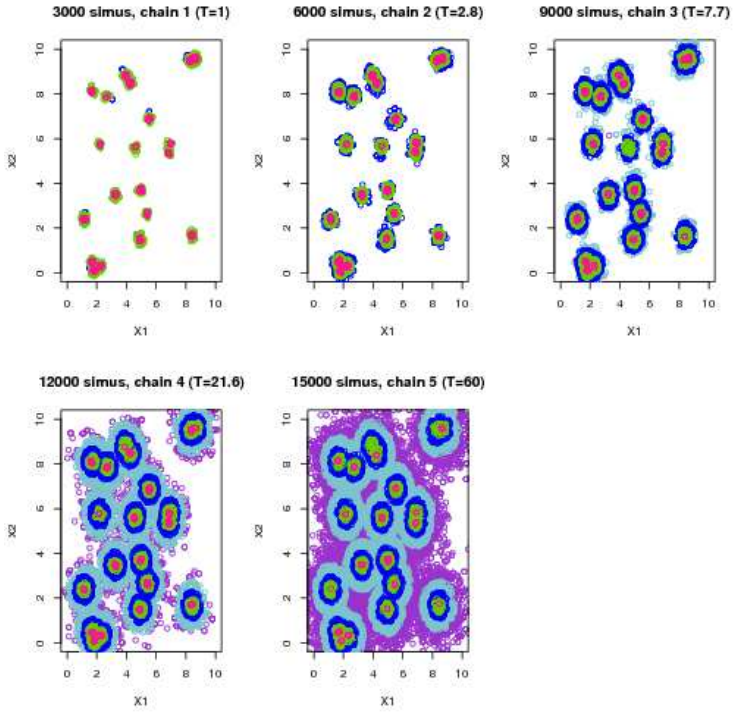

FIGURE 6.3 – Simulations générées par les cinq chaînes d'un run de l'EES. Les couleurs correspondent aux cinq niveaux d'énergie possibles. La première chaîne a visité tous les modes. Notons que plus l'ordre de la chaîne augmente et plus le nombre de simulations effectuées augmente.

Le tableau 6.6 permet d'observer avec quelles chaînes les chaînes 1, 10 et 20 ont effectué des mouvements Equi-Energy lors des 100 runs de PTEEM.

Nous pouvons voir que la chaîne 1 a plus souvent effectué un mouvement Equi-Energy avec une chaîne d'indice faible qu'avec une chaîne d'indice élevé. Cela parait logique, car les chaîne d'indices proches ont plus de chances d'être dans des états d'énergies similaires. De même, nous remarquons que la chaîne 10 a plus souvent échangé son état avec une chaîne d'indice proche de 10, tout comme la chaîne 20 a plus souvent échangé son état avec une chaîne d'indice élevé. Nous pouvons remarquer que des mouvements Equi-Energy ont été proposés et acceptés pour toutes les paires de chaînes possibles.

Cas de variances et de minima d'énergie non égaux

Dans ce qui vient d'être présenté, tous les composants du mélange ont les mêmes niveaux d'énergie minimum et les mêmes variances.

Dans l'optique d'étudier le comportement du PTEEM et de l'EES dans le cas de variances et de minima d'énergie non égaux, nous prenons $\sigma_1 = \ldots = \sigma_5 = 0.4$, $\sigma_6 = \ldots = \sigma_{13} = 0.1$ et $\sigma_{14} = \ldots = \sigma_{20} = 0.05$. Les modes associés avec de petites variances ont des niveaux d'énergie

	chaîne 1	chaîne 10	chaîne 20
chaîne 1	0.00	4.63	0.50
chaîne 2	16.32	4.33	0.62
chaîne 3	14.34	4.29	0.64
chaîne 4	11.98	4.64	0.70
chaîne 5	9.96	4.89	0.76
chaîne 6	8.26	5.46	1.03
chaîne 7	6.57	5.76	1.17
chaîne 8	6.01	6.26	1.50
chaîne 9	4.96	6.74	2.03
chaîne 10	4.32	0.00	2.30
chaîne 11	3.25	7.11	3.13
chaîne 12	2.85	6.67	4.33
chaîne 13	2.42	6.65	5.61
chaîne 14	1.98	6.11	7.38
chaîne 15	1.61	5.76	8.75
chaîne 16	1.44	5.13	10.64
chaîne 17	1.15	4.64	13.72
chaîne 18	1.08	4.32	16.09
chaîne 19	0.87	3.63	19.10
chaîne 20	0.62	2.99	0.00

TABLE 6.6 – Proportions (%) d'échanges Equi-Energy acceptés impliquant la chaîne 1 avec chacune des autres chaînes (moyenne sur 100 runs du PTEEM). Idem pour les chaînes 10 et 20.

plus faibles que ceux associés avec des variances plus élevées.

Les algorithmes PTEEM et EES sont lancés 100 fois chacun. Pour chaque run, l'algorithme PTEEM est lancé sur le même nombre d'itérations et avec les mêmes étapes de Metropolis-Hastings que précédemment. Pour le PTEEM, nous utilisons toujours 20 chaînes, mais avec 6 niveaux d'énergie : nous prenons $H_1 = 0.5$ et H_2 à H_6 régulièrement espacés sur une échelle logarithmique entre 1.5 et 20. Les températures sont toujours régulièrement espacées sur une échelle logarithmique entre 1 et 60. Pour l'EES, 6 chaînes sont utilisées avec les mêmes niveaux d'énergie que le PTEEM, et les températures sont aussi régulièrement espacées sur une échelle logarithmique entre 1 et 60, et $p_{ee} = 0.1$. C'est particulièrement intéressant, car les 20 modes sont répartis dans trois anneaux d'énergie différents, et le second anneau d'énergie contient des modes avec différentes variances (0.4^2 et 0.1^2). La figure de gauche de la Figure 6.4 représente 100000 simulations obtenues suivant le mélange décrit plus haut, les couleurs représentant les anneaux d'énergie.

Concernant les estimations des premiers et seconds moments, le tableau 6.2 (B) montre que les estimations du PTEEM sont toujours plus justes et précises que celles de l'EES. A propos du nombre de modes visités, l'algorithme PTEEM en a visité 18.91 en moyenne sur les 100 runs,

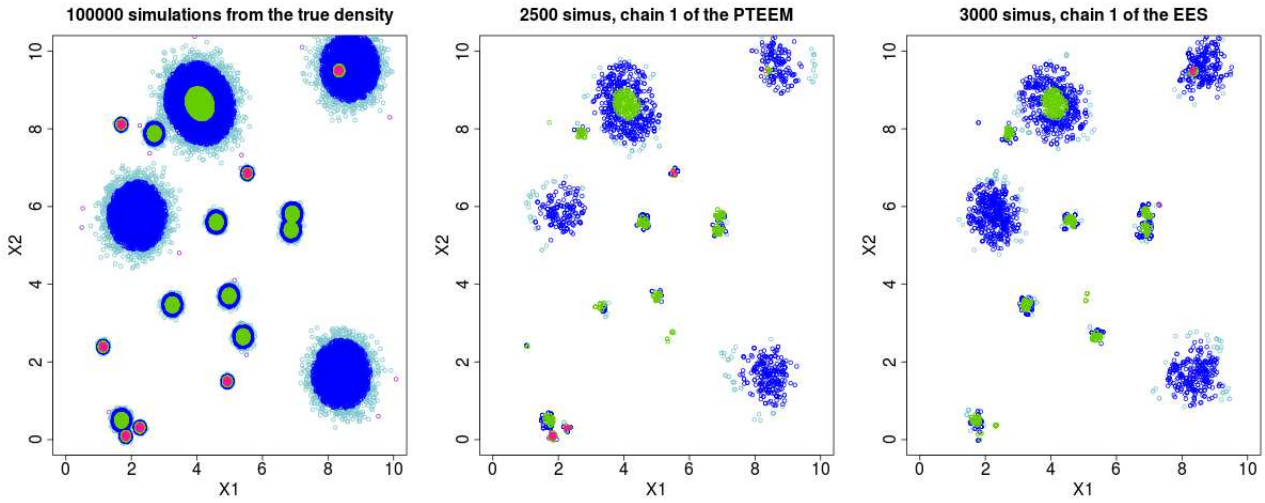

FIGURE 6.4 – A gauche, 100000 simulations obtenues suivant le mélange d'intérêt. Au milieu, simulations générées par la première chaîne d'un run de l'algorithme PTEEM. A droite, simulations générées par la première chaîne d'un run de l'EES. Les couleurs correspondent aux niveaux d'énergie.

contre 17.83 en moyenne pour l'EES. Enfin, en ce qui concerne les erreurs absolues de fréquence, nous observons une amélioration du PTEEM par rapport à l'EES, avec la moyenne des ratios *médiane EES/médiane PTEEM* sur les 20 modes de 1.213, et la moyenne des ratios *maximum EES/maximum PTEEM* sur les 20 modes de 1.203.
Notons que si certains composants du mélange sont associés à de très petites variances ($\sigma = 0.01$ par exemple), il devient difficile pour le PTEEM et l'EES de détecter ces modes quand ils sont isolés, éloignés de modes plus « larges ».

6.2.2 En combinaison avec un échantillonneur de Gibbs

Nous avons vu dans la partie 5.4.3 que nous ne pouvons pas utiliser un échantillonneur de Gibbs pour les actualisations locales de l'EES, sans mettre en oeuvre une méthode pour simuler des vraisemblances « étêtées ».

Dans cette partie, nous comparons donc seulement les résultats des méthodes PT et PTEEM. Deux exemples de mélanges de lois normales sont utilisés : l'exemple avec des données simulées qui a été présenté dans la partie 5.1.2, ainsi que l'exemple bien connu du jeu de données galaxy (voir Richardson et Green [184] parmi d'autres).

L'échantillonneur de Gibbs pour la $i^{\text{ème}}$ chaîne

La modélisation d'un mélange de lois normales est donnée dans la partie 5.1.2. Notre objectif est d'estimer les paramètres inconnus du modèle, soit les moyennes μ, les variances σ^2, les poids w, le vecteur des étiquettes c et le paramètre β. C'est classiquement un échantillonneur de Gibbs

qui est utilisé pour ce type de problème, c'est donc cet échantillonneur que nous utilisons pour les mouvements locaux.

Densités a posteriori jointes

Nous notons $y = (y_i)_{i=1\ldots n}$, $\mu = (\mu_p)_{p=1\ldots k}$, $\sigma^2 = (\sigma_p^2)_{p=1\ldots k}$, $w = (w_p)_{p=1\ldots k}$, $c = (c_i)_{i=1\ldots n}$, et m_p est le nombre d'observations classées dans la classe p : $m_p = \sum_{i=1}^{n} \mathbb{1}_{c_i=p}$. La densité cible suivant laquelle nous voulons échantillonner est la densité a posteriori jointe des paramètres :

$$\pi(x) \quad = \quad p(\mu, \sigma^{-2}, w, c, \beta \mid y) \quad \propto \quad p(y \mid \mu, \sigma^{-2}, c) p(\mu, \sigma^{-2}, c, \beta, w).$$

Concernant la $i^{\text{ème}}$ chaîne, elle doit normalement être échantillonnée suivant la distribution tempérée suivante :

$$\pi_i(x) \quad \propto \quad \pi(x)^{\frac{1}{T_i}} \quad \propto \quad p(y \mid \mu, \sigma^{-2}, c)^{\frac{1}{T_i}} p(\mu, \sigma^{-2}, w, c, \beta)^{\frac{1}{T_i}}.$$

Cependant, comme l'ont fait remarqué Jasra et al. [108] ainsi que Behrens et Hurn [21], en tempérant ainsi l'ensemble de l'a posteriori nous n'avons aucune garantie que la distribution tempérée sera propre. Par conséquent, il est préférable de ne tempérer que la vraisemblance, en laissant les a priori non tempérés. Nous évitons ainsi d'être confrontés à une distribution impropre. Nous échantillonnons donc la $i^{\text{ème}}$ chaîne suivant (formule plus détaillée en annexe (D.3)) :

$$\pi_i'(x) \quad \propto \quad p(y \mid \mu, \sigma^{-2}, c)^{\frac{1}{T_i}} p(\mu, \sigma^{-2}, c, \beta, w). \tag{6.8}$$

Distributions conditionnelles complètes

Les distributions conditionnelles complètes à utiliser dans l'échantillonneur de Gibbs pour la $i^{\text{ème}}$ chaîne sont facilement obtenues par conjugaison.

Nous utilisons les notations suivantes, avec $i = 1, \ldots N$ l'indice de la chaîne, $p = 1, \ldots, k$ l'indice du composant et $l = 1, \ldots, n$ l'indice de l'observation :

$$\begin{aligned} x_i = (\mu_i, \sigma_i^{-2}, w_i, c_i, \beta_i), \qquad & \mu_i = (\mu_{i1}, \mu_{i2}, \ldots, \mu_{ik}), \\ \sigma_i^{-2} = (\sigma_{i1}^{-2}, \sigma_{i2}^{-2}, \ldots, \sigma_{ik}^{-2}), \qquad & w_i = (w_{i1}, w_{i2}, \ldots, w_{ik}), \\ c_i = (c_{i1}, c_{i2}, \ldots, c_{in}), \qquad & m_i = (m_{i1}, m_{i2}, \ldots, m_{ik}). \end{aligned} \tag{6.9}$$

Pour μ_i, σ_i^{-2} et w_i, les distributions conditionnelles complètes sont les suivantes :

$$\mu_{ip} \mid \sigma_{ip}^{-2}, y, c_i, \xi, \kappa^{-1} \sim \mathcal{N}\left(\left(\frac{m_{ip}\sigma_{ip}^{-2}}{T_i} + \kappa \right)^{-1} \left(\frac{\sigma_{ip}^{-2}}{T_i} \sum_{l:c_{il}=p} y_l + \xi\kappa \right), \left(\frac{m_{ip}\sigma_{ip}^{-2}}{T_i} + \kappa \right)^{-1} \right). \tag{6.10}$$

$$\sigma_{ip}^{-2} \mid \mu_{ip}, y, c_i, \alpha, \beta_i \sim \Gamma\Big(\alpha + \frac{m_{ip}}{2T_i}, \beta_i + \sum_{l:c_{il}=p} \frac{(y_l - \mu_{ip})^2}{2T_i}\Big). \tag{6.11}$$

$$w_i \mid c_i, \delta \sim D(\delta + m_{i1}, \delta + m_{i2}, \ldots, \delta + m_{ik}). \tag{6.12}$$

Pour le vecteur des étiquettes c_i, sa distribution conditionnelle complète est une multinomiale avec les probabilités suivantes :

$$p_i(c_{il} = p \mid y, \mu_i, \sigma_i^{-2}, w_i) \propto \frac{1}{\sigma_{ip}^{\frac{1}{T_i}}} \exp\Big(-\frac{(y_l - \mu_{ip})^2}{2\sigma_{ip}^2 T_i}\Big) w_{ip}. \tag{6.13}$$

Enfin la distribution conditionnelle complète du paramètre β_i est la suivante :

$$\beta_i \mid \sigma_i^{-2}, \alpha, g, h \sim \Gamma\Big(g + k\alpha, h + \sum_{p=1}^{k} \sigma_{ip}^{-2}\Big). \tag{6.14}$$

Probabilité d'acceptation du mouvement Equi-Energy

Soit deux chaînes i et j sélectionnées dans un anneau d'énergie pour effectuer un mouvement Equi-Energy, la probabilité d'acceptation de ce mouvement est donnée par :

$$1 \wedge \frac{\pi_i'(x_j)\pi_j'(x_i)}{\pi_i'(x_i)\pi_j'(x_j)} = 1 \wedge \Big(\frac{p(y|x_i)}{p(y|x_j)}\Big)^{(1/T_j - 1/T_i)},$$

avec

$$\frac{\pi_i'(x_j)\pi_j'(x_i)}{\pi_i'(x_i)\pi_j'(x_j)} = \Big[\frac{\prod_{p=1}^k \sigma_{jp}^{-m_{jp}}}{\prod_{p=1}^k \sigma_{ip}^{-m_{ip}}}\Big]^{\frac{1}{T_i} - \frac{1}{T_j}} \exp\Big[-\frac{1}{2}\Big(\frac{1}{T_i} - \frac{1}{T_j}\Big)\Big(\sum_{l=1}^{n}(y_l - \mu_{jc_{jl}})^2\sigma_{jc_{jl}}^{-2} - \sum_{l=1}^{n}(y_l - \mu_{ic_{il}})^2\sigma_{ic_{il}}^{-2}\Big)\Big]. \tag{6.15}$$

Critère de comparaison entre les échantillonneurs

Même si un échantillonneur explore très mal l'espace des paramètres et reste bloqué dans un mode local (voire global), l'estimation des paramètres qui s'ensuit peut être satisfaisante. Nous ne pouvons donc pas comparer les échantillonneurs sur leurs capacités à correctement estimer les paramètres. Par contre, le phénomène de « label-switching » permet d'avoir une idée sur la capacité d'exploration de l'espace des paramètres (voir annexe C.1). En effet, comme les paramètres du mélange suivent des distributions a priori échangeables, la distribution a posteriori des paramètres a $k!$ modes qui doivent tous être visités en proportions égales. C'est ce critère que nous utilisons pour comparer les échantillonneurs.

Application sur un jeu de données simulé

Nous reprenons l'exemple du mélange de 4 lois normales présenté dans la partie 5.1.2, et lançons 100 fois chacun des échantillonneurs PT et PTEEM. Comme dans l'exemple précédent, dans l'algorithme PT ce sont des chaînes adjacentes qui sont choisies pour être échangées. Chaque run se compose de 5000 itérations dont 1000 de burn-in. Nous utilisons $N = 20$ chaînes, $K = 5$ niveaux d'énergie, et les hyper-paramètres suivants (voir 5.2) : $\alpha = 2$, $\xi = \bar{y}$, $\delta = 1$, $\kappa = 1/R^2$, $g = 0.2$ et $h = 10/R^2$, avec $R = \max(data) - \min(data)$. Concernant les niveaux d'énergies, ils sont régulièrement espacés sur une échelle logarithmique entre $H_1 = 105$ et $H_5 = 285$. Concernant les températures, elles sont également réparties en échelle logarithmique de la manière suivante : 4 températures entre 1 et 1.17, 10 températures entre 1.18 et 1.28 et 6 températures entre 1.30 et 10.

REMARQUE 6.2.1 *Nous avons été confronté à un problème de calibration comme illustré dans le tableau 6.1.3 : il y a un « saut d'énergie » entre les chaînes de températures inférieures à 1.18 et celles de températures supérieures à 1.28. En effet, celles de températures inférieures à 1.18 sont la majorité du temps dans le premier anneau, tandis que celles de températures supérieures à 1.28 sont dans les deux derniers anneaux. Nous avons donc introduit de nombreuses températures entre 1.18 et 1.28.*

Nous vérifions sur un run de PTEEM que ces choix de températures et d'énergies sont pertinents, qu'il n'y a pas de sauts d'énergie entre les chaînes, voir le tableau 6.7 pour la répartition des états de quelques unes de ces chaînes.

	$(-\infty, 134.8)$	$[134.8, 173)$	$[173, 222)$	$[222, 285)$	$[285, +\infty)$
chaîne 1	3994	6	0	0	0
chaîne 4	3661	253	84	2	0
chaîne 8	1223	332	1336	1046	63
chaîne 10	408	324	1534	1618	116
chaîne 12	36	92	756	2694	422
chaîne 16	0	0	0	1205	2795
chaîne 20	0	0	0	34	3966

TABLE 6.7 – Répartition des états issus de 4000 itérations d'un run de l'algorithme PTEEM dans les différents anneaux d'énergie, pour les chaînes 1, 4, 8, 10 12, 16 et 20.

La table suivante montre qu'en moyenne sur 100 runs de PTEEM la chaîne d'intérêt parvient à visiter plus de modes que sur 100 runs de PT.

Pour chaque run des algorithmes PT et PTEEM, nous examinons pour chaque mode le nombre de fois qu'il a été visité par la chaîne d'intérêt, et calculons son erreur absolue de fréquence $err_m = |\widehat{f_m} - 1/4!|$, où $\widehat{f_m}$ est la fréquence empirique du nombre de visites du mode m. Puis nous calculons la moyenne et la médiane de cette erreur absolue de fréquence sur l'ensemble

	moyenne	écart-type	min	max
PT	12.42	1.77	8	16
PTEEM	17.18	2.05	13	22

TABLE 6.8 – Moyenne, écart-type, valeurs minimale et maximale du nombre de modes visités lors de 100 runs de PT et PTEEM.

.

des modes et sur les 100 runs. Cela donne une moyenne et une médiane de 4,9% et 4.2% pour PT, et de 3.9% et 3.8% pour PTEEM.

Le tableau 6.9 donne le détail des mouvements Equi-Energy du PTEEM. Nous observons que les mouvements Equi-Energy ont plus souvent lieu entre chaînes de températures proches.

	chaîne 1	chaîne 10	chaîne 20
chaîne 1	0.00	1.00	0.00
chaîne 2	23.29	1.14	0.00
chaîne 3	20.07	1.48	0.00
chaîne 4	14.41	3.12	0.00
chaîne 5	12.45	4.24	0.00
chaîne 6	10.52	5.36	0.00
chaîne 7	8.02	7.97	0.01
chaîne 8	5.44	11.06	0.01
chaîne 9	3.17	13.91	0.01
chaîne 10	1.49	0.00	0.08
chaîne 11	0.75	14.00	0.13
chaîne 12	0.28	11.79	0.14
chaîne 13	0.08	9.81	0.18
chaîne 14	0.04	8.21	0.41
chaîne 15	0.00	5.80	0.43
chaîne 16	0.00	0.58	4.62
chaîne 17	0.00	0.23	13.43
chaîne 18	0.00	0.15	29.55
chaîne 19	0.00	0.10	51.01
chaîne 20	0.00	0.03	0.00

TABLE 6.9 – proportions (%) d'échanges Equi-Energy acceptés impliquant la chaîne 1 avec chacune des autres chaînes (moyenne sur 100 runs *PTEEM*). Idem pour chaînes 10 et 20.

Les taux d'acceptation moyens des mouvements d'échange sur les cents runs sont de 61% et 53% respectivement pour les algorithmes PT et PTEEM. Il est intéressant de noter qu'apparemment PTEEM explore mieux l'espace des paramètres que PT, bien qu'en moyenne moins d'échanges sont acceptés pour PTEEM que pour PT. Cela laisse penser que les mouvements

proposés par PTEEM sont plus pertinents et permettent plus de passer d'un mode à l'autre. En effet, pour PT un mode découvert par la dernière chaîne mettra un certain temps à traverser l'ensemble des chaînes, car les échanges se font entre chaînes adjacentes. Dans le cas de PTEEM il est probable qu'un mode découvert par la chaîne 20 mette moins de temps à traverser l'ensemble des chaînes, car la dernière chaîne peut échanger avec des chaînes de températures moyennes par exemple. Nous pouvons utiliser un PT avec des échanges entre chaînes non adjacentes, mais dans ce cas le taux d'acceptation moyen d'un mouvement d'échange est beaucoup plus faible. En comparaison PTEEM a l'avantage de proposer des échanges entre chaînes pas forcément adjacentes, mais ayant toujours des états d'énergies similaires.

Application au jeu de données Galaxy

Nous utilisons le jeu de données Galaxy qui a été étudié de nombreuses fois (voir par exemple Richardson et Green [184] ou Behrens et Hurn [21]), et qui comprend des mesures de vélocité de 82 galaxies. Le nombre de composants est fixé à $k = 6$, qui est le nombre de composants ayant la plus forte probabilité a posteriori (voir Richardson et Green [184]).
Nous lançons 100 fois chacun des échantillonneurs PT et PTEEM. Chaque run se compose de 12000 itérations dont 2000 de burn-in. Nous utilisons $N = 20$ chaînes, $K = 5$ niveaux d'énergie, et les hyper-paramètres suivants (voir 5.2) : $\alpha = 3$, $\xi = 20$, $\delta = 1$, $\kappa = 1/R^2$, $g = 0.2$ et $h = 10/R^2$, avec $R = 10$. Concernant les niveaux d'énergies, ils sont régulièrement espacées sur une échelle logarithmique entre $H_1 = 180$ et $H_5 = 260$. Concernant les températures, nous prenons des températures entre 1 et 4, ayant leurs inverses régulièrement espacés entre 0.25 et 1.

Le tableau 6.10 montre qu'en moyenne sur 100 runs de PTEEM la chaîne d'intérêt parvient à visiter plus de modes que sur 100 runs de PT.

	moyenne	écart-type	min	max
PT	645.04	13.52	610	683
PTEEM	666.52	9.23	641	692

TABLE 6.10 – Moyenne, écart-type, valeurs minimale et maximale du nombre de modes visités lors de 100 runs de PT et PTEEM.

Pour chaque run des algorithmes PT et PTEEM, nous examinons pour chaque mode le nombre de fois qu'il a été visité par la chaîne d'intérêt, et calculons son erreur absolue de fréquence. Les erreurs sont légèrement plus faibles pour l'algorithme PTEEM que pour l'algorithme PT : moyenne et médiane de 0.126% et 0.099% pour PT, et de 0.119% et 0.099% pour PTEEM. Dans cet exemple aussi il est observé que les mouvements Equi-Energy ont plus souvent lieu entre chaînes de températures proches (voir le tableau D.1 en annexe). Les taux d'acceptation moyens des mouvements d'échange sur les 100 runs sont de 61% et 49% respectivement pour les algorithmes PT et PTEEM.

6.2.3 En combinaison avec un échantillonneur à sauts réversibles

Le modèle utilisé est le même que celui de la partie 6.2.2, nous sommes toujours dans le cas d'un mélange de lois normales. La seule différence est que le nombre de composants du mélange k est considéré inconnu. Nous supposons qu'a priori k suit une loi uniforme,

$$k \sim \mathcal{U}_{[1,k_{max}]}.$$

Le vecteur des paramètres inconnus est $x = (\mu, \sigma^{-2}, w, c, \beta, k)$. Par rapport à la partie 6.2.2, nous avons un paramètre de plus : k le nombre de composants. Dans les formules, l'a priori de k intervient par $\mathbb{1}_{[1,k_{max}]}(k)$.

L'échantillonneur à sauts réversibles

Concernant la $i^{\text{ème}}$ chaîne, les actualisations des paramètres $\mu_i, \sigma_i^{-2}, w_i, c_i$ et β_i se font avec les distributions conditionnelles complètes données dans la partie 6.2.2, en remplaçant k par k_i et x_i par $(\mu_i, \sigma_i^{-2}, w_i, c_i, \beta_i, k_i)$. Pour les mouvements à dimensions variables, en nous basant sur les comparaisons d'échantillonneurs trans-dimensionnels effectuées par Cappé et al. [33] (voir annexe B.2.4), nous choisissons d'utiliser un échantillonneur à sauts réversibles ne contenant que des mouvements de création ou de destruction de composants vides (mouvements « birth-and-death »).

Choix entre une création et une destruction

La probabilité de choisir une création de composant vide est b_{k_i}, et la probabilité de choisir une destruction de composant vide est $d_{k_i} = 1 - b_{k_i}$. Avec, évidemment, $d_1 = 0$ et $b_{k_{max}} = 0$. Sinon, $b_{k_i} = d_{k_i} = 0.5$.

Cas d'une création

Trois variables auxiliaires sont utilisées : le poids w_{ip^*}, la moyenne μ_{ip^*} et la variance $\sigma^2_{ip^*}$ associés au nouveau composant p^*. Nous générons :

$$\mu_{ip^*} \sim \mathcal{N}(\xi, \kappa^{-1}), \qquad \sigma^{-2}_{ip^*}, \sim \Gamma(\alpha, \beta) \qquad w_{ip^*} \sim be(1, k).$$

Puis nous pondérons les poids de manière à ce que leur somme vaille toujours 1. Ainsi les anciens poids w_{ip} correspondants aux k_i anciens composants deviennent, pour $k_i + 1$ composants (les anciens plus le nouveau de poids w_{ip^*}), $w_{ip}(1 - w_{ip^*})$.
La naissance proposée est acceptée avec probabilité $\rho_i = min(1, A_i)$, avec :

$$A_i = \frac{p(k_i+1)}{p(k_i)}(k_i+1)\frac{1}{B(\delta, k_i\delta)}(1-w_{ip^*})^{n+k_i\delta}w_{ip^*}^{\delta-1}\frac{d_{k_i+1}}{(k_{i0}+1)b_{k_i}} \times \frac{1}{be_{1,k_i}(w_{ip^*})}. \tag{6.16}$$

Cas d'une destruction
Nous devons choisir aléatoirement quel composant vide supprimer. On le supprime, puis les poids restants sont pondérés de manière à ce que leur somme vaille toujours 1. La mort proposée est acceptée avec probabilité $\rho_i = min(1, A_i^{-1})$.

Probabilité d'acceptation du mouvement Equi-Energy

Les deux chaînes proposées pour un mouvement Equi-Energy peuvent être de dimension différentes, si leurs nombres de composants ne sont pas égaux. Soit deux chaînes i et j sélectionnées dans un anneau d'énergie pour effectuer un mouvement Equi-Energy, la probabilité d'acceptation de ce mouvement est donnée par :

$$1 \wedge \frac{\pi'_i(x_j)\pi'_j(x_i)}{\pi'_i(x_i)\pi'_j(x_j)} = 1 \wedge \Big(\frac{p(y|x_i)}{p(y|x_j)}\Big)^{(1/T_j - 1/T_i)},$$

avec

$$\frac{\pi'_i(x_j)\pi'_j(x_i)}{\pi'_i(x_i)\pi'_j(x_j)} = \left[\frac{\prod_{p=1}^{k_j}\sigma_{jp}^{-m_{jp}}}{\prod_{p=1}^{k_i}\sigma_{ip}^{-m_{ip}}}\right]^{\frac{1}{T_i}-\frac{1}{T_j}} \exp\left[-\frac{1}{2}\Big(\frac{1}{T_i}-\frac{1}{T_j}\Big)\Big(\sum_{l=1}^{n}(y_l-\mu_{jc_{jl}})^2\sigma_{jc_{jl}}^{-2}-\sum_{l=1}^{n}(y_l-\mu_{ic_{il}})^2\sigma_{ic_{il}}^{-2}\Big)\right] \tag{6.17}$$

La différence avec la formule de la partie 6.2.2 est que k_i peut être différent de k_j.

Application à la sensibilité de *Plasmodium Falciparum* à la doxycycline

Nous disposons de données de sensibilité du parasite *Plasmodium Falciparum* (responsable du paludisme), à la molécule de Doxycycline. Ces données nous ont été fournies par S. Briolant (médecin militaire à l'Institut de Médecine Tropicale du Service de Santé des Armées) et F. Nosten (directeur de l'unité de recherche sur la malaria de Shoklo (SMRU) en Thaïlande). Des échantillons de sang de 723 patients atteints du paludisme ont été receuillis en Thaïlande, puis exposés à différentes concentrations de Doxycycline. Pour chacun, nous notons la Concentration d'Exposition à laquelle 50% de la population de parasites est tuée (CE50). A la vue des données, les médecins pensent que les échantillons sont issus de plusieurs populations ayant différentes sensibilités à la doxycycline. Cependant, ils n'ont pas d'idée fixe sur le nombre de ces populations. Nous appliquons donc l'algorithme PTEEM en combinaison avec un échantillonneur à sauts réversibles sur ces données (au préalable centrées et réduites).

Nous utilisons $N = 20$ chaînes, $K = 5$ niveaux d'énergie, et les hyper-paramètres suivants (voir 5.2) : $R = \max(data) - \min(data)$, $\xi = mean(data)$, $\alpha = 3$, $g = 0.2$, $\delta = 1$, $h = 10/R^2$, $\kappa = 1/R^2$ et $k_{max} = 10$. Concernant les températures : cinq sont régulièrement espacées sur une échelle logarithmique entre 1 et 1.013, trois le sont entre 1.04 et 1.19, trois entre 1.27 et 1.36, cinq entre 1.45 et 1.9, et quatre entre 2 et 10. Concernant les niveaux d'énergies, $H_1 = 525$, et

les quatre autres niveaux sont régulièrement espacés sur une échelle logarithmique entre 740 et 1030.

Nous lançons trois runs du PTEEM en combinaison avec un échantillonneur à sauts réversibles. La figure 6.5 donne la distribution a posteriori obtenue pour le nombre de composants k sur les trois runs et pour la première chaîne. D'après cette figure, nous décidons de fixer le nombre de composants à 4. Remarquons qu'en utilisant simplement un échantillonneur à sauts réversibles au lieu du PTEEM, certains runs restent bloqués dans des modes locaux (par exemple, dans un mode où $k = 2$ et où les deux composants ne sont pas vraiment pertinents). D'où l'intérêt ici d'utiliser PTEEM.

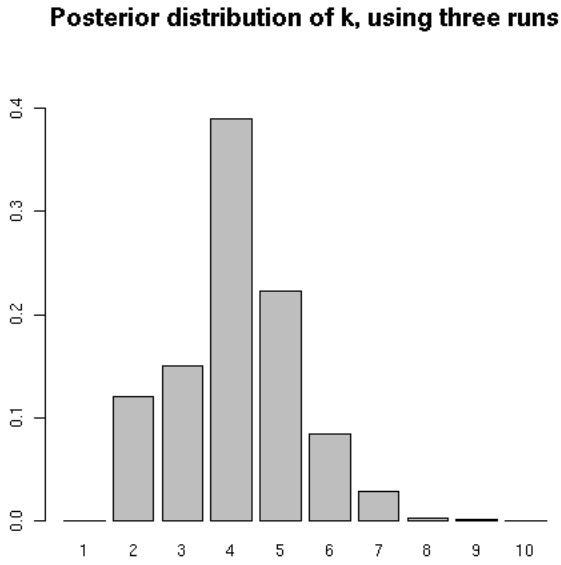

FIGURE 6.5 – Distribution a posteriori obtenue pour le nombre de composants k (moyenne sur trois runs du PTEEM en combinaison avec un échantillonneur à sauts réversibles).

Nous lançons ensuite trois runs du PTEEM en combinaison avec un échantillonneur de Gibbs, en ayant fixé $k = 4$. L'inférence n'étant pas directement possible à cause du phénomène de « label-switching », nous labellisons les différents composants. Pour cela nous utilisons une contrainte d'identifialibilité sur les moyennes (moyennes en ordre croissant) : c'est ici la façon de labelliser la plus pertinente d'un point de vue biologique. Les résultats des trois runs sont très similaires, les quatre composants estimés à partir de la première chaîne par les trois runs sont très proches. Les paramètres estimés sont donnés en supplément dans le tableau D.2, et la figure 6.6 représente les données ainsi que les composants estimés a posteriori et le mélange associé. Le mélange obtenu est assez proche des données observées, ce qui est satisfaisant. De plus, comme attendu nous trouvons une population de parasites plutôt « sensibles » à la doxycycline (en violet, CE50

moyenne faible), et une population de parasites plutôt « résistants » (en bleu clair, CE50 moyenne élevée). Un article est actuellement en préparation, dans lequel nous essayons de caractériser les composants estimés par certains profils génétiques.

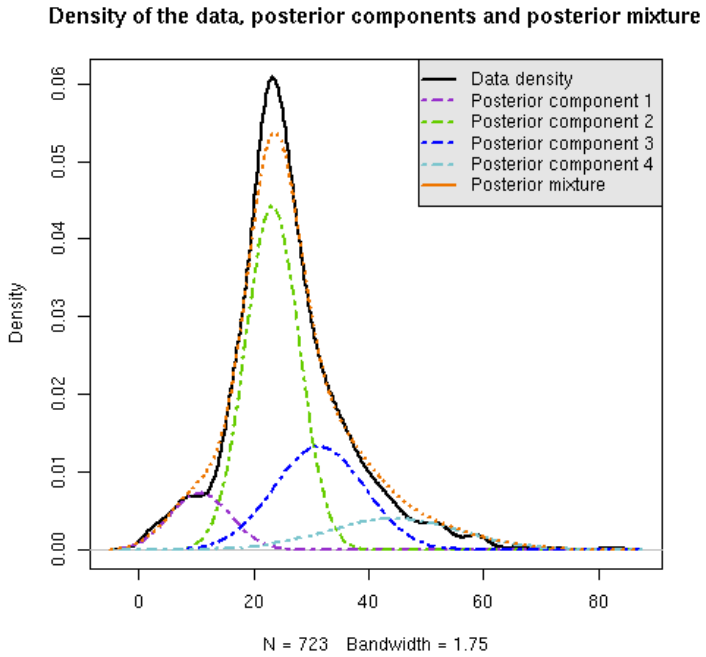

FIGURE 6.6 – Densité des données de sensibilité de *Plasmodium Falciparum* à la doxycycline (en noir), les 4 composants estimés a posteriori (en violet, vert, bleu clair et bleu foncé) ainsi que le mélange associé (en orange).

6.2.4 Recherche de sites promoteurs de facteurs de transcription

En biologie, de nombreux chercheurs veulent comprendre comment l'expression des différents gènes est controlée, régulée. La compréhension d'un tel mécanisme peut notamment déboucher sur des applications médicales. En résumé, à partir d'une séquence d'ADN, un phénomène de transcription produit une séquence d'ARN, qui va elle-même être traduite en une chaîne polypeptidique (une protéine) constituée d'acides aminés. Le code de l'ADN n'est pas entièrement utilisé, seules quelques régions de l'ADN vont être transcrites et utilisées pour former l'ARN. Ces régions constituent les gènes. Mais tous les gènes ne sont pas transcrits dans toutes les circonstances. La transcription de certains gènes va dépendre de l'environnement, du stade de développement, du type cellulaire,.... Par exemple, certains gènes ne seront trancrits que dans des cellules musculaires. Le rôle d'un promoteur est alors de déterminer à quel endroit doit commencer ou pas la transcription de l'ADN. Afin de réguler l'expression d'un gène, des facteurs de

transcription (ou éléments trans-régulateurs) vont se lier à des sites de fixation des promoteurs afin de favoriser la transcription du gène associé ou bien de la bloquer. La compréhension de l'interaction entre ces sites de fixation et les facteurs de transcription est donc d'un grand intérêt. Une étape nécessaire à cette compréhension est l'identification de ces sites de fixation. Pour cela, des séquences d'ADN homologues (provenant d'espèces différentes) sont utilisées. Mais à cause des phénomènes d'évolution, les sites de fixation des facteurs de transcription d'un même gène ne sont pas exactement identiques entre ces différentes séquences, ce qui rend leur identification délicate. Les premières méthodes développées pour les identifier sont des méthodes d'alignement multiples (voir Stormo et Hartzell [213]). Puis des méthodes plus statistiques ont été développées, utilisant notamment des algorithmes Expectation-Maximisation (Lawrence et Reilly [127]) ou des échantillonneurs de Gibbs (Lawrence et al. [126], Liu et al. [143]). Voir Jensen et al. [109] pour une revue des méthodes bayésiennes utilisées pour ce type de problème. L'identification de tels sites de fixation est considérée comme un problème complexe et difficile. En effet, d'un point de vue combinatoire le nombre de possibilités est très grand, et un échantillonneur de Gibbs a tendance à rester coincé dans un des nombreux modes locaux.
Afin de comparer les performances de l'EES avec celles du PTEEM, tout comme Kou et al. [118] nous nous sommes intéressés au problème de l'identification de sites de fixations dans des séquences homologues, sous certaines hypothèses : les sites de fixation sont supposés en un seul bloc, la longueur des sites est connue, et nous nous plaçons dans le cas d'un seul type de site. Nous voulons comparer des algorithmes aussi simples que possibles dans ce type de problème, afin de ne pas introduire de biais de comparaison. Cependant, chacune de ces hypothèse peut être affaiblie sans trop de difficultés (voir Liu et al. [143], Jensen et al. [109] ou Liu et al. [145]). Nous considérons que le nombre de sites est inconnu, et les séquences peuvent être de longueurs différentes.

Le modèle et les données

Soit S un ensemble de M séquences contenant potentiellement un ou plusieurs sites de fixation. Chaque séquence est constituée d'une succession de quatre nucléotides : A, C, G et T. Nous notons $|A|$ le nombre total de sites de fixation, inconnu. Les longueurs des séquences sont notées $L_1, L_2, \ldots, L_M$, et w est la longueur des sites de fixations. L'objectif est de trouver les positions initiales des sites de fixations. Pour la séquence m, il y a $L_m^* = L_m - w + 1$ positions initiales possibles pour des sites de fixation (la séquence ne peut contenir que des sites en entier). Comme le nombre total de sites est inconnu, nous considérons que les M séquences (amputées de leurs w dernières bases) mises à la suite les unes des autres constituent une seule grande séquence, contenant $|A|$ sites de fixation. Nous nous intéressons alors au vecteur $A = (a_1, a_2, \ldots, a_{L^*})$ constitué de 0 et de 1, avec $a_i = 1$ si la $i^{\text{ème}}$ position de la grande séquence est le point de départ d'un site, et $a_i = 0$ sinon. Comme il y a L_m^* positions initales possibles pour chacune des séquences,

A est de taille $L^* = \sum_{m=1}^{M} L_m^*$. L'objectif devient donc l'estimation de A.
L'ensemble des séquences est constitué de deux sous-ensembles (voir figure 6.7) : l'ensemble des nucléotides constituant les motifs des différents sites de fixation, et l'ensemble des nucléotides ne faisant pas partie de ces sites de fixation (appelé « background-sequence »).

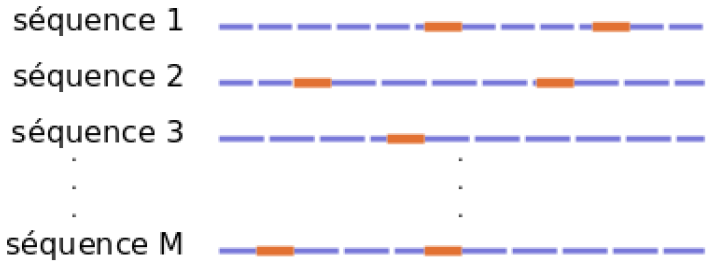

FIGURE 6.7 – Représentation des séquences d'ADN. En orange sont représentés les sites de fixation. En bleu pointillé sont représentées les morceaux de séquence ne faisant pas partie d'un site de fixation. Les sites oranges alignés sont notés $S(A)$, et les morceaux bleus les uns à la suite des autres forment la « background-sequence », notée $S(A^C)$.

L'ensemble des motifs des différents sites de fixation alignés est noté $S(A)$, et la « background-sequence » est notée $S(A^C)$. Ces deux sous-ensembles sont modélisés différemment. Au début, l'ensemble $S(A^C)$ était supposé suivre un modèle multinomial produit (voir Liu et al. [143]). Cependant il a été montré un peu plus tard qu'un modèle de Markov est plus adapté aux séquences réellement observées, et apporte une amélioration, mais il complique la tâche des échantillonneurs (Jensen et al. [109]). En effet, il est plus difficile de détecter des sites de fixation dans des séquences du type AAATTTTCGGGC, que dans des séquences du type ACTGTACGTGT, car les motifs répétés peuvent constituer des modes locaux. Dans cet exemple, nous considérons comme Kou et al. [118] que l'ensemble $S(A^C)$ suit un modèle de Markov d'ordre 1, avec la matrice de transition suivante entre les bases :

$$\theta_0 = \begin{array}{c} \\ \text{A} \\ \text{C} \\ \text{G} \\ \text{T} \end{array} \begin{array}{c} \begin{array}{cccc} \text{A} & \text{C} & \text{G} & \text{T} \end{array} \\ \begin{pmatrix} 1-3\alpha & \alpha & \alpha & \alpha \\ \alpha & 1-3\alpha & \alpha & \alpha \\ \alpha & \alpha & 1-3\alpha & \alpha \\ \alpha & \alpha & \alpha & 1-3\alpha \end{pmatrix} \end{array},$$

avec $\alpha = 0.12$. Cette matrice θ_0 peut être supposée connue. En effet, elle peut être estimée assez précisemment depuis la séquence dont nous disposons, si les sites de fixation n'en constituent qu'une petite partie.
En ce qui concerne l'ensemble des sites de fixation alignés $S(A)$, il peut être vu comme une matrice de dimensions $|A| \times w$, avec les sites de fixation en ligne et les différentes positions de ces sites en colonne, la colonne k contenant les nucléotides en $k^{\text{ème}}$ position des sites. Nous

introduisons un vecteur de comptage $C = (C_1, ..., C_w)$, où $C_k = (C_{kA}, C_{kC}, C_{kG}, C_{kT})$, avec C_{kG} par exemple le nombre total de G en $k^{\text{ème}}$ position sur tous les sites de fixation. Les motifs de $S(A)$ sont supposés être obtenus d'après une loi multinomiale produit de paramètres $\Theta = (\theta_1, \ldots, \theta_w)$, avec :

$$\Theta = \begin{array}{c} \begin{array}{cccc} \theta_1 & \theta_2 & \ldots & \theta_w \end{array} \\ \begin{pmatrix} \theta_{1A} & \theta_{2A} & \ldots & \theta_{wA} \\ \theta_{1C} & \theta_{2C} & \ldots & \theta_{wC} \\ \theta_{1G} & \theta_{2G} & \ldots & \theta_{wG} \\ \theta_{1T} & \theta_{2T} & \ldots & \theta_{wT} \end{pmatrix} \end{array}.$$

D'après le modèle, chacun des C_k suit une multinomiale de paramètre θ_k, et les différents C_k sont supposés indépendants entre eux. Pour notre exemple, nous prenons :

$$\Theta = \begin{pmatrix} 0.6 & 0.1 & 0 & 0.6 & 0,1 & 0 & 0.3 & 0 & 0.2 & 0 & 0.5 & 0 \\ 0 & 0 & 0.8 & 0 & 0 & 0 & 0 & 0 & 0.2 & 0.5 & 0.25 & 0.7 \\ 0 & 0.2 & 0 & 0.1 & 0.8 & 0.7 & 0 & 0.9 & 0 & 0 & 0.25 & 0.2 \\ 0.4 & 0.7 & 0.2 & 0.3 & 0,1 & 0.3 & 0.7 & 0.1 & 0.6 & 0.5 & 0 & 0.1 \end{pmatrix},$$

ce qui correspond au WebLogo [50] de la figure 6.8.

FIGURE 6.8 – WebLogo associé aux données simulées.

A partir de θ_0 et Θ, nous générons $M = 10$ morceaux de « background-sequence » de tailles 200, $|A| = 20$ sites de fixation de longueur $w = 12$, et nous introduisons 2 sites dans chacune des 10 séquences, ce qui donne au final 10 séquences de longueur 224. Le tableau 6.11 montre $S(A)$ l'ensemble des sites de fixation alignés que nous obtenons.

	1	2	3	4	5	6	7	8	9	10	11	12
1	T	T	C	A	A	T	A	G	A	T	A	T
2	A	A	C	A	G	G	A	G	T	C	A	C
3	A	T	C	T	G	G	T	T	T	T	C	T
4	T	A	C	A	G	G	T	G	T	C	A	C
5	A	T	C	G	G	T	T	G	A	C	A	C
6	T	T	C	A	G	T	T	G	C	C	C	C
7	T	T	C	A	G	G	A	G	T	T	A	C
8	T	T	C	A	G	G	A	T	T	C	A	G
9	A	G	C	A	G	T	A	G	A	T	G	T
10	T	G	C	T	T	T	A	G	T	T	C	C
11	A	G	T	A	A	G	T	G	T	T	C	C
12	A	T	T	G	G	T	A	G	T	T	A	C
13	T	T	C	A	G	G	A	G	A	C	A	C
14	T	T	C	G	G	G	T	G	T	T	A	C
15	A	T	C	T	G	T	T	G	T	T	A	C
16	T	T	C	G	G	T	T	T	A	T	A	C
17	T	A	T	A	G	G	A	G	T	T	A	C
18	A	T	C	G	G	T	A	G	T	C	G	C
19	A	T	C	A	G	G	T	G	T	T	A	C
20	A	T	C	A	G	G	T	T	C	T	C	C

TABLE 6.11 – Les 20 sites de fixation de longueurs 12 générés.

La première séquence entière obtenue est la suivante :
TTTTAAAAATGGGCCCCGTAAAAAA**TTCAATAGATAT**TTAATGAAG
AGGGGTCCGGAAGGGCCCCCGGTTCGGGGGGTTTTTGGAAAAAGTAAA
AAGGGGTTTAAACCCCCTTTAGGGGGGTGGGGCCCGCCA**AACAGGAG**
TCACAACCTGGGGTAGGGGCCCGTTTTGGGTTGGCCCCTTTTGGGACC
CT**TTTTCCAAAACG**GCCTTTTAATTTTTTGGGGGG.

Les deux sites de fixation (en rouge) commencent aux positions 26 et 134. Nous voyons qu'il n'est pas facile à première vue de les distinguer de la « background-sequence ». D'autres motifs donnent l'impression de plus se ressembler et de provenir d'une même modélisation, par exemple les deux motifs en gras.

Distributions a priori :
La matrice de paramètres Θ est supposée suivre un a priori de produit de Dirichlet de paramètres $\beta_1, \beta_2, \ldots, \beta_w$. Soit $\theta_k \sim \mathcal{D}(\theta_k)$, avec $\beta_k = (\beta_{kA}, \beta_{kC}, \beta_{kG}, \beta_{kT})$:

$$\pi(\Theta) \propto \prod_{k=1}^{w} \theta_k^{\beta_k - 1}, \qquad \text{avec} \qquad \theta_k^{\beta_k} = \prod_{j=\{A,C,G,T\}} \theta_{kj}^{\beta_{kj}}.$$

Ensuite, la probabilité a priori de valoir 1 pour un élément a_i de A vaut p_0, p_0 étant appelé le

paramètre d'abondance :

$$\pi(A \mid p_0) = p_0^{|A|}(1-p_0)^{L^*-|A|}.$$

Enfin, ce paramètre d'abondance p_0 est supposé suivre a priori une distribution beta $Be(a,b)$:

$$\pi(p_0) \propto p_0^{a-1}(1-p_0)^{b-1}.$$

Echantillonneur de Gibbs

L'échantillonneur classiquement utilisé pour estimer A est un échantillonneur de Gibbs ayant la distribution a posteriori de A comme distribution stationnaire. Nous développons ici les formules utilisées par cet échantillonneur de Gibbs.
Les données complètes sont constituées de A et S, l'ensemble des séquences S est constitué de l'union de $S(A)$ et de $S(A^C)$, et θ_0 est supposé connu. Alors la vraisemblance des données S s'écrit :

$$\pi(S \mid A, \Theta) = \pi(S(A) \mid A, \Theta)\pi(S(A^C) \mid A, \theta_0).$$

Avec

$$\pi(S(A^C) \mid A, \theta_0) = \frac{\pi(S \mid A, \theta_0)}{\pi(S(A) \mid A, \theta_0)},$$

nous obtenons

$$\pi(S \mid A, \Theta) \quad = \quad \frac{\pi(S(A) \mid A, \Theta)\pi(S \mid A, \theta_0)}{\pi(S(A) \mid A, \theta_0)} \quad \propto \quad \frac{\pi(S(A) \mid A, \Theta)}{\pi(S(A) \mid A, \theta_0)}. \tag{6.18}$$

La vraisemblance des données complètes (A,S) peut s'écrire, avec $\theta_k^{C_k} = \prod_{j=\{A,C,G,T\}} \theta_{kj}^{C_{kj}}$:

$$\begin{aligned} \pi(S, A|\Theta, p_0) &\propto \pi(S|A,\Theta)\pi(A|p_0) \\ &\propto \frac{\pi(S(A)|A,\Theta)}{\pi(S(A)|A,\theta_0)}\pi(A|p_0) \\ &\propto \frac{1}{\pi(S(A)|A,\theta_0)} \prod_{k=1}^{w} \theta_k^{C_k} \times p_0^{|A|}(1-p_0)^{L^*-|A|}. \end{aligned} \tag{6.19}$$

La distribution jointe vaut alors, avec $\theta_k^{C_k+\beta_k-1} = \prod_{j=\{A,C,G,T\}} \theta_{kj}^{C_{kj}+\beta_{kj}-1}$:

$$\begin{aligned} \pi(S, A, \Theta, p_0) &\propto \pi(S, A|\Theta, p_0)\pi(\Theta)\pi(p_0) \\ &\propto \frac{1}{\pi(S(A)|A,\theta_0)} \prod_{k=1}^{w} \theta_k^{C_k+\beta_k-1} \times p_0^{|A|+a-1}(1-p_0)^{L^*-|A|+b-1}. \end{aligned} \tag{6.20}$$

La distribution a posteriori complète est :

$$\pi(A, \Theta, p_0 \mid S) \propto \pi(S, A, \Theta, p_0),$$

or nous ne sommes intéressés que par la distribution a posteriori du vecteur A. Nous utilisons donc la technique de « collapsing » de Liu [141] (voir annexe A), et intégrons les paramètres Θ et p_0. Nous obtenons la formule (18) de Kou et al. [118], avec $\Gamma(C_k+\beta_k) = \prod_{j=\{A,C,G,T\}} \Gamma(C_{kj}+\beta_{kj})$:

$$\pi(A \mid S) \quad \propto \quad \frac{1}{\pi(S(A)|A,\theta_0)} \frac{\Gamma(|A|+a)\Gamma(L^*-|A|+b)}{\Gamma(L^*+a+b)} \prod_{k=1}^{w} \frac{\Gamma(C_k+\beta_k)}{\Gamma(|A|+|\beta_k|)}. \tag{6.21}$$

Nous désirons implémenter un échantillonneur de Gibbs ayant $\pi(A \mid S)$ comme distribution stationnaire. Etant donné que $A = (a_1, a_2, \ldots, a_{L^*})$ et que chacun des a_i vaut 0 ou 1, il est plus facile d'actualiser A élément par élément, chacun des a_i suivant sa distribution conditionnelle complète. Nous utilisons les identités suivantes (avec des notations évidentes) :

$$A \quad = \quad a_i \cup A_{[-i]} \qquad \Rightarrow \qquad \begin{cases} |A| = |A_{[-i]}| + 1 & \text{si} \quad a_i = 1 \\ |A| = |A_{[-i]}| & \text{si} \quad a_i = 0 \end{cases}$$

$$C_k = C_{k(-i)} + C_{k(i)}.$$

Rappelons que chacun des $C_k = (C_{kA}, C_{kC}, C_{kG}, C_{kT})$ est le vecteur de comptage des nucléotides en $k^{\text{ème}}$ position des sites de fixation. Alors chacun des $C_{k(-i)} = (C_{k(-i)A}, C_{k(-i)C}, C_{k(-i)G}, C_{k(-i)T})$ est le vecteur de comptage des nucléotides en $k^{\text{ème}}$ position des sites de fixation, sans prendre en compte le site commançant en position i. Et chacun des $C_{k(i)} = (C_{k(i)A}, C_{k(i)C}, C_{k(i)G}, C_{k(i)T})$ est le vecteur de comptage des nucléotides en $k^{\text{ème}}$ position du site commençant en position i de la grande séquence, constitué de trois 0 et d'un 1. Nous nous intéressons à $\pi(S(A) \mid A, \theta_0)$ intervenant dans (6.21). $S(a_m)$ représente le site de $S(A)$ commençant en position m, et $S(A_{[-m]})$ représente $S(A)$ auquel on a enlevé le site commençant en position m. Nous avons :

$$\pi(S(A) \mid A, \theta_0) \quad = \quad \prod_{m=1}^{|A|} \pi(S(a_m) \mid A, \theta_0). \tag{6.22}$$

Alors,

$$\text{Si } a_i = 1, \qquad \pi(S(A) \mid A, \theta_0) = \pi(S(a_i) \mid A, \theta_0) \times \pi(S(A_{[-i]}) \mid A, \theta_0). \tag{6.23}$$

Nous pouvons en déduire les probabilités conditionnelles complètes suivantes pour chacun des éléments de A :

$$p(a_i = 1|A_{[-i]}, S) \quad \propto \quad \frac{1}{\pi(S(a_i) \mid A, \theta_0) \times \pi(S(A_{[-i]}) \mid A, \theta_0)} \frac{\Gamma(|A_{[-i]}| + 1 + a)\Gamma(L^* - A_{[-i]} - 1 + b)}{\Gamma(L^* + a + b)} \times \prod_{k=1}^{w} \frac{\Gamma(C_{k(-i)} + C_{k(i)} + \beta_k)}{\Gamma(|A_{[-i]}| + 1 + |\beta_k|)},$$

$$p(a_i = 0|A_{[-i]}, S) \quad \propto \quad \frac{1}{\pi(S(A_{[-i]}|\theta_0)} \frac{\Gamma(|A_{[-i]}| + a)\Gamma(L^* - A_{[-i]} + b)}{\Gamma(L^* + a + b)} \times \prod_{k=1}^{w} \frac{\Gamma(C_{k(-i)} + \beta_k)}{\Gamma(|A_{[-i]}| + |\beta_k|)},$$

ce qui donne :

$$\begin{aligned} \frac{p(a_i = 1|A_{[-i]}, S)}{p(a_i = 0|A_{[-i]}, S)} &= \frac{1}{\pi(S(a_i) \mid A, \theta_0)} \frac{\Gamma(|A_{[-i]}| + 1 + a)\Gamma(L^* - A_{[-i]} - 1 + b)}{\Gamma(|A_{[-i]}| + a)\Gamma(L^* - A_{[-i]} + b)} \\ &\quad \times \prod_{k=1}^{w} \frac{\Gamma(C_{k(-i)} + C_{k(i)} + \beta_k)}{\Gamma(C_{k(-i)} + \beta_k)} \prod_{k=1}^{w} \frac{\Gamma(|A_{[-i]}| + |\beta_k|)}{\Gamma(|A_{[-i]}| + 1 + |\beta_k|)} \\ &= \frac{1}{\pi(S(a_i) \mid A, \theta_0)} \times \frac{|A_{[-i]}| + a}{L^* - A_{[-i]} - 1 + b} \times \prod_{k=1}^{w} \left(\frac{C_{k(-i)} + \beta_k}{|A_{[-i]}| + |\beta_k|} \right)^{C_{k(i)}}. \end{aligned} \tag{6.24}$$

Algorithme EES

L'algorithme EES utilise K chaînes. La $l^{\text{ème}}$ chaîne a comme distribution cible :

$$\tilde{\pi}_l(A) \propto \exp\Big(- \frac{h(A) \vee H_l}{T_l} \Big) \qquad \text{avec} \qquad h(A) = -log(\pi(A \mid S)).$$

Nous détaillons ici les mouvements locaux des différentes chaînes.
En ce qui concerne la première chaîne, $\tilde{\pi}_1(A) = \pi(A \mid S)$. Tout comme Kou et al. [118], nous actualisons localement cette chaîne à l'aide d'un échantillonneur de Gibbs en utilisant la formule d'actualisation (6.24). Par contre, en ce qui concerne les chaînes d'ordres supérieurs, elles ne peuvent être actualisées localement par un échantillonneur de Gibbs, car les distributions sont « étêtées » (voir 5.4.3). Comme Kou et al. [118], nous actualisons localement ces chaînes à l'aide d'algorithmes de Metropolis-Hastings. Soit A l'état actuel de la $l^{\text{ème}}$ chaîne. A l'aide des vecteurs de comptage des différents nucléotides dans les sites de fixation $C_k, k = 1, \ldots, w$, la matrice Θ des paramètres de la loi multinomiale produit est estimée, puis des pseudo-comptes lui sont rajoutés, pour obtenir $\hat{\Theta}$. Plus l'ordre de la chaîne est élevé et plus les pseudo-comptes ajoutés sont importants, ce qui va faire tendre $\hat{\Theta}$ vers une uniforme. Ensuite, $\hat{p_0} = 1/\bar{L^*}$ est défini, où $\bar{L^*}$ est la moyenne des L^*_m. Alors, chaque élément de A est actualisé indépendemment : si $x_1 x_2 \ldots x_w$ sont les nucléotides d'un site de fixation qui commencerait en position i, $a^*_i = 1$ est proposé avec

probabilité :

$$q_i = \frac{\hat{p_0} \prod_{k=1}^{w} \hat{\Theta}_{kx_k}}{\hat{p_0} \prod_{k=1}^{w} \hat{\Theta}_{kx_k} + (1-\hat{p_0}) P(x_1 x_2 \dots x_w \mid \theta_0)},$$

avec $P(x_1 x_2 \dots x_w \mid \theta_0)$ la probabilité que la succession de nucléotides $x_1 x_2 \dots x_w$ fasse partie de la « background-sequence ». Proposer chacun des a_i^* indépendemment revient à proposer un nouveau vecteur A^*, qui est accepté avec la probabilité suivante :

$$\rho = \frac{\tilde{\pi}_l(A^*) P(A \mid \hat{\Theta}^*)}{\tilde{\pi}_l(A) P(A^* \mid \hat{\Theta})},$$

où $P(A^* \mid \hat{\Theta}) = \prod_{i=1}^{L^*} q_i^{a_i^*} (1-q_i)^{1-a_i^*}$ représente la probabilité de générer A^* à partir de l'état actuel A.

Algorithme PTEEM

L'algorithme PTEEM utilise N chaînes. La $l^{\text{ème}}$ chaîne a comme distribution cible :

$$\pi_l(A \mid S) = \pi(A \mid S)^{\frac{1}{T_l}}$$

Contrairement à l'EES, chacune des chaînes est actualisée localement à l'aide d'un échantillonneur de Gibbs. Pour la première chaîne, l'actualisation de chacun des éléments se fait en utilisant la formule d'actualisation (6.24). Pour la chaîne l, la formule d'actualisation utilisée est la suivante :

$$\frac{p(a_i = 1 | A_{[-i]}, S)}{p(a_i = 0 | A_{[-i]}, S)} = \left(\frac{1}{\pi(S(a_i) \mid A, \theta_0)} \times \frac{|A_{[-i]}| + a}{L^* - A_{[-i]} - 1 + b} \times \prod_{k=1}^{w} \left(\frac{C_{k(-i)} + \beta_k}{|A_{[-i]}| + |\beta_k|} \right)^{C_{k(i)}} \right)^{\frac{1}{T_l}} \quad (6.25)$$

Un mouvement d'échange Equi-Energy entre deux chaînes l_1 et l_2 d'états actuels A_{l_1} et A_{l_2} est accepté avec la probabilité suivante :

$$\rho = 1 \wedge \frac{\pi_{l_1}(A_{l_2} \mid S) \pi_{l_2}(A_{l_1} \mid S)}{\pi_{l_1}(A_{l_1} \mid S) \pi_{l_2}(A_{l_2} \mid S)} = 1 \wedge \left(\frac{\pi(A_{l_2} \mid S)}{\pi(A_{l_1} \mid S)} \right)^{\frac{1}{T_{l_1}} - \frac{1}{T_{l_2}}},$$

avec

$$\left(\frac{\pi(A_{l_2} \mid S)}{\pi(A_{l_1} \mid S)} \right)^{\frac{1}{T_{l_1}} - \frac{1}{T_{l_2}}} = \left[\frac{\pi(S(A_{l_1}) \mid A_{l_1}, \theta_0)}{\pi(S(A_{l_2}) \mid A_{l_2}, \theta_0)} \times \frac{\mathcal{B}(|A_{l_2}| + a, L^* - |A_{l_2}| + b)}{\mathcal{B}(|A_{l_1}| + a, L^* - |A_{l_1}| + b)} \times \prod_{k=1}^{w} \frac{\Gamma(C_{l_2 k} + \beta_k)}{\Gamma(C_{l_1 k} + \beta_k)} \times \prod_{k=1}^{w} \frac{\Gamma(|A_{l_1}| + |\beta_k|)}{\Gamma(|A_{l_2}| + |\beta_k|)} \right]^{\frac{1}{T_{l_1}} - \frac{1}{T_{l_2}}}. \quad (6.26)$$

Résultats

Dix runs de l'EES et du PTEEM sont lançés sur les données présentées dans la partie 6.2.4. Pour chaque run de PTEEM, $N = 15$ chaînes sont utilisées, avec un burn-in de 200 itérations, et 800 itérations post-burn-in. Ce qui donne au final 15000 mouvements locaux et 1000 mouvements globaux proposés. Pour chaque run de l'EES, $K = 9$ chaînes sont utilisées, avec $p_{ee} = 0.1$, un burn-in de 200 itérations, et 800 itérations post-burn-in dont 100 itérations pour le remplissage initial des anneaux. Ce qui donne au final 18160 mouvements locaux et 1640 mouvements globaux proposés en moyenne, ce qui se rapproche des nombres de mouvements utilisés par un run de PTEEM. Les températures et niveaux d'énergie utilisés sont données en annexe D.2.

Mouvements effectués :

En ce qui concerne le PTEEM, les mouvements Equi-Energy proposés ont été acceptés dans 55.71% des cas, ce qui a permis un bon mélange de l'ensemble des 15 chaînes. Sur les 10 runs, la première chaîne arrivait assez bien à échanger avec les chaînes proches, et parvenait à échanger jusqu'aux chaînes 10, voire 11.

En ce qui concerne l'EES, la première chaîne était actualisée localement par un échantillonneur de Gibbs, mais les 8 autres l'étaient par un algorithme de Metropolis-Hastings. Or les nouveaux états proposés pour les différentes chaînes étaient relativement peu acceptés : 15% en moyenne des états proposées étaient acceptés pour les chaînes 2 à 5, mais seulement 2.9% l'étaient pour la chaîne 8 et 0.7% l'étaient pour la chaîne 9. Les sauts Equi-Energy proposés étaient en grande majorité acceptés (86% en moyenne), mais il faut noter qu'assez peu étaient proposés entre la première chaîne et la seconde (26 sauts proposés en moyenne sur les 1000 itérations). En effet, les états des chaînes 1 et 2 avaient souvent des energies éloignées. Par exemple, la chaîne 2 n'a jamais réussi a avoir un état dans l'anneau d'énergie de niveau le plus faible.

Identification des sites de fixation :

Les résultats des 10 runs de PTEEM étaient très similaires entre eux, ceux des 10 runs de l'EES l'étaient un peu moins. La figure 6.2.4 représente deux boxplots : un boxplot représentant les probabilités a posteriori $P(a_i = 1 \mid S), i \in \{1, \ldots, L^*\}$ obtenues lors d'un run du PTEEM, et le second donnant les mêmes probabilités lors d'un run de l'EES. Ces boxplots représentent donc les probabilités a posteriori pour chaque position possible d'être le point de départ d'un site de fixation. Seules les positions ayant des probabilités a posteriori associées élevées sont conservées. Sur les boxplots de la figure 6.2.4, nous conservons les positions ayant des probabilités a posteriori associées supeŕieures à 0.8. En effet, ces positions ont alors des probabilités a posteriori élevées d'être le point de départ d'un site de fixation. Le tableau 6.2.4 donne les nombres de sites identifiés par l'EES et le PTEEM, en moyenne sur 10 runs.

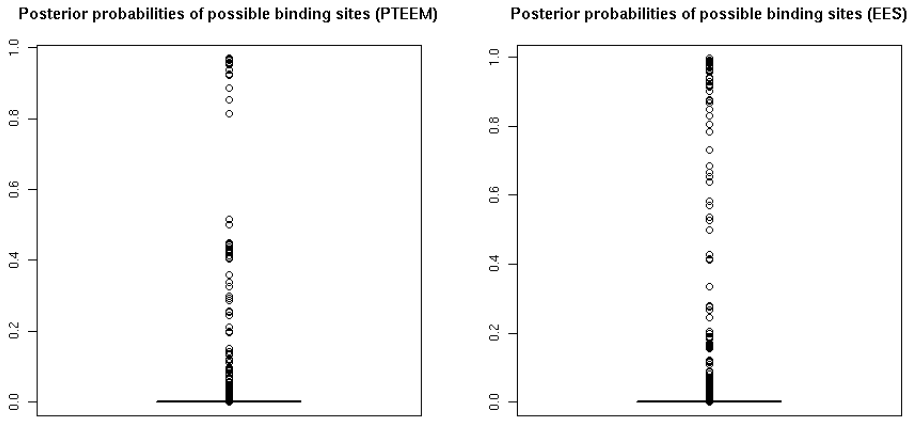

FIGURE 6.9 – Boxplots représentant les probabilités a posteriori $P(a_i = 1 \mid S), i = 1, \ldots, L^*$ obtenues lors d'un run du PTEEM, et lors d'un run de l'EES.

	EES	PTEEM
Sites identifiés	15.6	16
Sites identifiés avec bonne position	9.6	15
Sites identifiés avec décalage ou plusieurs positions	6	1

TABLE 6.12 – Nombres de sites identifiés par l'EES et le PTEEM, en moyenne sur 10 runs.

Concernant les 10 runs du PTEEM, tous ont permis d'identifiés 16 sites, dont 15 ont été identifiés avec les bonnes positions de départ, et 1 a été identifié avec 3 positions (les positions 877, 880 et 883 ont été conservées, la vraie position étant 880). Le fait que 15 positions de départ aient été correctement identifiées nous montre la force de l'algorithme PTEEM qui a pu s'échapper de la plupart des modes locaux. En effet, les positions décalées par rapport aux vraies positions de départ consituent des modes locaux.
Concernant les 10 runs de l'EES, ils ont permis d'identifier en moyenne 15.6 sites. Parmi ceux-ci, 9.6 sites ont été identifiés avec les bonnes positions de départ, et 6 sites ont été identifiés avec des positions décalées ou bien avec plusieurs positions. Par exemple, un site identifié par les positions 1784 et 1791 alors que la vraie position de départ est 1784, ou un site identifié par la position 27 alors que la vraie position de départ est 26. Notons que 5 runs de l'EES ont fait aussi bien que les 10 runs du PTEEM.

Discussion :
Les résultats des runs de l'EES pourraient être améliorés en utilisant plus de chaînes (au prix

d'un temps de calcul augmenté). Cependant, nous avons remarqué le nombre relativement faible de sauts Equi-Energy proposés entre les chaînes 1 et 2. Cela pourrait être dû à des températures très différentes, mais étant donné que $T_1 = 1$ et $T_2 = 1.001$, nous pensons plutôt que cela est dû à l'algorithme de Metropolis-Hastings utilisé pour actualiser la chaîne 2, qui ne permettrait pas de trouver assez souvent des états pertinents. En effet, il n'est pas évident de déterminer une bonne façon de proposer un état A^* à partir d'un état actuel A, et la méthode proposée par Kou et al. [118] pourrait être discutée. Les difficultés que nous avons rencontré pour bien calibrer l'EES et l'algorithme de Metropolis-Hastings associé sont un inconvénient à l'utilisation de cet échantillonneur.
En comparaison, la calibration du PTEEM est plus facile. En effet, l'échantillonneur de Gibbs n'a pas besoin d'être calibré, et comme toutes les chaînes sont actualisées par cet échantillonneur, les échanges entre chaînes sont assez nombreux tant que les énergies et les températures sont bien choisies. En ce qui concerne ces énergies et températures, nous n'avons pas rencontré de difficultés à les calibrer, les suggestions de la partie 6.1.3 ont donné de bons résultats.
Un mouvement local de l'échantillonneur de Gibbs est plus rapide qu'un mouvement local de Metropolis-Hastings, ce qui favorise l'algorithme PTEEM d'un point de vue temps de calcul.
En ce qui concerne les résultats de cet exemple, ceux obtenus avec PTEEM sont légèrement meilleurs que ceux obtenus avec l'EES. Le mélange de la chaîne d'intérêt était plus efficace dans un run de PTEEM que dans un run de l'EES. Par conséquent, PTEEM est parvenu à identifier exactement les positions de départ de la plupart des sites de fixation, alors que l'EES a plus eu tendance à identifier ces sites avec un décalage, ce qui signifie qu'il restait plus souvent bloqué dans des modes locaux.
Notons que ce problème de décalage (« phase-shift problem ») est rencontré par la plupart des méthodes servant à identifier des sites de fixation, et que des solutions ont été proposées, voir notamment Liu [141] ou Lawrence et al. [126]. L'implémentation de ces solutions dans les algorithmes PTEEM et EES est tout à fait possible. De même, des améliorations pourraient être apportées à ces algorithmes pour permettre d'avoir des sites de fixations de longueur inconnue, plusieurs types de sites de fixation, ou bien encore des sites constitués de plusieurs blocs non contigus (voir Jensen et al. [109] et Liu et al. [145]). Notons que ces améliorations semblent plus faciles à apporter dans le cadre d'un échantillonneur de Gibbs, et donc dans un PTEEM, que dans un Metropolis-Hastings.

6.3 Discussion et perspectives

Grâce à un mouvement d'échange Equi-Energy pertinent, l'algorithme PTEEM proposé permet de mieux explorer l'espace des paramètres que l'algorithme PT. Il est nécessaire de spécifier des niveaux d'énergie, mais avec un peu d'expérience il est souvent assez simple et rapide d'obtenir des niveaux pertinents (voir la partie 6.1.3). En comparaison avec l'EES, le PTEEM

donne d'aussi bons résultats, voire légèrement meilleurs, tout en ayant de bonnes propriétés. En particulier, sous des hypothèses simples la convergence du processus Markovien généré vers la distribution d'intérêt π^* est assurée. D'autre part, la variance asymptotique de l'EES est théoriquement au moins aussi élevée que celle du PTEEM (voir Atchadé [8]). Un point fort du PTEEM est qu'il nécessite très peu de stockage, contrairement à l'EES. De plus, il peut être facilement combiné avec un échantillonneur de Gibbs, ce qui n'est pas le cas de l'EES (voir la partie 5.4.3). Cela peut être très avantageux pour certains problèmes où les échantillonneurs de Gibbs sont faciles d'utilisation, comme dans l'exemple sur la recherche des sites de fixation (partie 6.2.4). Enfin, le PTEEM est assez facile à appréhender même pour des non-statisticiens.

L'algorithme tel que nous l'avons décrit n'utilise que le mouvement d'échange Equi-Energy comme mouvement global. Cependant, rien n'empêche de rajouter d'autres types de mouvements, comme les mouvements crossover de l'Evolutionary Monte Carlo [139] par exemple (voir la partie 5.3.2). Nous aurions également pu faire des mouvements à rejet retardé comme dans le Parallel Tempering (voir la partie 5.3.1). Cependant cela ne nous paraît pas forcément très pertinent, car si les niveaux d'énergie sont bien choisis et assez nombreux, la probabilité d'acceptation de l'échange Equi-Energy est assez élevée.

D'un point de vue théorique, les hypothèses des propositions 6.1.1 et 6.1.3 pourraient être allégées. De plus, il serait intéressant d'obtenir des vitesses de convergence.

Un inconvénient du PTEEM est que les simulations générées par les chaînes autres que la première ne sont pas directement utilisées. Elles permettent à la première chaîne qui est celle d'intérêt de correctement explorer l'espace des paramètres, mais elles ne sont pas utilisées pour l'inférence. Une perspective est de proposer une méthode permettant de tirer parti de ces simulations, grâce à des arguments d'échantillonnage préférentiel par exemple.

Enfin, une perspective est d'adapter de manière automatique les températures et les niveaux d'énergie au fur et à mesure des itérations de l'échantillonneur. Cependant, l'algorithme obtenu serait un échantillonneur MCMC adaptatif, donc certaines propriétés seraient perdues, et il paraît difficile a priori de gérer les « sauts d'énergie » (voir par exemple le tableau 6.1.3).

Chapitre 7

Méthodes sans vraisemblance

Nous nous plaçons dans un cadre bayésien, et notons $x \in \mathbb{D} \subseteq \mathbb{R}^n$ un vecteur de données observées. Ces données sont supposées être issues d'un modèle de paramètre $\theta \in \Theta \subseteq \mathbb{R}^d$, et de fonction de vraisemblance $f(x \mid \theta)$. C'est la distribution a posteriori de θ sachant les données x qui est d'intérêt :

$$\pi(\theta \mid x) = \frac{f(x \mid \theta)\pi(\theta)}{\int f(x \mid \theta)\pi(\theta)d\theta},$$

où $\pi(\theta)$ est l'a priori sur les paramètres, et $\int f(x \mid \theta)\pi(\theta)d\theta$ est la constante de normalisation. L'obtention de cet a posteriori requière donc le calcul de la vraisemblance, au moins à une constante près. Cependant dans le cas de modélisations complexes la vraisemblance n'est parfois pas disponible sous forme explicite. Les méthodes de Monte Carlo classiques ne sont alors pas utilisables. De même, si la vraisemblance est disponible sous forme explicite mais que son calcul est très long, les méthodes classiques sont difficilement envisageables. A l'inverse, il est souvent beaucoup plus facile de simuler un jeu de données artificiel z suivant le modèle, étant donné un vecteur de paramètres θ.

Des méthodes sans vraisemblance (ou méthodes ABC, pour « Approximated Bayesian Computation ») ont alors été proposées, utilisables lorsque les vraisemblances ne sont pas disponibles, mais qu'il est facile de simuler suivant le modèle. Au départ, l'idée à la base des méthodes ABC a été évoquée par Rubin [191], puis utilisée la première fois par Tavaré et al. [216] en génétique des populations. La méthode qu'ils ont développée correspond en fait à un simple algorithme acceptation-rejet. Etant donné que leur méthode ne peut s'appliquer que sous certaines conditions restrictives, Pritchard et al. [177] ont proposé une méthode dont la cible n'est plus la distribution a posteriori d'intérêt, mais une approximation de cette dernière. Cette méthode constitue en quelque sorte la genèse des méthodes ABC, et beaucoup d'améliorations ont été proposées depuis. En effet, Marjoram et al. [153] ont ensuite proposé un alogrithme ABC-MCMC, basé sur la théorie des chaînes de Markov. Bortot et al. [27] ont développé un algorithme ABC-MCMC avec augmentation, en faisant une analogie avec la méthode du simulated tempering. En paral-

lèle des méthodes séquentielles ont été proposées, et notamment l'algorithme Population Monte Carlo-ABC (PMC-ABC) proposé par Beaumont et al. [18].

Dans ce chapitre nous passons rapidement en revue les principales méthodes ABC qui ont été développées, ainsi que la manière de les calibrer et de traiter les résultats obtenus. Des revues récentes et plus complètes ont été effectuées par Grelaud [86], Marin et al. [149] et Beaumont [17].

7.1 Les premiers algorithmes ABC

7.1.1 ABC exact

En 1984, Rubin [191] est le premier à remarquer qu'un algorithme acceptation-rejet peut être utilisé pour simuler exactement suivant une distribution a posteriori, sans avoir à calculer la vraisemblance. Cet algorithme est repris en 1997 par Tavaré et al. [216], et est le suivant :

ABC exact

`Pour obtenir` n `simulations suivant l'a posteriori.`
`Pour` i `allant de 1 à` n`:`

1. `Générer` θ' `suivant sa distribution a priori` $\pi(.)$.
2. `Générer` z' `sachant` θ'`, suivant le modèle défini par` $f(. \mid \theta')$.

`Si` $z' = x$`, poser` $\theta_i = \theta'$`. Sinon retourner en 1.`

Les paramètres θ_i acceptés sont ceux à partir desquels des données z' simulées suivant le modèle sont égales aux données observées x.

Il est évident que pour pouvoir utiliser cet algorithme x doit prendre ses valeurs dans un espace fini ou dénombrable. En effet, si ce n'est pas le cas alors $\mathbb{P}(z' = x) = 0$, et aucun paramètre θ' ne peut être accepté.

Cet algorithme fournit un échantillon i.i.d. suivant la distribution a posteriori d'intérêt. En effet,

$$\begin{aligned} \mathbb{P}(\theta_i) &= \sum_{z' \in \mathbb{D}} \pi(\theta_i) f(z \mid \theta_i) \mathbb{1}_{\{x\}}(z') \\ &= \pi(\theta_i) f(x \mid \theta_i) \\ &\propto \pi(\theta_i \mid x). \end{aligned}$$

Notons qu'un grand nombre de simulations peut être nécessaire pour accepter un paramètre θ_i.

7.1.2 Algorithme de Pritchard et al.

L'ABC exact ne peut être utilisé que dans le cadre d'espaces des états finis ou dénombrables. Une extension possible dans le cadre d'espaces des états continus est la suivante :

ABC-1

Pour obtenir n simulations.
Pour i allant de 1 à n:

1. Générer θ' suivant sa distribution a priori $\pi(.)$.
2. Générer z' sachant θ', suivant le modèle défini par $f(. \mid \theta')$.

Si $\rho(z', x) < \epsilon$, poser $\theta_i = \theta'$. Sinon retourner en 1.

Il y a introduction d'une distance ρ, et d'un seuil de tolérance ϵ. Les paramètres θ_i acceptés sont ceux à partir desquels des données z' simulées suivant le modèle sont assez « proches » des données observées x.
Mais il peut être assez difficile et long de calculer une distance entre les données simulées et observées, surtout si les données sont de grandes dimensions. Afin de pallier ce problème, l'algorithme ABC-1 peut encore être étendu de la façon suivante :

ABC-2 ou ABC standard

Pour obtenir n simulations.
Pour i allant de 1 à n:

1. Générer θ' suivant sa distribution a priori $\pi(.)$.
2. Générer z' sachant θ', suivant le modèle défini par $f(. \mid \theta')$.

Si $\rho(S(z'), S(x)) < \epsilon$, poser $\theta_i = \theta'$. Sinon retourner en 1.

Une statistique S pouvant être de dimension supérieure à 1 et permettant de résumer les données est introduite sur l'espace des paramètres $\mathbb{D}$. Cette extension correspond à l'algorithme proposée par Pritchard et al. [177] dans le cadre d'applications génétiques, alors qu'ils se trouvaient dans un espace des états continu, et qu'ils avaient à disposition des statistiques résumant les données bien connues et utilisées des généticiens.

Cet algorithme fournit un échantillon i.i.d. suivant la distribution jointe :

$$\pi_\epsilon(z, \theta \mid x) \propto \pi(\theta) f(z \mid \theta) \mathbb{1}_{\{\rho(S(z), S(x)) < \epsilon\}}(z). \tag{7.1}$$

Nous sommes en général intéressés par les simulations de θ, soit par la marginale :

$$\pi_\epsilon(\theta \mid x) \propto \int \pi_\epsilon(z, \theta \mid x) dz, \qquad \text{souvent notée} \qquad \pi(\theta \mid \rho(S(z), S(x)) < \epsilon). \tag{7.2}$$

Si la statistique S résume bien les données et si ϵ est choisi assez petit, alors $\pi_\epsilon(\theta \mid x)$ est une approximation raisonnable de la distribution a posteriori d'intérêt $\pi(\theta \mid x)$. En particulier, si S est exhaustive et $\epsilon = 0$, nous avons $\pi_\epsilon(\theta \mid x) = \pi(\theta \mid x)$.
Lorsque S résume bien les données, si ϵ tend vers 0, alors les paramètres simulés θ_i peuvent être considérés comme issus de $\pi(\theta \mid x)$. Mais un grand nombre de simulations seront nécessaires en moyenne pour accepter un paramètre θ_i. A l'inverse, si ϵ tend vers ∞, alors les paramètres simulés θ_i peuvent être considérés comme issus de l'a priori $\pi(\theta)$, et quasiment toutes les simulations effectuées seront acceptées.

Dans le cas d'une statistique S exhaustive, utiliser $S(z')$ et $S(x)$ revient au même que d'utiliser directement les données simulées et observées z' et x, et l'algorithme ABC-1 est équivalent à l'algorithme ABC-2. Dans le cas où S n'est pas exhaustive, son utilisation fait nécessairement perdre de l'information par rapport au cas où nous utiliserions l'ensemble des données, et cela rajoute à l'approximation due à ρ et ϵ. Cependant il n'est en général pas possible de trouver de telles statistiques, surtout si la vraisemblance n'est pas connue sous forme explicite (exception faite du papier de Grelaud et al. [87] dans le cadre de champs de Gibbs). Il est donc important de choisir des statistiques résumant au mieux les données compte tenu de l'objectif fixé.

7.2 ABC et MCMC

7.2.1 ABC-MCMC

Le taux d'acceptation d'un paramètre θ' généré lors de l'étape 1 de l'algorithme ABC-2 peut être très petit si l'a priori sur θ est diffus par rapport à l'a posteriori. En effet, dans ce cas la grande majorité des paramètres proposés va engendrer des données simulées éloignées des données observées. L'idée de Marjoram et al. [153] est d'utiliser le paramètre accepté à l'itération précédente pour proposer un nouveau paramètre qui aura plus de chance d'être accepté, dans l'esprit d'un algorithme de Metropolis-Hastings [94]. Ils proposent alors l'algorithme suivant :

ABC-MCMC

Pour obtenir une chaîne de Markov de longueur $(n+1)$:

1. Utiliser l'algorithme ABC-2 pour obtenir une réalisation $(\theta^{(0)}, z^{(0)})$ suivant la distribution $\pi_\epsilon(z, \theta \mid x)$.
2. Pour t allant de 1 à n:

(a) `Générer` θ' `suivant` $q(. \mid \theta^{(t-1)})$.

(b) `Générer` z' `sachant` θ', `suivant le modèle défini par` $f(. \mid \theta')$.

`Poser` $(\theta^{(t)}, z^{(t)}) = (\theta', z')$ `avec la probabilité suivante:`

$$\alpha = min\left\{1, \frac{\pi(\theta')q\left(\theta^{(t-1)} \mid \theta'\right)}{\pi(\theta^{(t-1)})q\left(\theta' \mid \theta^{(t-1)}\right)} \mathbb{1}_{\{\rho(S(z'),S(x))<\epsilon\}}(z')\right\}.$$

`Sinon poser` $(\theta^{(t)}, z^{(t)}) = (\theta^{(t-1)}, z^{(t-1)})$.

La chaîne de Markov obtenue pour (z, θ) a pour distribution stationnaire $\pi_\epsilon(z, \theta \mid x)$, et la chaîne obtenue pour θ a pour distribution stationnaire $\pi(\theta \mid \rho(S(z), S(x)) < \epsilon)$. En effet, il suffit de remarquer que la probabilité α correspond tout simplement à la probabilité d'acceptation d'un algorithme de Metropolis-Hastings de noyau de transition $q\left(\theta' \mid \theta^{(t-1)}\right) f\left(z' \mid \theta'\right)$ et de distribution stationnaire $\pi_\epsilon(z, \theta \mid x)$:

$$\begin{aligned}
&\frac{\pi_\epsilon(z', \theta' \mid x)}{\pi_\epsilon(z^{(t-1)}, \theta^{(t-1)} \mid x)} \frac{q\left(\theta^{(t-1)} \mid \theta'\right) f\left(z^{(t-1)} \mid \theta^{(t-1)}\right)}{q\left(\theta' \mid \theta^{(t-1)}\right) f\left(z' \mid \theta'\right)} \\
&\quad = \frac{\pi(\theta') f(z' \mid \theta') \mathbb{1}_{\{\rho(S(z'),S(x))<\epsilon\}}(z')}{\pi(\theta^{(t-1)}) f(z^{(t-1)} \mid \theta)^{(t-1)} \mathbb{1}_{\{\rho(S(z^{(t-1)}),S(x))<\epsilon\}}(z^{(t-1)})} \frac{q\left(\theta^{(t-1)} \mid \theta'\right) f\left(z^{(t-1)} \mid \theta^{(t-1)}\right)}{q\left(\theta' \mid \theta^{(t-1)}\right) f\left(z' \mid \theta'\right)} \\
&\quad = \frac{\pi(\theta')q\left(\theta^{(t-1)} \mid \theta'\right)}{\pi(\theta^{(t-1)})q\left(\theta' \mid \theta^{(t-1)}\right)} \mathbb{1}_{\{\rho(S(z'),S(x))<\epsilon\}}(z')
\end{aligned}$$

Marjoram et al. [153] font remarquer que le « prix à payer » pour avoir des taux d'acceptation plus élevés qu'un algorithme ABC classique est l'obtention de paramètres corréles entre eux. Ils notent également les deux cas particuliers suivants :

1. Si $q\left(\theta^{(t-1)} \mid \theta'\right) = q\left(\theta' \mid \theta^{(t-1)}\right)$, alors la probabilité α n'utilise que les probabilités a priori de θ' et $\theta^{(t-1)}$.
2. Si q est réversible par rapport à la distribution a priori π, c'est à dire $\pi(\theta')q\left(\theta^{(t-1)} \mid \theta'\right) = \pi(\theta^{(t-1)})q\left(\theta' \mid \theta^{(t-1)}\right)$, alors la probabilité α vaut 1 et l'algorithme ABC-MCMC revient à un algorithme d'acceptation-rejet avec des paramètres obtenus corrélés entre eux, ce qui n'aurait aucun intérêt.

Dans l'algorithme ABC-MCMC, la distribution obtenue peut être très éloignée de la distribution a posteriori d'intérêt si ϵ n'est pas assez petit, voir Tanaka et al. [215] par exemple. A l'inverse, si ϵ est trop petit, l'algorithme ABC-MCMC peut avoir des difficultés à se déplacer

dans l'espace des paramètres. En effet, même si en théorie l'algorithme converge, la probabilité d'acceptation α est proportionnelle à la probabilité de simuler des données z' telles que $\rho(S(z'), S(x)) < \epsilon$ (elle même proportionnelle à la vraisemblance, voir Sisson [199]). Or cette probabilité est très faible si ϵ est très petit. En pratique, lorsque l'échantillonneur ABC-MCMC se situe dans une queue de distribution, les paramètres proposés à partir du paramètre où il se trouve aboutissent souvent à des données simulées éloignées des données observées. Alors le taux de rejet est élevé et l'échantillonneur peut rester bloqué dans la queue un long moment. Par conséquent nous obtenons des probabilités a posteriori trop élevées pour certaines zones des queues de distribution, voir par exemple Sisson [199]. Ce phénomène s'accentue lorsque le paramètre initial est choisi dans une queue de la distribution a posteriori. L'utilisation de la théorie des MCMC lorsque la vraisemblance n'est pas exploitable n'est donc pas forcément avantageuse en ce qui concerne la capacité de la chaîne générée à se déplacer hors des queues de distribution.

Remarque 7.2.1 *C'est l'inverse qui se passe lorsque la vraisemblance est connue, car alors la probabilité d'acceptation est proportionnelle au ratio des vraisemblances des états proposés et actuels. Ce ratio est élevé si la chaîne se trouve dans une queue de distribution et si on propose de s'en éloigner, et il est facile de s'échapper des queues de la distribution a posteriori.*

7.2.2 ABC-MCMC augmenté, analogie avec le simulated tempering

Bortot et al. [27] ont proposé une amélioration de l'algorithme ABC-MCMC en faisant une analogie avec le Simulated Tempering (voir partie 5.2.3). Ils n'utilisent pas de températures, mais font varier le seuil de tolérance ϵ. Pour des seuils élevés les paramètres proposés sont plus facilement acceptés, ce qui permet à la chaîne de se déplacer et éventuellement de partir d'une zone de faible probabilité a posteriori. Les paramètres obtenus pour des seuils faibles permettent quant à eux d'avoir une approximation précise de la distribution a posteriori cible. L'espace augmenté est alors $\Theta \times \mathbb{R}^+$, car le seuil de tolérance ϵ est considéré comme un paramètre. Le paramètre θ n'est plus simulé seul, mais conjointement avec le seuil ϵ, c'est le vecteur augmenté (θ, ϵ) qui est utilisé. Une distribution a priori $\pi(\epsilon)$ est supposée sur ϵ.
L'algorithme est le suivant :

ABC-MCMC augmenté

Pour obtenir une chaîne de Markov de longueur $(n+1)$:

1. Simuler $\epsilon^{(0)} \sim \pi(\epsilon)$. Utiliser l'algorithme ABC-2 pour obtenir une réalisation $(\theta^{(0)}, z^{(0)})$.
2. Pour t allant de 1 à n:
   (a) Générer (θ', ϵ') suivant $q\big((\theta', \epsilon') \mid (\theta^{(t-1)}, \epsilon^{(t-1)})\big)$.
   (b) Générer z' sachant θ', suivant le modèle défini par $f(. \mid \theta')$.

`Poser` $(\theta^{(t)}, \epsilon^{(t)}, z^{(t)}) = (\theta', \epsilon', z')$ `avec la probabilité suivante:`

$$\alpha = min\left\{1, \frac{\pi(\theta')\pi(\epsilon')q\Big((\theta^{(t-1)}, \epsilon^{(t-1)}) \mid (\theta', \epsilon')\Big)}{\pi(\theta^{(t-1)})\pi(\epsilon^{(t-1)})q\Big((\theta', \epsilon') \mid (\theta^{(t-1)}, \epsilon^{(t-1)})\Big)}\right\}\mathbb{1}_{\{\rho(S(z'),S(x))<\epsilon'\}}(z').$$

`Sinon poser` $(\theta^{(t)}, \epsilon^{(t)}, z^{(t)}) = (\theta^{(t-1)}, \epsilon^{(t-1)}, z^{(t-1)})$.

Cet algorithme génère une chaîne de Markov pour (z, θ, ϵ) de distribution stationnaire :

$$\pi(z, \theta, \epsilon \mid x) \propto \pi(\theta)\pi(\epsilon)f(z \mid \theta)\mathbb{1}_{\{\rho(S(z),S(x))<\epsilon\}}(z).$$

La distribution de (θ, ϵ) est :

$$\pi(\theta, \epsilon \mid x) \propto \int \pi(z, \theta, \epsilon \mid x)dz, \qquad \text{notée} \qquad \pi(\theta, \epsilon \mid \rho(S(z), S(x)) < \epsilon).$$

Afin de favoriser de petites valeurs pour le seuil de tolérance ϵ, Bortot et al. [27] préconisent de prendre comme a priori une loi exponentielle de paramètre λ assez grand. Les paramètres θ obtenus pour des valeurs du seuil de tolérance élevées ne sont pas pertinents pour approximer la distribution a posteriori cible $\pi(\theta \mid x)$. Bortot et al. suggèrent donc de filtrer la chaîne obtenue pour (θ, ϵ), en ne gardant que les couples avec un seuil ϵ inférieur à une valeur seuil ϵ_T. La distribution stationnaire de la chaîne filtrée est alors :

$$\int_0^{\epsilon_T} \pi(\theta, \epsilon \mid \rho(S(z), S(x)) < \epsilon)d\epsilon,$$

qui est une bonne approximation de la distribution a posteriori d'intérêt $\pi(\theta \mid x)$ si ϵ_T est assez petit et S bien choisie.

7.2.3 Algorithme de Ratmann et al. (2007)

Ratmann et al. [181] ont proposé deux améliorations intéressantes de l'algorithme ABC-MCMC :

1. Ils proposent que durant la période de burn-in la statistique calculée sur les données observée ne soit pas simplement comparée à la statistique calculée sur un jeu de données simulé, mais à la moyenne de statistiques obtenues sur plusieurs jeux simulés à partir du même paramètre proposé. Plus le nombre de jeux simulés est important et plus la variance de la moyenne calculée est faible. Par conséquent plus le nombre de jeux simulés est important et plus les paramètres acceptés lors de cette période de burn-in sont proches du mode a posteriori.

2. Ils proposent l'utilisation d'une séquence de seuils de tolérance et d'une séquence de distributions instrumentales, afin que l'échantillonneur ne reste pas bloqué dans les queues de la distribution a posteriori. Le seuil de tolérance est diminué graduellement à chaque itération, jusqu'à parvenir à un seuil minimal pré-établi (cela rappelle le « simulated annealing », voir partie 5.2.2). De même pour les distributions instrumentales servant à proposer un paramètre θ' à partir du paramètre actuel $\theta^{(t)}$: une séquence de distributions de plus en plus précises est pré-établie, et à chaque itération la distribution instrumentale utilisée est modifiée. Par exemple, les distributions instrumentales sont des distributions normales tronquées, et les variances de ces distributions diminuent à chaque itération jusqu'à atteindre une variance minimale. Par conséquent la chaîne peut facilement se déplacer au début, tandis que les paramètres conservés à la fin sont proches du mode a posteriori.

7.3 ABC et méthodes séquentielles

7.3.1 ABC-PRC

Les méthodes sans vraisemblance de type acceptation-rejet ne sont pas optimales en temps de calcul, car de nombreuses simulations peuvent être nécessaires pour en accepter une. Les méthodes existantes de type MCMC ne sont pas non plus totalement satisfaisantes, car les simulations obtenues peuvent être très corrélées et les échantillonneurs peuvent rester bloqués dans les queues de distribution.
Sisson et al. [199] se sont alors intéressés à des méthodes séquentielles (filtrage particulaire), notamment à l'algorithme Population Monte Carlo (PMC) de Cappé et al. [32] et aux algorithmes de Monte Carlo séquentiels (SMC) de Del Moral et al. [54]. Le principe est le suivant : une population de paramètres $\theta_1, \ldots, \theta_n$ (aussi appelés particules) est obtenue à partir d'une distribution initiale spécifiée par l'utilisateur, puis ces particules sont propagées et pondérées à travers une séquence de distributions intermédiaires. L'objectif est d'obtenir un échantillon pondéré dont la distribution empirique converge asymptotiquement vers la distribution cible. Dans le cadre des méthodes ABC, la distribution cible est bien évidemment l'a posteriori $\pi(\theta \mid \rho(S(z), S(x)) < \epsilon)$, et les distributions intermédiaires sont naturellement les $\pi_{\epsilon_t} = \pi(\theta \mid \rho(S(z), S(x)) < \epsilon_t)$, avec $\epsilon_1 > \epsilon_2 > \ldots > \epsilon_T = \epsilon$. Les échantillonneurs SMC étant présentés par Del Moral et al. [54] comme généralisant l'algorithme Population Monte Carlo (PMC), Sisson et al. [199] ont proposé un algorithme combinant des arguments SMC avec les méthodes ABC, appelé algorithme ABC-Partial Rejection Control. Cette méthode utilise une distribution initiale π_1 suivant laquelle il est facile d'échantillonner, des noyaux de transition markoviens « forward » K_t, et des noyaux de transitions markoviens « backward » L_t.
L'algorithme est donné en supplément D.3.

Pour définir les poids des particules (voir formule (D.4)), Sisson et al. [199] se sont basés sur

le travail de Del Moral et al. [54], qui préconisent d'utiliser les poids suivants à l'itération t (θ_i^* échantillonné depuis la population $\{\theta_j^{(t-1)}\}_{j=1,\dots,n}$ munie des poids $\{w_j^{(t-1)}\}_{j=1,\dots,n}$) :

$$w_i^{(t)} = \frac{\pi(\theta_i^{(t)} \mid x) L_{t-1}(\theta_i^* \mid \theta_i^{(t)})}{\pi(\theta_i^* \mid x) K_t(\theta_i^{(t)} \mid \theta_i^*)} = \frac{f(x \mid \theta_i^{(t)}) \pi(\theta_i^{(t)}) L_{t-1}(\theta_i^* \mid \theta_i^{(t)})}{f(x \mid \theta_i^*) \pi(\theta_i^*) K_t(\theta_i^{(t)} \mid \theta_i^*)}, \quad i = 1, \dots, n.$$

Ces poids ne pouvant pas être directement utilisés dans le cadre des méthodes ABC à cause de la présence de vraisemblances, Sisson et al. [199] ont en fait remplacé le ratio de ces vraisemblances par 1 lorsque $\rho(S(z'), S(x)) < \epsilon_t$. Mais il a alors été noté par Beaumont et al. [18], Sisson et al. [200], et Toni et al. [223] que cela implique un biais dans l'approximation de la distribution a posteriori $\pi(\theta \mid \rho(S(z), S(x)) < \epsilon)$ (calculs détaillés dans Beaumont et al. [18]). Une condition pour qu'il n'y ait pas de biais est que les noyaux de transitions $L_{t-1}(\theta_i^* \mid \theta_i^{(t)})$ ne dépendent pas des $\theta_i^{(t)}$, or Sisson et al. [199] ont proposé de prendre des noyaux gaussiens centrés autour des $\theta_i^{(t)}$.

7.3.2 ABC-PMC

Beaumont et al. [18] ayant noté que l'algorithme de Sisson et al. [199] donne des résultats biaisés, ils ont proposé un nouvel algorithme se basant sur des arguments d'échantillonnage préférentiel (« importance sampling »), dans l'esprit de l'algorithme Population Monte Carlo (voir Cappé et al. [32] et Douc et al. [60]). Ils remarquent qu'un poids naturellement associé avec une particule $\theta_i^{(t)}$ acceptée est :

$$w_i^{(t)} \propto \pi(\theta_i^{(t)}) / \hat{\pi}_t(\theta_i^{(t)}), \qquad \text{avec} \qquad \hat{\pi}_t(\theta_i^{(t)}) \propto \sum_{j=1}^{n} w_j^{(t-1)} K_t(\theta_i^{(t)} \mid \theta_j^{(t-1)}).$$

En effet, $\theta_j^{(t-1)}$ est échantillonné depuis la population précédente $\{\theta_i^{(t-1)}\}_{i=1,\dots,n}$ munie des poids $\{w_i^{(t-1)}\}_{i=1,\dots,n}$, puis $\theta_i^{(t)}$ est obtenu à partir de cette particule échantillonnée grâce au noyau K_t. Contrairement à l'ABC-PRC, l'utilisation de ces poids n'engendre pas de biais (voir Beaumont et al. [18]).

Beaumont et al. [18] suggèrent de plus que les noyaux de transition $K_t(\theta_j^{(t)} \mid \theta_j^{(t-1)})$ soient modifiés, actualisés à chaque itération. En ce sens, l'algorithme ABC-PMC qu'ils proposent est adaptatif. Ils suggèrent l'utilisation de marches aléatoires commes noyaux de transition K_t, soit :

$$\theta_i^{(t)} \mid \theta_i^* \sim \mathcal{N}(\theta_i^*, \tau_i^2).$$

Concernant le choix de τ_i^2, un noyau pertinent minimise à chaque itération la dissimilarité entre la distribution $\hat{\pi}_t$ et la distribution a posteriori exacte. En effet, $\hat{\pi}_t(.)$ approxime la distribution intermédiaire $\pi_{\epsilon_t}(. \mid x)$ qui elle-même approxime $\pi(. \mid x)$. Comme Douc et al. [60], Beaumont et al. [18] choissisent de prendre la divergence de Kullback-Leibler comme critère de dissimilarité.

La variance τ_i^2 optimale vaut alors $2\text{var}(\theta_i^{(t)} \mid x)$.
L'algorithme qu'ils proposent est donné en supplément D.3. Il peut être vu comme une approximation de l'algorithme SMC de Del Moral et al. [54] avec le noyau de transition « backward » optimal.

Notons que parallèlement à Beaumont et al. [18], Sisson et al. [200] ont publié un errata dans lequel ils reconnaissent que l'algorithme ABC-PRC donne des résultats biaisés, et dans lequel ils proposent un nouvel algorithme quasiment identique à l'algorithme ABC-PMC. Toujours à la même période, Toni et al. [223] ont également proposé un algorithme combinant les méthodes ABC avec des arguments d'échantillonage préférentiel. Leur algorithme est très similaire à celui de Beaumont et al., si ce n'est qu'ils ne proposent pas d'adapter les noyaux de transitions K_t au fur et à mesure des itérations. Ils ne peuvent donc jouer que sur la séquence des niveaux de tolérance $\epsilon_1 > \epsilon_2 > \ldots > \epsilon_T = \epsilon$. Récemment, Filippi et al. [64] ont proposé trois classes pour les noyaux de transition utilisés dans ces algorithmes, et ont étudié leurs efficacités (notamment en utilisant la distance de Kullback-Leibler, dans le même esprit que Beaumont et al. [18]).

7.3.3 ABC-SMC

Del Moral et al. [55] ont également proposé un algorithme ABC séquentiel adaptatif. Tout comme Sisson et al. [199], ils se sont inspirés des échantillonneurs SMC de Del Moral et al. [54] et ont combiné des arguments SMC avec les méthodes ABC. Cependant, contrairement à eux, ils parviennent à un algorithme donnant des résultats non biaisés. Ils ont fait le constat que les algorithmes développés par Beaumont et al. [18], Toni et al. [223] et Sisson et al. [200] ont des complexités quadratiques par rapport au nombre de particules utilisées, et ne permettent pas un ajustement automatique de la séquence des niveaux de tolérance. Leurs objectifs pour le développement de leur algorithme qu'ils ont appelé ABC-SMC ont alors été les suivants :

1. Complexité linéaire par rapport au nombre de particules utilisées.
2. Ajustement automatique de la séquence des niveaux de tolérance.
3. Ajustement automatique des noyaux de transition (déjà proposé dans Beaumont et al. [18]).

Génération de multiples jeux de donnés

Ils proposent également de générer, pour chaque particule sélectionnée θ, non pas un jeu de données z qui est comparé au jeu observé x, mais M jeux de données notés $z_{1:M} = (z_1, z_2, \ldots, z_M)$, avec $M > 1$ (idée similaire à celle de Ratmann et al. [181]). Par conséquent, au lieu de générer un échantillon i.i.d. suivant la distribution jointe

$$\pi_\epsilon(z, \theta \mid x) \propto \pi(\theta) f(z \mid \theta) \mathbb{1}_{\{\rho(S(z), S(x)) < \epsilon\}}(z),$$

ils ont développé un algorithme générant un échantillon i.i.d. suivant la distribution :

$$\pi_\epsilon(z_{1:M}, \theta \mid x) \propto \pi(\theta) \prod_{k=1}^{M} f(z_k \mid \theta) \Big(\frac{1}{M} \sum_{k=1}^{M} \mathbb{1}_{\{\rho(S(z_k),S(x))<\epsilon\}}(z_k) \Big). \tag{7.3}$$

La distribution a posteriori cible est la marginale (identique pour tout M) :

$$\pi_\epsilon(\theta \mid x) \propto \int \pi_\epsilon(z_{1:M}, \theta \mid x) dz_{1:M}.$$

La séquence de distributions intermédiaires qu'ils utilisent est naturellement

$$\pi_{\epsilon_1}(z_{1:M}, \theta \mid x), \quad \pi_{\epsilon_2}(z_{1:M}, \theta \mid x), \quad \ldots, \quad \pi_{\epsilon_T}(z_{1:M}, \theta \mid x),$$

avec la séquence des niveaux de tolérance $\epsilon_1 > \epsilon_2 > \ldots > \epsilon_T = \epsilon$.

Del Moral et al. [55] montrent l'intérêt de générer plusieurs jeux de données au lieu d'un seul. En effet, leur algorithme contient des étapes d'un algorithme ABC-MCMC modifié, et la variance de la probabilité d'acceptation d'une telle étape diminue quand M augmente (voir étape 3.c de l'algorithme et la formule (D.5) en supplément).

Adapter l'algorithme SMC à un cadre ABC

La performance des algorithmes SMC dépend de la séquence des distributions intermédiaires (soit des niveaux de tolérance dans le cadre ABC), ainsi que des noyaux de transitions markoviens « forward » K_t et « backward » L_t utilisés à chaque itération ($1 \leq t \leq T$). Suivant les recommandations de Del Moral et al. [54], Del Moral et al. [55] décident de prendre pour K_t un noyau de transition markovien de distribution stationnaire $\pi_{\epsilon_t}(z_{1:M}, \theta \mid x)$. Une fois que K_t est défini, ils posent :

$$L_{t-1}\Big((z'_{1:M}, \theta') \mid (z_{1:M}, \theta)\Big) = \frac{\pi_{\epsilon_t}(z'_{1:M}, \theta' \mid x) K_t\Big((z_{1:M}, \theta) \mid (z'_{1:M}, \theta')\Big)}{\pi_{\epsilon_t}(z_{1:M}, \theta \mid x)}.$$

Alors la formule d'actualisation des poids utilisée par Del Moral et al. [54] dans leur algorithme SMC devient, pour la particule $(z^{(t)}_{1:M,i}, \theta^{(t)}_i)$:

$$w_i^{(t)} \quad \propto \quad w_i^{(t-1)} \frac{\pi_{\epsilon_t}\Big(z^{(t-1)}_{1:M,i}, \theta^{(t-1)}_i \mid x\Big)}{\pi_{\epsilon_{t-1}}\Big(z^{(t-1)}_{1:M,i}, \theta^{(t-1)}_i \mid x\Big)}.$$

Et en utilisant (7.3), ces poids deviennent :

$$w_i^{(t)} \propto w_i^{(t-1)} \frac{\sum_{k=1}^{M} \mathbb{1}_{\{\rho\left(S(z_{k,i}^{(t-1)}),S(x)\right)<\epsilon_t\}}(z_{k,i}^{(t-1)})}{\sum_{k=1}^{M} \mathbb{1}_{\{\rho\left(S(z_{k,i}^{(t-1)}),S(x)\right)<\epsilon_{t-1}\}}(z_{k,i}^{(t-1)})}. \tag{7.4}$$

Ajustement automatique de la séquence des niveaux de tolérance

Pour ajuster automatiquement la séquence des niveaux de tolérance, Del Moral et al. [55] utilisent la proportion de particules « vivantes », c'est à dire ayant un poids non nul. Cette proportion mesure en quelque sorte la qualité de l'approximation SMC (tout comme l'ESS lorsque $M = 1$), et est définie par :

$$PA\Big(\{w_i^{(t)}\}_{i=1,\dots,n}, \epsilon_t\Big) = \frac{\sum_{i=1}^{n} \mathbb{1}_{w_i^{(t)}>0}(w_i^{(t)})}{n},$$

A l'itération t le seuil de tolérance ϵ_t est alors choisi de manière à ce que la proportion de particules vivantes soit égale à un certain pourcentage α de ce qu'elle est à l'itération $t-1$:

$$PA\Big(\{w_i^{(t)}\}_{i=1,\dots,n}, \epsilon_t\Big) = \alpha PA\Big(\{w_i^{(t-1)}\}_{i=1,\dots,n}, \epsilon_{t-1}\Big), \qquad 0 < \alpha < 1. \tag{7.5}$$

Si $\alpha \approx 1$, l'échantillonneur va se déplacer lentement vers la distribution cible $\pi_{\epsilon_T}(z_{1:M}, \theta \mid x)$ et l'approximation obtenue sera très bonne. A l'inverse, si $\alpha \approx 0$, l'échantillonneur va se déplacer très rapidement vers la distribution cible, et l'approximation ne sera pas fiable.

REMARQUE 7.3.1 *Le critère (7.5) n'est valide que dans le cas d'une distance ρ à valeurs continues. Dans le cas d'une distance ρ à valeurs discrètes, Del Moral et al. [55] proposent une alternative.*

Ajustement automatique des noyaux de transition

Del Moral et al. [55] prennent pour les K_t des noyaux de transition markoviens de distributions stationnaires $\pi_{\epsilon_t}(z_{1:M}, \theta \mid x)$, voir l'étape 3.c de l'algorithme.

Algorithme

L'algorithme est donné en supplément D.3.

Un algorithme similaire à cet algorithme ABC-SMC a également été proposé par Drovandi et al. [61]. Cet algorithme n'apporte pas grand chose par rapport à l'ABC-SMC, si ce n'est que les étapes MCMC (correspondant à 3.c dans l'algorithme ABC-SMC) sont répétées plus d'une fois, afin d'avoir plus de chances que les particules bougent. Le temps de calcul est alors augmenté.

7.4 Calibration des algorithmes ABC

Choix des statistiques

Il n'est en général pas possible de trouver des statistiques exhaustives (hormis dans Grelaud et al. [87]). Or, le choix des statistiques représentant les données est crucial, voir par exemple McKinley et al. [156] et Hamilton et al. [92]).

Il y a en général plusieurs statistiques disponibles pour un problème donné, fournies par les spécialistes du domaine en question. Il faut choisir celle(s) résumant au mieux les donnéees compte tenu de l'objectif. Si nous décidons d'utiliser plusieurs statistiques, d par exemple, $S(x)$ s'écrit $(S_1(x), \ldots, S_d(x))$ et lorsqu'un jeu de données z' est simulé, nous devons comparer chacune des statistiques $S_j(x)$ avec $S_j(z')$. Cependant, augmenter le nombre de statistiques utilisées n'est pas forcément avantageux, car plus d augmente, et plus il est difficile d'accepter des simulations (z', θ') (d'où les méthodes proposées par Beaumont et al. [19] et Blum et François [26], voir partie 7.5).

En 2005, Hamilton et al. [92] ont proposé de pondérer les différentes statistiques disponibles, en fonction de leurs associations avec les paramètres d'intérêt. Ensuite, en 2008, Joyce et Marjoram [152] ont proposé une méthode pour sélectionner itérativement les statistiques à utiliser, à l'aide de tests de rapports de vraisemblances, jusqu'à ce que l'inclusion d'une nouvelle statistique n'apporte plus rien. Cependant, cette méthode dépend du sens d'inclusion des statistiques, de paramètres choisis par l'utilisateur, et ne prend pas en compte les corrélations éventuelles entre statistiques, qui sont fréquentes dans le cas de très nombreuses statistiques possibles. Dans le même esprit, Ratmann a présenté une méthode d'inclusion de statistiques dans sa thèse [180]. En 2010, Nunes et Balding [169] ont également développé une méthode pour sélectionner des statistiques pertinentes, en se basant sur un critère d'entropie.

Afin de prendre en compte les corrélations des statistiques tout en réduisant la dimension de l'espace des statistiques à utiliser, Leuenberger et al. [133] et Bazin et al. [16] proposent d'utiliser une analyse en composantes principales sur les statistiques disponibles, et de ne conserver que les premiers composants. Si un nombre suffisant de composants est conservé, la perte d'information peut être minime. Dans le même esprit, Wegmann et al. [229] proposent une approche de type régression « partial least-squares » : les variables à expliquer sont les paramètres du modèle et les statistiques disponibles sont les variables explicatives. Cette approche nécessite une étape de calibration pour déterminer les composants à utiliser.

Récemment, Fearnhead et Prangle [63] proposent une construction semie-automatique de statistiques appropriées. Au lieu de considérer un algorithme ABC comme performant lorsque l'ensemble de la distribution a posteriori d'intérêt $\pi(\theta \mid x)$ est bien approchée, ils le considèrent comme performant seulement quand certains paramètres de cette distribution sont bien estimés. Leur algorithme ABC n'est donc utilisé que dans une optique inférentielle, pour estimer certains paramètres d'intérêt. Leur approche consiste à calibrer correctement l'algorithme de façon à ce

qu'il donne de bons estimateurs pour ces paramètres d'intérêt. Ils montrent de plus que dans le cas de fonction de perte quadratique, les statistiques optimales sont les moyennes a posteriori de ces paramètres d'intérêt.

Choix de la distance

Le choix d'une distance est lui aussi important, voir encore McKinley et al. [156]. Que nous utilisions une distance entre deux jeux de données (soit $\rho(z, x)$), ou une distance entre les statistiques de deux jeux de données (soit $\rho(S(z), S(x))$), la pertinence d'une distance dépendra du type de données utilisées. Par exemple, pour des données temporelles, des fonctions d'autocovariance peuvent être utilisées pour définir des distances pertinentes.
Des zones d'acceptation rectangulaires peuvent être utilisées (voir par exemple Pritchard et al. [177]), du type : (z', θ') est acceptée si :

$$|S_1(z') - S_1(x)| < \epsilon_1 \quad \text{et} \quad |S_2(z') - S_2(x)| < \epsilon_2 \quad \text{et} \quad \dots \quad \text{et} \quad |S_d(z') - S_d(x)| < \epsilon_d.$$

Mais c'est la distance euclidienne qui est la plus utilisée (voir par exemple Cornuet et al. [48] ou Beaumont et al [19]), ce qui donne des zones d'acceptation sphériques : (z', θ') est acceptée si :

$$\sqrt{\sum_{j=1}^{d} \big(S_j(z') - S_j(x)\big)^2} < \epsilon.$$

Les statistiques utilisées sont en général normalisées par leurs écart-types sous le modèle supposé, car elles ne sont pas toutes du même ordre de grandeur. Une estimation de ces écart-types nécessite en général des simulations préliminaires au lancement de l'algorithme.

Choix du seuil de tolérance

Le choix du seuil de tolérance ϵ détermine la qualité de l'approximation de la distribution a posteriori d'intérêt, puisque $\pi(\theta \mid \rho(S(z), S(x)) < \epsilon)$ est une approximation de $\pi(\theta \mid x)$. A la limite, si ϵ tend vers 0, les simulations obtenues pourront être considérées comme issues de $\pi(\theta \mid x)$, mais alors le temps de calcul d'un algorithme ABC sera très long. A l'inverse, si ϵ tend vers ∞, les simulations obtenues pourront être considérées comme issues de l'a priori $\pi(\theta)$ et l'algorithme sera très rapide puisque toutes les simulations seront acceptées.
En pratique, le seuil ϵ est souvent déterminé comme un quantile des distances simulées, cette approche est expliqué par Wegmann et al. [229]. Des simulations préliminaires sont lancées : n paramètres θ sont générés suivant $\pi(.)$ et pour chacun un jeu de données associé z est généré suivant $f(. \mid \theta)$. Pour chacun de ces n jeux de données, les statistiques $S(z)$ sont calculées, ainsi que leurs distances ρ par rapport à $S(x)$. Soit p un petit pourcentage, le seuil ϵ peut être défini comme le quantile d'ordre p des distances simulées. Si $p = 1\%$ par exemple, cela nous assure

qu'environ 1% des simulations générées par ABC-1 ou ABC-2 seront acceptées.

REMARQUE 7.4.1 *Dans l'ABC-SMC de Del Moral et al. [55] (voir la partie 7.3.3), une séquence de seuils de tolérance est construite de manière adaptative, mais cela ne dispense pas de spécifier un seuil minimal à atteindre.*

7.5 Post-processing des résultats obtenus par une méthode ABC

7.5.1 Méthode de Beaumont et al. (2002)

Nous nous plaçons dans le cadre de l'algorithme ABC-2 (ABC standard). En 2002, Beaumont et al. [19] ont proposé deux améliorations de cet algorithme. La méthode qu'ils proposent ne modifie pas le processus de simulation (étapes 1 et 2), mais utilise les simulations obtenues lors de ces étapes pour améliorer l'approximation de la distribution a posteriori cible. Dans l'ABC-2 standard, si une simulation (θ', z') est telle que $\rho(S(z'), S(x)) < \epsilon$, alors elle est retenue, sinon elle est rejetée. La méthode de Beaumont et al. est plus subtile : si $\rho(S(z'), S(x))$ est trop élevé, la simulation n'est pas retenue. Mais si $\rho(S(z'), S(x))$ est en dessous d'un seuil noté δ (qui peut être plus grand que ϵ utilisé dans l'algorithme standard), alors la simulation est retenue et un poids lui est affecté, poids qui dépend de $\rho(S(z'), S(x))$. En comparaison, l'analyse classique des résultats de l'ABC-2 revient à n'avoir que des poids de 0 ou 1. De plus, la méthode qu'ils proposent ajuste les simulations retenues grâce à une régression linéaire locale.

Beaumont et al. [19] considèrent le problème d'approximation de la distribution a posteriori cible $\pi(\theta \mid x)$ comme un problème d'estimation de la densité conditionnelle associée. Si n itérations des étapes 1 et 2 de l'algorithme ABC-2 sont effectuées, n simulations (θ'_i, z'_i) sont obtenues. Beaumont et al. supposent que pour des θ'_i associés à des z'_i assez proches de x observé (soit $\rho(S(z'_i), S(x))$ assez petit), le modèle de régression suivant est vérifié :

$$\theta'_i = \alpha + (S(z'_i) - S(x))^T \beta + \nu_i, \qquad i = 1, \ldots, n. \tag{7.6}$$

avec les ν_i supposés non corrélés, de moyennes nulles, et de variances identiques. Supposons que $S(x)$ est de dimension d, soit $S(x) = (S_1(x), \ldots, S_d(x))$ avec les $S_j(x)$ qui sont des scalaires. En notation matricielle, le modèle de régression linéaire est :

$$\begin{pmatrix} \theta'_1 \\ \vdots \\ \theta'_n \end{pmatrix} = \begin{pmatrix} 1 & S_1(z'_1) - S_1(x) & \ldots & S_d(z'_1) - S_d(x) \\ \vdots & & & \\ 1 & S_1(z'_n) - S_1(x) & \ldots & S_d(z'_n) - S_d(x) \end{pmatrix} \begin{pmatrix} \alpha \\ \beta_1 \\ \vdots \\ \beta_d \end{pmatrix} + \begin{pmatrix} \nu_1 \\ \vdots \\ \nu_n \end{pmatrix}, \tag{7.7}$$

ce que nous notons :

$$\theta' = X(\alpha, \beta) + \nu. \tag{7.8}$$

Les estimateurs des moindres carrés $(\hat{\alpha}, \hat{\beta})$ peuvent s'obtenir par $(X'X)^{-1}X'\theta'$. Cependant, Beaumont et al. [19] soulignent que ce modèle de régression linéaire ne peut être supposé valide que si les θ'_i sont associés à des z'_i assez proches de x observé (soit $\rho(S(z'_i), S(x))$ assez petit). C'est donc une régression locale qu'ils veulent effectuer, en donnant plus de poids à des θ'_i associés à des z'_i proches de x qu'à des θ'_i associés à des z'_i plus éloignés de x. Ils préconisent donc l'utilisation d'une régression linéaire pondérée, avec l'utilisation de poids $K_\delta\big(\rho(S(z'_i), S(x))\big)$, K_δ étant un noyau non-paramétrique (souvent le noyau d'Epanechnikov). L'idée est que plus $\rho(S(z'_i), S(x))$ est petit et plus le θ'_i associé a un poids important. A l'opposé, dés que $\rho(S(z'_i), S(x))$ dépasse un seuil δ, alors θ'_i n'intervient plus dans la régression locale, son poids est nul. Les estimateurs $(\hat{\alpha}, \hat{\beta})$ utilisés sont alors ceux des moindres carrés généralisés de (α, β).
Afin d'estimer la densité a posteriori d'intérêt, Beaumont et al. [19] n'utilisent pas l'échantillon des simulations θ'_i, ils se basent plutôt sur l'échantillon ajusté des θ^*_i, avec :

$$\theta^*_i = \theta'_i - (S(z'_i) - S(x))^T \hat{\beta} \qquad i = 1, \ldots, n. \tag{7.9}$$

Soit δ le paramètre utilisé dans la méthode de Beaumont et al. [19] et ϵ le seuil de tolérance utilisé dans l'algorithme ABC-2 standard. Beaumont et al. montrent que l'utilisation de cette méthode en « post-processing » permet d'améliorer les résultats obtenus par l'algorithme ABC-2 (voir également Marin et al. [149]). En effet, si $\delta = \epsilon$, l'approximation de la distribution a posteriori cible obtenue par la méthode de Beaumont et al. est meilleure que celle obtenue par l'algorithme ABC-2. La différence se fait d'autant plus sentir que les seuils ϵ et δ sont élevés. Avec un $\delta > \epsilon$, des approximations similaires peuvent être obtenues par l'algorithme ABC-2 standard et par la méthode de Beaumont et al., qui nécessite donc moins de simulations.

7.5.2 Autres méthodes

En 2010, plusieurs méthodes de « post-processing » ont été proposées. Blum [25] a proposé une modification de l'approche de Beaumont et al. [19], en utilisant une régression quadratique plutôt qu'une régression linéaire.

En parallèle, Blum et François [26] ont généralisé la méthode de Beaumont et al. [19]. Plus précisemment, ils proposent de modéliser la relation entre les statistiques et les paramètres par un modèle non-linéaire et hétéroscédastique, contrairement à Beaumont et al. [19] qui utilisent un modèle de régression linéaire locale homoscédastique. Dans l'approche de Beaumont et al., une partie des simulations rejetées par l'ABC-2 standard sont conservées, puis pondérées et ajustées. Cependant, toutes les simulations produites ne peuvent pas être conservées car le modèle de régression linéaire n'est valide que localement autour des statistiques observées. L'approche

de Blum et François utilisant un modèle non-linéaire, un plus grand nombre des simulations produites peut être conservé (parfois même toutes), car le modèle est valable dans une bonne partie ou l'ensemble de l'espace des statistiques. Pour estimer la moyenne et la variance de leur modèle non linéaire, Blum et François utilisent des réseaux de neurones. Ceux-ci permettent une réduction de la dimension de l'ensemble des statistiques utilisées, en utilisant des projections sur des sous-ensembles de plus petites dimensions. Cela est particulièrement avantageux lorsque de nombreuses statistiques doivent être utilisées.

Leuenberger et al. [133] ont remarqué que le fait de modéliser les paramètres θ en fonction des statistiques $S(z)$ comme le font Beaumont et al. [19] a un défaut principal : l'a priori de θ n'est pas utilisé, et l'on peut obtenir des approximations de la loi a posteriori mal spécifiées (des zones ayant une probabilité a posteriori approximée non nulle alors que la probabilité a priori l'est). Pour pallier à ce problème, ils préconisent plutôt de modéliser les statistiques $S(z)$ sachant les paramètres θ en fonction des paramètres θ. Le modèle linéaire qu'ils utilisent permet de pouvoir prendre en compte la corrélation entre les différentes statistiques. Enfin, contrairement à Beaumont et al., leur méthode peut être utilisée en combinaison avec un algorithme ABC-MCMC qui génère des observations corrélées.

Chapitre 8

Parallel Tempering sans vraisemblance

L'algorithme ABC-MCMC a longtemps été l'algorithme sans vraisemblance de référence. Cependant, les méthodes sans vraisemblance séquentielles (présentées en 7.3) sont en général plus performantes que les méthodes sans vraisemblance combinées avec des méthodes MCMC (présentées en 7.2), voir par exemple Beaumont et al. [18] et McKinley et al. [156] pour des comparaisons.

En faisant une analogie avec l'algorithme Parallel Tempering (présenté en 5.3.1), nous proposons une nouvelle méthode sans vraisamblance basée sur la théorie des MCMC, utilisant une population de chaînes, ainsi que des séquences de températures et de seuils de tolérance. Dans ce chapitre, nous développons cet algorithme que nous appellons ABC-Parallel Tempering, et le comparons avec les méthodes sans vraisemblance existantes, et notamment avec l'ABC-PMC qui a tendance à devenir le nouvel algorithme de référence.

8.1 ABC et Parallel Tempering

8.1.1 Analogie avec le Parallel Tempering

Les mêmes notations que dans le chapitre précédent sont utilisées : $x \in \mathbb{D} \subseteq \mathbb{R}^n$ est un vecteur de données observées, supposées être issues d'un modèle de paramètre $\theta \in \Theta \subseteq \mathbb{R}^d$, et de fonction de vraisemblance $f(x \mid \theta)$. La distribution a posteriori d'intérêt est $\pi(\theta \mid x)$, et $\pi(\theta)$ est l'a priori sur les paramètres.

Par analogie avec le Parallel Tempering, nous souhaitons utiliser une population de N chaînes en parallèle. Certaines doivent se déplacer facilement dans l'espace des paramètres, tandis que d'autres doivent être plus précises et permettre une bonne approximation de la distribution a posteriori cible $\pi(\theta \mid x)$. Des échanges entre ces deux types de chaînes doivent permettre aux dernières de s'échapper de zones de faibles probabilités a posteriori.
Tout comme Bortot et al. [27] nous proposons de faire varier le seuil de tolérance ϵ. Nous proposons également de faire varier les noyaux de transition utilisés dans les mouvements MCMC,

en les tempérant (de tels noyaux tempérés ont été utilisés par Ratmann et al. [181]).

Une séquence de températures $T_1 = 1 < T_2 < \ldots < T_N$ est pré-définie, ainsi qu'une séquence de seuils de tolérance $\epsilon_{min} = \epsilon_1 < \epsilon_2 < \ldots < \epsilon_N$, avec ϵ_{min} qui est le seuil de tolérance cible, souhaité pour l'approximation finale de $\pi(\theta \mid x)$.
Chacune des chaînes est actualisée localement par un algorithme ABC-MCMC, avec un seuil de tolérance et un noyau de transition qui lui sont propres. En effet, la $i^{\text{ème}}$ chaîne est actualisée par une itération d'un ABC-MCMC avec le seuil de tolérance ϵ_i et le noyau de transition $q_i(. \mid \theta)$ dépendant de T_i. Par exemple, $q_i(. \mid \theta)$ peut être un noyau de transition gaussien tempéré par T_i. Plus l'indice de la chaîne augmente, et plus le noyau de transition doit permettre de proposer des paramètres éloignés des paramètres actuels.

8.1.2 Le mouvement d'échange

La séquence des seuils de tolérance associée à l'ensemble des N chaînes est notée $\underline{\epsilon} = (\epsilon_1, \epsilon_2, \ldots, \epsilon_N)$, l'ensemble des paramètres associés aux N chaînes est noté $\underline{\theta} = (\theta_1, \theta_2, \ldots, \theta_N)$ avec $\theta_i \in \Theta \subseteq \mathbb{R}^d$, et l'ensemble des jeux de données simulés associés aux N chaînes est noté $\underline{z} = (z_1, z_2, \ldots, z_N)$. L'état associé à la chaîne d'indice i ($i \in \{1, \ldots, N\}$) est noté (z_i, θ_i), et cette chaîne est associée à la distribution jointe suivante :

$$\pi_{\epsilon_i}(z_i, \theta_i \mid x) \propto \pi(\theta_i) f(z_i \mid \theta_i) \mathbb{1}_{\{\rho(S(z_i), S(x)) < \epsilon_i\}}(z_i). \tag{8.1}$$

Nous définissons

$$\pi^*_{\underline{\epsilon}}(\underline{z}, \underline{\theta} \mid x) = \prod_{i=1}^{N} \pi_{\epsilon_i}(z_i, \theta_i \mid x). \tag{8.2}$$

Pour effectuer un échange entre deux chaînes, une paire de chaînes est choisie uniformément. Les indices des deux chaînes choisies sont notés i et j ($i < j$), et les états proposés lors de l'échange sont notés $\underline{\theta}'$ et $\underline{z}'$. Nous notons $q(\underline{z}', \underline{\theta}' \mid \underline{z}, \underline{\theta})$ la probabilité de proposer $(\underline{z}', \underline{\theta}')$ à partir de $(\underline{z}, \underline{\theta})$, et $q(\underline{\theta}' \mid \underline{\theta})$ la probabilité de proposer $\underline{\theta}'$ à partir de $\underline{\theta}$.

Mouvement intuitif ne satisfaisant pas aux exigences de l'ABC

Un mouvement d'échange intuitif consiste à simplement échanger l'état (θ_i, z_i) de la chaîne i avec l'état (θ_j, z_j) de la chaîne j. Cela revient à proposer, pour l'ensemble des N chaînes :

$$\underline{\theta}' = (\theta_1, \ldots, \theta_j, \ldots, \theta_i, \ldots, \theta_N) \qquad \text{et} \qquad \underline{z}' = (z_1, \ldots, z_j, \ldots, z_i, \ldots, z_N).$$

Afin d'assurer la réversibilité du mouvement par rapport à $\pi^*_{\underline{\epsilon}}$, cet échange doit être accepté avec la probabilité suivante :

$$\alpha \quad = \quad 1 \wedge \frac{\pi^*_{\underline{\epsilon}}(\underline{z}', \underline{\theta}' \mid x) q(\underline{z}, \underline{\theta} \mid \underline{z}', \underline{\theta}')}{\pi^*_{\underline{\epsilon}}(\underline{z}, \underline{\theta} \mid x) q(\underline{z}', \underline{\theta}' \mid \underline{z}, \underline{\theta})} \quad = \quad 1 \wedge \frac{\pi^*_{\underline{\epsilon}}(\underline{z}', \underline{\theta}' \mid x)}{\pi^*_{\underline{\epsilon}}(\underline{z}, \underline{\theta} \mid x)}. \tag{8.3}$$

En effet, pour ce mouvement $q(\underline{z}', \underline{\theta}' \mid \underline{z}, \underline{\theta})$ représente simplement la probabilité de choisir la paire (i, j) parmi l'ensemble des paires possibles. Comme la paire de chaînes est choisie uniformément, $q(\underline{z}, \underline{\theta} \mid \underline{z}', \underline{\theta}') = q(\underline{z}', \underline{\theta}' \mid \underline{z}, \underline{\theta})$. Nous avons :

$$\begin{aligned} \frac{\pi^*_{\underline{\epsilon}}(\underline{z}', \underline{\theta}' \mid x)}{\pi^*_{\underline{\epsilon}}(\underline{z}, \underline{\theta} \mid x)} &= \frac{\pi_{\epsilon_i}(z'_i, \theta'_i \mid x) \pi_{\epsilon_j}(z'_j, \theta'_j \mid x)}{\pi_{\epsilon_i}(z_i, \theta_i \mid x) \pi_{\epsilon_j}(z_j, \theta_j \mid x)} \\ &= \frac{\pi(\theta'_i) f(z'_i \mid \theta'_i) \mathbb{1}_{\{\rho(S(z'_i), S(x)) < \epsilon_i\}}(z'_i)}{\pi(\theta_i) f(z_i \mid \theta_i) \mathbb{1}_{\{\rho(S(z_i), S(x)) < \epsilon_i\}}(z_i)} \times \frac{\pi(\theta'_j) f(z'_j \mid \theta'_j) \mathbb{1}_{\{\rho(S(z'_j), S(x)) < \epsilon_j\}}(z'_j)}{\pi(\theta_j) f(z_j \mid \theta_j) \mathbb{1}_{\{\rho(S(z_j), S(x)) < \epsilon_j\}}(z_j)}. \end{aligned} \tag{8.4}$$

Le calcul de la probabilité d'acceptation (8.3) ne satisfait pas aux exigences d'une méthode sans vraisemblance, car il nécessite le calcul des ratios de vraisemblances $f(z'_i \mid \theta'_i)/f(z_i \mid \theta_i)$ et $f(z'_j \mid \theta'_j)/f(z_j \mid \theta_j)$.

Mouvement satisfaisant aux exigences de l'ABC

Nous proposons alors d'échanger le paramètre θ_i de la chaîne i avec le paramètre θ_j de la chaîne j, et de rajouter une étape d'actualisation des jeux de données simulés associés z_i et z_j. Plus précisément, les deux étapes suivantes sont effectuées :

1. Echanger le paramètre θ_i de la chaîne i avec le paramètre θ_j de la chaîne j. Cela revient à proposer :

$$\underline{\theta}' = (\theta_1, \ldots, \theta_j, \ldots, \theta_i, \ldots, \theta_N). \tag{8.5}$$

2. Actualiser z_i et z_j par z'_i et z'_j, à partir de θ_j pour la chaîne i, et de θ_i pour la chaîne j. Cela revient à proposer :

$$\underline{z}' = (z_1, \ldots, z'_i, \ldots, z'_j, \ldots, z_N). \tag{8.6}$$

En procédant ainsi, nous avons (avec $\theta'_i = \theta_j$ et $\theta'_j = \theta_i$) :

$$q(\underline{z}', \underline{\theta}' \mid \underline{z}, \underline{\theta}) \quad = \quad q(\underline{z}', \underline{\theta}' \mid \underline{\theta}) \quad \propto \quad f(z'_i \mid \theta'_i) f(z'_j \mid \theta'_j) q(\underline{\theta}' \mid \underline{\theta}).$$

Afin d'assurer la réversibilité de ce mouvement, la probabilité d'acceptation doit être la suivante :

$$\begin{aligned}\alpha &= 1 \wedge \frac{\pi^*_{\underline{\epsilon}}(\underline{z}', \underline{\theta}' \mid x) q(\underline{z}, \underline{\theta} \mid \underline{z}', \underline{\theta}')}{\pi^*_{\underline{\epsilon}}(\underline{z}, \underline{\theta} \mid x) q(\underline{z}', \underline{\theta}' \mid \underline{z}, \underline{\theta})} \\ &= 1 \wedge \frac{\pi^*_{\underline{\epsilon}}(\underline{z}', \underline{\theta}' \mid x) f(z_i \mid \theta_i) f(z_j \mid \theta_j)}{\pi^*_{\underline{\epsilon}}(\underline{z}, \underline{\theta} \mid x) f(z'_i \mid \theta'_i) f(z'_j \mid \theta'_j)}. \end{aligned} \tag{8.7}$$

En effet, si la paire de chaînes est choisie uniformément parmi l'ensemble des paires possibles, $q(\underline{\theta} \mid \underline{\theta}') = q(\underline{\theta}' \mid \underline{\theta})$. Alors, en utilisant (8.4) la formule de la probabilité d'acceptation (8.7) devient :

$$\begin{aligned}\alpha &= 1 \wedge \frac{\pi(\theta'_i)\pi(\theta'_j)\mathbb{1}_{\{\rho(S(z'_i),S(x))<\epsilon_i\}}(z'_i)\mathbb{1}_{\{\rho(S(z'_j),S(x))<\epsilon_j\}}(z'_j)}{\pi(\theta_i)\pi(\theta_j)\mathbb{1}_{\{\rho(S(z_i),S(x))<\epsilon_i\}}(z_i)\mathbb{1}_{\{\rho(S(z_j),S(x))<\epsilon_j\}}(z_j)} \\ &= \mathbb{1}_{\{\rho(S(z'_i),S(x))<\epsilon_i\}}(z'_i)\mathbb{1}_{\{\rho(S(z'_j),S(x))<\epsilon_j\}}(z'_j). \end{aligned} \tag{8.8}$$

Ce mouvement d'échange en deux étapes satisfait aux exigences de l'ABC, puisque le calcul de la vraisemblance n'intervient nulle part. Remarquons que cette probabilité dépend des deux seuils de tolérance ϵ_i et ϵ_j.

8.1.3 Algorithme

L'algorithme est le suivant :

ABC-PT

Pour obtenir une chaîne de Markov de longueur $(n+1)$.

Définir une séquence de seuils de tolérance $\epsilon_1 < \epsilon_2 < \ldots < \epsilon_N$, ainsi qu'une séquence de températures $T_1 = 1 < T_2 < \ldots < T_N$.

1. Utiliser l'algorithme ABC-2 pour obtenir les N réalisations suivantes:
   1) $(\theta_1^{(0)}, z_1^{(0)})$ suivant la distribution $\pi_{\epsilon_1}(z, \theta \mid x)$.
   2) $(\theta_2^{(0)}, z_2^{(0)})$ suivant la distribution $\pi_{\epsilon_2}(z, \theta \mid x)$.
   i) ...
   N) $(\theta_N^{(0)}, z_N^{(0)})$ suivant la distribution $\pi_{\epsilon_N}(z, \theta \mid x)$.

   Poser $\underline{\theta}^{(0)} = (\theta_1^{(0)}, \theta_2^{(0)}, \ldots, \theta_N^{(0)})$ et $\underline{z}^{(0)} = (z_1^{(0)}, z_2^{(0)}, \ldots, z_N^{(0)})$.

2. Pour t allant de 1 à n:
   (a) MOUVEMENTS LOCAUX ABC-MCMC: Pour i allant de 1 à N:
      i. Générer θ'_i suivant $q_i(. \mid \theta_i^{(t-1)})$.
      ii. Générer z'_i sachant θ'_i, suivant le modèle défini par $f(. \mid \theta'_i)$.

Poser $(\theta_i^{(t)}, z_i^{(t)}) = (\theta_i', z_i')$ avec la probabilité suivante:

$$\alpha = min\left\{1, \frac{\pi(\theta_i')q_i\left(\theta_i^{(t-1)} \mid \theta_i'\right)}{\pi(\theta_i^{(t-1)})q_i\left(\theta_i' \mid \theta_i^{(t-1)}\right)}\right\}\mathbb{1}_{\{\rho(S(z_i'),S(x))<\epsilon_i\}}(z_i').$$

Sinon poser $(\theta_i^{(t)}, z_i^{(t)}) = (\theta_i^{(t-1)}, z_i^{(t-1)})$.

(b) MOUVEMENTS D'ECHANGE: N mouvements d'échange sont proposés: N paires de chaînes sont choisies uniformément dans l'ensemble des paires possibles, avec remise. Pour chacune des paires choisies :
Nous notons i et j $(i < j)$ les indices des chaînes de la paire choisie, et $\underline{z}$ et $\underline{\theta}$ les états actuels de l'ensemble des chaînes. Nous proposons un nouvel état $(\underline{z}', \underline{\theta}')$ construit de la manière suivante:

i. Echanger θ_i de la chaîne i avec θ_j de la chaîne j. Cela revient à avoir $\underline{\theta}' = (\theta_1, \ldots, \theta_j, \ldots, \theta_i, \ldots, \theta_N)$.

ii. Actualiser z_i et z_j par z_i' et z_j', à partir de θ_j pour la chaîne i, et de θ_i pour la chaîne j. Cela revient à avoir $\underline{z}' = (z_1, \ldots, z_i', \ldots, z_j', \ldots, z_N)$.

iii. Ce nouvel état $(\underline{z}', \underline{\theta}')$ est accepté pour remplacer $(\underline{z}, \underline{\theta})$ avec la probabilité d'acceptation suivante:

$$\alpha = \mathbb{1}_{\{\rho(S(z_i'),S(x))<\epsilon_i\}}(z_i')\mathbb{1}_{\{\rho(S(z_j'),S(x))<\epsilon_j\}}(z_j').$$

8.1.4 Convergence

Cet algorithme génère une chaîne de Markov pour $(\underline{z}, \underline{\theta})$ de distribution stationnaire $\pi_{\underline{\epsilon}}^*(\underline{z}, \underline{\theta} \mid x)$, avec :

$$\begin{aligned}\pi_{\underline{\epsilon}}^*(\underline{z}, \underline{\theta} \mid x) &= \prod_{i=1}^{N} \pi_{\epsilon_i}(z_i, \theta_i \mid x) \\ &\propto \prod_{i=1}^{N} \pi(\theta_i) f(z_i \mid \theta_i) \mathbb{1}_{\{\rho(S(z_i),S(x))<\epsilon_i\}}(z_i). \end{aligned} \tag{8.9}$$

En effet, Marjoram et al. [153] ont montré que l'algorithme ABC-MCMC utilisé en local pour actualiser la $i^{\text{ème}}$ chaîne génère une chaîne de Markov pour (z_i, θ_i) de distribution stationnaire $\pi_{\epsilon_i}(z_i, \theta_i \mid x)$. De plus, les mouvements d'échange sont définis de manière à être réversibles par rapport à $\pi_{\underline{\epsilon}}^*$, nous en déduisons facilement que l'ensemble des N chaînes est une chaîne de

Markov de distribution stationnaire $\pi^*_{\underline{\epsilon}}(\underline{z}, \underline{\theta} \mid x)$. La preuve de cette convergence est similaire à la preuve de la propriété 6.1.1 de l'algorithme PTEEM.
La première chaîne associée à $\epsilon_1 = \epsilon_{min}$ est celle qui nous intéresse, car ses réalisations sont issues de $\pi_{\epsilon_1}(z_1, \theta_1 \mid x)$. Nous sommes en général intéressés par les simulations de θ_1, soit par la marginale :

$$\pi_{\epsilon_1}(\theta_1 \mid x) \propto \int \pi_{\epsilon_1}(z_1, \theta_1 \mid x) dz, \qquad \text{notée} \qquad \pi(\theta_1 \mid \rho(S(z), S(x_1)) < \epsilon_1). \tag{8.10}$$

Cette distribution $\pi_{\epsilon_1}(\theta_1 \mid x)$ approxime correctement l'a posteriori d'intérêt $\pi(\theta \mid x)$ si ϵ_1 est assez petit et S bien choisie pour nos données.

8.1.5 En pratique

Nous avons besoin de définir le nombre de chaînes à utiliser, la séquence des seuils de tolérance ainsi que la séquence des températures pour les noyaux de transition.

Choix de la séquence des seuils de tolérance Le seuil le plus petit associé à la première chaîne ϵ_{min} détermine la qualité de notre approximation, et la chaîne associée au seuil le plus élevé ϵ_N doit pouvoir facilement s'échapper des zones de faibles probabilités a posteriori. D'après notre expérience, une fois que ces seuils ϵ_{min} et ϵ_N sont choisis, deux manières de calibrer la séquence donnent de bons résultats :

1. Utiliser une séquence de constantes de proportionnalité. Si par exemple les constantes de proportionalité sont 1.1, 1.2 et 1.3, la séquence est définie par : $\epsilon_2 = 1.1\epsilon_{min}$, $\epsilon_3 = 1.2\epsilon_2$ et $\epsilon_4 = 1.3\epsilon_3$ (approche similaire à celle de Beaumont et al. [18]).
2. Espacer régulièrement les seuils sur une échelle logarithmique.

Dans le choix de cette séquence et du nombre de chaînes à utiliser, nous devons tenir compte du fait qu'un trop grand nombre de chaînes associées à de petits seuils de tolérance est contre-productif. En effet, deux chaînes associées à de petits seuils de tolérance n'accepteront que rarement un mouvement d'échange (voir (8.8)). Or l'algorithme ABC-PT est plus efficace lorsque plus de mouvements d'échange sont acceptés. Si toutes les chaînes étaient associées à de petits seuils, un cercle vicieux s'intallerait : moins les mouvements d'échange sont acceptés, plus les chaînes ont tendance à rester coincées dans les queues de distribution, et moins un mouvement d'échange peut être accepté. A l'inverse, il est intéressant d'avoir un certain nombre de chaînes associées à des seuils de tolérance élevés.

Choix de la séquence des températures Les mêmes recommandations que pour l'algorithme PTEEM s'appliquent, voir la partie 6.1.3. La chaîne d'indice N doit pouvoir facilement explorer l'espace des paramètres. Pour cela, le noyau de transition tempéré par la température

la plus élevée T_N doit proposer des paramètres suffisamment éloignés des paramètres actuels, et le seuil ϵ_N doit être assez élevé.

Mouvements d'échange entre chaînes adjacentes Comme expliqué ci-dessus, deux chaînes associées à de petits seuils de tolérance n'accepteront que rarement un mouvement d'échange. Par conséquent l'algorithme ABC-PT donne de moins bons résultats si nous imposons que les mouvements d'échange ne peuvent se faire qu'entre chaînes adjacentes. Le mélange des chaînes d'indices élevés serait favorisé, mais celui des chaînes d'indices plus faibles, et donc de la première chaîne d'intérêt, serait défavorisé.

REMARQUE 8.1.1 *A l'inverse, imposer des mouvements d'échange entre chaînes adjacentes pour l'algorithme Parallel Tempering classique est souvent avantageux.*

Sur le nombre de simulations Si n_{iter} itérations sont lancées sur les N chaînes en parallèle, alors le nombre de jeux de données qui auront été simulés à la fin de l'algorithme se situe entre $2n_{iter}N$ et $3n_{iter}N$.
En effet, à chaque itération chacune des N chaînes est actualisée localement, donc N jeux sont simulés, puis N mouvements d'échanges sont proposés. Lors d'un mouvement d'échange, si les deux chaînes choisies sont i et j $(i < j)$, nous devons simuler un jeu z_i' pour la chaîne i, et un jeu z_j' pour la chaîne j. Le mouvement est accepté si $\rho(S(z_i'), S(x)) < \epsilon_i$ et $\rho(S(z_j'), S(x)) < \epsilon_j$. En pratique, nous pouvons simuler un jeu z_i' et vérifier si la condition $\rho(S(z_i'), S(x)) < \epsilon_i$ est vérifiée. Si oui, nous simulons un jeu z_j'. Sinon, nous savons que le mouvement sera refusé, et nous pouvons passer directement au mouvement ou à l'itération suivante sans simuler z_j'. Dans le premier cas deux jeux sont simulés lors du mouvement d'échange, tandis que dans le deuxième cas un seul jeu est simulé.

8.2 Illustrations

8.2.1 Exemple jouet

Nous nous intéressons à l'exemple jouet utilisé notamment par Sisson et al. [199] et Beaumont et al. [18]. Le modèle est de la forme :

$$\begin{aligned} \theta &\sim \mathcal{U}_{[-10,10]}, \\ x \mid \theta &\sim 0.5\mathcal{N}(\theta, 1) + 0.5\mathcal{N}(\theta, 1/100). \end{aligned}$$

La distribution a posteriori associée à une seule observation $x = 0$ est :

$$\theta \mid x = 0 \sim \Big(0.5\mathcal{N}(0,1) + 0.5\mathcal{N}(0,1/100)\Big)\mathbb{1}_{[-10,10]}(\theta).$$

Nous désirons approximer $\pi(\theta \mid x = 0)$. En utilisant $S(x) = x$ et $\rho(x, z) = |z - x|$, nous obtenons $\pi_\epsilon(\theta \mid x = 0) = \pi(\theta \mid |x| < \epsilon)$. L'intérêt de cet exemple est que cette approximation s'exprime explicitement :

$$\begin{aligned}
\pi(\theta \mid |x| < \epsilon) &= \pi(\theta \mid x < \epsilon) - \pi(\theta \mid x < -\epsilon) \\
&\propto \pi(x < \epsilon \mid \theta) - \pi(x < -\epsilon \mid \theta) \\
&\propto \Phi(\epsilon - \theta) + \Phi(10(\epsilon - \theta)) - \Phi(-\epsilon - \theta) - \Phi(10(-\epsilon - \theta)) \\
&\propto \Phi(\epsilon - \theta) - \Phi(-\epsilon - \theta) + \Phi(10(\epsilon - \theta)) - \Phi(-10(\epsilon + \theta)),
\end{aligned}$$

avec Φ la fonction de répartition de la loi normale centrée réduite. Nous étudions le comportement de l'algorithme ABC-PT sur cet exemple, et le comparons aux algorithmes ABC standard, ABC-MCMC, ABC-MCMC augmenté, et ABC-PMC. Le seuil de tolérance cible est de 0.025, car alors l'approximation cible π_ϵ n'est pas visuellement discernable de l'a posteriori exact π.

ABC standard

La figure 8.1 montre l'approximation obtenue sur 5000 simulations (environ 45 secondes).
L'approximation est très bonne, cependant de nombreuses simulations sont nécessaires avant d'en accepter une. En effet, en moyenne 410 simulations sont nécessaires pour $\epsilon = 0.025$ (100 sont nécessaires si nous prenons $\epsilon = 0.1$).

ABC-MCMC

Comme Sisson et al. [199] et Beaumont et al. [18], nous utilisons un noyau gaussien $q(\theta' \mid \theta^{(t-1)}) \sim \mathcal{N}(\theta^{(t-1)}, 0.15^2)$. La performance de l'ABC-MCMC dépend de cette loi de proposition, mais nous ne discutons pas son choix ici, pour cet exemple tous les algorithmes ABC générant des chaînes de Markov utilisent ce noyau de transition.
La figure 8.1 illustre bien le problème de l'algorithme ABC-MCMC qui a tendance à rester bloqué dans les queues de distribution lorsque le seuil de tolérance est petit. Nous voyons que la chaîne est restée bloquée un moment dans une queue de distribution. Pour $\epsilon = 0.025$ et 600 000 itérations dont 150 000 de burn-in, l'algorithme met environ 1 minute 30 secondes.
Nous examinons les taux d'actualisation des chaînes générées en fonction du seuil de tolérance choisi, soit les proportions d'états proposés qui sont acceptés. Le tableau 8.1 montre que plus le seuil est petit et plus ces taux d'actualisation sont faibles.

Seuil de tolérance	2	1.5	1	0.5	0.1	0.025
Taux d'actualisation	82%	76%	66%	45%	13%	3.5%

TABLE 8.1 – ABC-MCMC : taux d'actualisation obtenus pour des seuils de tolérance valant 2, 1.5, 1, 0.5, 0.1 et 0.025 (600 000 itérations dont 150 000 de burn-in).

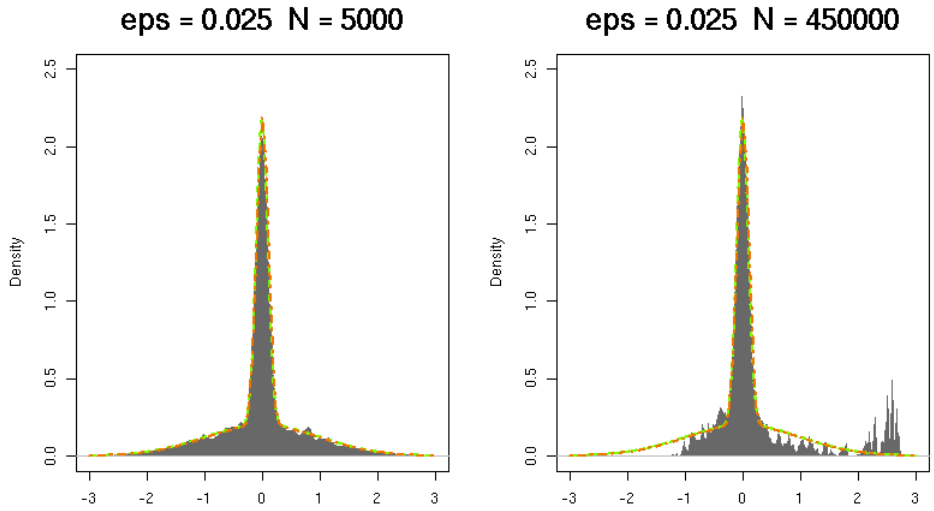

FIGURE 8.1 – Simulations obtenues avec l'ABC standard (à gauche, 5000 valeurs) et avec l'ABC-MCMC (à droite, 600 000 itérations dont 150 000 de burn-in) , pour $\epsilon = 0.025$. L'a posteriori exact est en pointillés oranges, l'approximation cible est en pointillés verts, et les densités des simulations obtenues sont grisées.

ABC-MCMC augmenté

Nous devons définir un noyau de transition pour générer (θ', ϵ') à partir de $(\theta^{(t-1)}, \epsilon^{(t-1)})$ (étape 2.a de l'algorithme). Nous considérons ϵ' et θ' indépendamment l'un de l'autre. Concernant θ', nous utilisons le noyau gaussien $q(\theta' \mid \theta^{(t-1)}) \sim \mathcal{N}(\theta^{(t-1)}, 0.15^2)$. Concernant ϵ', nous le proposons indépendemment de $\epsilon^{(t-1)}$ à partir d'un noyau exponentiel de paramètre τ. Avec $\tau = 10$, ϵ vaut en moyenne 0.1. Pour 600 000 itérations dont 150 000 de burn-in, l'algorithme met environ 2 minutes. Nous obtenons différentes chaînes par filtrage, en utilisant différentes valeurs seuils ϵ_T. La figure 8.2 montre les simulations obtenues pour les chaînes filtrées avec ϵ_T valant 0.5, 0.1 et 0.025. Il est clair que le résultat obtenu n'est pas satisfaisant. Pour $\epsilon_T = 0.025$, la chaîne reste bloquée à certains endroits, tandis que d'autres ne sont pas visités.

Le tableau 8.2 donne les pourcentages de simulations retenues pour chacune de ces chaînes. Pour l'ensemble de la chaîne, le taux d'actualisation obtenu est de 13%.

ϵ_T	2	1.5	1	0.5	0.1	0.025
Pourcentage de simulations retenues	100%	100%	99.96%	95.9%	27.0%	2.9%

TABLE 8.2 – ABC-MCMC augmenté : pourcentages de simulations retenues après filtrage.

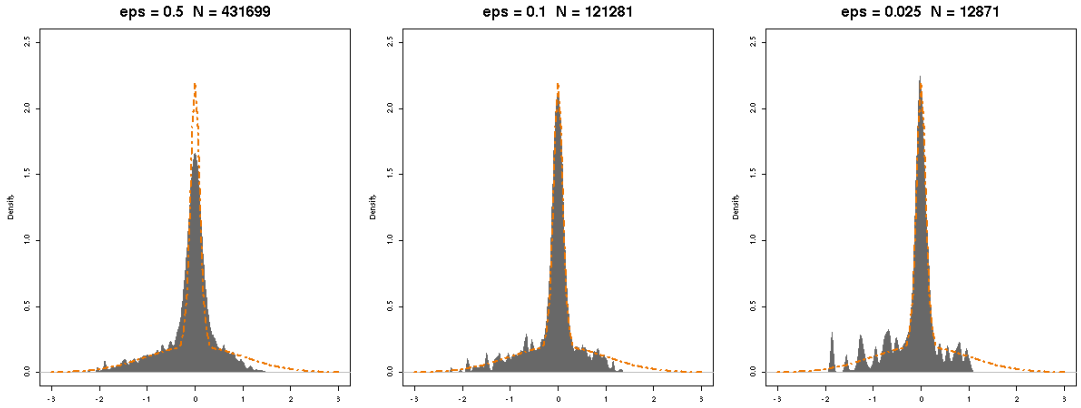

FIGURE 8.2 – ABC-MCMC augmenté : simulations obtenues pour ϵ_T valant 0.5, 0.1 et 0.025 (600 000 itérations dont 150 000 de burn-in). L'a posteriori exact est en pointillés oranges, et les densités des simulations obtenues sont grisées.

ABC-PMC

Nous utilisons la séquence de seuils de tolérance suivante : 2, 1.5, 1, 0.5, 0.1 et 0.025, et générons six populations de particules de tailles 5000. La figure 8.3 représente chacune de ces populations. Le résultat est bien plus satisfaisant que ceux de tous les autres algorithmes présentés jusqu'à

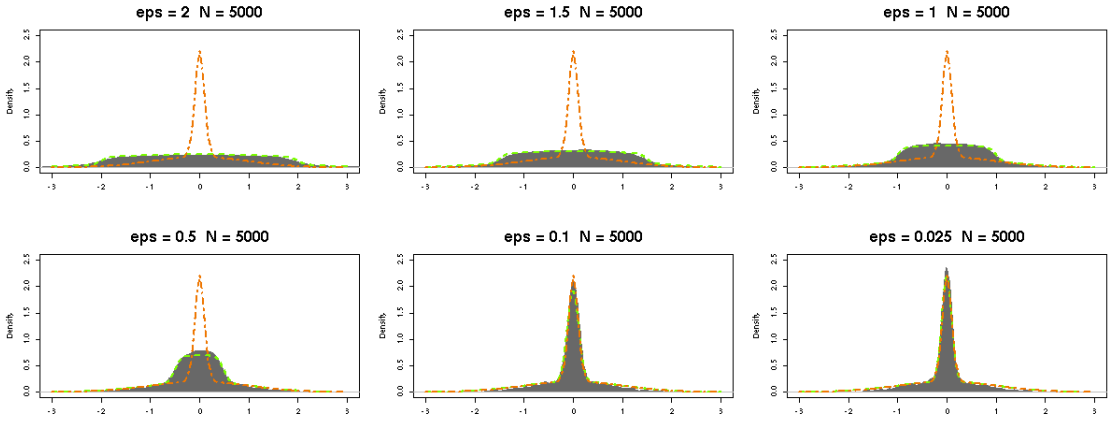

FIGURE 8.3 – ABC-PMC : les six populations associées aux seuils de tolérance 2, 1.5, 1, 0.5, 0.1 et 0.025 (5000 particules). L'a posteriori exact est en pointillés oranges, l'approximation cible est en pointillés verts, et les densités des populations pondérées obtenues sont grisées.

présent. Nous remarquons cependant que plus le seuil de tolérance diminue et plus les queues de distribution semblent être sous-représentées, voir la figure 8.4 pour un seuil de 0.025.

Le tableau 8.3 donne le nombre de simulations nécessaires en moyenne pour en accepter une, pour chaque population.

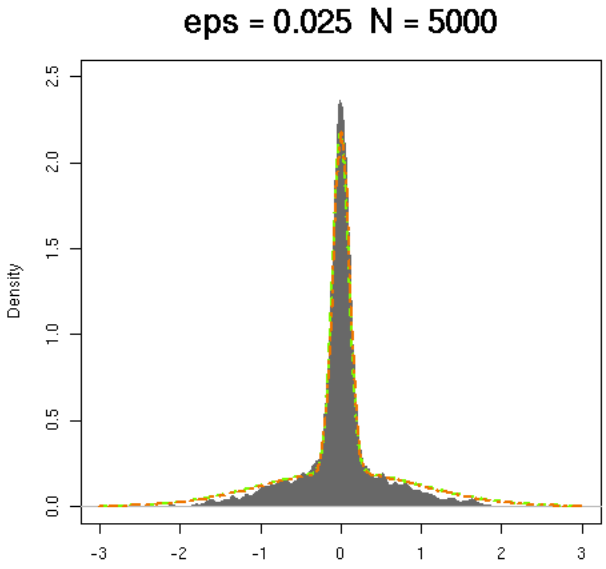

FIGURE 8.4 – ABC-PMC Zoom : population associée à un seuil de 0.025 (5000 particules). L'a posteriori exact est en pointillés oranges, l'approximation cible est en pointillés verts, et la densité de la population pondérée obtenue est grisée.

Seuil de tolérance	2	1.5	1	0.5	0.1	0.025	Total
Nb simus moyen	4.9	2.2	2.7	4.3	18.7	67.3	
Nb simus total	24684	10775	13678	21380	93332	336558	500407

TABLE 8.3 – ABC-PMC : nombres de simulations nécessaires en moyenne pour en accepter une, et nombres de simulations totales, pour chacune des six populations (calcul sur 5000 simulations acceptées).

Bien que les algorithmes précédemment présentés tournent rapidement sous R (environ deux minutes), cet algorithme est assez long (41 minutes). Cela est certainement dû aux tirages aléatoires des particules à chaque itération. Cependant, nous pourrions certainement diminuer le temps de calcul en prenant une plus petite séquence de seuils de tolérance.

ABC-PT

Nous prenons $N = 15$ chaînes en parallèle.

- Noyaux de transition : nous utilisons $q_i(\theta'_i \mid \theta_i^{(t-1)}) \sim \mathcal{N}(\theta_i^{(t-1)}, 0.15^2 * T_i)$.
- Températures : nous les prenons de 1 à 4, régulièrement espacées sur une échelle logarithmique.
- Séquence des seuils de tolérance : nous la définissons soit à l'aide de constantes de proportionnalité, soit en espaçant régulièrement les seuils sur une échelle logarithmique. Dans le

premier cas, nous prenons des constantes de proportionnalité régulièrement espacées entre 1 et 1.7 et $\epsilon_1 = 0.025$, ce qui donne $\epsilon_{15} = 1.85$. Dans le second, nous définissons les seuils comme étant régulièrement espacés entre 0.025 et 2. Les deux types de calibration donnent des résultats très similaires. Nous présentons ici des résultats avec les seuils régulièrement espacés sur une échelle logarithmique.

Pour 600 000 itérations dont 150 000 de burn-in, l'algorithme met environ 15 minutes. La figure 8.5 montre les simulations obtenues pour chacune des 15 chaînes en parallèle. La figure 8.6 se concentre sur les deux chaînes associées aux seuils de tolérance 0.025 et 0.087. Sur cette figure, il apparaît que les approximations obtenues par les chaînes associées à de petits seuils de tolérance sont satisfaisantes : nous devinons que les chaînes restent bloquées de petits moments dans les queues de distribution, mais jamais trop longtemps grâce aux mouvements d'échanges. Contrairement à l'ABC-PMC, l'approximation n'est pas toujours bien lisse au niveau des queues de distributions, mais celles-ci apparaissent bien remplies.
Le tableau 8.4 donne les taux d'actualisation et les nombres de mouvements d'échanges acceptés pour six des quinze chaînes.

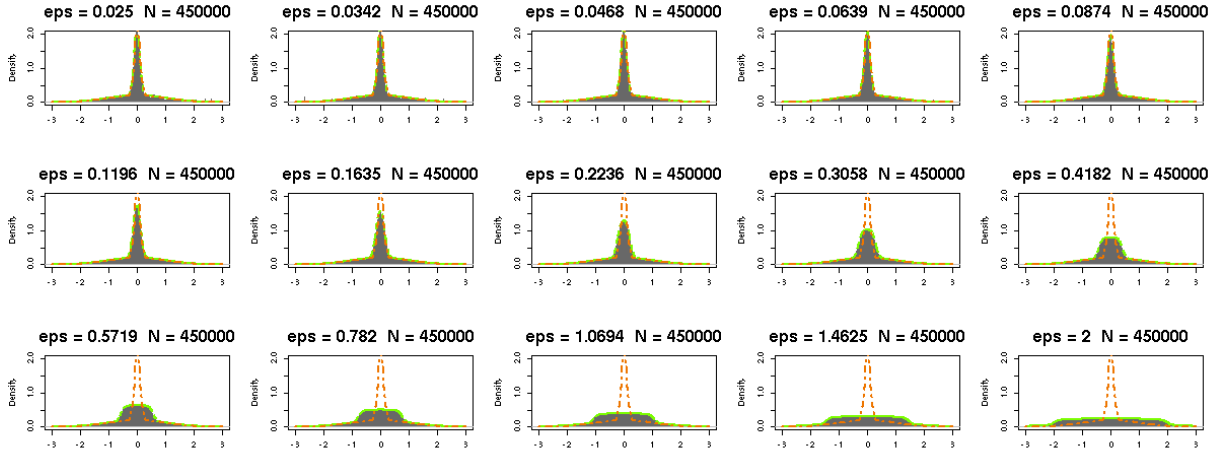

FIGURE 8.5 – ABC-PT : simulations obtenues par chacune des N chaînes, les seuils de tolérance associés allant de 0.025 à 2 (600 000 itérations dont 150 000 de burn-in). L'a posteriori exact est en pointillés oranges, l'approximation cible est en pointillés verts, et les densités des simulations obtenues sont grisées.

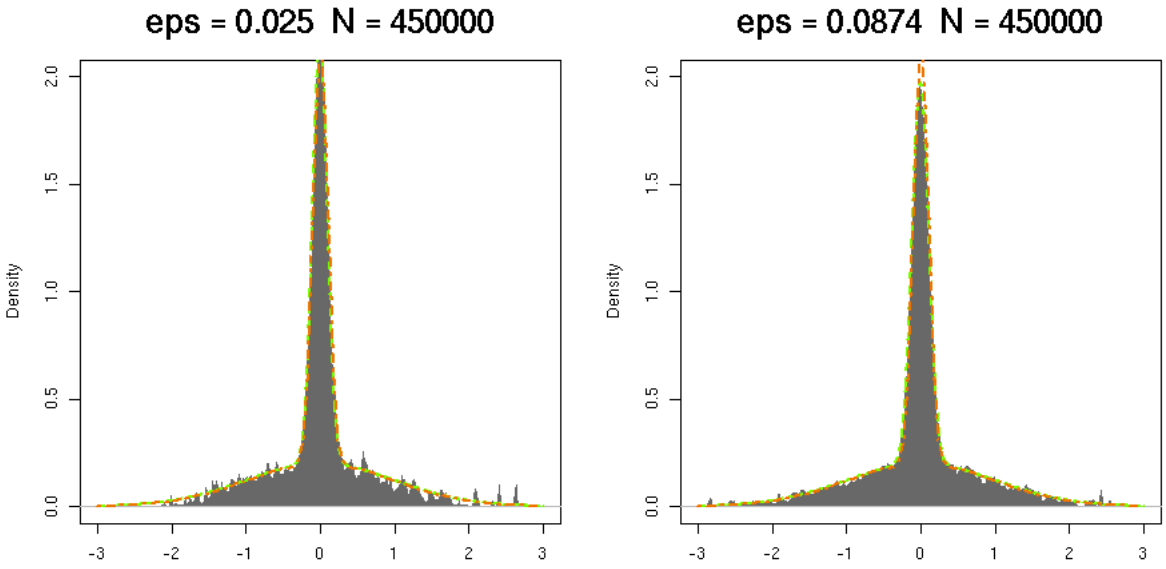

FIGURE 8.6 – ABC-PT Zoom : simulations obtenues pour les deux chaines associées aux seuils 0.025 et 0.087 (600 000 itérations dont 150 000 de burn-in). L'a posteriori exact est en pointillés oranges, l'approximation cible est en pointillés verts, et les densités des simulations obtenues sont grisées.

Seuil de tolérance	2	0.78	0.31	0.12	0.047	0.025
Taux d'actualisation	81.1%	56.7%	32.2%	15.5%	6.8%	3.8%
Mvts d'échange acceptés	193 458	182 987	111 152	53 938	22 933	12 567

TABLE 8.4 – Algorithme ABC-PT : taux d'actualisation et nombres de mouvements d'échanges acceptés, pour les chaînes 15, 12, 9, 6, 3 et 1.

REMARQUE 8.2.1 *Sur cet exemple où les simulations sont très rapides, l'algorithme ABC-PT est plus rapide que l'ABC-PMC. Cependant, cela n'est pas vrai de manière générale. Lorsque les simulations sont longues à obtenir, la grande partie du temps de calcul des algorithmes est consacrée à ces simulations. Dans ce cas, nous pouvons alors directement comparer les algorithmes en termes de nombres de simulations nécessaires.*

REMARQUE 8.2.2 *Si nous lançons l'algorithme ABC-PT sur seulement 40 000 itérations dont 5000 de burn-in, il met environ 1 minute, et la figure 8.7 montre les résultats obtenus. Les résultats apparaissent toujours meilleurs que ceux de l'ABC-MCMC et de l'ABC-MCMC augmenté, pour un temps plus court.*

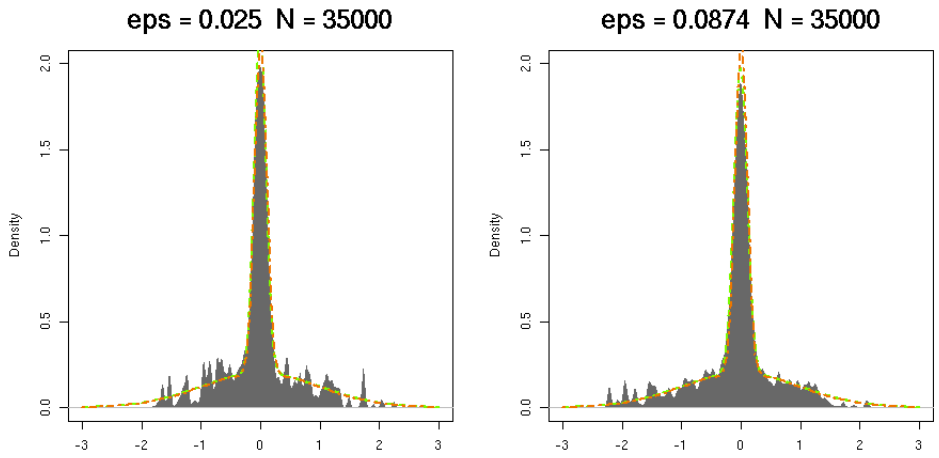

FIGURE 8.7 – ABC-PT : simulations obtenues pour les deux chaînes associées aux seuils 0.025 et 0.087 (40 000 itérations dont 5000 de burn-in). L'a posteriori exact est en pointillés oranges, l'approximation cible est en pointillés verts, et les densités des simulations obtenues sont grisées.

8.2.2 Exemple jouet modifié

Sur l'exemple jouet, les échantillonneurs les plus performants sont l'ABC-PMC et l'ABC-PT. Nous modifions un peu cet exemple afin d'étudier le comportement de ces deux échantillonneurs au niveau des queues de distribution. Nous ajoutons un petit mode dans une queue de distribution, et les deux modèles modifiés sont les suivants :

$$\begin{aligned} \theta &\sim \mathcal{U}_{[-10,10]}, \\ x \mid \theta &\sim 0.45\mathcal{N}(\theta, 1) + 0.45\mathcal{N}(\theta, 1/100) + 0.1\mathcal{N}(\theta - 5, 1), \end{aligned} \tag{8.11}$$

et

$$\begin{aligned} \theta &\sim \mathcal{U}_{[-10,10]}, \\ x \mid \theta &\sim 0.45\mathcal{N}(\theta, 1) + 0.45\mathcal{N}(\theta, 1/100) + 0.1\mathcal{N}(\theta - 9, 1). \end{aligned} \tag{8.12}$$

ABC-MCMC

La figure 8.8 montre les résultats de deux runs de l'ABC-MCMC en utilisant le modèle (8.11), pour des seuils de tolérance de 0.025. Lors du premier run la chaîne reste bloquée dans la zone de faible densité a posteriori autour de 5, et lors du second cette même zone n'est pas visitée. Dans le cas du modèle (8.12), le petit mode est encore plus éloigné du mode global, et il est encore plus difficile de bien approximer la distribution a posteriori d'intérêt.

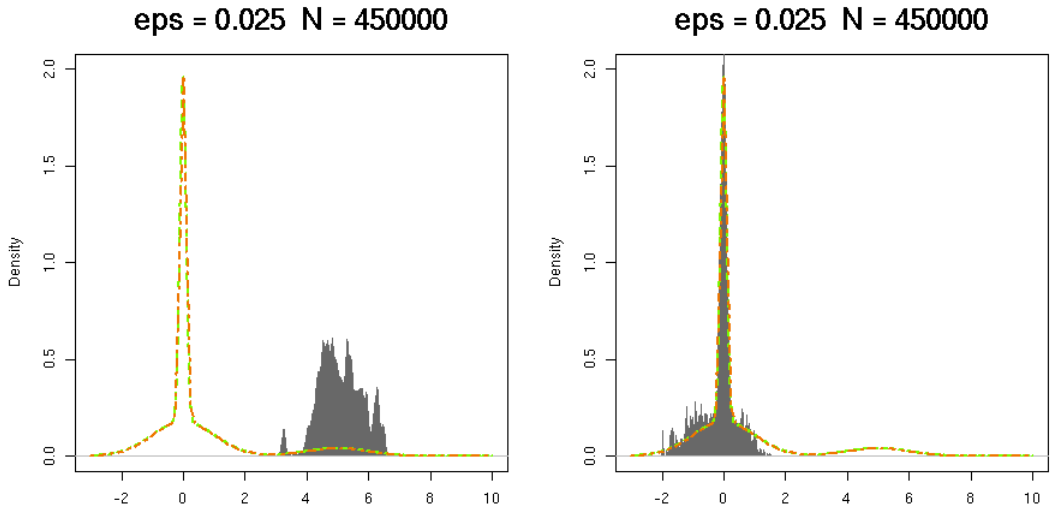

FIGURE 8.8 – ABC-MCMC : simulations obtenues lors de deux runs pour ϵ valant 0.025, en utilisant le modèle (8.11) (600 000 itérations dont 150 000 de burn-in). L'a posteriori exact est en pointillés oranges, l'approximation cible est en pointillés verts, et les densités des simulations obtenues sont grisées.

ABC-PMC

Nous utilisons toujours la séquence de seuils de tolérance 2, 1.5, 1, 0.5, 0.1 et 0.025, et générons six populations de particules de tailles 5000. La figure 8.9 représente la population associée au seuil 0.025, en utilisant le modèle (8.11). Comme pour la figure 8.4, nous remarquons que les zones de faible densité a posteriori sont sous-représentées. Pour obtenir les six populations, 1 104 603 simulations ont été nécessaires.

La figure D.1 en supplément représente la population associée au seuil 0.025 en utilisant le modèle (8.12). La zone de faible densité a posteriori autour du second mode paraît encore moins bien visitée que dans le cas du modèle (8.11).

ABC-PT

Nous lançons un algorithme ABC-PT avec les mêmes paramètres que dans l'exemple jouet, en utilisant le modèle (8.11). Pour 600 000 itérations dont 150 000 de burn-in, la première chaîne effectue 7738 échanges. Pour l'ensemble des 15 chaînes, il y a en moyenne 0.8 échange accepté par itération (sur 15, soit environ 5%), et 1.11 jeux de données sont simulés en moyenne par échange proposé. La figure 8.10 montre les résultats obtenus pour les deux chaînes associées aux seuils 0.025 et 0.06 : les zones de faible densité a posteriori sont bien explorées. En particulier, le petit mode autour de 5 est bien détecté, contrairement à l'ABC-PMC. Cependant, l'approximation obtenue est moins lisse que celle obtenue par l'ABC-PMC.

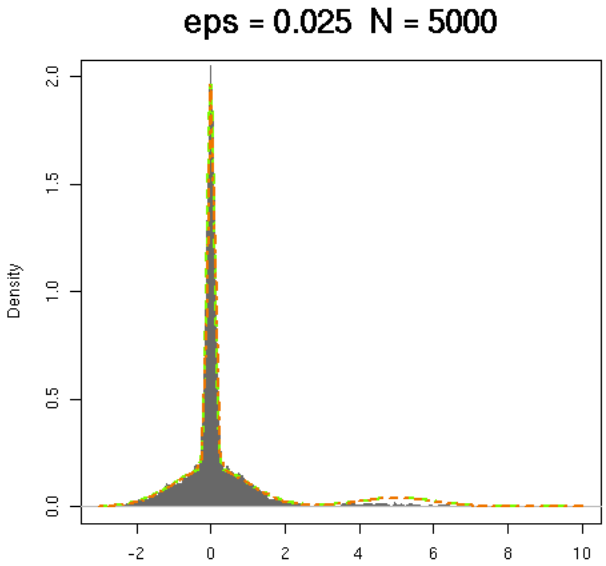

FIGURE 8.9 – ABC-PMC : population associée à un seuil de 0.025, en utilisant le modèle (8.11) (5000 particules). L'a posteriori exact est en pointillés oranges, l'approximation cible est en pointillés verts, et la densité de la population pondérée obtenue est grisée.

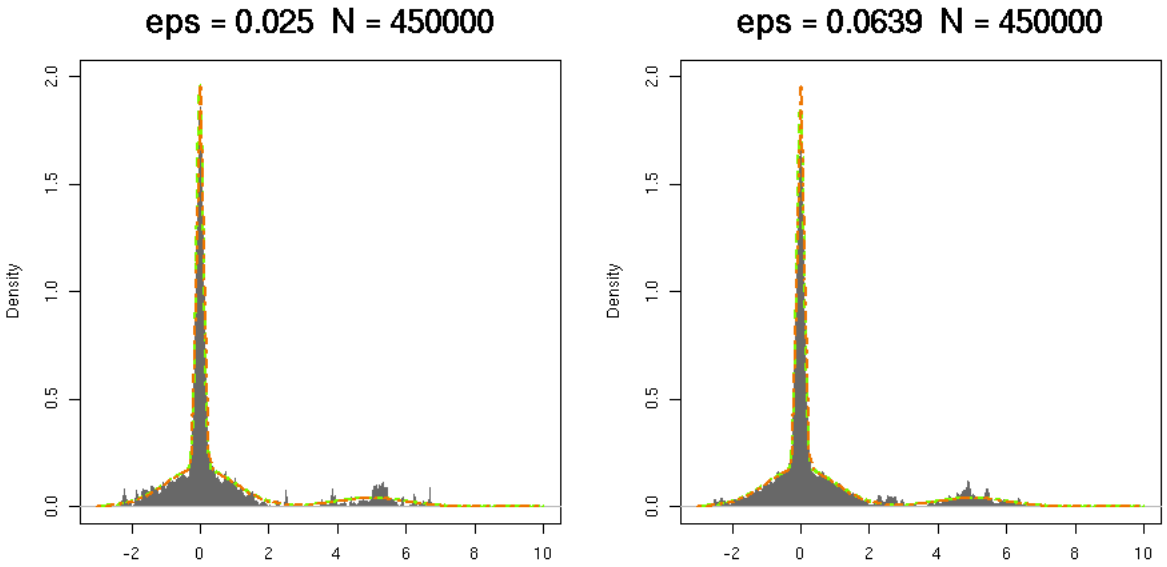

FIGURE 8.10 – ABC-PT : simulations obtenues pour les deux chaines associées aux seuils 0.025 et 0.06 (600 000 itérations dont 150 000 de burn-in, pour le modèle (8.11)). L'a posteriori exact est en pointillés oranges, l'approximation cible est en pointillés verts, et les densités des simulations obtenues sont grisées.

La figure D.2 en supplément représente les simulations de deux chaînes associées au seuil 0.025 (deux runs différents), en utilisant le modèle (8.12). La zone de faible densité a posteriori autour du second mode est toujours visitée, même si la fréquence n'est pas toujours celle attendue.

8.2.3 Exemple sur la transmission de la tuberculose

Nous examinons les performances de l'algorithme ABC-PT sur un exemple réél concernant la transmission de la tuberculose. Cet exemple a notamment été utilisé par Tanaka et al. [215] et par Sisson et al. [199]. L'objectif est d'estimer trois paramètres biologiques d'intérêt : le taux de reproduction, le temps de doublement de la prévalence, et le taux de transmission. Nous disposons de données génotypiques dénombrant les génotypes observés, soit les souches de tuberculose observées. Le modèle mathématique utilisé est un modèle de transmission et mutation. C'est une extension des modèles ne contenant que des morts et des naissances de cas. La naissance d'un cas correspond à une nouvelle infection, et une mort à un hôte mort ou guéri. En modélisant en plus les mutations du germe ou de la batérie pathogène, nous pouvons tirer parti de l'observation des différentes souches. Une mutation correspond au remplacement d'une souche de pathogène par une autre au sein d'un hôte. Ce dernier reste donc infecté, seul le génotype du pathogène change. Trois évènements sont donc possibles au cours de la transmission : naissance ou mort d'un cas, et mutation.
Nous utilisons les mêmes hypothèses biologiques que Tanaka et al. [215] : toute mutation donne naissance à un génotype qui n'avait jamais existé auparavant, et tous les génotypes ont les mêmes propriétés épidémiologiques. Nous supposons de plus que la probabilité de chaque évènement sur un court intervalle de temps est proportionnelle au nombre de cas, et les taux par individu restent constants au cours du temps. Nous avons donc affaire à des processus homogènes dans le temps. Enfin, nous supposons qu'initialement l'épidémie commence avec un seul cas introduit dans une population saine.

Modélisation

En reprenant les notations de Tanaka et al. [215], le nombre de cas ayant le génotype i du pathogène au temps t est noté $X_i(t)$, $G(t)$ est le nombre de génotypes qui ont existé jusqu'au temps t, et $N(t)$ est le nombre de cas qui ont été infectés avant ou au temps t. Les génotypes sont notés $1, 2, 3, \ldots$ par convenance, bien qu'il n'y ait pas d'ordre entre eux, sauf pour $i = 1$ qui représente le génotype initial, dont tous les autres descendent.

$$N(t) = \sum_{i=1}^{G(t)} X_i(t).$$

Trois taux définissent le système : le taux de naissance α, le taux de mort δ et le taux de mutation θ (taux par cas par année). Nous définissons :

$$\begin{aligned} P_{i,x}(t) &= P(X_i(t) = x), \\ \bar{P}_n(t) &= P(N(t) = n), \\ \tilde{P}_g(t) &= P(G(t) = g). \end{aligned}$$

L'évolution de $P_{i,x}(t)$ est définie par les équations différentielles suivantes :

$$\frac{dP_{i,x}(t)}{dt} = \underbrace{-(\alpha+\delta+\theta)xP_{i,x}(t)}_{\text{aucun évènement}} + \underbrace{\alpha(x-1)P_{i,x-1}(t)}_{\text{naissance}} + \underbrace{(\delta+\theta)(x+1)P_{i,x+1}(t)}_{\text{mort ou mutation}}, \quad (8.13)$$

$$\text{pour} \qquad x = 1, 2, 3, \ldots$$

$$\frac{dP_{i,0}(t)}{dt} = (\delta+\theta)P_{i,1}(t). \quad (8.14)$$

Comme nous avons initialement une seule copie du génotype ancestral, les conditions initiales sont : $P_{i,x}(0) = 0$ pour tout (i, x), sauf $P_{1,1}(0) = 1$, et pour $i = 2, 3, 4, \ldots$, $P_{i,0}(0) = 1$.

Pour prendre en compte la création de nouveaux génotypes, la probabilité $\tilde{P}_g(t)$ suit les équations différentielles suivantes (seule une mutation peut créer un nouveau génotype) :

$$\frac{d\tilde{P}_g(t)}{dt} = \underbrace{-\theta N(t)\tilde{P}_g(t)}_{\text{pas de mutation}} + \underbrace{\theta N(t)\tilde{P}_{g-1}(t)}_{\text{mutation}}, \qquad \text{pour} \qquad g = 2, 3, 4, \ldots \quad (8.15)$$

$$\frac{d\tilde{P}_1(t)}{dt} = -\theta N(t)\tilde{P}_1(t). \quad (8.16)$$

La condition initiale est $G(0) = 1$. Soit t_g le temps auquel un nouveau génotype g a été créé, nous avons $P_{g,1}(t_g) = P(X_g(t_g) = 1) = 1$, et $P_{g,x}(t_g) = P(X_g(t_g) = x) = 0$ pour $x \neq 1$.

Le nombre total de cas $N(t)$ est quant à lui modélisé par les équations différentielles suivantes (seule une mort ou une naissance influe le nombre de cas, lors d'une mutation le nombre de cas reste inchangé) :

$$\frac{d\bar{P}_n(t)}{dt} = \underbrace{-(\alpha+\delta)n\bar{P}_n(t)}_{\text{pas mort, pas naissance}} + \underbrace{\alpha(n-1)\bar{P}_{n-1}(t)}_{\text{naissance}} + \underbrace{\delta(n+1)\bar{P}_{n+1}(t)}_{\text{mort}}, \quad (8.17)$$

$$\text{pour} \qquad n = 1, 2, 3, \ldots$$

$$\frac{d\bar{P}_0(t)}{dt} = \delta\bar{P}_1(t). \quad (8.18)$$

Les conditions initiales sont $\bar{P}_1(0) = 1$ et $\bar{P}_n(0) = 0$ pour $n \neq 1$.

Cette modélisation est gouvernée par trois taux : les taux de naissance, mort et mutation, α, δ et θ. Cependant, les quantités d'intérêt pour les biologistes et médecins sont le taux de transmission, le temps de doublement de la prévalence, et le taux de reproduction R. Le taux de transmission permet d'appréhender la vitesse à laquelle le nombre de cas augmente, et est défini par $\alpha - \delta > 0$. Un paramètre associé est le temps de doublement de la prévalence, qui est $\ln(2)/(\alpha - \delta)$ pour le modèle utilisé. Le taux de reproduction R correspond quant à lui au nombre de nouvelles infections générées par un seul cas, et est défini par le ratio α/δ. Plus de détails sur ces paramètres d'intérêt et sur la manière de les obtenir peuvent être trouvés dans Tanaka et al. [215].

Simulation d'un jeu de données :

Nous ne pouvons écrire de manière explicite la vraisemblance associée à ce modèle. Par contre, ce modèle ne contient que des évènements de naissance, mort ou mutation et il est très facile, bien que long, de simuler des jeux de données à partir de paramètres α, δ et θ.

Pour simuler à partir de paramètres α, δ et θ donnés, la population doit être initialisée : $X_1(0) = 1$, $X_i(0) = 0$ pour $i \neq 1$, $N(0) = 1$ et $G(0) = 1$.
Chaque temps est caractérisé par un évènement : naissance, mort ou mutation. Nous pouvons modéliser le temps entre deux évènements, mais ce n'est pas utile ici, nous considérons donc simplement que t augmente de 1 lorsqu'un évènenent a lieu.
Pour simuler, à chaque temps nous devons choisir un génotype parmi les génotypes encore existants, puis choisir un évènement parmi les trois possibles. Plus précisément :

1. Pour chaque génotype, la probabilité de le choisir est proportionnelle au nombre de cas portant ce génotype : $X_i(t)/N(t)$ pour le génotype i.

2. En ce qui concerne le choix d'un évènement parmi les trois, les probabilités de choisir une naissance, une mort ou une mutation sont proportionelles à α, δ et θ respectivement.

Une fois ces deux choix faits, il est très facile d'actualiser les X_i, N et G.
La taille de la population simulée va donc varier au fur et à mesure du temps. Si la population s'éteint, soit $N(t) = 0$ pour un certain t, alors la simulation est écartée, nous n'en tenons pas compte. Si la population atteint une taille prédéfinie N_{stop}, alors la simulation est stoppée et un échantillon de taille n de la population obtenue est tiré aléatoirement sans remise. La valeur N_{stop} doit refléter la taille d'une population d'un niveau de diversité approprié par rapport aux données observées. En effet, plus les données sont observées longtemps après le début de l'épidémie et plus il y a de diversité dans la population des infectés, population dont nous observons un échantillon.

Les données

Tout comme Tanaka et al. [215] et Sisson et al. [199], nous utilisons des données de Small et al. [201] receuillies lors d'une épidémie de tuberculose à San Francisco entre 1991 et 1992. Plus précisément, 473 isolats ont été receuillis, et analysés pour le marqueur IS61110 par la méthode « finger-print ». Ces 473 isolats correspondent à 326 génotypes différents, et se répartissent de la manière suivante :

$$30^1 23^1 15^1 10^1 8^1 5^2 4^4 3^{13} 2^{20} 1^{282},$$

où n^k signifie que k groupes de génotypes de taille n sont observés. Nous prenons $N_{stop} = 10000$ pour les étapes de simulation. En effet, prendre $N_{stop} = 1000$ donne des échantillons de taille 473 de diversité insuffisante, tandis que prendre $N_{stop} = 10^5$ n'est pas réaliste par rapport aux tailles de populations infectées observées en général dans une région donnée. Cependant, Tanaka et al. [215] ont montré que les résultats obtenus par des méthodes ABC ne sont pas très sensibles à ce paramètre.

Statistiques et distance utilisées :
Un échantillon de taille n peut être résumé à l'aide de deux statistiques considérées pertinentes par les biologistes : le nombre de génotypes distincts présents dans cet échantillon, noté g (potentiellement différent du G de la population totale), ainsi que la diversité génétique H, définie par :

$$H = 1 - \sum_{i=1}^{g} \left(\frac{n_i}{n}\right)^2,$$

avec n_i le nombre de génotypes i présents dans cet échantillon. Sur les données de Small et al. [201], nous avons $n = 473$, $g_{obs} = 326$ et $H_{obs} = 0.9892236$.

Soit g_{obs} et H_{obs} les valeurs obtenues sur les données observées, un échantillon de taille n obtenu par simulation et associé à g et H est considéré proche des données observées si :

$$\frac{1}{n}|g_{obs} - g| + |H_{obs} - H| < \epsilon,$$

avec ϵ un seuil de tolérance approprié. Nous normalisons g et g_{obs} par $1/n$ car ils peuvent prendre des valeurs entre 0 et n tandis que H et H_{obs} prennent des valeurs entre 0 et 1.

Seuil de tolérance cible :
En utilisant les mêmes données que nous, Tanaka et al. [215] ont estimé les trois paramètres biologiques d'intérêt en utilisant un algorithme ABC-MCMC. Ils ont utilisé un seuil de tolérance de 0.0025, et ont lancé 2.5 millons d'itérations après burn-in. Cela a nécessité une semaine de calculs sur un cluster computationnel. Etant donné que nous n'avons pas accés à un cluster com-

putationnel mais seulement à un serveur, l'obtention de résultats avec le même seuil de tolérance pour 2.5 millions d'itérations nous prendrait plus de 40 jours. Cela n'étant pas raisonnable pour nous, nous avons décidé de prendre un seuil de tolérance cible plus grand, valant 0.01.
De plus, Tanaka et al. [215] ont étudié l'influence du choix du seuil ϵ en testant les valeurs 0.0025, 0.005, 0.015 et 0.025. Les taux d'acceptation pour les chaînes générées par l'algorithme ABC-MCMC sont alors de 0.03%, 1.3%, 5.9% et 10.3% respectivement. Ainsi, en prenant un seuil cible de 0.01, le taux d'acceptation sera plus élevé que pour 0.0025, et nous pouvons réduire le nombre d'itérations.

Distributions a priori :
Les paramètres de naissance et mort α et δ sont supposés suivre a priori des uniformes entre 0 et 5, avec $\alpha > \delta$. Concernant le paramètre de mutation θ, de nombreuses études ont été effectuées pour l'estimer (références dans Tanaka et al. [215]), ce qui nous permet de prendre l'a priori informatif suivant :

$$\theta \sim \mathcal{N}(0.198, 0.06735^2), \qquad \text{tronquée à gauche en 0.}$$

Résultats

ABC-PT :
Les paramètres gouvernant le modèle sont notés $\phi = (\alpha, \delta, \theta)$. Nous notons $\phi_i^{(t)}$ les paramètres de la chaîne d'indice i au temps t, $i = 1, \ldots, N$. Nous utilisons $N = 7$ chaînes, et pour le noyau de transition de la première chaîne nous prenons comme Tanaka et al. [215] $q(. \mid \phi_1^{(t-1)})$ correspondant à la densité d'une gaussienne $\mathcal{N}(\phi_1^{(t-1)}, \Sigma)$, avec :

$$\Sigma = \begin{pmatrix} 0.5^2 & 0.225 & 0 \\ 0.225 & 0.5^2 & 0 \\ 0 & 0 & 0.015^2 \end{pmatrix}.$$

Pour les autres chaînes, nous prenons $q_i(. \mid \phi_i^{(t-1)})$ correspondant à la densité d'une gaussienne $\mathcal{N}(\phi_i^{(t-1)}, \Sigma_i)$, avec $\Sigma_i = \Sigma^{1/T_i}$. Nous prenons des températures comprises entre 1 et 2 régulièrement espacées sur une échelle logarithmique. En ce qui concerne la séquence des seuils de tolérance nous prenons $\epsilon_1 = 0.01$, et ϵ_2 à ϵ_7 régulièrement espacés de 0.03 à 0.1 sur une échelle logarithmique. Nous démarrons de 0.03 pour ne pas avoir trop de chaînes associées à de trop petits seuils de tolérance Nous lançons l'algorithme ABC-PT sur 20 000 itérations, dont 2 000 de burn-in. Cela a pris environ 11 jours de temps de calcul.

Sur les 20 000 itérations, la première chaîne a 3151 valeurs de paramètres différentes (15,8%). Cette chaîne est actualisée localement dans 3,8% des itérations, et elle effectue en moyenne 0.16 mouvement d'échange par itération (3229 mouvements au total).

En ce qui concerne l'ensemble des $N = 7$ chaînes, il y a en moyenne 2.29 échanges acceptés par itération (sur 7, soit environ 33%), et 1.38 jeux de données sont simulés en moyenne par échange proposé. Le tableau 8.5 donne les pourcentages de mouvements d'échange acceptés en fonction des paires de chaînes choisies, ainsi que les taux d'actualisation locale de chacune des chaînes. La première chaîne n'échange pas plus avec les chaînes d'indices faibles qu'avec celles d'indices plus élevés (contrairement aux échantillonneurs PT et PTEEM).

	chaîne 1	chaîne 2	chaîne 3	chaîne 4	chaîne 5	chaîne 6	chaîne 7
chaîne 1	3.8	7.2	8.6	9.1	9.1	8.2	6.2
chaîne 2	.	13.8	30.8	35	34.6	30.7	26.4
chaîne 3	.	.	16.6	41.4	42.4	39.9	34.0
chaîne 4	.	.	.	17.9	51.2	49.8	43.8
chaîne 5	.	.	.	.	18.7	59.9	54.6
chaîne 6	.	.	.	.	.	19.7	64.2
chaîne 7	.	.	.	.	.	.	20

TABLE 8.5 – ABC-PT : la diagonale donne taux d'actualisation locale des chaînes, et hors diagonale sont donnés les pourcentages de mouvements d'échange acceptés en fonction des paires de chaînes choisies.

Les estimations des paramètres d'intérêt obtenues sur cette première chaîne sont données dans le tableau 8.6. Ces estimations sont du même ordre de grandeur que celles obtenues par Tanaka et al. [215], Sisson et al. [199] et Blum [25]. La moyenne du taux de transmission est autour de 0.60. Le temps de doublement de la prévalence fournit la même information car est une simple transformation du taux de transmission, on l'utilise car il est plus facile d'interprétation par les médecins (temps au bout duquel le pourcentage de malades a doublé dans la population). Ici, ce temps est estimé autour de 1.35 années. Si nous examinons le taux de reproduction (nombre de nouvelles infections générées par un seul cas), son intervalle de crédibilité est très large. En effet, nous avons des probabilités a posteriori positives pour de très petites valeurs de δ, soit de très grandes valeurs de ce taux. Cependant, la majorité des valeurs a posteriori sont autour de la médiane (voir la figure 8.11), qui doit donc être privilégiée par rapport à la moyenne qui n'est pas très pertinente car trop instable.

	Moyenne	Médiane	Int. crédibilité à 95%
Taux de transmission	0.59	0.58	(0.29,0.93)
Temps de doublement	1.35	1.20	(0.75,2.43)
Taux de reproduction	17.44	2.26	(1.22,20.52)
Taux de mutation	0.25	0.24	(0.15,0.35)

TABLE 8.6 – ABC-PT : estimations a posteriori des paramètres d'intérêt sur les 18 000 itérations post-burn-in de la première chaîne : moyenne, médiane et intervalle de crédibilité à 95%.

Comparaisons avec ABC standard, ABC-MCMC et ABC-PMC :

Nous lançons des échantillonneurs ABC standard, ABC-MCMC et ABC-PMC avec les mêmes seuils de tolérance cibles. Pour l'ABC standard, 1000 particules sont générées, pour l'ABC-MCMC 350 000 itérations sont effectuées dont 50 000 de burn-in, et pour l'ABC-PMC 1000 particules sont générées en utilisant une séquence de seuils de tolérance de taille 10, avec $\epsilon_1 = 1$, $\epsilon_{10} = 0.01$ et $\epsilon_k = 0.5(\epsilon_{k-1} + \epsilon_{10})$, comme Sisson et al. [199]. De plus, pour l'ABC-PMC nous utilisons pour l'étape 3.a.ii de l'algorithme (annexe D.3), des noyaux de transitions gaussiens centrés sur les particules échantillonnées dans la population précédente, et de matrices de covariance Σ_t. Dans l'étape 3.c, Σ_t est pris égal à deux fois la covariance empirique pondérée des $\{\phi_j^{(t)}\}_{j=1,\ldots,1000}$ (voir Filippi et al. [64]).

Les estimations des paramètres d'intérêt obtenues avec ces trois algorithmes sont similaires à celles obtenues par l'ABC-PT, voir la figure 8.11 montrant les densités a posteriori obtenues pour les paramètres d'intérêt, ainsi que le tableau D.3 en supplément.

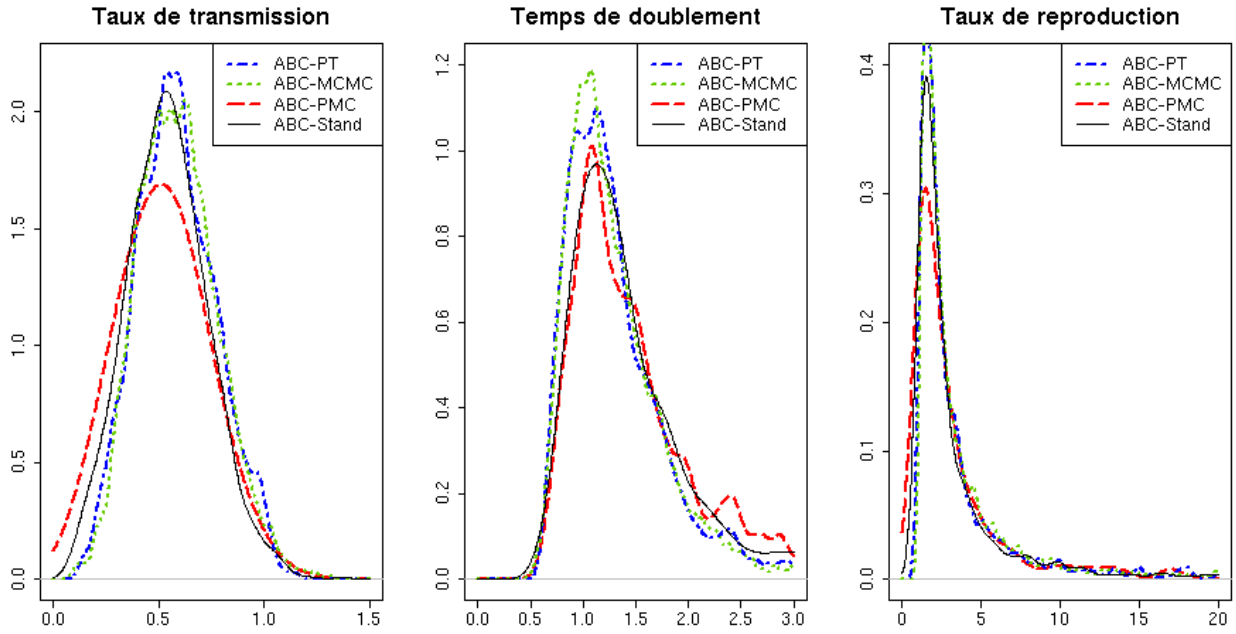

FIGURE 8.11 – Densités a posteriori du taux de transmission, du temps de doublement et du taux de reproduction, obtenues par l'ABC-PT (première chaîne), l'ABC-MCMC, l'ABC-PMC (dernière population) et l'ABC standard.

- Concernant l'ABC standard, 408 simulations sont nécessaires en moyenne pour obtenir une particule. L'échantillonneur a mis environ 6.5 jours.
- Concernant l'ABC-PMC, 203 simulations sont nécessaires en moyenne pour obtenir une particule. L'échantillonneur a mis environ 4,5 jours.

- Concernant l'ABC-MCMC, L'échantillonneur a mis environ 9 jours. La chaîne a été actualisée dans 3,8% des itérations, ce qui donne 13290 valeurs de paramètres différentes sur les 350 000 itérations. Les simulations de cette chaîne sont plus corrélées que les simulations de la première chaîne de l'ABC-PT. Cela se devine à l'oeil nu : pour le paramètre α, la figure 8.12 montre les tracés des chaînes obtenues par l'ABC-MCMC et l'ABC-PT (première chaîne) sur 10 000 itérations post-burn-in. Le tableau 8.7 donne pour ce paramètre les autocorrélations d'ordre 1, 10 et 20 de ces deux chaînes, en retenant toutes ou une partie des itérations (résultats similaires pour les paramètres δ et θ). Afin d'avoir des autocorrélations raisonnables (décroissant rapidement et négligeables à l'ordre 4), il faut retenir 1 itération sur 500 pour la chaîne de l'ABC-MCMC, et 1 sur 15 pour la première chaîne de l'ABC-PT. Par conséquent, 583 simulations sont nécessaires en moyenne pour en obtenir une pour l'ABC-MCMC (350 000/600), et 277 sont nécessaires en moyenne pour en obtenir une pour l'ABC-PT (20 000 * 7 * (1+1.38) / 1200). En supplément, la figure D.3 montre la fonction d'autocorrélation de la première chaîne de l'ABC-PT pour laquelle 1 itération sur 15 est retenue.

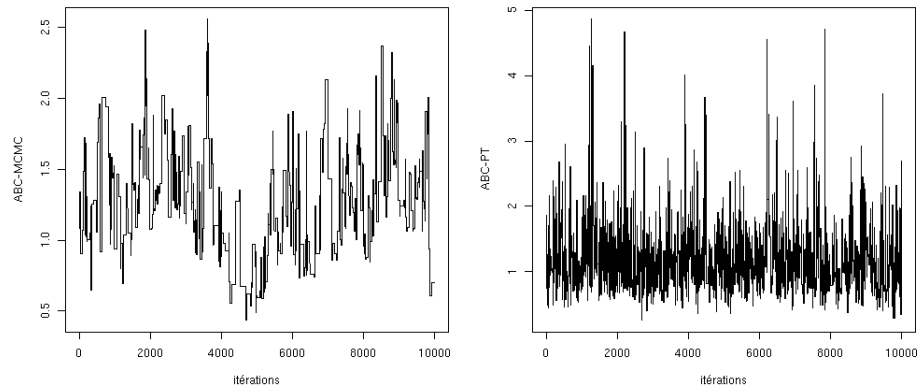

FIGURE 8.12 – Pour le paramètre α, sur 10 000 itérations post-burn-in. A gauche le tracé de la chaîne obtenue par l'ABC-MCMC, et à droite le tracé de la première chaîne obtenue par l'ABC-PT.

	Itérations conservées	Nb itérations	Ordre 1	Ordre 10	Ordre 20
ABC-MCMC	toutes	300 000	0.991	0.919	0.850
ABC-MCMC	1 sur 100	3 000	0.513	0.107	0.032
ABC-MCMC	1 sur 500	600	0.162	0.056	0.017
ABC-PT (chaîne 1)	toutes	18 000	0.841	0.272	0.161
ABC-PT (chaîne 1)	1 sur 10	1 800	0.318	0.013	-0.003
ABC-PT (chaîne 1)	1 sur 15	1 200	0.201	-0.024	-0.008

TABLE 8.7 – Paramètre α : autocorrélations d'ordre 1, 10 et 20 de la chaîne obtenue par l'ABC-MCMC, et de la première chaîne de l'ABC-PT (toutes ou une partie seulement des itérations retenues).

Notons que l'obtention de simulations corrélées est inhérente à n'importe quelle méthode MCMC, et cela n'empêche pas d'obtenir de bonnes estimations des paramètres d'intérêt. Nous pouvons par exemple décider de conserver 1 itération sur 10 pour l'ABC-PT, et dans ce cas l'ABC-PT et l'ABC-PMC nécessitent des nombres de simulations équivalents pour en conserver une (185 vs 203 simulations).

8.3 Discussion et perspectives

8.3.1 Sur la méthode ABC-PT

L'algorithme ABC-PT développé est une nouvelle méthode sans vraisemblance basée sur la théorie des chaînes de Markov. Sur les exemples étudiés, il donne de meilleurs résultats que l'ABC-MCMC, notamment en termes d'exploration de l'espace des paramètres et d'autocorrélations de la chaîne générée. Sa performance apparaît équivalente à celle de l'algorithme ABC-PMC, qui est une méthode séquentielle. L'inconvénient de l'APC-PT par rapport à l'ABC-PMC est que l'approximation de l'a posteriori obtenue est moins lisse que celle obtenue par l'ABC-PMC (pour de très petits seuils de tolérance). Par contre, l'ABC-PT a l'avantage de mieux explorer les zones de faible densité a posteriori, et en particulier de mieux détecter de petits modes dans ces zones. Suivant l'objectif recherché, l'un ou l'autre des deux algorithmes sera le plus pertinent. Il faudrait confirmer ces résultats sur d'autres exemples.

La méthode développée n'utilise que le mouvement d'échange comme mouvement global. Cependant, rien ne nous empêche de définir d'autres types de mouvements, comme ceux de l'Evolutionary Monte Carlo [139], voir la partie 5.3.2. Il faut simplement conserver l'idée d'un mouvement en deux étapes (proposition des nouveaux paramètres et actualisation des jeux de données associés), afin de ne pas faire intervenir la vraisemblance dans les probabilités d'acceptation. Nous pouvons également définir des mouvements à rejet retardé, voir la partie 5.3.1.

Enfin, une perspective est d'utiliser les simulations générées par les chaînes autres que la première, en développant une méthode dans le même esprit que celles de Beaumont et al. [19] ou Blum et François [26].

8.3.2 Perspective sur la construction semie-automatique de statistiques dans le cas d'un n-échantillon

Pour l'ensemble des méthodes sans vraisemblance, le choix de la statistique permettant de résumer les données est crucial, car $\pi(\theta \mid \rho(S(z), S(x)) < \epsilon)$ est une approximation raisonnable de la distribution a posteriori d'intérêt $\pi(\theta \mid x)$ si ϵ est suffisamment petit, et si S résume correctement les données. Dans certains problèmes, les connaissances des spécialistes du domaine peuvent permettre de définir des statistiques pertinentes, voir les nombreux exemples donnés par Csilléry et al. [51]. Lorsque de nombreuses statistiques sont disponibles, plusieurs méthodes de

sélection ou de combinaison de statistiques ont été proposées, voir la partie 7.4. Mais la plupart du temps il est assez difficile de définir de telles statistiques. A cause de cette difficulté, Sousa et al. [205] ont par exemple décidé de ne pas utiliser de statistique.

Nous pouvons donc être confrontés à un problème où aucune statistique pertinente n'est proposée par les spécialistes du domaine, et où très peu d'information est disponible sur la forme de la distribution a posteriori cible. Nous pourrions utiliser des statistiques classiques, comme des moyennes ou des variances par exemple. Cependant, dans beaucoup de situations les distributions a posteriori peuvent être multimodales, et ces statistiques alors très peu pertinentes. Dans ce cas, il est intéressant de pouvoir construire de manière semi-automatique une statistique permettant de comparer les données observées à des données simulées. Cette approche revient à comparer deux populations à travers une statistique résumé S, et en fonction d'un seuil de tolérance ϵ. En nous basant sur le principe des tests lisses introduits par Neyman [167] (voir plus récemment Rayner et al.[183]), nous proposons une statistique S pouvant être utilisée lorsque nous disposons d'un n-échantillon de données quantitatives, uni ou multi-dimensionnelles. Ces tests ont l'avantage d'avoir de bonnes propriétés aymptotiques et certaines propriétés d'optimalité comme décrites dans Rayner et Best [182].

Nous supposons que le jeu de données observé est un n-échantillon de variables aléatoires $X_{(1)}, X_{(2)}, \ldots, X_{(n)}$ indépendantes, de même loi de densité f_θ, et à valeurs dans $\mathbb{R}^d$ (les $_{(i)}$ sont des notations, ils ne correspondent pas à des statistiques d'ordre). De même, le jeu de données simulées est un n-échantillon de variables aléatoires $Z_{(1)}, Z_{(2)}, \ldots, Z_{(n)}$ indépendantes, de même loi de densité $f_{\theta'}$, et à valeurs dans $\mathbb{R}^d$. Nous voulons construire un test automatique comparant les densités des populations observées et simulées f_θ et $f_{\theta'}$. Nous supposons que si elles peuvent être considérées comme proches suivant un certain seuil ϵ, le paramètre θ' ayant servi à générer les données simulées peut être considéré comme proche de θ et conservé. Le principe est de développer f_θ et $f_{\theta'}$ dans une base de fonctions, pour ensuite pouvoir les comparer facilement. La base étant de dimension infinie, en pratique nous ne conservons que les k premières dimensions. La principale difficulté va résider dans le choix de cette dimension k.

Supposons le cas simple où le support $\mathcal{X}$ des $X_{(i)}, i \in \{1, \ldots, n\}$ est inclu dans $\mathbb{R}$. Nous devons choisir une mesure de référence ν de telle sorte que la loi des $X_{(i)}$ soit dans $L^2_\nu(\mathbb{R})$ muni du produit scalaire classique sur les fonctions. Nous prenons ensuite une base dénombrable de fonctions orthogonales associée $\{P_i, i \in \mathbb{N}\}$ telle que :

$$\int_{\mathbb{R}} P_i(x) P_{i'}(x) d\nu(x) = \delta_{ii'},$$

avec $\delta_{ii'} = 1$ si $i = i'$ et $\delta_{ii'} = 0$ sinon. Si $\mathcal{X}$ est borné, nous pouvons par exemple prendre la mesure de Lebesgue et les polynômes de Legendre associés. Si le support des $X_{(i)}$ est $\mathbb{R}$, nous pouvons nous munir de la mesure ayant pour densité la loi normale par rapport à la mesure de Lebesgue, et prendre les polynômes de Hermite associés. Les premiers polynômes de Hermite

orthonormés par rapport à la loi normale standard sont les suivants :

$$\begin{aligned} P_0(x) &= 1, & P_1(x) &= x \\ P_2(x) &= x^2 - 1, & P_3(x) &= x^3 - 3x \\ P_4(x) &= x^4 - 6x^2 + 3 & P_5(x) &= x^5 - 10x^3 + 15x. \end{aligned}$$

Supposons maintenant que $\mathcal{X}$ est inclu dans $\mathbb{R}^d$. Nous notons $x = (x_1, x_2, \ldots, x_d) \in \mathbb{R}^d$. Nous devons choisir une mesure de référence μ telle que la loi des $X_{(i)}$ soit dans $L^2_\mu(\mathbb{R}^d)$, et nous munir d'une base orthogonale associée $\{Q_j, j \in \mathbb{N}^d\}$:

$$\int_{\mathbb{R}^d} Q_j(x) Q_{j'}(x) d\mu(x) = \delta_{jj'}.$$

Nous pouvons simplement prendre :

$$\begin{aligned} \mu(dx) &= \nu(dx_1)\nu(dx_2)\cdots\nu(dx_d) \\ Q_j(x) &= P_{j_1}(x_1)P_{j_2}(x_2)\cdots P_{j_d}(x_d), \end{aligned}$$

avec $j = (j_1, \cdots, j_d)$. Nous notons $|j| = j_1 + j_2 + \cdots + j_d$ et C_k le cardinal de l'ensemble $\{j \in \mathbb{N}^d; |j| \leq k\}$. Nous avons, en utilisant cette base orthogonale,

$$f_\theta = \sum_{j \in \mathbb{N}^d} a_j Q_j \qquad \text{et} \qquad f_{\theta'} = \sum_{j \in \mathbb{N}^d} b_j Q_j,$$

avec

$$a_j = \int Q_j(x) f_\theta(x) d\mu(x) \qquad \text{et} \qquad b_j = \int Q_j(x) f_{\theta'}(x) d\mu(x).$$

Etant donné que nous observons des échantillons i.i.d. $X_{(1)}, X_{(2)}, \ldots, X_{(n)}$ et $Z_{(1)}, Z_{(2)}, \ldots, Z_{(n)}$ suivant f_θ et $f_{\theta'}$ respectivement, nous pouvons estimer a_j et b_j par :

$$\widehat{a}_j = \frac{1}{n} \sum_{i=1}^{n} Q_j(X_{(i)}) \qquad \text{et} \qquad \widehat{b}_j = \frac{1}{n} \sum_{i=1}^{n} Q_j(Z_{(i)}).$$

Enfin, comme les séquences $(a_j)_{j \in \mathbb{N}^d}$ et $(b_j)_{j \in \mathbb{N}^d}$ tendent vers 0, nous pouvons approximer les densités f_θ et $f_{\theta'}$ par les estimateurs d'ordre k suivants, k arbitrairement fixé :

$$f_\theta(x) \approx \sum_{|j| \leq k} \widehat{a}_j Q_j(x) \qquad \text{et} \qquad f_{\theta'}(x) \approx \sum_{|j| \leq k} \widehat{b}_j Q_j(x).$$

Nous notons $U_k = (\widehat{a}_1 - \widehat{b}_1, \ldots, \widehat{a}_{C_k} - \widehat{b}_{C_k})$. Asymptotiquement, si les deux échantillons (observés et simulés) viennent d'une même population, leurs densités sont égales et nous avons

la convergence en loi suivante quand $n \to \infty$:

$$\sqrt{n}U_k \to N(0, V_k),$$

avec V_k la matrice de covariance de $\sqrt{n}U_k$, Nous avons alors, pour tout estimateur convergent $\widehat{V_k}$ de V_k :

$$T_k = nU_k^t \widehat{V_k}^{-1} U_k \to \chi^2_{C_k},$$

Il est important de bien choisir l'ordre k utilisé pour les approximations de f et g. D'après D'Agostino et Stephens [53] et Rayner et Best [182], il est raisonnable de prendre k entre 1 et 4, suivant la taille de notre échantillon. Cependant, imposer un nombre de dimensions à conserver n'est pas toujours pertinent, voir Inglot et al. [102]. Ledwina [128] a alors proposé de choisir k de manière automatique en maximisant un critère pénalisé de type Schwartz (méthode étendue par Kallenberg et Ledwina [111] et par Inglot et al. [103], ainsi que par Janic-Wróblewska et Ledwina [105] et Ghattas et al. [79] dans un contexte de comparaison de deux échantillons). En nous inspirant de ces méthodes, nous choisissons k_{opt} maximisant l'ensemble $\{T_k - C_k \log(n); k = 1, \ldots, K\}$, en ayant fixé K assez grand. Cette procédure est très facile à implémenter. L'inconvénient est qu'elle nécessite le calcul de toutes les statistiques $T_1, \ldots, T_K$ (bien qu'en pratique il est souvent suffisant de prendre $K = 4$ ou 5).
Une version d'un algorithme ABC utilisant cette approche est alors :

ABC semi-automatique

Pour obtenir N simulations.
Le n-échantillon observé est $(x_{(1)}, x_{(2)}, \ldots, x_{(n)})$. On a fixé K.
Pour i allant de 1 à N:

1. Générer θ' suivant sa distribution a priori $\pi(.)$.
2. Générer $(z_{(1)}, z_{(2)}, \ldots, z_{(n)})$ sachant θ', suivant le modèle défini par $f(. \mid \theta')$.
3. Prendre k_{opt} maximisant l'ensemble $\{T_k - C_k \log(n); k = 1, \ldots, K\}$. On a $T_{k_{opt}}$.

Si $T_{k_{opt}} < \epsilon$, poser $\theta_i = \theta'$. Sinon retourner en 1.

Pour déterminer le seuil de tolérance ϵ à utiliser, nous utilisons la proposition suivante :

Proposition 8.3.1 *Sous l'hypothèse nulle d'égalité des densités f_θ et $f_{\theta'}$, $T_{k_{opt}}$ converge en loi vers une χ^2_1 lorsque $n \to \infty$.*

Preuve : Il est clair que $T_k \to \chi^2_{C_k}$. Nous montrons que $k_{opt} \to 1$, soit que $P(k_{opt} \geq 2) \to 0$

quand $n \to \infty$. Nous avons :

$$\begin{aligned} P(k_{opt} \geq 2) &= P\Big(\exists k, 2 \leq k \leq K : T_k - C_k \log(n) > T_1 - \log(n)\Big) \\ &= P\Big(\exists k, 2 \leq k \leq K : T_k - T_1 > (C_k - 1)\log(n)\Big) \\ &= P\Big(\exists k, 2 \leq k \leq K : T_k > (C_k - 1)\log(n)\Big) \\ &\leq \sum_{k=2}^{K} P\Big(T_k \geq (C_k - 1)\log(n)\Big) \\ &\leq \sum_{k=2}^{K} \frac{\mathbb{E}(T_k)}{(C_k - 1)\log(n)}, \end{aligned}$$

qui tend vers 0. Pour passer de la deuxième à la troisième ligne nous utilisons le fait que $T_1 > 0$. Pour passer de la quatrième à la cinquième, nous utilisons l'inégalité de Markov.

■

Nous pouvons donc prendre pour le seuil ϵ un fractile d'une χ^2_1 d'un niveau approprié.

Cas de données dépendantes

Soit $\{Y_{(t)}; t \in \mathbb{Z}\}$ un processus réel mesurable défini sur un espace probabilisé muni de la probabilité P. Notons $\mathcal{F}_u^v$ la tribu générée par $\{Y_{(t)}; u \leq t \leq v\}$. Le processus est alpha-mélangeant (ou fortement mélangeant) si (voir Rosenblatt [190]) :

$$\alpha(n) = \sup_{s \in \mathbb{R}} \sup_{A \in \mathcal{F}_{-\infty}^{s}, B \in \mathcal{F}_{s+n}^{\infty}} |P(A \cap B) - P(A)P(B)| \underset{n \to +\infty}{\longrightarrow} 0.$$

Supposons que les données observées sont issues d'un processus à valeurs dans $\mathbb{R}$ stationnaire et alpha-mélangeant $\{X_{(t)}; t \in \mathbb{Z}\}$, tel que la fonction de répartition de $X_{(t)}$ vaut F_0 connue pour tout t. Nous voulons savoir si un processus simulé $\{Z_{(t)}; t \in \mathbb{Z}\}$ a la même fonction de répartition pour les $Z_{(t)}$, afin de savoir si des paramètres proches ont été utilisés pour obtenir ces deux processus. Nous proposons pour cela d'utiliser la statistique de test développée par Munk et al. [162] :

$$N_k = (12\hat{\sigma}^2)^{-1} \sum_{j=1}^{k} \Big[\sum_{i=1}^{n} n^{-1/2} \phi_j(Z_{(i)}) \Big]^2, \quad k = 1, 2, \ldots,$$

avec ϕ_j le j^e polynôme de Legendre, et $\hat{\sigma}^2$ un estimateur consistant de :

$$\sigma^2 = \sum_{t=-\infty}^{+\infty} cov(Z_{(0)}, Z_{(t)}).$$

Sous l'hypothèse nulle que la fonction de répartition pour les $Z_{(t)}$ est une $\mathcal{U}_{[0,1]}$ (nous pouvons toujours nous ramener à ce cas en utilisant les $F_0(Z_{(t)})$ au lieu des $Z_{(t)}$), N_k suit une combinaison

linéaire de k distributions χ^2_1. Donc pour $k=1$, N_k suit une χ^2_1. Comme précédemment, une difficulté est de choisir l'ordre k. Comme Munk et al. [162], nous proposons de choisir l'ordre k_{opt} maximisant l'ensemble $\{N_k - k\log(n); k=1,\ldots,K\}$, en ayant fixé K assez grand. Pour le choix du seuil de tolérance, nous pouvons utiliser la proposition suivante.

Proposition 8.3.2 *Supposons que $\sigma^2 > 0$, qu'il existe $a > 0$ et $o < \rho < 1$ tels que $\alpha(n) \leq a\rho^n$, et que $\mathbb{E}|X^{\gamma}_{(t)}| < \infty$ pour un $\gamma < 2$. Alors sous l'hypothèse nulle que la distribution marginale de $\{Z_{(t)}; t \in \mathbb{Z}\}$ est une $\mathcal{U}_{[0,1]}$, $N_{k_{opt}}$ converge en loi vers une χ^2_1 lorsque $n \to \infty$.*

Preuve : Comme dans la preuve de la proposition 8.3.1, nous voulons montrer que $P(k_{opt} \geq 2) \to 0$ quand $n \to \infty$. Nous notons :

$$N_{j^*} = \Big[\sum_{i=1}^{n} n^{-1/2}\phi_j(Z_{(i)})\Big]^2, \quad \text{soit} \quad N_k = (12\hat{\sigma}^2)^{-1}\sum_{j=1}^{k} N_{j^*}.$$

Nous avons :

$$\begin{aligned}
P(k_{opt} \geq 2) &= P\Big(\exists k, 2 \leq k \leq K : N_k - k\log(n) > N_1 - \log(n)\Big) \\
&= P\Big(\exists k, 2 \leq k \leq K : N_k - N_1 > (k-1)\log(n)\Big) \\
&= P\Big(\exists j, 2 \leq j \leq K : N_{j^*} > (12\hat{\sigma}^2)\log(n)\Big) \\
&\leq \sum_{j=2}^{K} P\Big(N_{j^*} > (12\hat{\sigma}^2)\log(n)\Big) \\
&\leq \sum_{j=2}^{K} \frac{\mathbb{E}(N_{j^*})}{(12\hat{\sigma}^2)\log(n)}.
\end{aligned}$$

Pour passer de la deuxième à la troisième ligne nous remarquons que si $\forall j, 2 \leq j \leq K$ nous avons $N_{j^*} \leq (12\hat{\sigma}^2)\log(n)$, alors :

$$(12\hat{\sigma}^2)(N_k - N_1) = \sum_{j=2}^{k} N_{j^*} \leq (k-1)(12\hat{\sigma}^2)\log(n),$$

soit $N_k - N_1 < (k-1)\log(n)$, ce qui n'est pas possible. Pour passer de la quatrième à la cinquième ligne, nous utilisons l'inégalité de Markov. Grâce à la stationnarité, nous avons :

$$\mathbb{E}(N_{k^*}) = \frac{1}{n}\sum_{ij}\mathbb{E}(\phi_k(Z_{(i)}))\mathbb{E}(\phi_k(Z_{(j)})) \leq 2\sum_{i=0}^{n}\Big|cov\Big(\phi_k(Z_{(0)}, \phi_k(Z_{(i)})\Big)\Big|.$$

Et en utilisant un résultat de Rio [185], il existe $0 < C < \infty$ tel que :

$$\left| cov\Big(\phi_k(Z_{(0)}, \phi_k(Z_{(i)})\Big) \right| \leq C \log(k).$$

Alors,

$$P(k_{opt} \geq 2) \leq \sum_{j=2}^{K} \frac{C \log(k)}{(12\hat{\sigma}^2) \log(n)},$$

qui tend vers 0.

■

Une limite importante de ce résultat est que nous comparons la distribution marginale de $\{Z_{(t)}; t \in \mathbb{Z}\}$ à la distribution marginale de $\{X_{(t)}; t \in \mathbb{Z}\}$ supposée connue. Par conséquent, il apparaît nécessaire d'étendre le résultat au cas où la distribution marginale de $\{X_{(t)}; t \in \mathbb{Z}\}$ n'est pas connue, ce qui est souvent le cas en pratique. Voir par exemple les travaux de Reboul et Yao (2011, en cours).

Conclusion générale

La quantité croissante de données en biologie, et notamment en génomique, nécessite le développement de méthodes statistiques adaptées. En particulier, les nouvelles technologies comme les puces microarray ou le séquençage haut-débit génèrent des dizaines, voire des centaines de milliers de variables par individu. Que ce soit dans une optique de prédiction, de diagnostic, ou de compréhension de mécanismes biologiques, il est alors nécessaire de sélectionner les variables les plus pertinentes. De nombreuses méthodes ont été développées, nous en avons évoquées un certain nombre dans le chapitre 1. Cependant, au début de cette thèse la société Ipsogen ne disposait pas d'une méthode facile à mettre en oeuvre et permettant de prendre en compte des effets aléatoires. Nous avons donc apporté notre contribution en développant une méthode bayésienne de sélection de variables dans un modèle probit mixte, qui a été présentée dans le chapitre 2. En appliquant cette méthode à différentes études, l'équipe bioinformatique d'Ipsogen a été confrontée au problème de variables quasi-colinéaires, et à la singularité computationnelle d'une matrice nécessitant d'être inversée. Nous avons alors étendu la méthode précédente au cas de variables combinaisons linéaires d'autres variables (voir le chapitre 3). En effet, à notre connaissance aucune méthode n'avait été proposée pour solutionner spécifiquement ce problème. Finalement, dans la première partie de cette thèse nous avons développé une méthode de sélection de variables facile d'utilisation, considérant à la fois la présence d'effets aléatoires et la présence de variables combinaisons linéaires d'autres variables. Elle a répondu aux attentes de la société Ipsogen qui l'utilise fréquemment grâce à un package R qui a été réalisé.

Cette méthode bayésienne de sélection de variables nécessite d'échantillonner suivant la distribution a posteriori des paramètres du modèle utilisé. Cela nous a amené à considérer les différents échantillonneurs possibles, notamment les méthodes MCMC existantes ainsi que l'Equi-Energy Sampler [118], que nous avons présentés dans le chapitre 5. L'étude de ces méthodes nous a inspiré un nouvel algorithme MCMC, que nous avons appelé Parallel Tempering with Equi-Energy Moves, et que nous avons développé dans le chapitre 6. Les performances de cet algorithme sont satisfaisantes et son champ d'application est très vaste. Nous l'avons par exemple appliqué sur un problème de recherche de facteurs de transcription, ou encore dans le cadre d'une étude de la sensibilité de *Plasmodium Falciparum* à la doxycycline. Cependant nous savons que dans certains problèmes, et notamment en biologie ou génétique, la vraisemblance des données n'est

pas disponible sous forme explicite. La méthode développée dans le chapitre 6 ne peut donc pas être utilisée dans ces cas là. En étudiant en détails les méthodes sans vraisemblance existantes, que nous avons présentées dans le chapitre 7, nous avons remarqué qu'aucune n'utilisait un principe similaire à celui du Parallel Tempering [76]. Cela peut s'expliquer par l'engouement actuel pour les méthodes sans vraisemblance séquentielles, qui jusqu'à présent apparaissaient plus performantes que les méthodes sans vraisemblance générant des chaînes de Markov. Or à l'époque où le Parallel Tempering a été développé, le principe d'avoir plusieurs chaînes en parallèle inter-agissant grâce à des mouvements d'échanges a apporté une amélioration par rapport aux échantillonneurs existants à ce moment. Nous avons alors préssenti que l'utilisation d'un tel principe pouvait améliorer les méthodes sans vraisemblance existantes basées sur la théorie des chaînes de Markov. Nous avons alors développé l'algorithme ABC-PT, présenté dans le chapitre 8. Effectivement, nous avons vérifié que cet algorithme est plus performant que l'algorithme ABC-MCMC, ou que l'Algorithme ABC-MCMC augmenté. Bien que nous ne pouvons pas prétendre qu'il soit plus performant que les méthodes sans vraisemblance séquentielles, nous avons pu tout de même apprécier la capacité de l'ABC-PT à détecter de petits modes dans les distributions a posteriori cibles. En définitive, dans la seconde partie de cette thèse nous avons proposé deux nouvelles méthodes d'échantillonnage MCMC performantes, dont une ne nécessitant pas le calcul de la vraisemblance.

Les perspectives des trois méthodes développées dans cette thèse sont nombreuses et ont été présentées dans les parties 4.3, 6.3 et 8.3. Certaines peuvent être approfondies à court terme.
En particulier, concernant les méthodes de sélection de variables, il serait particulièrement intéressant d'étudier l'influence de la taille des groupes de variables liées, et de comparer la méthode développée avec les méthodes fréquentistes de type LASSO ou Elastic Net. De plus, la méthode de sélection de variables dans un mélange de modèles à temps de sortie accéléré mixtes développée en annexe B peut être appliquée telle quelle, seuls des jeux de données pertinents sont nécessaires.
En ce qui concerne les échantillonneurs PTEEM et ABC-PT, des méthodes permettant de tirer partie des simulations des chaînes autres que la chaîne d'intérêt pourraient apporter beaucoup en termes de précision et de temps de calcul. De même, pour ces deux échantillonneurs, proposer d'ajuster automatiquement les températures et/ou les niveaux d'énergie et les seuils de tolérance au fur et à mesure des itérations éviterait les difficultés rencontrées en cas de mauvaise calibration. Enfin, un article appliqué utilisant l'échantillonneur PTEEM est en préparation, approfondissant l'étude de la sensibilité de *Plasmodium Falciparum* à la doxycycline.
Pour finir, nous souhaitons proposer une manière semi-automatique de construire des statistiques utilisables dans les méthodes sans vraisemblance, lorsque peu d'information est disponible sur la forme de la distribution a posteriori cible, et notamment lorsque les données observées sont issues d'un processus.

Bibliographie

[1] F. Al-Awadhi, M. Hurn, and C. Jennison. Improving the acceptance rate of reversible jump MCMC proposals. *Statistics and Probability Letters*, 69 :189–198, 2004.

[2] J. Albert and S. Chib. Bayesian analysis of binary and polychotomous response data. *Journal of the American Statistical Association*, 88(422) :669–679, 1993.

[3] D. F. Andrews and C. L. Mallows. Scale mixtures of normal distributions. *Journal of the Royal Statistical Society B*, 36 :99–102, 1974.

[4] C. Andrieu, A. Jasra, A. Doucet, and P. Del Moral. Convergence of the Equi-Energy sampler. *ESAIM : Proceedings*, 19 :1–5, 2007.

[5] C. Andrieu, A. Jasra, A. Doucet, and P. Del Moral. Non-linear Markov chain Monte Carlo. *ESAIM : Proceedings*, 19 :79–84, 2007.

[6] C. Andrieu, A. Jasra, A. Doucet, and P. Del Moral. A note on convergence of the Equi-Energy sampler. *Stochastic Analysis and Applications*, 26(2) :298–312, 2008.

[7] Y. Atchadé. Resampling from the past to improve on MCMC algorithms. *Far East Journal of Theoretical Probability*, 27(1) :81–99, 2009.

[8] Y. Atchadé. A cautionary tale on the efficiency of some adaptive Monte Carlo schemes. *Annals of Applied Probability*, 20(3) :841–868, 2010.

[9] Y. Atchadé, G. Fort, E. Moulines, and P. Priouret. *Inference and learning in dynamic models*, chapter Adaptive Markov chain Monte Carlo : theory and methods, pages 33–53. Cambridge University Press, 2011.

[10] Y. Atchadé and J. Liu. Discussion of Equi-Energy sampler by Kou, Zhou and Wong. *The Annals of Statistics*, 34(4) :1620–1628, 2006.

[11] Y. Atchadé and J. Liu. The Wang-Landau algorithm for Monte Carlo computation in general state spaces. *Statistica Sinica*, 20 :209–233, 2010.

[12] Y. Atchadé, G. Roberts, and S. Rosenthal. Towards optimal scaling of Metropolis-coupled Markov chain Monte Carlo. *Statistics and Computing*, in press, 2010.

[13] K. Athreya, H. Doss, and J. Sethuraman. On the convergence of the Markov chain simulation method. *The Annals of Statistics*, 24(1) :69–100, 1996.

[14] K. Bae and B. Mallick. Gene selection using a two-level hierarchicalBayesian model. *Bioinformatics*, 20(18) :3423–3430, 2004.

[15] M. Baragatti and D. Pommeret. Comment on "Bayesian variable selection for disease classification using gene expression data". *Bioinformatics*, 27(8) :1194, 2011.

[16] E. Bazin, K. Dawson, and M. Beaumont. Likelihood-free inference of population structure and local adaptation in a Bayesian hierarchical model. *Genetics*, 185(2) :587–602, 2010.

[17] M. Beaumont. Approximate Bayesian computation in evolution and ecology. *Annual Review of Ecology Evolution and Systematics*, 41 :379–406, 2010.

[18] M. Beaumont, J. Cornuet, J. M. Marin, and C. Robert. Adaptive approximate Bayesian computation. *Biometrika*, 96(4) :983–990, 2009.

[19] M. Beaumont, W. Zhang, and D. Balding. Approximate Bayesian computation in population genetics. *Genetics*, 162 :2025–2035, 2002.

[20] E. Bedrick, R. Christensen, and W. Johnson. Bayesian accelerated failure time analysis with application to veterinary epidemiology. *Statistics in Medicine*, 19 :221–237, 2000.

[21] G. Behrens, N. Friel, and M. Hurn. Tuning tempered transitions. *Statistics and Computing*, in press, 2011.

[22] B. Berg and T. Neuhaus. Multicanonical ensemble : A new approach to simulate first-order phase transitions. *Physical Review Letters*, 68 :9–12, 1992.

[23] J. Berger and L. Pericchi. The intrinsic Bayes factor for model selection and prediction. *Journal of the American Statistical Association*, 91(433) :109–122, 1996.

[24] D. Bertsimas and J. Tsitsiklis. Simulated annealing. *Statistical Science*, 8 :10–15, 1993.

[25] M. Blum. Approximate Bayesian computational : a non-parametric perspective. *Journal of the American Statistical Association*, 491 :1178–1187, 2010.

[26] M. Blum and O. François. Non-linear regression models for Approximate Bayesian computation. *Statistics and Computing*, 20(1) :63–73, 2010.

[27] P. Bortot, S. Coles, and S. Sisson. Inference for stereological extremes. *Journal of the American Statistical Association*, 102 :84–92, 2007.

[28] L. Bottolo and S. Richardson. Evolutionary stochastic search for Bayesian model exploration. *Bayesian Analysis*, 5(3) :583–618, 2010.

[29] S. Brooks, P. Giudici, and G. Roberts. Efficient construction of reversible jump Markov chain Monte Carlo proposal distributions (with discussion). *Journal of the Royal Statistical Society B*, 65 :3–55, 2003.

[30] P. Brown, M. Vannucci, and T. Fearn. Multivariate Bayesian variable selection and prediction. *Journal of the Royal Statistical Society B*, 60(3) :627–641, 1998.

[31] B. Cai and D. Dunson. Bayesian covariance selection in generalized linear mixed models. *Biometrics*, 62 :446–457, 2006.

[32] O. Cappé, A. Guillin, J. Marin, and C. Robert. Population Monte Carlo. *Journal of Computational and Graphical Statistics*, 13(4) :907–929, 2004.

[33] O. Cappé, C. Robert, and T. Rydén. Reversible jump, birth-and-death and more general continuous time Markov chain Monte Carlo samplers. *Journal of the Royal Statistical Society B*, 65(3) :679–700, 2003.

[34] G. Celeux. Bayesian inference for mixtures : the label-switching problem. In *COMPSTAT 98 - Proceedings in Computational Statistics, Physica-Verlag*, pages 227–232, 1998.

[35] G. Celeux, M. El Anbari, J. M. Marin, and C. Robert. Regularization in regression : comparing Bayesian and frequentist methods in a poorly informative situation. *arXiv :1010.0300,v2*, 2011.

[36] G. Celeux, M. Hurn, and C. Robert. Computational and inferential difficulties with mixture posterior distributions. *Journal of the American Statistical Association*, 95(451) :957–970, 2000.

[37] G. Celeux, J. M. Marin, and C. Robert. Sélection bayésienne de variables en régression linéaire. *Journal de la Société Française de Statistique*, 147 :59–79, 2006.

[38] M. Chen and D. Dey. Variable selection for multivariate logistic regression models. *Journal of Statistical Planning and Inference*, 111 :37–55, 2003.

[39] Z. Chen and D. Dunson. Random effects selection in linear mixed models. *Biometrics*, 59 :762–769, 2003.

[40] W. Cheng, M. Tsai, C. Chang, C. Huang, C. Chen, W. Shu, Y. Lee, T. Wang, J. Hong, C. Li, and I. Hsu. Microarray meta-analysis database (M2DB) : a uniformly pre-processed, quality controlled, and manually curated human clinical microarray database. *BMC Bioinformatics*, 11, 2010.

[41] H. Chipman, E. George, and R. McCulloch. The practical implementation of Bayesian model selection. In *Model selection - IMS Lecture Notes*. P. LAHIRI. Institute of Mathematical Statistics, 2001.

[42] N. Chopin. A sequential particle filter method for static models. *Biometrika*, 89 :539–552, 2002.

[43] N. Chopin. Inference and model choice for time-ordered hidden Markov models. *Journal of the Royal Statistical Society B*, 69(2) :269–284, 2007.

[44] N. Chopin. Fast simulation of truncated Gaussian distributions. *Statistics and Computing*, 21(2) :275–288, 2011.

[45] R. Christensen and W. Johnson. Modelling accelerated failure time with a Dirichlet process. *Biometrika*, 75 :693–704, 1988.

[46] H. Chung, E. Loken, and J. Schafer. Difficulties in drawing inferences with finite mixture models. *The American Statistician*, 58 :152–158, 2004.

[47] D. Cimino, L. Fuso, C. Sfiligoi, N. Biglia, R. Ponzone, F. Maggiorotto, G. Russo, L. Cicatiello, A. Weisz, D. Taverna, P. Sismondi, and M. De-Bortoli. Identification of new genes associated with breast cancer progression by gene expression analysis of predefined sets of neoplastic tissues. *International Journal of Cancer*, 123(6) :1327–1338, 2008.

[48] J. Cornuet, J. Santos, M. Beaumont, C. Robert, J. Marin, D. Balding, T. Guillemaud, and A. Estoup. Inferring population history with DIYABC : a user-friendly approach to approximate Bayesian computation. *Bioinformatics*, 24 :2713–2719, 2008.

[49] D. Cox. Regression models and life-tables (with discussion). *Journal of the Royal Statistical Society B*, 34 :187–220, 1972.

[50] G. Crooks, G. Hon, J. Chandonia, and S. Brenner. WebLogo : a sequence logo generator. *Genome Research*, 14 :1188–1190, 2004.

[51] K. Csilléry, M. Blum, O. Gaggiotti, and O. François. Approximate Bayesian computation (ABC) in practice. *Trends in Ecology and Evolution*, 25 :490–491, 2010.

[52] W. Cui and E. George. Empirical Bayes vs. fully Bayes variable selection. *Journal of Statistical Planning and Inference*, 138(4) :888–900, 2008.

[53] R. D'Agostino and M. Stephens. *Goodness-of-fit techniques*. Marcel Dekker, New York, 1986.

[54] P. Del Moral, A. Doucet, and A. Jasra. Sequential Monte Carlo samplers. *Journal of the Royal Statistical Society B*, 68(3) :411–436, 2006.

[55] P. Del Moral, A. Doucet, and A. Jasra. An adaptive sequential Monte Carlo Method for approximate Bayesian computation. *Statistics and Computing*, in press, 2011.

[56] P. Dellaportas, J. Forster, and I. Ntzoufras. On Bayesian model and variable selection using MCMC. *Statistics and computing*, 12 :27–36, 2002.

[57] L. Devroye. *Non uniform random variate generation*. Springer, 1986.

[58] J. Díaz-García and R. Ramos-Quiroga. Generalised natural conjugate prior densities : singular multivariate linear model. Technical report, Univesidad Autónoma Agraria Antonio Narro and Centro de Investigación en Matemáticas, 2003.

[59] J. Diebolt and C. Robert. Estimation of finite mixture distributions through Bayesian sampling. *Journal of the Royal Statistical Society B*, 56 :363–375, 1994.

[60] R. Douc, A. Guillin, J. M. Marin, and C. Robert. Convergence of adaptive mixtures of importance sampling schemes. *Annals of Statistics*, 35(1) :420–448, 2007.

[61] C. Drovandi and A. Pettitt. Estimation of parameters for macroparasite population evolution using approximated Bayesian computation. *Biometrics*, 67 :225–233, 2011.

[62] R. Edgar, M. Domrachev, and A. Lash. Gene Expression Omnibus : NCBI gene expression and hybridization array data repository. *Nucleic Acids Research*, 30(1) :207–210, 2002.

[63] P. Fearnhead and D. Prangle. Constructing summary statistics for approximate Bayesian computation : semi-automatic ABC. *arxiv :1004.1112,v2*, 2011.

[64] S. Filippi, C. Barnes, and M. Stumpf. On optimal kernels for ABC SMC. *arXiv :1106.6280,v2*, 2011.

[65] G. Fort, E. Moulines, and P. Priouret. Convergence of adaptive MCMC algorithms : ergodicity and law of large numbers. Technical report, 2010.

[66] D. Fouskakis, I. Ntzoufras, and D. Draper. Population-based reversible jump Markov chain Monte Carlo methods for Bayesian variable selection and evaluation under cost limit restrictions. *Journal of the Royal Statistical Society C*, 58(3) :483–503, 2009.

[67] B. Fridley. Bayesian variable and model selection methods for genetic association studies. *Genetic Epidemiology*, 33 :27–37, 2009.

[68] S. Frühwirth-Schnatter. Markov chain Monte Carlo estimation of classical and dynamic switching and mixture models. *Journal of the American Statistical Association*, 96 :194–209, 2001.

[69] S. Frühwirth-Schnatter and H. Wagner. Bayesian variable selection for random intercept modeling of Gaussian and non-Gaussian data. In J. Bernardo et al., editors, *Bayesian Statistics 9*, Proceedings of the 9th Valencia International Meeting, pages 165–200. Oxford University Press, June 2010.

[70] A. Gelfand and A. Smith. Sampling-based approaches to calculating marginal densities. *Journal of the American Statistical Association*, 85(410) :398–409, 1990.

[71] A. Gelman. Prior distributions for variance parameters in hierarchical models (Comment on Article by Browne and Draper). *Bayesian Analysis*, 1(3) :515–534, 2006.

[72] A. Gelman and D. Rubin. Inference from iterative simulation using multiple sequences. *Statistical Science*, 7 :457–511, 1992.

[73] E. George and D. Foster. Calibration and empirical Bayes variable selection. *Biometrika*, 87(4) :731–747, 2000.

[74] E. George and R. McCulloch. Variable selection via Gibbs sampling. *Journal of the American Statistical Association*, 88(423) :881–889, 1993.

[75] E. George and R. McCulloch. Approaches for Bayesian variable selection. *Statistica Sinica*, 7 :339–373, 1997.

[76] C. Geyer. Markov chain Monte Carlo maximum likelihood. *Computing Science and Statistics : Proceedings of the 23rd Symposium on the interface*, pages 156–163, 1991.

[77] C. Geyer. Reweighting Monte Carlo mixtures. Technical Report 568, University of Minnesota, 1991.

[78] C. Geyer and E. Thompson. Annealing Markov chain Monte Carlo with applications to ancestral inference. *Journal of the American Statistical Association*, 90 :909–920, 1995.

[79] B. Ghattas, D. Pommeret, L. Reboul, and A. Yao. Smooth test for paired populations. *Journal of Statistical Planning and Inference*, 141 :262–275, 2010.

[80] W. Gilks, G. Roberts, and E. George. Adaptive direction sampling. *Statistician*, 43 :179–189, 1994.

[81] J. Gosh, A. Herring, and A. Siega-Riz. Bayesian variable selection for latent class models. *Biometrics*, in press, 2011.

[82] G. Goswami and J. Liu. On learning strategies for evolutionary Monte Carlo. *Statistics and Computing*, 17 :23–28, 2007.

[83] P. Green. Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika*, 82 :711–732, 1995.

[84] P. Green. *Highly structured stochastics systems*, chapter Trans-dimensional Markov Chain Monte Carlo, pages 179–196. 2003.

[85] P. Green and A. Mira. Delayed rejection in reversible jump Metropolis-Hastings. *Biometrika*, 88 :1035–1053, 2001.

[86] A. Grelaud. *Méthodes sans vraisemblance appliquées à l'étude de la sélection naturelle et à la prédiction de structure tridimensionnelle des protéines.* PhD thesis, Université Paris Dauphine, 2009.

[87] A. Grelaud, J. M. Marin, C. Robert, F. Rodolphe, and F. Tally. Likelihood-free methods for model choice in Gibbs random fields. *Bayesian Analysis*, 3(2) :427–442, 2009.

[88] R. Guo and P. Speckman. Bayes factor consistency in linear models. *The 2009 International Workshop on Objective Bayes Methodology, Philadelphia*, 2009.

[89] M. Gupta and J. Ibrahim. Variable selection in regression mixture modeling for the discovery of gene regulatory networks. *Journal of the American Statistical Association*, 102(479) :867–880, 2007.

[90] M. Gupta and J. Ibrahim. An information matrix prior for Bayesian analysis in generalized linear models with high dimensional data. *Statistica Sinica*, 19(4) :1641–1663, 2009.

[91] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, (46) :389–422, 2002.

[92] G. Hamilton, M. Currat, N. Ray, G. Heckel, M. Beaumont, and L. Excoffier. Bayesian estimation of recent migration rates after a spatial expansion. *Genetics*, 170 :409–417, 2005.

[93] C. Hans. Bayesian Lasso regression. *Biometrika*, 96(4) :835–845, 2009.

[94] W. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 88 :1035–1053, 1970.

[95] R. Holley, S. Kusukoa, and D. Stroock. Asymptotics of the spectral gap with applications to the theory of simulated annealing. *Journal of Functional Analysis*, 83 :333–347, 1989.

[96] C. Holmes and L. Held. Bayesian auxiliary variable models for binary and multinomial regression. *Bayesian Analysis*, 1(1), 2006 145-168.

[97] P. Hougaard. Fundamentals of survival data. *Biometrics*, 55 :13–22, 1999.

[98] P. Hougaard. *Analysis of multivariate survival data.* Springer-Verlag, New York, 2000.

[99] X. Hua and S. Kou. Convergence of the Equi-Energy sampler and its application to the Ising model. *Statistica Sinica*, 21(4), 2011.

[100] K. Hukushima and K. Nemoto. Exchange Monte Carlo Method and application to spin glass simulations. *Journal of the Physical Society of Japan*, 65 :1604–1608, 1996.

[101] M. Hurn, A. Justel, and C. Robert. Estimating mixtures of regressions. *Journal of Computational and Graphical Statistics*, 12 :55–79, 2003.

[102] T. Inglot, T. Jurlewicz, and T. Ledwina. On Neyman-type smooth tests of fit. *Statistics*, 21 :549–568, 1990.

[103] T. Inglot, W. Kallenberg, and T. Ledwina. Data-driven smooth tests for composite hypotheses. *The Annals of Statistics*, 25 :1222–1250, 1997.

[104] R. Irizarry, B. Hobbs, F. Collin, Y. Beazer-Barclay, K. Antonellis, U. Scherf, and T. Speed. Exploration, normalization, and summaries of high density oligonucleotide array probe level data. *Biostatistics*, 4(2) :249–264, 2003.

[105] A. Janic-Wróblewska and T. Ledwina. Data driven rank tests for two-sample problem. *The scandinavian Journal of Statistics*, 27 :281–297, 2000.

[106] A. Jasra, C. Holmes, and D. Stephens. Markov chain Monte Carlo methods and the label switching problem in Bayesian mixture modeling. *Statistical Science*, 20(1) :50–67, 2005.

[107] A. Jasra, D. Stephens, and C. Holmes. On population-based simulation for static inference. *Statistics and Computing*, 17 :263–279, 2007.

[108] A. Jasra, D. Stephens, and C. Holmes. Population-based reversible jump Markov chain Monte Carlo. *Biometrika*, 94 :787–807, 2007.

[109] S. Jensen, X. Liu, Q. Zhou, and J. Liu. Computational discovery of gene regulatory binding motifs : a Bayesian perspective. *Statistical Science*, 19 :188–294, 2004.

[110] J. D. Kalbfleisch and R. L. Prentice. *The statistical analysis of failure time data.* Wiley, Chichester, second edition, 2002.

[111] W. Kallenberg and T. Ledwina. Consistency and Monte Carlo simulation of a data driven version of smooth goodness-of-fit tests. *The Annals of Statistics*, 23 :1594–1608, 1995.

[112] R. Kass and A. Raftery. Bayes factors. *Journal of the American Statistical Association*, 90(430) :773–795, 1995.

[113] N. Keiding, P. Andersen, and J. Klein. The role of frailty models and accelerated failure time models in describing heterogeneity due to omitted covariates. *Statistics in Medicine*, 16 :215–225, 1997.

[114] S. Kim, M. Tadesse, and M. Vannucci. Variable selection in clustering via Dirichlet process mixture models. *Biometrika*, 93(4) :877–893, 2006.

[115] S. Kinney and D. Dunson. Fixed and random effects selection in linear and logistic models. *Biometrics*, 63 :690–698, 2007.

[116] S. Kirkpatrick, C. Gelatt, and M. Vecchi. Optimization by simulated annealing. *Science*, 220 :671–680, 1983.

[117] A. Komarek and E. Lesaffre. Bayesian accelerated failure time model for correlated interval-censored data with a normal mixture as error distribution. *Statistica Sinica*, 17 :549–569, 2007.

[118] S. Kou, Q. Zhou, and W. Wong. Equi-Energy sampler with application in statistical inference and statistical mechanics. *The Annals of Statistics*, 34(4) :1581–1619, 2006.

[119] L. Kuncheva. A stability index for feature selection. In *Proceedings of the 25th IASTED International Multi-Conference, Artificial Intelligence and Applications.*, Innsbruck, Austria, February 2007.

[120] L. Kuo and B. Mallick. Bayesian semiparametric inference for the accelerated failure time model. *Canadian Journal of Statistics*, 25 :457–472, 1997.

[121] L. Kuo and B. Mallick. Variable selection for regression models. *Sankhya, series B*, 60(1) :65–81, 1998.

[122] D. Kwon, M. Landi, M. Vannucci, H. Isaaq, D. Prieto, and R. Pfeiffer. An efficient stochastic search for Bayesian variable selection with high-dimensional correlated predictors. *Computational Statistics and Data Analysis*, 55 :2807–2818, 2011.

[123] M. Kyung, J. Gill, and G. Casella. Penalized regression, standard errors, and Bayesian Lassos. *Bayesian Analysis*, 5(2) :369–412, 2010.

[124] P. Lambert, D. Collett, A. Kimber, and R. Johnson. Parametric accelerated failure time models with random effects and an application to kidney transplant survival. *Statistics in Medicine*, 23 :3177–3192, 2004.

[125] D. Lamnisos, J. Griffin, and M. Steel. Transdimensional sampling algorithms for Bayesian variable selection in classification problems with many more variables than observations. *Journal of Computational and Graphical Statistics*, 18(3) :592–612, 2009.

[126] C. Lawrence, S. Altschul, M. Boguski, J. Liu, and A. Neuwald. Detecting subtle sequence signals : a Gibbs sampling strategy for multiple alignment. *Science*, 262 :208–214, 1993.

[127] C. Lawrence and A. Reilly. An expectation maximization (EM) algorithm for the identification and characterization of common sites in unaligned biopolymer sequences. *PROTEINS : Structure, Fonction and Genetics*, 7 :41–51, 1990.

[128] T. Ledwina. Data-driven version of Neyman's smooth test of fit. *Journal of the American Statistical Association*, (89) :1000–1005, 1994.

[129] K. Lee, S. Chakraborty, and J. Sun. Bayesian variable selection in semiparametric proportional hazards model for high dimensional survival data. *The International Journal of Biostatistics*, 7(1), 2011.

[130] K. Lee and B. Mallick. Bayesian methods for variable selection in survival models with application to DNA microarray data. *Sankhya.*, 66 :756–778, 2004.

[131] K. Lee, N. Sha, E. Dougherty, M. Vannucci, and B. Mallick. Gene selection : a Bayesian variable selection approach. *Bioinformatics*, 19(1) :90–97, 2003.

[132] A. Legarra, C. Robert-Granié, P. Croiseau, F. Guillaume, and S. Fritz. Improved Lasso for genomic selection. *Genetics Research*, 93(1) :77–87, 2011.

[133] C. Leuenberger, D. Wegmann, and L. Excoffier. Bayesian computation and model selection in population genetics. *Genetics*, 184 :243–252, 2010.

[134] F. Li and N. Zhang. Bayesian variable selection in structured high-dimensional covariate spaces with applications in genomics. *Journal of the American Statistical Association*, 105(491) :1202–1214, 2010.

[135] J. Li, K. Das, G. Fu, R. Li, and R. Wu. The Bayesian Lasso for genome-wide association studies. *Bioinformatics*, 27(4) :516–523, 2011.

[136] Q. Li and N. Lin. The Bayesian Elastic Net. *Bayesian Analysis*, 5(1) :151–170, 2010.

[137] F. Liang, R. Paulo, G. Molina, M. Clyde, and J. Berger. Mixtures of g priors for Bayesian variable selection. *Journal of the American Statistical Association*, 103(481) :410–423, 2008.

[138] F. Liang and W. Wong. Evolutionary Monte Carlo sampling : applications to Cp model sampling and change-point problem. *Statistica Sinica*, 10 :317–342, 2000.

[139] F. Liang and W. Wong. Real-parameter evolutionary Monte Carlo with applications to Bayesian Mixture models. *Journal of the American Statistical Association*, 96 :653–666, 2001.

[140] C. Liedtke, C. Hatzis, W. Symmans, C. Desmedt, H.-K. B., V. Valero, H. Kuerer, G. Hortobagyi, M. Piccart-Gebhart, C. Sotiriou, and L. Pusztai. Genomic Grade Index is associated with response to chemotherapy in patients with breast cancer. *Journal of Clinical Oncology*, 27(19) :3185–3191, 2009.

[141] J. Liu. The collapsed Gibbs sampler in Bayesian computations with application to a gene regulation problem. *Journal of the American Statistical Association*, 89(427) :958–966, 1994.

[142] J. Liu. *Monte Carlo strategies in scientific computing*. Springer, 2001.

[143] J. Liu, A. Neuwald, and C. Lawrence. Bayesian models for multiple local sequence alignment and Gibbs sampling strategies. *Journal of the American Statistical Association*, 90(432) :1156–1170, 1995.

[144] J. Liu, W. Wong, and A. Kong. Covariance structure and convergence rate of the Gibbs sampler with applications to the comparisons of estimators and augmentation schemes. *Biometrika*, 81 :27–40, 1994.

[145] X. Liu, D. Brutlag, and J. Liu. Bioprospector : discovering conserved DNA motifs in upstream regulatory regions of co-expressed genes. *Pacific Symposium on Biocomputing*, 6 :127–138, 2001.

[146] D. Madigan and A. Raftery. Model selection and accounting for model uncertainty in graphical models using Occam's window. *Journal of the American Statistical Association*, 89 :1535–1546, 1994.

[147] N. Madras and Z. Zheng. On the swapping algorithm. *Random Structures and Algorithms*, 22(1) :66–97, 2003.

[148] J. M. Marin, K. Mengersen, and C. Robert. *Handbook of Statistics 25*, chapter Bayesian modelling and inference on mixtures of distributions. 2005.

[149] J. M. Marin, P. Pudlo, C. Robert, and R. Ryder. Approximate Bayesian computational methods. *arxiv :1101.0955,v2*, 2011.

[150] J. M. Marin and C. Robert. *Bayesian core : a practical approach to computational Bayesian statistics*. New-York, Springer-Verlag edition, 2007.

[151] E. Marinari and G. Parisi. Simulated tempering : a new Monte Carlo scheme. *Europhysics Letters*, 19 :451–458, 1992.

[152] P. Marjoram and P. Joyce. Approximately sufficient statistics and Bayesian computation. *Statistical Applications in Genetics and Molecular Biology*, 7(1), 2008.

[153] P. Marjoram, J. Molitor, V. Plagnol, and S. Tavaré. Markov chain Monte Carlo without likelihoods. *Proceedings of the National Academy of Sciences of the USA*, 100(26) :15324–15328, 2003.

[154] D. Marquardt. Generalized inverses, ridge regression, biaised linear estimation, and nonlinear estimation. *Technometrics*, 3 :591–612, 1970.

[155] Y. Maruyama and E. George. gBF : a fully Bayes factor with a generalized g-prior. *arxiv :0801.4410,v3*, 2011.

[156] T. McKinley, A. Cook, and R. Deardon. Inference in epidemic models without likelihoods. *The international Journal of Biostatistics*, 5(1), 2009.

[157] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller. Equations of state calculations by fast computing machines. *Journal of Chemical Physics*, 21(6) :1087–1092, 1953.

[158] P. Minary and M. Levitt. Discussion of Equi-Energy sampler by Kou, Zhou and Wong. *The Annals of Statistics*, 34(4) :1636–1641, 2006.

[159] A. Mira. *Ordering, slicing and splitting Monte Carlo Markov chains*. PhD thesis, University of Minnesota, 1998.

[160] T. Mitchell and J. Beauchamp. Bayesian variable selection in linear regression. *Journal of the American Statistical Association*, 83(404) :1023–1032, 1988.

[161] A. Mitsutake, Y. Sugita, and Y. Okamoto. Replica-exchange multicanonical and multicanonical replica-exchange Monte Carlo simulations of peptides. I. Formulation and benchmark test. *The Journal of Chemical Physics*, 118(14) :6664–6675, 2003.

[162] A. Munk, J. Stockis, J. Valeinis, and G. Giese. Neyman smooth goodness-of-fit tests for the marginal distribution of dependant data. *The Annals of the Institute of Statistical Mathematics*, 63(5) :939–959, 2010.

[163] G. Nagaraja, M. Othman, B. Fox, R. Alsaber, C. Pellegrino, Y. Zeng, R. Khanna, P. Tamburini, A. Swaroop, and R. Kandpal. Gene expression signatures and biomarkers of noninvasive and invasive breast cancer cells : comprehensive profiles by representational difference analysis, microarrays and proteomics. *Oncogene*, 25(16) :2328–2338, 2006.

[164] K. Nagata and S. Watanabe. Asymptotic behavior of exchange ratio in exchange Monte Carlo method. *Neural Networks*, 21 :980–988, 2008.

[165] Y. Naoi, K. Kishi, T. Tanei, R. Tsunashima, N. Tominaga, Y. Baba, S. Kim, T. Taguchi, Y. Tamaki, and S. Noguchi. High Genomic Grade Index associated with poor prognosis for lymph node-negative and estrogen receptor-positive breast cancers and with good response to chemotherapy. *Cancer*, 117(3) :472–479, 2011.

[166] R. Neal. Sampling from multimodal distributions using tempered transitions. *Statistics and computing*, 6(4) :353–366, 1996.

[167] J. Neyman. Smooth test for goodness of fit. *Skandinavisk Aktuarietidskrift*, 20 :149–199, 1937.

[168] I. Ntzoufras, P. Dellaportas, and J. Forster. Bayesian variable and link determination for generalised linear models. *Journal of Statistical Planning and Inference*, 111 :165–180, 2003.

[169] A. Nunes and D. Balding. On optimal selection of summary statistics for approximate Bayesian computation. *Statistical Applications in Genetics and Molecular Biology*, 9(1) :art. 34, 2010.

[170] A. O'Hagan. Fractional Bayes factors for model comparison. *Journal of the Royal Statistical Society B*, 57 :99–138, 1995.

[171] R. O'Hara and M. Sillanpää. A review of Bayesian variable selection methods : What, How and Which. *Bayesian Analysis*, 4(1) :85–118, 2009.

[172] W. Pan and J. Connett. A multiple imputation approach to linear regression with clustered censored data. *Lifetime Data Analysis*, 7 :111–123, 2001.

[173] W. Pan and T. Louis. A linear mixed-effects model for multivariate censored data. *Biometrics*, 56 :160–166, 2000.

[174] P. Papastamoulis and G. Iliopoulos. An artificial allocations based solution to the label switching problem in Bayesian analysis of mixtures of distributions. *Journal of Computational and Graphical Statistics*, 19(2) :313–331, 2010.

[175] T. Park and G. Casella. The Bayesian Lasso. *Journal of the American Statistical Association*, 103 :681–686, 2008.

[176] P. Peskun. Optimum Monte Carlo sampling using Markov chains. *Biometrika*, 60 :607–612, 1973.

[177] J. Pritchard, M. Seielstad, A. Perez-Lezaun, and M. Feldman. Population growth of human Y chromosomes : a study of Y chromosome microsatellites. *Molecular Biology and Evolution*, 16 :1791–1798, 1999.

[178] K. Puolamäki and S. Kaski. *Advances in intelligent data analysis VIII, Lecture notes in computer science*, chapter Bayesian solutions to the label switching problem, pages 381–392. 2009.

[179] J. Rae, M. Johnson, J. Scheys, K. Cordero, J. Larios, and M. Lippman. GREB 1 is a critical regulator of hormone dependent breast cancer growth. *Breast Cancer Research and Treatment*, 92(2) :141–149, 2005.

[180] O. Ratmann. *ABC under model uncertainty*. PhD thesis, Imperial College, London, 2009.

[181] O. Ratmann, O. Jorgensen, T. Hinkley, M. Stumpf, S. Richardson, and C. Wiuf. Using likelihood-free inference to compare evolutionary dynamics of the protein networks of H. pylori and P. falciparum. *PLoS Computational Biology*, 3(11) :2266–2278, 2007.

[182] J. Rayner and D. Best. *Smooth Tests of Goodness of Fit*. Oxford University Press, New York, 1989.

[183] J. Rayner, D. Best, and O. Thas. Smooth tests of goodness of fit. *Wiley Interdisciplinary Reviews : Computational Statistics*, 2011.

[184] S. Richardson and P. Green. On Bayesian analysis of mixtures with an unknown number of components (with discussion). *Journal of the Royal Statistical Society B*, 59 :731–792, 1997.

[185] E. Rio. Covariance inequalities for strongly mixing processes. *Annales de l'institut Henri Poincaré (B) Probabilités et Statistiques*, 29 :587–597, 1993.

[186] C. Robert and G. Casella. *Monte Carlo statistical methods*. Springer, second edition, 2004.

[187] G. Roberts and W. Gilks. Convergence of adaptive direction sampling. *Journal of multivariate analysis*, 49 :287–298, 1994.

[188] G. Roberts and J. Rosenthal. Harris recurrence of Metropolis-within-Gibbs and transdimensional Markov chains. *The Annals of Applied Probability*, 16(4) :2123–2139, 2006.

[189] G. Roberts and R. Tweedie. Exponential convergence of Langevin distributions and their discrete approximation. *Bernoulli*, 2(4) :341–363, 1996.

[190] M. Rosenblatt. A central limit theorem and a strong mixing condition. *Proceedings of the National Academy of Sciences of the USA*, 42 :43–47, 1956.

[191] D. Rubin. Bayesianly justifiable and relevant frequency calculations for the applied statistician. *Annals of Statistics*, 12 :1151–1172, 1984.

[192] D. Sabanés Bové and L. Held. Hyper-g priors for generalized linear models. *Bayesian Analysis*, 6(1), 2011.

[193] T. Savitsky, M. Vannucci, and N. Sha. Variable selection for nonparametric Gaussian process priors : models and computational strategies. *Statistical Science*, 26(1) :130–149, 2011.

[194] G. Schwartz. Estimating the dimension of a model. *Annals of Statistics*, 6 :461–464, 1978.

[195] M. Schwartz, Q. Mo, M. Murphy, J. Lupton, M. Turner, M. Hong, and M. Vannucci. Bayesian variable selection in clustering high-dimensional data with substructure. *Journal of agricultural, biological and environmental statistics*, 13(4) :407–423, 2008.

[196] N. Sha, M. Tadesse, and M. Vannucci. Bayesian variable selection for the analysis of microarray data with censored outcomes. *Bioinformatics*, 22(18) :2262–2268, 2006.

[197] N. Sha, M. Vannucci, M. Tadesse, P. Brown, I. Dragoni, N. Davies, T. Roberts, A. Contestabile, M. Salmon, C. Buckley, and F. Falciani. Bayesian variable selection in multinomial probit models to identify molecular signatures of disease stage. *Biometrics*, 60 :812–819, 2004.

[198] D. Singh and R. Chaturvedi. Recent patents on genes and gene sequences useful for developing breast cancer detection systems. *Recent Patents on DNA and Gene Sequences*, 3(2) :139–147, 2009.

[199] S. Sisson, Y. Fan, and M. Tanaka. Sequential Monte Carlo without likelihood. *Proceedings of the National Academy of Sciences of the USA*, 104 :1760–1765, 2007.

[200] S. Sisson, Y. Fan, and M. Tanaka. Sequential Monte Carlo without likelihood : Errata. *Proceedings of the National Academy of Sciences of the USA*, 106, 2009.

[201] P. Small, P. Hopewell, S. Singh, A. Paz, J. Parsonnet, D. Ruston, G. Schecter, C. Daley, and G. Schoolnik. The epidemiology of tuberculosis in San Francisco : a population-based study using conventional and molecular methods. *The New England Journal of Medicine*, 330 :1703–1709, 1994.

[202] M. Smith and R. Kohn. Non parametric regression using Bayesian variable selection. *Journal of Econometrics*, 75 :317–344, 1997.

[203] P. Somol and J. Novovicova. Evaluating the stability of feature selectors that optimize feature subset cardinality. In N. da Vitora Lobo et al., editor, *Lecture Notes in Computer Science, vol 5342*, pages 956–966. Springer-Verlag Berlin Heidelberg, 2008.

[204] C. Sotiriou, P. Wirapati, S. Loi, A. Harris, S. Fox, J. Smeds, H. Nordgren, P. Farmer, V. Praz, H.-K. B., C. Desmedt, D. Larsimont, F. Cardoso, H. Peterse, D. Nuyten, M. Buyse, M. Van de Vijver, J. Bergh, M. Piccart, and M. Delorenzi. Gene expression profiling in breast cancer : Understanding the molecular basis of histologic grade to improve prognosis. *Journal of the National Cancer Institute*, 98(4) :262–272, 2006.

[205] V. Sousa, M. Fritz, M. Beaumont, and L. Chikhi. Approximate Bayesian computation without summary statistics : the case of admixture. *Genetics*, 181(4) :1507–1519, 2009.

[206] M. Sperrin, T. Jaki, and E. Wit. Probabilistic relabelling strategies for the label switching problem in Bayesian mixture models. *Statistics and Computing*, 20(3) :357–366, 2010.

[207] M. Srivastava. Singular Wishart and multivariate beta distributions. *The Annals of Statistics*, 31(5) :1537–1560, 2003.

[208] M. Stephens. *Bayesian methods for mixtures of normal distributions*. PhD thesis, University of Oxford, 1997.

[209] M. Stephens. Discussion of "On Bayesian analysis of mixtures with an unknown number of components" by Richardson and Green. *Journal of the Royal Statistical Society B*, 59 :768–769, 1997.

[210] M. Stephens. Bayesian analysis of mixtures with an unknown number of components - an alternative to reversible jump methods. *Annals of Statistics*, 28 :40–74, 2000.

[211] M. Stephens. Dealing with label switching in mixture models. *Journal of the Royal Statistical Society B*, 62 :795–809, 2000.

[212] F. Stingo and M. Vannucci. *Bayesian Statistics 9*, chapter Bayesian models for variable selection that incorporate biological information. Oxford University Press, 2010.

[213] G. Stormo and G. Hartzell. Identifying protein-binding sites from unaligned DNA fragments. *Proceedings of the National Academy of Sciences of the USA*, 86 :1183–1187, 1989.

[214] M. Tadesse, N. Sha, and M. Vannucci. Bayesian variable selection in clustering high-dimensional data. *Journal of the American Statistical Association*, 100 :602–617, 2005.

[215] M. Tanaka, A. Francis, F. Luciani, and S. Sisson. Using approximate Bayesian computation to estimate tuberculosis transmission parameters form genotype data. *Genetics*, 173 :1511–1520, 2006.

[216] S. Tavaré, D. Balding, R. Griffith, and P. Donnelly. Inferring coalescence times from DNA sequence data. *Genetics*, 145 :505–518, 1997.

[217] T. Therneau and P. M. Grambsch. *Modeling survival data : extending the Cox model.* Springer-Verlag, New York, 2000.

[218] R. Tibshirani. Regression shrinkage and selection via the Lasso. *Journal of the Royal Statistical Society B*, 58(1) :267– 288, 1996.

[219] R. Tibshirani, M. Saunders, S. Rosset, J. Zhu, and K. Knight. Sparsity and smoothness via the fused Lasso. *Journal of the Royal Statistical Society B*, 67 :91–108, 2005.

[220] L. Tierney. Markov chains for exploring posterior distributions. *Annals of Statistics*, 22 :1701–1762, 1994.

[221] L. Tierney. A note on Metropolis-Hastings kernels for general state spaces. *Annals of Applied Probability*, 8 :1–9, 1998.

[222] L. Tierney and A. Mira. Some adaptive Monte Carlo methods for Bayesian inference. *Statistics in Medicine*, 18 :2507–2015, 1999.

[223] T. Toni, D. Welch, N. Strelkowa, A. Ipsen, and M. Stumpf. Approximate Bayesian computation scheme for parameter inference and model selection in dynamical systems. *Journal of the Royal Society Interface*, 6 :187–202, 2009.

[224] S. Townson and P. O'Connell. Identification of estrogen receptor alpha variants in breast tumors : implications for predicting response to hormonal therapies. *Journal of Surgical Oncology*, 94(4) :271–273, 2006.

[225] R. Tüchler. Bayesian variable selection for logistic models using auxiliary mixture sampling. *Journal of Computational and Graphical Statistics*, 17(1) :76–94, 2008.

[226] G. Tutz and J. Ulbrich. Penalized regression with correlation based penalty. *Statistics and Computing*, 19(3) :239–253, 2009.

[227] D. van Dyk and T. Park. Partially collapsed Gibbs samplers : theory and methods. *Journal of the American Statistical Association*, 103 :790–796, 2008.

[228] A. Warnes. *The normal kernel coupler : an adaptive Markov chain Monte Carlo method for efficiently sampling from multimodal distributions.* PhD thesis, University of Washington, 2001.

[229] D. Wegmann, C. Leuenberger, and L. Excoffier. Efficient approximate Bayesian computation coupled with Markov chain Monte Carlo without likelihood. *Genetics*, 182 :1207–1218, 2009.

[230] M. West. *Bayesian Statistics 7*, chapter Bayesian factor regression models in the 'large p, small n' paradigm. Oxford University Press, 2003.

[231] A. Yang and X. Song. Bayesian variable selection for disease classification using gene expression data. *Bioinformatics*, 26(2) :215–222, 2010.

[232] A. Yang and X. Song. Supplementary material to Bayesian variable selection for disease classification using gene expression data. *Bioinformatics*, 26(2), 2010.

[233] W. Yao. *On using mixtures and modes of mixtures in data analysis.* PhD thesis, The Pennsylvania State University, 2007.

[234] W. Yao and B. Lindsay. Bayesian mixture labelling by highest posterior density. *Journal of the American Statistical Association*, 104(486) :758–767, 2009.

[235] N. Yi and S. Xu. Bayesian Lasso for quantitative trait loci mapping. *Genetics*, 179 :1045–1055, 2008.

[236] M. Yuan and Y. Lin. Efficient empirical Bayes variable selection and estimation in linear models. *Journal of the American Statistical Association*, 100(472) :1215–1225, 2005.

[237] M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society B*, 68 :49–67, 2006.

[238] A. Zellner. *Bayesian inference and decision techniques – essays in honour of Bruno De Finetti*, chapter On assessing prior distributions and Bayesian regression analysis with g-prior distributions., pages 233–243. Amsterdam, 1986.

[239] A. Zellner and A. Siow. Posterior odds ratios for selected regression hypotheses. In *Bayesian Statistics : Proceedings of the First International Meeting Held in Valencia*, pages 585–603. University of Valencia Press, 1980.

[240] Z. Zheng. On swapping and simulated tempering algorithms. *Stochastic Processes aand their Applications*, 104 :131–154, 2003.

[241] X. Zhou, K. Liu, and S. Wong. Cancer classification and prediction using logistic regression with Bayesian gene selection. *Journal of Biomedical Informatics*, 37 :249–259, 2004.

[242] X. Zhou, X. Wang, and E. Dougherty. A Bayesian approach to non linear probit gene selection and classification. *Journal of the Franklin Institute*, (341) :137–156, 2004.

[243] X. Zhou, X. Wang, and E. Dougherty. Gene prediction using multinomial probit regression with Bayesian gene selection. *EURASIP Journal on Applied Signal Processing*, (1) :115–124, 2004.

[244] H. Zou and T. Hastie. Regularization and variable selection via the Elastic Net. *Journal of the Royal Statistical Society B*, 67 :301–320, 2005.

Annexes

Annexe A

Echantillonneurs de Gibbs « partially collapsed »

A.1 Echantillonneurs « partially collapsed Gibbs samplers »

Dans leur article de 2008, van Dyk et Park [227] ont proposé des échantillonneurs généralisant les échantillonneurs de Gibbs, ainsi que les techniques de « grouping » et de « collapsing » de Liu [141] et Liu et al. [144].
Soit un échantillonneur de Gibbs à balayage systématique de paramètres W, X, Y et Z. A chaque itération cet échantillonneur génère chacune des variables suivant sa distribution conditionnelle complète. Les étapes suivantes sont donc effectuées :

Echantillonneur 1 :

1. Générer W suivant $W \mid X, Y, Z$.
2. Générer X suivant $X \mid W, Y, Z$.
3. Générer Y suivant $Y \mid W, X, Z$.
4. Générer Z suivant $Z \mid W, X, Y$.

Nous pouvons modifier les distributions suivant lesquelles chaque variable est générée pour obtenir un nouvel échantillonneur. Cependant, afin de conserver la distribution a posteriori cible comme distribution stationnaire, certaines règles doivent être respectées. Van Dyk et Park [227] donnent une procédure utilisant trois outils, appelés marginalisation, permutation et réduction (« trimming »), et montrent que les échantillonneurs obtenus via cette procédure conservent la distribution a posteriori cible comme distribution stationnaire.
L'étape de marginalisation lors d'une étape de l'échantillonneur consiste à faire passer un groupe de variables de l'état de « conditionnellement à » à l'état d'« échantillonnée ». Par exemple, si

aux étapes 3 et 4 de l'échantillonneur 1 nous faisons passer W de l'état de « conditionnellement à » à l'état d'« échantillonnée », cela donne :

Echantillonneur 2 :

1. Générer W^* suivant $W \mid X, Y, Z$.
2. Générer X suivant $X \mid W, Y, Z$.
3. Générer W^*, Y suivant $W, Y \mid X, Z$.
4. Générer W, Z suivant $W, Z \mid X, Y$.

L'exposant * désigne une quantité intermédiaire qui est échantillonnée mais qui ne fait pas partie du résultat final de l'itération. A chaque étape nous conditionnons par rapport aux valeurs des variables les plus récemment générées (mais pas générées lors de l'étape actuelle). Par exemple à l'étape 2 nous utilisons pour W la valeur W^* obtenue à l'étape 1. Cette approche peut être intéressante, si par exemple nous ne savons pas échantillonner suivant $Y \mid W, X, Z$, mais suivant $Y \mid X, Z$ et $W \mid X, Y, Z$. Alors, nous pouvons échantillonner suivant $W, Y \mid X, Z$, et l'étape 3 de l'échantillonneur 2 est réalisable, tandis que celle de l'échantillonneur 1 ne l'est pas. Cependant, un tel échantillonneur n'est pas très efficace dans le sens où la variable W est échantillonnée trois fois alors que seule la dernière valeur est conservée. Van Dyk et Park introduisent alors les outils de permutation et de réduction. La permutation consiste à permuter des étapes de l'échantillonneur avant d'effectuer une réduction qui supprime les échantillonnages redondants de certaines quantités échantillonnées plusieurs fois lors d'une itération de l'échantillonneur.
Nous pouvons par exemple permuter les étapes de l'échantillonneur 2 de la façon suivante :

Echantillonneur 3 :

1. Générer W^*, Y suivant $W, Y \mid X, Z$.
2. Générer W^*, Z suivant $W, Z \mid X, Y$.
3. Générer W suivant $W \mid X, Y, Z$.
4. Générer X suivant $X \mid W, Y, Z$.

Il est facile de voir que les outils de marginalisation et de permutation conservent la distribution stationnaire de la chaîne de Markov. Cela est moins évident pour l'outil de réduction, que nous présentons maintenant. En effet, la suppression des échantillonnages redondants doit se faire sous certaines conditions pour que la distribution stationnaire soit conservée, et notamment pour conserver la structure de corrélation. En effet, seules des quantités intermédiaires qui ne sont pas utilisées dans un conditionnement plus loin dans l'itération peuvent être supprimées. C'est le cas pour les quantités W^* des étapes 1 et 2 de l'échantillonneur 3. Ainsi, nous pouvons

les supprimer et obtenir l'échantillonneur suivant :

Echantillonneur 4 :

1. Générer Y suivant $Y \mid X, Z$.
2. Générer Z suivant $Z \mid X, Y$.
3. Générer W suivant $W \mid X, Y, Z$.
4. Générer X suivant $X \mid W, Y, Z$.

Les étapes 2 et 3 de l'échantillonneur 4 peuvent être regroupées, cela revient à générer W et Z ensemble :

Echantillonneur 5 :

1. Générer Y suivant $Y \mid X, Z$.
2. Générer W, Z suivant $W, Z \mid X, Y$.
3. Générer X suivant $X \mid W, Y, Z$.

Van Dyk et Park [227] ont montré que les échantillonneurs obtenus via cette procédure conservent la distribution a posteriori cible comme distribution stationnaire, et que leurs convergences sont améliorées par rapport à un échantillonneur de Gibbs classique. En effet, les autocorrélations de la chaîne sont réduites. Cependant, il est alors nécessaire de réaliser les différentes étapes de l'échantillonneur dans un certain ordre, et la chaîne générée n'est donc pas réversible.

A.2 Les techniques de « grouping » et de « collapsing »

Liu [141] et Liu et al. [144] ont introduit des techniques de « grouping » et de « collapsing ». Ces techniques sont des cas particuliers des échantillonneurs « partially collapsed Gibbs samplers » de van Dyk et Park [227]. Soit un échantillonneur de Gibbs à balayage systématique de paramètres X, Y et Z. A chaque itération cet échantillonneur génère chaque variable suivant sa distribution conditionnelle complète. Les étapes suivantes sont donc effectuées :

Echantillonneur 6 :

1. Générer X suivant $X \mid Y, Z$.
2. Générer Y suivant $Y \mid X, Z$.
3. Générer Z suivant $Z \mid X, Y$.

La technique de « grouping » (aussi appelée « blocking ») consiste à considérer ensemble deux variables. Par exemple :

Echantillonneur 7 :

1. Générer X suivant $X \mid Y, Z$.
2. Générer Y, Z suivant $Y, Z \mid X$.

Cet échantillonneur peut être avantageux si nous savons échantillonner suivant $Y \mid X, Z$ et $Z \mid X$ par exemple.
La technique de « collapsing » consiste quant à elle à supprimer totalement une des variables. Par exemple :

Echantillonneur 8 :

1. Générer X suivant $X \mid Y$.
2. Générer Y suivant $Y \mid X$.

Cela est avantageux si nous savons échantillonner directement suivant $X \mid Y$ et $Y \mid X$, en ayant tout intégré en Z.

De manière générale, quand sa mise en oeuvre est possible, la technique de « collapsing » est plus avantageuse que la technique de « grouping » (voir Liu [141]).

Annexe B

Sélection de variables dans un mélange de modèles AFT

Les modèles de mélange sont couramment utilisés, car ils permettent de modéliser de façon assez intuitive des données hétérogènes, pour lesquelles chaque observation est supposée provenir d'un groupe parmi plusieurs. L'apparition d'échantillonneurs bayésiens basés sur la théorie des « Monte Carlo Markov Chains » a permis une utilisation et une implémentation facile de ces modèles, dans le cas où le nombre de composants d'un mélange est supposé connu (voir Diebolt et Robert [59]), mais également dans le cas de mélanges avec nombres de composants inconnus (voir Richardson et Green [184] et Stephens [210]).

Le problème de la sélection bayésienne de variables dans ce cadre de modèles de mélange a été étudié par plusieurs auteurs, parmi lesquels Kim et al. [114] qui proposent une méthode de sélection de variables dans une optique de classification, ou Tadesse et al. [214] qui proposent une méthode de sélection de variables dans le cas de mélanges de lois normales. La méthode de ces derniers a été étendue au cas de données structurées par Schwartz et al. [195]. Récemment Gosh et al. [81] se sont intéressés aux modèles à classes latentes, en les modélisant également par des mélanges de lois normales.

Les données de survie suscitent également beaucoup d'intérêt, à une époque où les méthodes de diagnostic personnalisé se développent. La particularité de ces données est qu'elles sont souvent censurées.

Le problème de la sélection bayésienne de variables dans le cadre de données censurées a lui aussi été étudié (voir entre autres Sha et al. [196], Lee et Mallick [130] ou plus récemment Savitsky et al. [193] et Lee et al. [129]).

Cependant, il n'y a pas à notre connaissance de méthode bayésienne de sélection de variables dans le cadre de modèles de mélanges pour des données censurées.

Le Genomic Grade Index (GGI) commercialisé par Ipsogen est une signature génomique permettant de classifier les patientes ayant un cancer du sein en différents grades génomiques (voir Sotiriou et al. [204]). Jusqu'à présent les médecins utilisaient surtout le grade histologique (allant de 1 à 3), mais ce dernier a l'inconvénient de ne pas être très reproductible entre différents hôpitaux, et de nombreuses patientes sont classées en grade histologique 2. Or les patientes en grade histologique 2 sont suspectées être issues d'un mélange de grades 1 et 3, et sont donc en réalité de grade indéterminé. Un des avantages du GGI est qu'il permet de reclassifier les patientes de grade histologique 2 en grade génomique 1 ou 3, ce qui permet de prendre des décisions quant à leurs traitements.

De plus, le GGI serait associé à la réponse des patientes à la chimiothérapie (voir Liedtke et al. [140]). Enfin, nous suspectons également qu'il a de bonnes propriétés pronostiques (voir notamment Naoi et al. [165]). C'est ce qui nous a motivés à développer une méthode de sélection de variables dans le cadre de modèles de mélanges pour des données de survie. En effet, parmi les 128 probesets constituant le GGI nous désirons savoir si certains en particulier ont un rôle pronostique pour l'apparition de métastases à cinq ans. Nous suspectons de plus que toutes les patientes n'ont pas la même évolution, et cela serait dû entre autre aux statuts de leurs récepteurs hormonaux. Nous voulons donc nous placer dans le cadre d'un mélange avec nombre de composants inconnu. Enfin, nous voulons prendre en compte le fait que les patientes proviennent de différents jeux de données, et nous sommes donc dans le cadre de modèles mixtes.

Nous disposons de données de survie pour des patientes pour lesquelles un cancer a été diagnostiqué, mais sans métastase. Elles subissent alors une chirurgie, puis en général une hormonothérapie et/ou une radiothérapie. Nous observons alors l'apparition ou pas de métastases à cinq ans. Deux variables sont mises à notre disposition : une variable temps, et une variable évènement. Si l'évènement vaut 1, alors le temps correspond à la date d'apparition d'un métastase, et si l'évènement vaut 0, le temps correspond à la date de censure. De plus, nous disposons pour chacune des patientes des mesures d'expression des 128 probesets constituant le GGI.

B.1 Modéle hiérarchique

B.1.1 Choix du modèle

Pour la $i^{\text{ème}}$ patiente, S_i et C_i sont le temps de survie et la censure associée. Ils sont supposés indépendants. Nous observons le temps $T_i = \min(C_i, S_i)$ ainsi que la présence d'évènement ou pas au temps T_i, $\delta_i = \mathbb{1}_{\{S_i \leq C_i\}}$. La variable d'intérêt est la survie S. Nous allons l'étudier via $Y = log(S)$.

$$Y_i = log(S_i) \Leftrightarrow \begin{cases} Y_i = log(T_i) & \text{si } \delta_i = 1 \\ Y_i > log(T_i) & \text{si } \delta_i = 0 \end{cases} \qquad i = 1, \ldots, n. \tag{B.1}$$

Pour modéliser la survie, deux types de modèles sont principalement utilisés : les modèles semi-paramétriques de Cox [49], ou bien les modèles à temps de sortie accéléré (« Accelerated Failure Time », AFT) (voir par exemple [110] pour de tels modèles paramétriques). Le modèle de Cox est un modèle à risques proportionnels car le ratio des fonctions de risque de deux individus est supposé constant. Les covariables ont des effets multiplicatifs sur les risques instantanés. Dans le modèle AFT, les différentes covariables modifient la vitesse selon laquelle le temps s'écoule pour les individus, les covariables accélérent ou décélérent le temps en fonction de leurs coefficients et ont donc un effet multiplicatif directement sur les temps de survie. La distribution de l'erreur d'un modèle AFT conduit à différent types de modèles connus pour le temps de survie : ce dernier suit un modèle log-linéaire si les erreurs sont gaussiennes, un modèle de Weibull si les erreurs suivent la loi des valeurs extrêmes, ou encore un modèle log-logistique si les erreurs suivent une loi logistique.
Etant donné que nous sommes en présence d'un effet aléatoire (provenance des patientes), nous devons utiliser des modèles de Cox avec fragilité (voir Hougaard [98] et Therneau et Grambsch [217]), ou bien des « Mixed Effects Accelerated Failure Time models (MEAFT) » (voir par exemple Pan et Connett [172], Pan et Louis [173] et Komarek et Lesaffre [117]). Plusieurs arguments nous amènent à utiliser un modèle MEAFT plutôt qu'un modèle de Cox avec fragilité. Tout d'abord, la présence de plusieurs effets aléatoires est plus facile à prendre un compte avec un modèle MEAFT, et ce dernier est plus facile d'utilisation dans un cadre bayésien. Ensuite, le choix de la distribution de fragilité dans un modèle de Cox peut avoir une influence importante sur les paramètres de régression estimés (voir Hougaard [98]). Le modèle de Cox est également moins robuste à l'omission de covariables pertinentes que le modèle AFT (voir Hougaard [97]). Enfin, les paramètres de régression estimés par des modèles MEAFT ont l'avantage d'être assez robustes à une mauvaise spécification des effets aléatoires (voir Keiding et al. [113] et Lambert et al. [124]).
Dans le cadre bayésien, les modèles AFT ou MEAFT ont été estimés par des approches paramétriques (voir par exemple Bedrick et al. [20]) ou semi-paramétriques (voir entre autres Christensen et Johnson[45] et Kuo et Mallick [120]). Dans un cadre de sélection de variables bayésienne, ces modèles ont notamment été utilisés par Lee et Mallick [130] et Sha et al. [196]. Nous considérons ici une inférence paramétrique des modèles MEAFT, et nous supposons la normalité des erreurs de ces modèles afin d'avoir des a priori conjugués simples et pratiques à utiliser.

Afin de modéliser le fait que l'évolution des patientes peut être différente en fonction des statuts de leurs récepteurs hormonaux, nous supposons un modèle de mélange avec nombre de composants inconnu, noté K :

$$Y_i \mid \beta, \nu^2, U \sim \sum_{k=1}^{K} w_k \mathcal{N}(X_i'\beta_k + Z_i'U, \nu^2), \qquad i = 1, \ldots, n.$$

Les indices des composants sont notés $k = 1, \ldots, K$, et les tailles des différents composants sont proportionnelles aux w_k dont la somme vaut 1. Les vecteurs X_i et Z_i correspondent aux effets fixes et aléatoires associés à la $i^{\text{ème}}$ observation, le vecteur de paramètres β_k correspond aux coefficients des effets fixes pour le $k^{\text{ème}}$ composant et le vecteur de paramètres U correspond aux coefficients des effets aléatoires. Nous notons X et Z les matrices de design associées aux effets fixes et aléatoires respectivement. Les effets aléatoires sont supposés identiques dans les différents composants. A l'inverse, les paramètres de régression des effets fixes diffèrent entre les différents composants.

Chaque Y_i est supposé provenir d'un des composants du mélange. Nous introduisons alors un vecteur latent d'étiquettes c, avec $c_i = k$ si la $i^{\text{ème}}$ observation Y_i est issu du $k^{\text{ème}}$ composant. Les z_i sont supposés indépendants et issus des distributions :

$$p(c_i = k) = w_k, \qquad k = 1, \ldots, k.$$

Conditionnellement à c, nous savons de quelle population est issu y_i :

$$Y_i \mid \beta, \nu^2, U, z \sim \mathcal{N}(X_i'\beta_{c_i} + Z_i'U, \nu^2), \qquad i = 1, \ldots, n.$$

Sachant β, ν^2, U et z, nous avons bien un modèle MEAFT :

$$\log(S_i) = Y_i = X_i'\beta_k + Z_i'U + \epsilon_i, \qquad \text{avec} \qquad \epsilon_i \sim \mathcal{N}(0, \nu^2). \tag{B.2}$$

L'ensemble des variables disponibles est de taille $p \gg n$. Afin de pouvoir sélectionner des variables, un vecteur γ constitué de p variables indicatrices est introduit :

$$\gamma_j = \begin{cases} 1 & \text{si } \exists k / \beta_{kj} \neq 0, \quad \text{la variable } j \text{ est sélectionnée dans au moins un composant,} \\ 0 & \text{si } \forall k \quad \beta_{kj} = 0, \quad \text{la variable } j \text{ n'est sélectionnée dans aucun composant.} \end{cases}$$

Nous supposons que la variable d'intérêt S peut être expliquée par les mêmes prédicteurs dans chaque composant du mélange. Ce sont donc les mêmes variables qui sont sélectionnées pour l'ensemble des composants, mais les coefficients des variables diffèrent entre les composants.

Notations

Nous utilisons les notations suivantes :

- $Y = (Y_1, \ldots, Y_n)$.
- $\beta = (\beta_1, \ldots, \beta_K)$, et $\beta_k = (\beta_{k1}, \ldots, \beta_{kp})$ avec β_{kj} le coefficient associé à la $j^{\text{ème}}$ variable dans le $k^{\text{ème}}$ composant. Les β_k sont supposés indépendants entre eux.

- Soit L effets aléatoires à prendre en compte, nous notons $U = (U_1', \ldots, U_L')'$, avec U_l de taille q_l et $\sum_{l=1}^{L} q_l = q$.
- $w = (w_1, \ldots, w_K)$.
- n_k est le nombre d'observations classées dans le composant k, $\sum_{k=1}^{K} n_k = n$.
- Sachant les classes des observations, $X_{[k]}$ est la sous-matrice de X contenant seulement les lignes correspondant aux observations classées dans le composant k. Elle est de dimension $n_k \times p$. De même, nous introduisons $Z_{[k]}$ de dimension $n_k \times q$, ainsi que $Y_{[k]}$ sous-vecteur de Y de longueur n_k. Nous avons :

$$Y_{[k]} \mid \beta, \nu^2, U \sim \mathcal{N}(X_{[k]}\beta_k + Z_{[k]}U, \nu^2 I_{n_k}).$$

- Sachant γ, $\beta_{k\gamma}$ est le vecteur des éléments de β_k associés aux éléments non nuls de γ. Les matrices X_γ et Z_γ correspondent aux matrices X et Z respectivement, mais ne contiennent que les colonnes associées aux éléments non nuls de γ. La matrice $X_{[k]\gamma}$ correspond à la matrice $X_{[k]}$ avec seulement les colonnes associées aux éléments non nuls de γ.
- Les vecteurs $\beta_{k\gamma}$ sont de taille d_γ qui est le nombre d'éléments non nuls de γ, et ce qui correspond également au nombre de variables sélectionnées. Nous notons :

$$\begin{aligned} \beta_{k\gamma} &= (\beta_{k\gamma 1}, \ldots, \beta_{k\gamma d_\gamma}), \qquad k = 1, \ldots, K. \\ \beta_\gamma &= (\beta_{1\gamma}, \ldots, \beta_{K\gamma}), \qquad \text{de taille } d_\gamma \times K. \end{aligned}$$

B.1.2 Les distributions a priori

Distribution a priori de w

L'a priori sur w est une Dirichlet symétrique :

$$w \sim D(\delta, \delta, \ldots, \delta).$$

Distribution a priori de c

Les c_i sont supposés indépendants et issus de distributions multinomiales :

$$p(c_i = k) = w_k, \qquad k = 1, \ldots, k,$$

$$c_i \sim \mathcal{M}ult(1, w),$$

$$c \sim \mathcal{M}ult(n, w).$$

Distribution a priori de K

Le nombre de composants K est supposé suivre a priori une loi uniforme (nous aurions pu choisir

une loi de Poisson) :

$$K \sim \mathcal{U}_{[1,K_{max}]}.$$

Distribution a priori des $\beta_{k\gamma}$, $k = 1, \ldots, K$

En ce qui concerne $\beta_{k\gamma}$, $k = 1, \ldots, K$, nous supposons un a priori avec paramètre ridge, comme présenté dans le chapitre 3. Nous ajoutons seulement un paramètre ν^2 car Y suit un modèle linéaire :

$$\begin{aligned} \beta_{k\gamma} \mid \gamma &\sim \mathcal{N}_{d_\gamma}\Big(0, \nu^2 \Sigma_\gamma\Big), \qquad \text{avec} \qquad d_\gamma = \sum_{j=1}^{p} \gamma_j, \\ \Sigma_\gamma &= \Big(\frac{1}{c} X'_\gamma X_\gamma + \lambda I\Big)^{-1}. \end{aligned}$$

Les $\beta_{k\gamma}$ sont supposés indépendants entre eux.

Distribution a priori de U

Comme dans les chapitres 2 et 3, nous posons :

$$U \mid D \sim \mathcal{N}_q(0, D).$$

Dans le cas général, aucune structure n'est supposée pour la matrice de variance-covariance D, sa distribution a priori est une inverse-wishart $\mathcal{W}^{-1}(\Psi, m)$.

Si nous nous placons dans le cas d'une matrice D diagonale (cas de notre application), avec $D = diag(A_1, \ldots, A_K)$, $A_l = \sigma_l^2 I_{q_l}$, $l = 1, \ldots, K$ et I_{q_l} la matrice identité d'ordre q_l, alors les distributions a priori des σ_l^2 sont supposées être des inverse-gamma $\mathcal{IG}(a, b)$ (b correspond au facteur d'échelle).

Distribution a priori de ν^2

Le paramètre ν^2 est classiquement supposé suivre une inverse-Gamma :

$$\nu^2 \sim \mathcal{IG}(f, g).$$

Distribution a priori de γ

Les γ_j sont supposés être des variables de Bernoulli indépendantes, avec :

$$P(\gamma_j = 1) = \pi_j, \qquad 0 \leq \pi_j \leq 1.$$

Pour ne pas favoriser certaines variables par rapport à d'autres, nous posons $\forall j, \quad \pi_j = \pi$.

B.1.3 La densité jointe des paramètres du modèle

La densité jointe des paramètres du modèle est la suivante :

$$
\begin{aligned}
f(Y,\nu^2,\beta_\gamma,\gamma,w,c,K,U,D) &\propto \prod_{k=1}^{K} f(Y_{[k]} \mid \nu^2,\beta_\gamma,\gamma,U) \prod_{k=1}^{K} f(\beta_{k\gamma} \mid \gamma,\nu^2) f(\nu^2) f(\gamma) f(w \mid K) \\
&\times f(c \mid w,K) f(K) f(U \mid D) f(D) \\
&\propto \prod_{k=1}^{K} \Bigg\{ (2\pi)^{-\frac{n_k}{2}} |\nu^2 I_{n_k}|^{-\frac{1}{2}} \exp\Big[-\frac{1}{2\nu^2} \big(Y_{[k]} - X_{[k]\gamma}\beta_{k\gamma} - Z_{[k]}U\big)' \big(Y_{[k]} - X_{[k]\gamma}\beta_{k\gamma} - Z_{[k]}U\big) \Big] \\
&\times (2\pi)^{-\frac{d_\gamma}{2}} |\nu^2 \Sigma_\gamma|^{-\frac{1}{2}} \exp\Big[-\frac{1}{2\nu^2} \beta'_{k\gamma} \Sigma_\gamma^{-1} \beta_{k\gamma} \Big] \Bigg\} \times \Big(\frac{1}{\nu^2}\Big)^{f+1} \exp\Big(-\frac{g}{\nu^2} \Big) \times \prod_{j=1}^{p} \pi_j^{\gamma_j} (1-\pi_j)^{1-\gamma_j} \\
&\times \frac{\Gamma(K\delta)}{\Gamma(\delta)^K} \prod_{k=1}^{K} w_k^{\delta-1} \times \frac{n!}{\prod_{k=1}^{K} n_k!} \prod_{k=1}^{K} w_k^{n_k} \times \mathbb{1}_{[1,K_{max}}(K) \times (2\pi)^{-\frac{q}{2}} |D|^{-\frac{1}{2}} \exp\Big[\frac{1}{2} U' D^{-1} U \Big] \\
&\times |D|^{-\frac{m+q+1}{2}} \exp\Big[-\frac{1}{2} tr(\Psi D^{-1}) \Big]. \qquad \text{(B.3)}
\end{aligned}
$$

B.2 L'échantillonneur bayésien

B.2.1 Les distributions conditionnelles complètes

Distribution conditionnelle complète de Y

Pour $i = 1, \ldots, n$, si $\delta_i = 1$ alors Y_i est connu et vaut $\log(T_i)$. Nous n'avons donc besoin d'implémenter Y_i que si $\delta_i = 0$:

$$
\begin{cases}
\text{si } \delta_i = 1, & Y_i \;\; = \log(T_i), \\
\text{si } \delta_i = 0, & Y_i \;\; \mid \nu^2, \beta_\gamma, \gamma, c_i = k, U, T_i \sim \mathcal{N}(X'_{i\gamma}\beta_{k\gamma} + Z'_i U, \nu^2) \mathbb{1}_{\{\log(T_i) < Y_i\}}.
\end{cases} \qquad \text{(B.4)}
$$

Distribution conditionnelle complète de w

$$
w \mid K, c \sim \mathcal{D}(\delta + n_1, \ldots, \delta + n_K). \qquad \text{(B.5)}
$$

Distribution conditionnelle complète de c

$$
c_i \mid Y, \nu^2, \beta_\gamma, \gamma, w, U \sim \mathcal{M}ult\Big(1, (p_1, \ldots, p_K)\Big), \qquad \text{(B.6)}
$$

avec

$$
p_k = \frac{\exp\Big[-\frac{1}{2\nu^2} \big(Y_i - X'_{i\gamma}\beta_{k\gamma} - Z'_i U\big)^2 \Big] w_k}{\sum_{k=1}^{K} \exp\Big[-\frac{1}{2\nu^2} \big(Y_i - X'_{i\gamma}\beta_{k\gamma} - Z'_i U\big)^2 \Big] w_k}.
$$

Distribution conditionnelle complète des $\beta_{k\gamma}$
Nous avons, en notant $T_{k\gamma} = (\frac{1}{c}X'_{[k]\gamma}X_{[k]\gamma} + \Sigma_\gamma^{-1})^{-1}$:

$$\beta_{k\gamma} \mid Y_{[k]}, \nu^2, \gamma, c, U \sim \mathcal{N}_{d_\gamma}\big(T_{k\gamma}X'_{[k]\gamma}(Y_{[k]} - Z_{[k]}U), \nu^2 T_{k\gamma}\big). \tag{B.7}$$

Distribution conditionnelle complète de U
Nous avons, en notant $W_K = \Big(\nu^{-2}\sum_{k=1}^K Z'_{[k]}Z_{[k]} + D^{-1}\Big)^{-1}$:

$$U \mid Y, \nu^2, \beta_\gamma, \gamma, c, D \sim \mathcal{N}_q\Big(\nu^{-2}W_K \sum_{k=1}^K Z'_{[k]}\big(Y_{[k]} - X_{[k]\gamma}\beta_{k\gamma}\big), W_K\Big). \tag{B.8}$$

Distribution conditionnelle complète de D
Idem chapitre 2, voir la partie 2.2.1.

Distribution conditionnelle complète de ν^2

$$\nu^2 \mid Y, \beta_\gamma, \gamma, c, U, \sim \mathcal{IG}\left\{\frac{K + d_\gamma}{2} + f, g + \sum_{k=1}^K \Big[\big(Y_{[k]} - X_{[k]\gamma}\beta_{k\gamma} - Z_{[k]}U\big)'\big(Y_{[k]} - X_{[k]\gamma}\beta_{k\gamma} - Z_{[k]}U\big) + \beta'_{k\gamma}\Sigma_\gamma^{-1}\beta_{k\gamma}\Big]\right\}. \tag{B.9}$$

Distribution conditionnelle de γ

$$f(\gamma \mid Y, \nu^2, \beta_\gamma, c, U, K) \propto \prod_{k=1}^K \left\{ \exp\Big[-\frac{1}{2\nu^2}\big(Y_{[k]} - X_{[k]\gamma}\beta_{k\gamma} - Z_{[k]}U\big)'\big(Y_{[k]} - X_{[k]\gamma}\beta_{k\gamma} - Z_{[k]}U\big)\Big] \right.$$
$$\left. \times (2\pi)^{-\frac{d_\gamma}{2}}|\Sigma_\gamma|^{-\frac{1}{2}} \exp\Big[-\frac{1}{2\nu^2}\beta'_{k\gamma}\Sigma_\gamma^{-1}\beta_{k\gamma}\Big]\right\} \times \prod_{j=1}^p \pi_j^{\gamma_j}(1 - \pi_j)^{1-\gamma_j}. \tag{B.10}$$

B.2.2 Utilisation de la technique de « grouping »

Comme dans le chapitre 2 et pour les mêmes raisons (voir partie 2.2.2), nous utilisons la technique de « grouping » de Liu [141] (voir Annexe A). L'objectif est d'éliminer le paramètre de nuisance β_γ de la distribution conditionnelle complète de γ, en groupant les paramètres β_γ et

γ. Rappelons que les $\beta_{k\gamma}$ sont supposés indépendants entre eux.

$$\begin{aligned} f(\gamma \mid Y, \nu^2, c, U, K) &\propto \int_{\beta_\gamma} f(\gamma \mid Y, \nu^2, \beta_\gamma, c, U, K) d\beta_\gamma \\ &\propto \int_{\beta_{1\gamma}} \dots \int_{\beta_{K\gamma}} f(\gamma \mid Y, \nu^2, \beta_\gamma, c, U, K) d\beta_{1\gamma} \dots d\beta_{K\gamma} \\ &\propto \prod_{k=1}^{K} \Bigg\{ \int_{\beta_{k\gamma}} (2\pi)^{-\frac{d_\gamma}{2}} |\Sigma_\gamma|^{-\frac{1}{2}} \exp\Big[-\frac{1}{2\nu^2} \Big(\big(Y_{[k]} - X_{[k]\gamma}\beta_{k\gamma} - Z_{[k]}U\big)' \\ &\quad \big(Y_{[k]} - X_{[k]\gamma}\beta_{k\gamma} - Z_{[k]}U\big) + \beta'_{k\gamma}\Sigma_\gamma^{-1}\beta_{k\gamma} \Big) \Big] d\beta_{k\gamma} \Bigg\} \times \prod_{j=1}^{p} \pi_j^{\gamma_j} (1-\pi_j)^{1-\gamma_j}. \end{aligned}$$

Or, nous avons :

$$\begin{aligned} \int_{\beta_{k\gamma}} (2\pi)^{-\frac{d_\gamma}{2}} |\nu^2 T_{k\gamma}|^{-\frac{1}{2}} \exp\Big[-\frac{1}{2\nu^2} \Big(\beta_{k\gamma} - T_{k\gamma} X'_{[k]\gamma} (Y_{[k]} - Z_{[k]}U) \Big)' T_{k\gamma}^{-1} \\ \times \Big(\beta_{k\gamma} - T_{k\gamma} X'_{[k]\gamma} (Y_{[k]} - Z_{[k]}U) \Big) \Big] d\beta_{k\gamma} = 1. \end{aligned}$$

Alors,

$$\begin{aligned} f(\gamma \mid Y, \nu^2, c, U, K) \propto \prod_{k=1}^{K} \Big(\frac{|\nu^2 T_{k\gamma}|}{|\Sigma_\gamma|} \Big)^{\frac{1}{2}} \exp\Big[-\frac{1}{2\nu^2} (Y_{[k]} - Z_{[k]}U)' \Big(I_{n_k} - X_{[k]\gamma} T_{k\gamma} X'_{[k]\gamma} \Big) (Y_{[k]} - Z_{[k]}U) \Big] \\ \times \prod_{j=1}^{p} \pi_j^{\gamma_j} (1-\pi_j)^{1-\gamma_j}. \end{aligned} \tag{B.11}$$

B.2.3 Algorithme de Metropolis-Hastings pour générer γ

Nous voulons générer γ suivant sa distribution conditionnelle complète intégrée en β_γ (B.11). Notons $\gamma^{(i)}$ la valeur de γ à l'itération i de l'algorithme de Metropolis-Hastings et $q(.|.)$ un noyau de transition. A chaque itération l'algorithme effectue les étapes suivantes :

1. Générer une valeur $\gamma^* \sim q(\gamma^* | \gamma^{(i)})$.

2. Prendre

$$\gamma^{(i+1)} = \begin{cases} \gamma^* & \text{avec probabilité} \quad \rho(\gamma^{(i)}, \gamma^*) \\ \gamma^{(i)} & \text{avec probabilité} \quad 1 - \rho(\gamma^{(i)}, \gamma^*), \end{cases}$$

où

$$\rho(\gamma^{(i)}, \gamma^*) = \min \left\{ 1, \frac{f(\gamma^* \mid Y, \nu^2, c, U, K) q(\gamma^{(i)} | \gamma^*)}{f(\gamma^{(i)} \mid Y, \nu^2, c, U, K) q(\gamma^* | \gamma^{(i)})} \right\}.$$

Avec un noyau de transition symétrique la probabilité d'acceptation d'un candidat γ^* à partir de $\gamma^{(i)}$ se simplifie. En effet, dans ce cas nous avons, avec $\forall j\ \pi_j = \pi$:

$$\begin{aligned}\rho(\gamma^{(i)},\gamma^*) &= \min\left\{\left(\frac{\pi}{1-\pi}\right)^{\sum_1^p(\gamma_j^*-\gamma_j^{(i)})} \times (\nu^2)^{\frac{K}{2}(d_{\gamma^*}-d_{\gamma^{(i)}})} \times \prod_{k=1}^{K}\left(\frac{|T_{k\gamma^*}||\Sigma_{k\gamma^{(i)}}|}{|\Sigma_{k\gamma^*}||T_{k\gamma^{(i)}}|}\right)^{\frac{1}{2}}\right. \\ &\times \left. \exp\Big[-\frac{1}{2\nu^2}\sum_{k=1}^{K}(Y_{[k]}-Z_{[k]}U)'\Big(X_{[k]\gamma^{(i)}}T_{k\gamma^{(i)}}X'_{[k]\gamma^{(i)}} - X_{[k]\gamma^*}T_{k\gamma^*}X'_{[k]\gamma^*}\Big)(Y_{[k]}-Z_{[k]}U)\Big], 1\right\}. \end{aligned} \tag{B.12}$$

Comme dans le chapitre 3, partie 3.3, nous ne fixons pas le nombre de variables à sélectionner à chaque itération. A l'itération $i+1$, le vecteur candidat γ^* proposé à partir du vecteur $\gamma^{(i)}$ correspond simplement à $\gamma^{(i)}$ pour lequel r éléments ont été choisis au hasard et modifiés.

B.2.4 Choix d'un algorithme trans-dimensionnel

Le nombre de composants du mélange K est inconnu et peut varier, la dimension de l'espace des paramètres varie donc lorsque K varie. Classiquement, pour résoudre ce type de problème trans-dimensionnel, des échantillonneurs à sauts réversibles (RJMCMC) sont utilisés (voir Green [83] et Richardson et Green [184]). Dans le cadre de modèles de mélange, l'algorithme RJMCMC de Richardson et Green [184] propose des mouvements « split/combine » pour diviser ou fusionner des composants, ainsi que des mouvements « birth-and-death » pour créer ou détruire des composants vides.

Il existe deux alternatives principales au RJMCMC pour des modèles de mélange. Tout d'abord l'algorithme « Birth and Death Monte Carlo Markov Chain (BDMCMC) » de Stephens [210]. Il a deux modifications majeures par rapport au RJMCMC : cet algorithme est dans un cadre continu tandis que le RJMCMC est dans un cadre discret, et les mouvements entre espaces de dimensions différentes ne sont que des « birth-and-death ». Ensuite, les algorithmes « Continuous Time Monte Carlo Markov Chain (BDMCMC) » de Cappé et al. [33]. Ces derniers sont dans un cadre continu et peuvent se déplacer entre espaces de dimensions différentes avec des mouvements « birth-and-death » et « splits/combine ».

Si nous considérons le fait qu'un échantillonneur peut être soit à temps continu, soit à temps discret, et contenir soit seulement des mouvements « birth-and-death », soit des mouvements « birth-and-death » et « split/combine », alors il y a quatre types possibles d'échantillonneurs. En nous basant sur les comparaisons de ces échantillonneurs effectuées par Cappé et al. [33], nous choisissons d'utiliser un RJMCMC avec des mouvements « birth-and-death » seulement.

B.2.5 Sauts réversibles

L'actualisation du paramètre K (nombre de composants du mélange) implique un changement de dimension de l'espace des paramètres. Les mouvements de création et de destruction d'un

composant vide doivent être définis ensemble, afin d'assurer la réversibilité des mouvements.

Choix entre une création et une destruction

En pratique, il faut tout d'abord décider de façon aléatoire de créer ou détruire un composant vide. La probabilité de choisir une création est b_K, et la probabilité de choisir une destruction est $d_K = 1 - b_K$, avec $d_1 = 0$ et $b_{K_{max}} = 0$.

Dans le cas de K composants avec K_0 composants vides, la probabilité de choisir la création d'un composant vide est b_K. Si le mouvements est accepté il y a $K+1$ composants dont K_0+1 vides.
Dans le cas de $K+1$ composants avec K_0+1 composants vides, la probabilité de choisir la destruction d'un composant vide déterminé est $d_{K+1}/(K_0+1)$ (choix d'effectuer une mort, puis choix du composant à détruire). Si le mouvement est accepté il y a K composants dont K_0 vides.

Cas d'une destruction

Si nous avons choisi de détruire un composant vide, nous devons aussi choisir aléatoirement lequel supprimer, puis pondérer les poids restants de manière à ce que leur somme vaille toujours 1.
Cette destruction d'un composant vide est proposée, puis acceptée avec une certaine probabilité d'acceptation (voir la partie B.2.5).

Cas d'une création

Si nous avons choisi de créer un composant vide, nous devons utiliser deux variables auxiliaires pour vérifier la condition de « dimension-matching » : le poids w_{k^*} et le vecteur des paramètres de régression $\beta_{k^*\gamma}$ associés au nouveau composant k^*. Etant donné que nous avons comme a priori :

$$\beta_{k\gamma} \mid \gamma \sim \mathcal{N}_{d_\gamma}(0, \nu^2\Sigma_\gamma), \quad j \in \{1\ldots,k\} \quad \text{et} \quad w \sim D(\delta,\delta,\ldots,\delta),$$

nous générons :

$$\beta_{k^*\gamma} \mid \gamma \sim \mathcal{N}_{d_\gamma}(0, \nu^2\Sigma_\gamma) \quad \text{et} \quad w_{k^*} \sim be(1,k).$$

Puis nous devons pondérer les poids de manière à ce que leur somme vaille toujours 1. Ainsi les anciens poids w_k correspondants aux K anciens composants deviennent, pour $K+1$ composants (les anciens plus le nouveau de poids w_{k^*}), $w_k(1-w_{k^*})$.
Cette création d'un composant vide est proposée, puis acceptée avec une certaine probabilité d'acceptation (voir la partie B.2.5).

Probabilité d'acceptation

Notons $\theta = (Y, \beta_\gamma, \gamma, \nu^2, w, c, U, D)$, $x = (\theta, K)$ et $v = (w_{k^*}, \beta_{k^*\gamma})$ les variables auxiliaires générées pour une création de composant vide. Dans le cas d'une création de composant vide,

$$\begin{array}{rll} K & \text{devient} & K+1, \\ K_0 & \text{devient} & K_0+1, \\ \beta_\gamma & \text{devient} & \beta'_\gamma = (\beta_\gamma, \beta_{k^*\gamma}), \\ w & \text{devient} & w' = (w_1(1-w_{k^*}), \ldots, w_K(1-w_{k^*}), w_{k^*}), \\ \theta & \text{devient} & \theta' = (Y, \beta'_\gamma, \gamma, \nu^2, w', c, U, D), \\ x & \text{devient} & x' = (\theta', K+1) = (Y, \beta'_\gamma, \gamma, \nu^2, w', c, U, D, K+1). \end{array}$$

Nous utilisons la formule de la probabilité d'acceptation donnée par Green [83], qui permet d'assurer la réversibilité du mouvement :

$$\rho = \min(1, A), \qquad A = \frac{\pi(K+1, \theta')\pi_{x' \to x}}{\pi(K, \theta)\pi_{x \to x'} g(v)} \left| \frac{\partial T(\theta, v)}{\partial(\theta, v)} \right|.$$

Avec :

1.

$$\begin{aligned} \frac{\pi(K+1, \theta')}{\pi(K, \theta)} &= \frac{\pi(x')}{\pi(x)} = \frac{p(x'|T, \delta)}{p(x|T, \delta)} \\ &= \frac{p(Y, \beta'_\gamma, \gamma, \nu^2, w', c, U, D, K+1 \mid T, \delta)}{p(Y, \beta_\gamma, \gamma, \nu^2, w, c, U, D, K \mid T, \delta)}. \end{aligned}$$

Or nous proposons de créer un composant vide, donc aucune réallocation n'est nécessaire, la vraisemblance des données ne change pas :

$$\frac{p(Y \mid \beta'_\gamma, \gamma, \nu^2, U, c, T, \delta)}{p(Y \mid \beta_\gamma, \gamma, \nu^2, U, c, T, \delta)} = 1.$$

De plus comme $K \sim \mathcal{U}_{[1, K_{max}]}$, $p(K) = p(K+1)$.
Alors

$$\begin{aligned} \frac{\pi(K+1, \theta')}{\pi(K, \theta)} &= \frac{p(\beta'_\gamma \mid \gamma, K+1, \nu^2) p(w' \mid K+1) p(c \mid w', K+1)}{p(\beta_\gamma \mid \gamma, K+1, \nu^2) p(w \mid K+1) p(c \mid w, K+1)} \\ &= (K+1)(1-w_{k^*})^{K(\delta-1)+n} w_{k^*}^{\delta-1} \frac{1}{B(\delta, K\delta)} \\ &\times (2\pi)^{-\frac{d_\gamma}{2}} |\nu^2 \Sigma_\gamma|^{-\frac{1}{2}} \exp\Big[-\frac{1}{2\nu^2} \beta'_{k^*\gamma} \Sigma_\gamma^{-1} \beta_{k^*\gamma} \Big]. \end{aligned}$$

Les facteurs $(k+1)!$ et $k!$ permettent de tenir compte des différentes labellisations possibles, et $B(.,.)$ est la fonction beta.

2.
$$\frac{\pi_{x'\to x}}{\pi_{x\to x'}} = \frac{d_{K+1}/(K_0+1)}{b_K}.$$

3.
$$\begin{aligned}\frac{1}{g(v)} &= \frac{1}{g_1(w_{k^*})g_2(\beta_{k^*\gamma})} \\ &= \frac{1}{be_{1,K}(w_{k^*}) \times (2\pi)^{-\frac{d_\gamma}{2}}|\nu^2\Sigma_\gamma|^{-\frac{1}{2}}\exp\Big[-\frac{1}{2\nu^2}\beta'_{k^*\gamma}\Sigma_\gamma^{-1}\beta_{k^*\gamma}\Big]}.\end{aligned}$$

4. Le jacobien est défini par :
$$\begin{aligned}T : (w_1,\beta_{1\gamma},\ldots,w_K,\beta_{K\gamma},w_{k^*},\beta_{k^*\gamma}) &\longmapsto \Big(w'_1,\beta'_{1\gamma},\ldots,w'_K,\beta'_{K\gamma},w'_{K+1},\beta'_{(K+1)\gamma}\Big) \\ &\longmapsto \Big(w_1(1-w_{k^*}),\beta_{1\gamma},\ldots,w_K(1-w_{k^*}),\beta_{K\gamma},w_{k^*},\beta_{k^*\gamma}\Big).\end{aligned}$$

$$\begin{aligned}\left|\frac{\partial T(\theta,v)}{\partial(\theta,v)}\right| &= \left|\frac{\partial T(w_1,\beta_{1\gamma},\ldots,w_K,\beta_{K\gamma},w_{k^*},\beta_{k^*\gamma})}{\partial(w_1,\beta_{1\gamma},\ldots,w_K,\beta_{K\gamma},w_{k^*},\beta_{k^*\gamma})}\right| \\ &= (1-w_{k^*})^K.\end{aligned}$$

Nous avons donc finalement :
$$A = (K+1)\frac{1}{B(\delta,K\delta)}(1-w_{k^*})^{n+K\delta}w_{k^*}^{\delta-1}\frac{d_{K+1}}{(K_0+1)b_K} \times \frac{1}{be_{1,K}(w_{k^*})}. \tag{B.13}$$

Dans le cas de la mort d'un composant, $\rho = \min(1, A^{-1})$, en utilisant la même expression pour A.

B.2.6 Algorithme de l'échantillonneur

Algorithme Metropolis-within-RJMCMC modifié par la grouping technique

Nous commençons l'algorithme à partir de valeurs initiales: $\nu^{2(0)}, \beta_\gamma^{(0)}, \gamma^{(0)}, w^{(0)}, c^{(0)}, K^{(0)}, U^{(0)}$ et $D^{(0)}$. On impute ensuite les valeurs manquantes de Y à partir de (B.4) pour obtenir $Y^{(0)}$.
A l'itération t, nous avons $\nu^{2(t)}, \beta_\gamma^{(t)}, \gamma^{(t)}, w^{(t)}, c^{(t)}, K^{(t)}, U^{(t)}, D^{(t)}$ et $Y^{(t)}$.
L'itération $t+1$ se déroule en deux grandes étapes:

1. **Mouvements sans changement de dimension :**

   (a) Générer $\gamma^{(t+1)}$ suivant $f(\gamma \mid Y^{(t)}, \nu^{2(t)}, c^{(t)}, U^{(t)}, K^{(t)})$ (voir B.11), grâce à un algorithme de Metropolis-Hasting. Sachant $\gamma^{(t)}, Y^{(t)}, \nu^{2(t)}, c^{(t)}, U^{(t)}$ et $K^{(t)}$, l itérations de l'algorithme MH sont effectuées (l fixé arbitrairement). L'algorithme MH prend $\gamma^{(t)}$ comme valeur initiale. Puis à chaque itération $i+1$:

   i. Générer un candidat γ^*, en changeant les valeurs de r éléments de $\gamma^{(i)}$ (de 0 en 1 ou de 1 en 0).

   ii. Prendre

$$\gamma^{(i+1)} = \begin{cases} \gamma^* & \text{avec probabilité} \quad \rho(\gamma^{(i)}, \gamma^*) \quad \text{voir (B.12)} \\ \gamma^{(i)} & \text{avec probabilité} \quad 1 - \rho(\gamma^{(i)}, \gamma^*) \end{cases}$$

   $\gamma^{(t+1)}$ sera le $\gamma^{(l)}$ obtenu à la $l^{\text{ème}}$ itération de l'algorithme MH.

   (b) Générer β_γ^{temp} en générant $\beta_{k\gamma}^{temp}$ pour $k = 1, \ldots, K$, $f(\beta_{k\gamma} \mid Y^{(t)}, \nu^{2(t)}, U^{(t)}, c^{(t)}, \gamma^{(t+1)})$ (voir (B.7)).

   (c) Générer $\nu^{2(t+1)}$ suivant $f(\nu^2 \mid Y^{(t)}, \beta_\gamma^{temp}, \gamma^{(t+1)}, U^{(t)}, c^{(t)}, K^{(t)})$ (voir (B.9)).

   (d) Générer w^{temp} suivant $f(w \mid c^{(t)}, K^{(t)})$ (voir (B.5)).

   (e) Générer $c^{(t+1)}$ suivant $f(c \mid Y^{(t)}, \nu^{2(t+1)}, \beta_\gamma^{temp}, \gamma^{(t+1)}, w^{temp}, U^{(t)}, K^{(t)})$ (voir (B.6)).

   (f) Générer $U^{(t+1)}$ suivant $f(U \mid Y^{(t)}, \nu^{2(t+1)}, \beta_\gamma^{temp}, \gamma^{(t+1)}, c^{(t+1)}, D^{(t)})$ (voir (B.8)).

   (g) Générer $D^{(t+1)}$ suivant $f(D \mid U^{(t+1)})$ (voir (2.11) ou (2.12)).

   (h) Imputer les valeurs manquantes de Y pour obtenir $Y^{(t+1)}$. Pour i tel que $\delta_i = 0$, générer $Y_i \mid \beta_\gamma^{temp}, \gamma^{(t+1)}, \nu^{2(t+1)}, U^{(t+1)}, c^{(t+1)}, T$, voir (B.4).

2. **Mouvements avec changement de dimension :**

   (a) Choix entre une création et une destruction, à l'aide des probabilités $b_{K^{(t)}}$ et $d_{K^{(t)}}$.

   (b) Si le mouvement choisi est une destruction, choisir le composant vide à éliminer, puis calculer la probabilité d'acceptation $\rho = \min(1, A^{-1})$, voir (B.13).
   Si le mouvement est rejeté, $w^{(t+1)}$ est égal à w^{temp}, $\beta_\gamma^{(t+1)}$ est égal à β_γ^{temp}, et $K^{(t+1)}$ est égal à $K^{(t)}$.
   Si le mouvement est accepté, $w^{(t+1)}$ et $\beta_\gamma^{(t+1)}$ sont obtenus en actualisant w^{temp} et β_γ^{temp}, et $K^{(t+1)}$ est égal à $K^{(t)} - 1$.

(c) Si le mouvement choisi est une création, générer les variables auxiliaires w_{k^*} et $\beta_{k^*\gamma}$ puis calculer la probabilité d'acceptation $\rho = \min(1, A)$, voir (B.13).
Si le mouvement est rejeté, $w^{(t+1)}$ est égal à w^{temp}, $\beta_\gamma^{(t+1)}$ est égal à β_γ^{temp}, et $K^{(t+1)}$ est égal à $K^{(t)}$.
Si le mouvement est accepté, $w^{(t+1)}$ et $\beta_\gamma^{(t+1)}$ sont obtenus en actualisant w^{temp} et β_γ^{temp}, et $K^{(t+1)}$ est égal à $K^{(t)} + 1$.

B.3 Résultats et perspectives

Nous avons appliqué l'algorithme précédent à des données d'Ipsogen constituées de 93 évènements observés seulement sur 426. Nous n'avons pas obtenu de résultat pertinent et encourageant, nous ne sommes donc pas allés plus loin que le développement de la méthode. Cela est principalement dû au fait qu'il y a trop de censures dans nos données.

Une perspective de travail est alors de trouver des jeux de données pertinents pour appliquer la méthode développée. De plus, comme le modèle de mélange utilisé a de nombreux paramètres, il est probable que l'échantillonneur décrit dans la partie précédente ait du mal à bien explorer l'espace des paramètres. Nous envisageons donc d'utiliser l'algorithme PTEEM (chapitre 6).

Annexe C

Le phénomène du « label-switching »

Nous nous plaçons dans le cas d'observations indépendantes y_i, $i \in \{1, \ldots, n\}$, issues d'un mélange de k composants :

$$y_i \sim \sum_{j=1}^{k} w_j f(. \mid \theta_j), \qquad i = 1, \ldots, n, \tag{C.1}$$

où $f(. \mid \theta_i)$ est une famille paramétrique de densités, de paramètre θ_i. Les tailles des différents composants sont proportionnelles aux w_j, qui sont les poids des composants (leur somme vaut 1).
Chaque observation est supposée provenir d'un des composants du mélange. Un vecteur d'étiquettes c peut être introduit, avec $c_i = j$ si l'observation y_i est issue du $j^{\text{ème}}$ composant. Les c_i sont supposés indépendants et issus des distributions :

$$p(c_i = j) = w_j, \qquad j = 1, \ldots, k.$$

Conditionnellement à c, nous savons de quelle population est issu y_i :

$$y_i \mid c \sim f(. \mid \theta_{c_i}), \qquad i = 1, \ldots, n.$$

Nous notons $\theta = (\theta_1, \ldots, \theta_k)$ et $y = (y_1, \ldots, y_n)$.

C.1 Problème d'identifiabilité dans des modèles de mélange

Dans le cas du modèle C.1, la vraisemblance des données est :

$$p(y \mid \theta, k) = \prod_{i=1}^{n} \Big[\sum_{j=1}^{k} w_j f(y_i \mid \theta_j) \Big].$$

Soit σ une permutation de $\{1, \ldots, k\}$, et $\sigma(\theta) = (\theta_{\sigma(1)}, \ldots, \theta_{\sigma(k)})$. La vraisemblance est invariante pour toute permutation σ :

$$p(y \mid \theta, k) = \prod_{i=1}^{n} \Big[\sum_{j=1}^{k} w_{\sigma(j)} f(y_i \mid \theta_{\sigma(j)}) \Big].$$

Par conséquent, si les paramètres du mélange suivent des distributions a priori échangeables ($p(\theta) = p(\sigma(\theta))$), alors la distribution a posteriori des paramètres est invariante par permutation des labels des composants ($p(\theta \mid y) = p(\sigma(\theta) \mid y)$). Cette distribution a posteriori a donc $k!$ modes, et les distributions marginales des paramètres sont identiques pour chacun des composants. C'est pourquoi, lors de la génération de la chaîne MCMC, les labels des différents composants vont être normalement constamment échangés (« label-switching »). Jasra et al. [106] ou encore Marin et al. [148] ont illustré ce problème.

Inférence directe non pertinente

Etant donné que les distributions marginales des paramètres sont identiques pour chacun des composants, il n'est pas pertinent de mener directement une inférence sur ces distributions marginales en utilisant des estimateurs classiques tels que les moyennes : les estimations obtenues seront identiques pour les paramètres des différents composants.

Il est donc nécessaire de labelliser les composants du mélange pour chaque observation de l'échantillon généré. Plusieurs méthodes ont été proposées, quelques unes sont évoquées dans la partie C.2. L'application de ces méthodes permet une interprétation des simulations générées, mais nécessite un effort computationnel souvent non négligeable.

Critère de convergence

Bien que le phénomène de « label-switching » empêche une inférence directe sur les paramètres du mélange, il permet d'avoir une idée sur la convergence de la chaîne générée. Si ce phénomène est observé, cela ne veut pas nécessairement dire que la chaîne a convergé. Cependant, si la chaîne a convergé, alors ce phénomène doit être observé. En effet, dans ce cas elle doit avoir parcouru l'ensemble de l'espace des paramètres, et notamment l'ensemble des « labellisations » possibles.

Visiter chacun des $k!$ modes est redondant sur le plan statistique, car il suffit d'en visiter un et de permuter les labels des composants pour avoir les autres. Si une chaîne atteint ne serait-ce qu'un seul des modes, l'estimation des paramètres peut être correcte. Cependant, si le phénomène de « label-switching » n'est pas observé, cela indique nécessairement que la chaîne n'a pas visité tout l'espace, ce qui n'est pas satisfaisant car nous pouvons alors nous demander si elle a vraiment atteint un des modes.

C.2 Algorithmes de labellisation

Notons qu'il est possible d'imposer un « label-switching » entre les $k!$ modes d'un mélange à k composants, cela correspond à l'algorithme « constrained permutation sampling » de Frühwirth-Schnatter [68]. Une étape est ajoutée à chaque itération d'un échantillonneur classique, consistant à appliquer une permutation aléatoire des labels. Cependant, ce type de mouvement ne permet pas à la chaîne de converger plus vite, il va seulement masquer le fait que la chaîne ne converge pas. Son utilisation n'est donc pas recommandée.

Contrainte d'identifiabilité

La manière la plus naïve de régler le problème du « label-switching » est d'imposer une contrainte d'identifiabilité aux paramètres obtenus à chaque itération. Par exemple, imposer une contrainte d'ordre sur les moyennes des différents composants : $\mu_1 < \mu_2 < \ldots < \mu_k$.

Cette méthode peut être appliquée soit en cours de processing, soit en post-processing, une fois que toute la chaîne a été générée. En règle générale, il est préférable d'effectuer la labellisation des différents composants après le processing (voir Stephens [208]). En effet, si nous labellisons après chaque itération en cours de processing, l'espace où la chaîne peut se déplacer est contraint. Or Celeux et al. [36] ont montré que si l'espace des paramètres est contraint, la capacité de la chaîne a l'explorer est diminuée, et que cela peut même empêcher la chaîne de converger.

Imposer une contrainte d'identifiabilité, même en post-processing, a cependant plusieurs inconvénients, notamment dans le cas de problèmes multivariés. Tout d'abord la contrainte utilisée ne respecte pas forcément la topologie de l'a priori ou de la vraisemblance, et donc la méthode peut échouer à sélectionner un des $k!$ modes de la distribution a posteriori des paramètres. Des parties de différents modes peuvent être sélectionnées, et les moyennes a posteriori obtenues après labellisation peuvent se trouver dans des régions de faibles probabilités. De plus, la contrainte utilisée peut être valable pour certains composants et pas pour d'autres : dans un même modèle, des composants peuvent être proches en moyenne, tandis que d'autres seront proches en variance ou en poids. Enfin, nous pouvons remarquer que l'emploi de différentes contraintes d'identifiabilité mènent à des estimation des paramètres qui peuvent être très différentes (voir des exemples dans Jasra et al. [106] et Celeux et al. [36]).

Algorithmes de labellisation

Une bonne revue des algorithmes classiquement utilisés a été faite par Jasra et al. [106]. Celeux [34] et Stephens [208, 209, 211] ont notamment proposé des algorithmes de labellisation basés sur des fonction de coût et de risque, et utilisant des risques de Monte Carlo. Le premier algorithme présenté par Stephens dans [211] généralise les algorithmes précédemment proposés dans [34, 208, 209]. Le second algorithme qu'il présente dans [211] utilise la divergence de Kullback-Leibler

comme fonction de coût et permet de classifier les observations dans les différents composants du mélange. Jasra et al. [106] considèrent cependant que ces algorithmes induisent une contrainte d'identifiabilité, qu'ils ne sont qu'un moyen de trouver une contrainte d'identifiabilité de manière automatique.

Des méthodes utilisant des fonctions de coût invariantes par rapport aux labels ont ensuite été proposées (voir Celeux et al. [36] et Hurn et al. [101]). D'après Jasra et al. [106] ces méthodes sont plus satisfaisantes d'un point de vue bayésien que les méthodes induisant une contrainte d'identifiabilité. Cependant, elles ont un coût computationnel assez élevé et il peut être difficile de construire une classe de fonctions de coût invariantes adaptées à nos objectifs.

De nombreuses autres méthodes ont été proposées, voir notamment Chung et al. [46], Papastamoulis et Iliopoulos [174], Puolamäki et Kaski [178], Yao [233], Sperrin et al. [206] ou encore Yao et Lindsay [234].

Annexe D

Suppléments de la partie II

D.1 Suppléments du chapitre 5

D.1.1 Mouvements à rejet retardé

Mouvements Splitting Rejection

La technique des mouvements « Splitting Rejection » a été développée par Mira [159] et Tierney et Mira [222].
Dans un algorithme de Metropolis-Hastings classique de distribution stationnaire π, notons $X_t = x$ l'état de la chaîne de Markov au temps t. Un nouvel état y est proposé suivant un noyau de transition $q_1(x, dy)$. Si cet état est accepté nous posons $X_{t+1} = y$, et s'il est rejeté la chaîne reste dans le même état avec $X_{t+1} = x$.
Plus le nombre de propositions rejetées au cours de l'algorithme est élevé, et plus les autocorrélations de la chaîne augmentent. Les estimateurs obtenus d'après cette chaîne sont alors moins efficaces au sens de Peskun (voir Peskun [176] et Tierney [221] pour des justifications théoriques). Suivant ce constat, l'objectif de Tierney et Mira [222] est de diminuer le nombre de propositions rejetées au cours d'un algorithme de Metropolis-Hastings.

Le principe est simplement le suivant : lorsqu'une première proposition est rejetée, une seconde proposition est effectuée (dépendant éventuellement de la proposition rejetée), au lieu de passer directement à l'itération suivante. Si le deuxième candidat est également rejeté, soit nous posons $X_{t+1} = x$, soit nous faisons une troisième proposition, et ainsi de suite ...

Les probabilités d'acceptation des seconde, troisième, et autres propositions sont définies par Tierney et Mira [222] dans le but de conserver la distribution stationnaire de la chaîne. Ils utilisent pour cela la « detailed balance condition », qui est suffisante pour que π soit la distribution stationnaire de la chaîne (voir par exemple Robert et Casella [186]).
Le premier candidat proposé y suivant $q_1(x, dy)$ est accepté avec la probabilité d'acceptation

suivante :

$$\alpha_1(x,y) = 1 \wedge \frac{\pi(y)q_1(y,x)}{\pi(x)q_1(x,y)}. \tag{D.1}$$

Si ce candidat y est rejeté, un état z est proposé suivant un nouveau noyau de transition $q_2(x,y,dz)$. Pour passer de l'état x à un état z, soit z est proposé dés la première proposition et accepté, soit un état y est proposé lors de la première proposition, rejeté, puis z est proposé lors de la seconde proposition et accepté. Le noyau de transition pour passer de x à $z \neq x$ s'écrit :

$$q_1(x,z)\alpha_1(x,z) + \int q_1(x,y)[1-\alpha_1(x,y)]q_2(x,y,z)\alpha_2(x,y,z)dy.$$

La « detailed balance condition » est alors la suivante :

$$\pi(x)\int q_1(x,y)[1-\alpha_1(x,y)]q_2(x,y,z)\alpha_2(x,y,z)dy = \pi(z)\int q_1(z,y)[1-\alpha_1(z,y)]q_2(z,y,x)\alpha_2(z,y,x)dy.$$

Une condition suffisante pour la vérifier est l'égalité des intégrandes :

$$\pi(x)q_1(x,y)[1-\alpha_1(x,y)]q_2(x,y,z)\alpha_2(x,y,z) = \pi(z)q_1(z,y)[1-\alpha_1(z,y)]q_2(z,y,x)\alpha_2(z,y,x),$$

et la plus petite probabilité $\alpha_2(x,y,z)$ vérifiant cette équation est :

$$\alpha_2(x,y,z) = 1 \wedge \frac{\pi(z)q_1(z,y)[1-\alpha_1(z,y)]q_2(z,y,x)}{\pi(x)q_1(x,y)[1-\alpha_1(x,y)]q_2(x,y,z)}. \tag{D.2}$$

C'est la probabilité d'acceptation pour le second état proposé z, à partir de x, et en ayant refusé y. Notons que l'égalité des intégrandes est ici une hypothèse forte, puisque cela suppose que l'état y à l'aller pour passer de x à z est le même que l'état y au retour pour passer de z à x.
Le raisonnement est le même pour écrire la probabilité d'acceptation d'un troisième candidat si le second candidat z est rejeté, d'un $i^{\text{ème}}$ candidat si les $(i-1)$ candidats précédents ont tous été rejetés,.... Cependant, il est fréquent en pratique de s'arrêter à une seconde proposition. En effet, plus il y a de propositions en cas de rejet, et plus les calculs des probabilités d'acceptation sont fastidieux, contrebalançant le bénéfice de ne pas passer à l'itération suivante.
Mira [159] a montré que l'algorithme de Metropolis-Hastings utilisant ces mouvements « Splitting Rejection » donne de meilleurs résultats que l'algorithme de Metropolis-Hastings classique au sens de Peskun [176]. Cependant, une itération de cet algorithme est en moyenne plus longue qu'une itération de l'algorithme de Metropolis-Hastings classique.

Généralisation des mouvements Splitting Rejection : Delayed Rejection

Green et Mira [85] ont généralisé les mouvements « Splitting Rejection » au cas trans-dimensionnel, et notamment aux méthodes à sauts réversibles [83]. Dans le développement de

la probabilité d'acceptation (D.2) ci-dessus, il est supposé que l'état y à l'aller pour passer de x à z est le même que l'état y au retour pour passer de z à x, ce qui est une hypothèse forte, et parfois impossible à vérifier. Green et Mira [85] ont donc supprimé cette hypothèse, en permettant à l'aller de passer par un état y, et au retour par un $y*$, égal ou non à y. En effet, la distribution stationnaire de l'échantillonneur peut être conservée même si ce ne sont pas les mêmes états intermédiaires qui sont utilisés à l'aller et au retour. L'échantillonneur gagne alors en efficacité et permet de définir des passages entre deux états de manière plus intuitive dans le cadre de modèles à dimensions variables. Notons que ces techniques de « Splitting Rejection » et de « Delayed Rejection » diminuent les autocorrélations de la chaîne, mais ne permettent pas à la chaîne de mieux explorer l'espace des paramètres si celle-ci a tendance à rester bloquée dans les modes locaux.

D.1.2 Target Oriented Evolutionary Monte Carlo

Quatre autres mouvements globaux ont été proposés par Goswami et liu [82]. Pour trois de ces mouvements, une chaîne d'indice i associée à une température faible est choisie aléatoirement, puis une seconde chaîne d'indice j est choisie. Contrairement à un mouvement d'échange classique, la chaîne d'indice j n'est pas choisie uniformément parmi les $(N-1)$ chaînes restantes : les chaînes d'indices inférieurs à i (donc de températures inférieures à T_i) ont des probabilités nulles d'être choisies, et les chaînes d'indices supérieurs à i sont associées à des probabilités qui dépendent du mouvement choisi : $\pi_i(x_j)$ pour le « best chromosome exchange », $\pi_i(x_j)/\pi_j(x_j)$ pour le « best importance ratio exchange », et $\pi_i(x_j)\pi_j(x_i)$ pour le « best swap exchange ».
Le quatrième mouvement que Goswami et liu [82] proposent est un mouvement cyclique permettant d'actualiser les états de plus de deux chaînes en même temps (« cyclic exchange »).

D.2 Suppléments du chapitre 6

En combinaison avec un échantillonneur de Gibbs : densités a posteriori jointes
La densité cible de la $i^{\text{ème}}$ chaîne est la suivante :

$$\begin{aligned}
\pi_i'(x) &\propto p(y \mid \mu, \sigma^{-2}, c)^{\frac{1}{T_i}} p(\mu, \sigma^{-2}, c, \beta, w) \\
&\propto \Big[\prod_{p=1}^{k} (\sigma_p \sqrt{(2\pi)})^{-m_p} \exp\Big(-\sum_{l=1}^{n} \frac{(y_l - \mu_{c_l})^2}{2\sigma_{c_l}^2}\Big)\Big]^{\frac{1}{T_i}} \Big[\prod_{p=1}^{k} \frac{\kappa^{\frac{1}{2}}}{\sqrt{2\pi}} \exp\Big(-\frac{1}{2}(\mu_p - \xi)^2 \kappa\Big)\Big] \\
&\quad \times \Big[\prod_{p=1}^{k} \frac{\beta^\alpha}{\Gamma(\alpha)} \sigma_p^{-2(\alpha-1)} \exp(-\beta\sigma_p^{-2})\Big] \Big[\frac{n!}{\prod_{p=1}^{k} m_p!} \prod_{p=1}^{k} w_p^{m_p}\Big] \\
&\quad \times \Big[\frac{1}{B(\delta, \ldots, \delta)} \prod_{p=1}^{k} w_p^{\delta-1}\Big] \Big[\beta^{g-1} \exp(-h\beta) \frac{h^g}{\Gamma(g)}\Big], \qquad \text{(D.3)}
\end{aligned}$$

avec

$$B(\delta, \ldots, \delta) = \frac{\prod_{p=1}^{k} \Gamma(\delta)}{\Gamma(\sum_{p=1}^{k} \delta)} = \frac{\Gamma(\delta)^k}{\Gamma(k\delta)}.$$

Application au jeu de données Galaxy

	chaîne 1	chaîne 10	chaîne 20
chaîne 1	0.00	0.02	0.00
chaîne 2	63.65	0.10	0.00
chaîne 3	23.90	0.32	0.00
chaîne 4	7.75	0.87	0.00
chaîne 5	2.78	1.77	0.00
chaîne 6	1.12	3.30	0.00
chaîne 7	0.47	6.45	0.00
chaîne 8	0.23	12.65	0.01
chaîne 9	0.07	22.44	0.05
chaîne 10	0.02	0.00	0.23
chaîne 11	0.00	21.39	0.67
chaîne 12	0.00	13.92	1.52
chaîne 13	0.00	7.99	3.12
chaîne 14	0.00	4.29	5.46
chaîne 15	0.00	2.12	8.56
chaîne 16	0.00	1.11	12.18
chaîne 17	0.00	0.61	16.58
chaîne 18	0.00	0.34	22.37
chaîne 19	0.00	0.19	29.26
chaîne 20	0.00	0.13	0.00

TABLE D.1 – Proportions (%) d'échanges Equi-Energy acceptés impliquant la chaîne 1 avec chacune des autres chaînes (moyenne sur 100 runs PTEEM). Idem pour les chaînes 10 et 20.

Application à la sensibilité de *Plasmodium Falciparum* à la doxycycline

		run 1	run 2	run 3	moyenne
moyennes μ	μ_1	10.66	10.32	11.05	10.68
	μ_2	23.21	23.20	23.27	23.23
	μ_3	31.20	31.28	31.59	31.36
	μ_4	44.17	43.51	44.67	44.12
variances σ	σ_1	5.34	5.17	5.36	5.30
	σ_2	4.63	4.60	4.79	4.67
	σ_3	7.87	7.76	7.87	7.83
	σ_4	11.72	11.84	11.70	11.75
poids w	w_1	0.10	0.09	0.10	0.10
	w_2	0.51	0.52	0.52	0.52
	w_3	0.27	0.26	0.26	0.26
	w_4	0.12	0.12	0.12	0.12
nombre d'observations m	m_1	50	49	50	
	m_2	480	486	486	
	m_3	148	142	144	
	m_4	45	46	43	
paramètre β		0.87	0.86	0.88	0.87

TABLE D.2 – Composants a posteriori estimés à partir de la première chaîne, pour trois runs du PTEEM en combinaison avec un échantillonneur de Gibbs, en ayant fixé $k = 4$.

Recherche de sites promoteurs de facteurs de transcription

Calibration des températures et des niveaux d'énergies :

En ce qui concerne les runs de PTEEM : 5 anneaux d'énergie ont été utilisés, avec des énergies régulièrement réparties en échelle logarithmique entre 10 et 100, ce qui donne les seuils 10, 17.78, 31.62, 56.23 et 100. Les températures ont été choisies de façon à ce que leurs inverses soient régulièrement répartis entre 1 et 1/1.3, ce qui donne $T_{min} = 1$ et $T_{max} = 1.3$. Nous avons vérifié que les échanges se faisaient bien entre les chaînes, et que la chaîne associée à T_{max} parcourait bien l'espace des paramètres.

En ce qui concerne les runs de l'EES : 9 anneaux d'énergie ont été utilisés, avec des énergies régulièrement réparties en échelle logarithmique entre 10 et 100. Les températures utilisées sont les suivantes : 1, 1.001, 1.002, 1.005, 1.01, 1.02, 1.06, 1.1 et 1.3. Le choix de ces températures s'est fait de manière à permettre des sauts entre les chaînes. Si par exemple nous avions posé $T_1 = 1$ et $T_2 = 1.1$, aucun saut n'aurait été possible entre les chaînes 1 et 2, car les énergies des chaînes associées à ces températures sont trop éloignées pour cet exemple.

D.3 Suppléments du chapitre 7

ABC-PRC

Pour obtenir n particules:

1. Initialiser une séquence $\epsilon_1 > \epsilon_2 > \ldots > \epsilon_T = \epsilon$, spécifier la distribution initiale π_1, initialiser le compteur de population (itération) à $t = 1$.
2. Pour l'itération $t = 1$:
   (a) Pour chaque compteur de particules i allant de 1 à n:
      i) Générer $\theta_i^{(1)}$ suivant π_1.
      ii) Générer z' sachant $\theta_i^{(1)}$ suivant le modèle défini par $f(. \mid \theta_i^{(1)})$.
      iii) Répéter i) et ii) jusqu'à ce que $\rho(S(z'), S(x)) < \epsilon_1$.
      iv) Poser $w_i^{(1)} = \pi(\theta_i^{(1)})/\pi_1(\theta_i^{(1)})$.
   (b) Normaliser les poids $\{w_j^{(1)}\}_{j=1,\ldots,n}$.
3. Pour les itérations $t = 2, \ldots, T$:
   (a) Pour chaque compteur de particules i allant de 1 à n:
      i) Echantillonner θ_i^* depuis la population précédente $\{\theta_j^{(t-1)}\}_{j=1,\ldots,n}$ munie des poids $\{w_j^{(t-1)}\}_{j=1,\ldots,n}$.
      ii) Générer $\theta_i^{(t)}$ suivant $K_t(\theta \mid \theta_i^*)$, puis z' sachant $\theta_i^{(t)}$ suivant le modèle défini par $f(. \mid \theta_i^{(t)})$.
      iii) Répéter i) et ii) jusqu'à ce que $\rho(S(z'), S(x)) < \epsilon_t$.
      iv) Poser
      $$w_i^{(t)} = \frac{\pi(\theta_i^{(t)}) L_{t-1}(\theta_i^* \mid \theta_i^{(t)})}{\pi(\theta_i^*) K_t(\theta_i^{(t)} \mid \theta_i^*)}. \tag{D.4}$$
   (b) Normaliser les poids $\{w_j^{(t)}\}_{j=1,\ldots,n}$.
   (c) Si $EES = 1/\sum_{j=1}^n w_j^{(t)2} < E$, alors rééchantillonner avec remise les particules $\{\theta_j^{(t)}\}_{j=1,\ldots,n}$ munies des poids $\{w_j^{(t)}\}_{j=1,\ldots,n}$.
   On obtient une nouvelle population $\{\theta_j^{(t)}\}_{j=1,\ldots,n}$ et on définit les nouveaux poids comme $\{w_j^{(t)} = 1/n\}_{j=1,\ldots,n}$.

ABC-PMC

Pour obtenir n particules:

1. Initialiser une séquence $\epsilon_1 > \epsilon_2 > \ldots > \epsilon_T = \epsilon$, initialiser le compteur de population (itération) à $t = 1$.
2. Pour l'itération $t = 1$:
   (a) Pour chaque compteur de particules i allant de 1 à n:
      i) Générer $\theta_i^{(1)}$ suivant $\pi(.)$.
      ii) Générer z' sachant $\theta_i^{(1)}$ suivant le modèle défini par $f(. \mid \theta_i^{(1)})$.
      iii) Répéter i) et ii) jusqu'à ce que $\rho(S(z'), S(x)) < \epsilon_1$.
      iv) Poser $w_i^{(1)} = 1/n$.
   (b) Poser τ_2^2 comme étant égal à deux fois la variance empirique des $\{\theta_j^{(1)}\}_{j=1,\ldots,N}$.
3. Pour les itérations $t = 2, \ldots, T$:
   (a) Pour chaque compteur de particules i allant de 1 à n:
      i) Echantillonner θ_i^* depuis la population précédente $\{\theta_j^{(t-1)}\}_{j=1,\ldots,n}$ munie des poids $\{w_j^{(t-1)}\}_{j=1,\ldots,n}$.
      ii) Générer $\theta_i^{(t)} \sim \mathcal{N}(\theta_i^*, \tau_t^2)$, puis z' sachant $\theta_i^{(t)}$ suivant le modèle défini par $f(. \mid \theta_i^{(t)})$.
      iii) Répéter i) et ii) jusqu'à ce que $\rho(S(z'), S(x)) < \epsilon_t$.
      iv) Poser
      $$w_i^{(t)} = \frac{\pi(\theta_i^{(t)})}{\sum_{j=1}^n w_j^{(t-1)} \phi\Big(\tau_t^{-1}(\theta_i^{(t)} - \theta_j^{(t-1)})\Big)},$$
      ϕ étant la densité de probabilité de la loi normale centrée réduite.
   (b) Normaliser les poids $\{w_j^{(t)}\}_{j=1,\ldots,n}$.
   (c) Poser τ_{t+1}^2 comme étant égal à deux fois la variance empirique pondérée des $\{\theta_j^{(t)}\}_{j=1,\ldots,N}$.

ABC-SMC

Pour obtenir n particules:

1. Initialiser une séquence $\epsilon_1 = \infty > \epsilon_2 > \ldots > \epsilon_T = \epsilon$, initialiser le compteur de population (itération) à $t = 1$.
2. Pour l'itération $t = 1$: Pour chaque compteur de particules i allant de 1 à n:

(a) Générer $\theta_i^{(1)}$ suivant $\pi(.)$.

(b) Générer $z_{1:M,i}^{(1)}$ sachant $\theta_i^{(1)}$: on génère indépendamment les $z_{k,i}^{(1)}$ suivant le modèle défini par $f(. \mid \theta_i^{(1)})$, pour $k = 1, \ldots, M$.

(c) Poser $w_i^{(1)} = 1/n$, on a alors $PA\Big(\{w_i^{(1)}\}_{i=1,\ldots,n}, \epsilon_1\Big) = 1$.

3. Incrémenter l'itération t: $t \leftarrow t + 1$.

(a) Si $\epsilon_{t-1} = \epsilon$, STOP. Sinon, déterminer ϵ_t en résolvant

$$PA\Big(\{w_i^{(t)}\}_{i=1,\ldots,n}, \epsilon_t\Big) = \alpha PA\Big(\{w_i^{(t-1)}\}_{i=1,\ldots,n}, \epsilon_{t-1}\Big), \qquad 0 < \alpha < 1$$

En utilisant

$$w_i^{(t)} \quad \propto \quad w_i^{(t-1)} \frac{\sum_{k=1}^{M} \mathbb{1}_{\{\rho\left(S(z_{k,i}^{(t-1)}), S(x)\right) < \epsilon_t\}}(z_{k,i}^{(t-1)})}{\sum_{k=1}^{M} \mathbb{1}_{\{\rho\left(S(z_{k,i}^{(t-1)}), S(x)\right) < \epsilon_{t-1}\}}(z_{k,i}^{(t-1)})}.$$

L'obtention de ϵ_t se fait en utilisant une méthode de bissection.
Si nous obtenons $\epsilon_t < \epsilon$, poser $\epsilon_t = \epsilon$.

(b) Si $EES\Big[\{w_i^{(t)}\}_{i=1,\ldots,n}\Big] = 1/\sum_{i=1}^{n} w_i^{(t)2} < E$, alors rééchantillonner: tirer avec remise n particules parmi les $\Big\{(z_{1:M,i}^{(t-1)}, \theta_i^{(t-1)})\Big\}_{i=1,\ldots,n}$ munies des poids $\{w_i^{(t)}\}_{i=1,\ldots,n}$.
On obtient une nouvelle population notée $\Big\{(z_{1:M,i}^{(t)}, \theta_i^{(t)})\Big\}_{i=1,\ldots,n}$ et on définit les nouveaux poids comme $\{w_i^{(t)} = 1/n\}_{i=1,\ldots,n}$.

(c) Pour chaque compteur de particules i allant de 1 à n:
Si $w_i^{(t)} > 0$, échantillonner $(z_{1:M,i}^{(t)}, \theta_i^{(t)})$ suivant un noyau de transition $K_t\Big(. \mid (z_{1:M,i}^{(t-1)}, \theta_i^{(t-1)})\Big)$ de distribution stationnaire $\pi_{\epsilon_t}(z_{1:M}, \theta \mid x)$. En pratique, une modification de l'algorithme ABC-MCMC est utilisée:

i) Générer θ_i^* suivant $q_t(. \mid \theta_i^{(t-1)})$.

ii) Générer $z_{1:M,i}^*$ sachant θ_i^*: on génère indépendamment les $z_{k,i}^*$ suivant le modèle défini par $f(. \mid \theta_i^*)$, pour $k = 1, \ldots, M$.

iii) `Poser` $(z^{(t)}_{1:M,i}, \theta^{(t)}_i) = (z^*_{1:M,i}, \theta^*_i)$ `avec une probabilité`

$$\begin{aligned}\alpha &= min\left\{1, \frac{\pi_{\epsilon_t}(z^*_{1:M,i}, \theta^*_i \mid x) q_t(\theta^{(t-1)}_i \mid \theta^*_i) \prod_{k=1}^M f(z^{(t-1)}_{k,i} \mid \theta^{(t-1)}_i)}{\pi_{\epsilon_t}(z^{(t-1)}_{1:M,i}, \theta^{(t-1)}_i \mid x) q_t(\theta^*_i \mid \theta^{(t-1)}_i) \prod_{k=1}^M f(z^*_{k,i} \mid \theta^*_i)}\right\} \\ &= min\left\{1, \frac{q_t(\theta^{(t-1)}_i \mid \theta^*_i) \sum_{k=1}^M \mathbb{1}_{\{\rho\left(S(z^*_{k,i}), S(x)\right) < \epsilon_t\}}(z^{(t-1)}_{k,i})}{q_t(\theta^*_i \mid \theta^{(t-1)}_i) \sum_{k=1}^M \mathbb{1}_{\{\rho\left(S(z^{(t-1)}_{k,i}), S(x)\right) < \epsilon_t\}}(z^{(t-1)}_{k,i})}\right\}. \qquad \text{(D.5)}\end{aligned}$$

`Sinon, poser` $(z^{(t)}_{1:M,i}, \theta^{(t)}_i) = (z^{(t-1)}_{1:M,i}, \theta^{(t-1)}_i)$.

`Retourner à l'étape 3.`

D.4 Suppléments du chapitre 8

Exemple jouet modifié

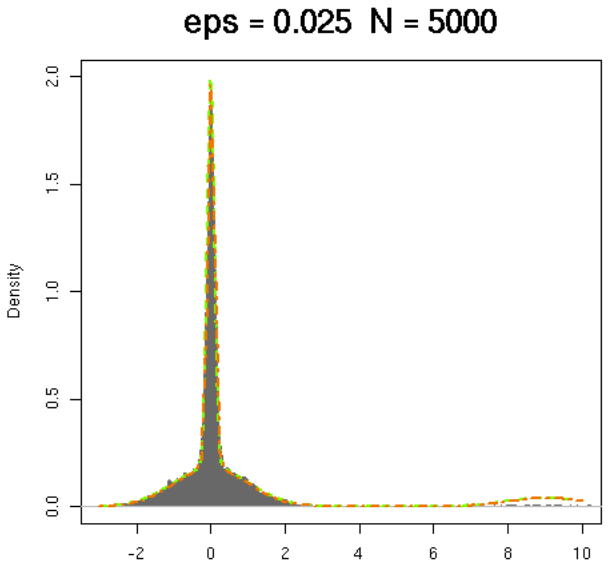

FIGURE D.1 – ABC-PMC : population associée à un seuil de 0.025, en utilisant le modèle (8.12) (5000 particules). L'a posteriori exact est en pointillés oranges, l'approximation cible est en pointillés verts, et la densité de la population pondérée obtenue est grisée.

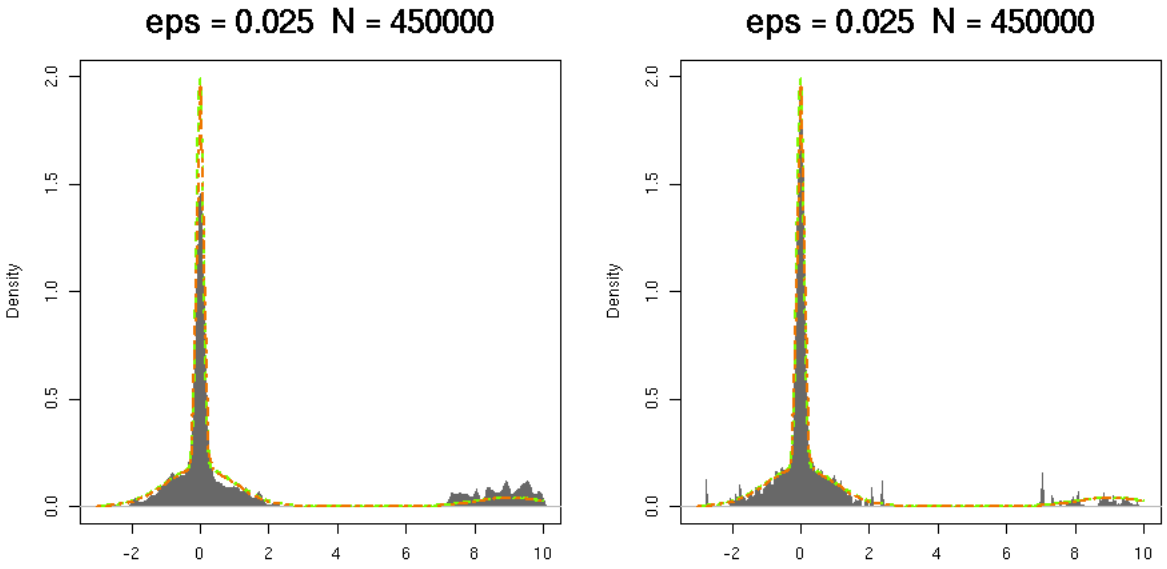

FIGURE D.2 – ABC-PT : pour deux runs différents, simulations obtenues pour la première chaîne associée au seuil 0.025 (600 000 itérations dont 150 000 de burn-in, pour le modèle (8.12)). L'a posteriori exact est en pointillés oranges, l'approximation cible est en pointillés verts, et les densités des simulations obtenues sont grisées.

Transmission de la tuberculose

	ABC standard			ABC-MCMC			ABC-PMC		
	Moy.	Méd.	IC 95%	Moy.	Méd.	IC 95%	Moy.	Méd.	IC 95%
Taux de transmission	0.54	0.54	(0.23,0.86)	0.59	0.58	(0.30,0.93)	0.52	0.51	(0.20-0.87)
Temps de doublement	1.53	1.29	(0.80,3.04)	1.33	1.19	(0.75,2.29)	1.67	1.36	(0.80-3.44)
Taux de reproduction	7.27	2.11	(1.11,19.71)	8.49	2.39	(1.25,19.72)	6.00	2.20	(1.08,17.20)
Taux de mutation	0.24	0.24	(0.14,0.34)	0.25	0.24	(0.15,0.35)	0.24	0.23	(0.13,0.34)

TABLE D.3 – Estimations a posteriori des paramètres d'intérêt obtenues par l'ABC standard, L'ABC-MCMC et la dernière population de l'ABC-PMC : moyenne, médiane et intervalle de crédibilité à 95%.

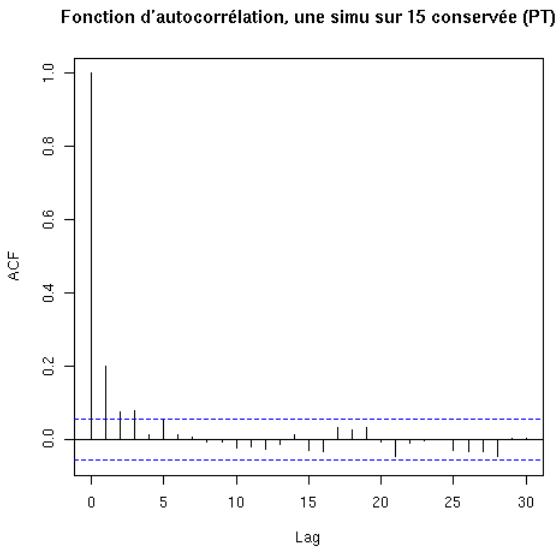

FIGURE D.3 – ABC-PT : pour le paramètre α, fonction d'autocorrélation de la première chaîne pour laquelle 1 itération sur 15 est retenue (20 000 itérations dont 2 000 de burn-in).

Publications et Communications

Articles publiés :

[1] Baragatti M. Bayesian variable selection for probit mixed models, applied to gene selection. *Bayesian Analysis*, 2011, 6(2) :209-230.

[2] Baragatti M, Pommeret D. Comments on Bayesian variable selection for disease classification using gene expression data. *Bioinformatics*, 2011, 27(8) :1194.

Articles en révision :

[3] Baragatti M, Grimaud A, Pommeret D. Parallel Tempering with Equi-Energy moves. *En révision*, 2011.

[4] Baragatti M, Pommeret D. A study of variable selection using g-prior distribution with ridge parameter. *En révision*, 2011.

Articles soumis :

[5] Baragatti M, Grimaud A, Pommeret D. Likelihood free Parallel Tempering. *Soumis*, 2011.

Articles en préparation :

[6] Baragatti M, Grimaud A, Pommeret D. A smooth statistic for ABC methods.

[7] Baragatti M, Briolant S, Nosten F. Article sur la chimiosensibilité de *Plasmodium Falciparum* à la doxycycline.

Conférences :

[1] Eurocancer, COBRED workshop : Palais des congrés de Paris, 21 juin 2011.

[2] 43èmes Journées de Statistique, SFdS : Tunis, du 23 au 27 mai 2011.

[3] The XXVth International Biometric Conference : Florianopolis, Brésil, du 5 au 10 décembre 2010.

[4] 42èmes Journées de Statistique, SFdS : Marseille, du 24 au 28 mai 2010.

[5] SMPGD'10, Statistical methods for post-genomic data : Marseille, 14 et 15 janvier 2010.

Séminaires :

[6] Réunion AppliBugs : AgroParisTech, 17 juin 2011.

[7] Séminaire statistique et applications, Institut de Mathématiques de Luminy : Marseille 6 juin 2011.

[8] Groupe de travail en Statistiques MAP5 : Paris Descartes, 1er avril 2011.

[9] Séminaire de Probabilités et Statistiques de l'université de Montpellier 2 : Montpellier, 28 mars 2011.

Résumé

Cette thèse se décompose en deux parties. Dans un premier temps nous nous intéressons à la sélection bayésienne de variables dans un modèle probit mixte. L'objectif est de développer une méthode pour sélectionner quelques variables pertinentes parmi plusieurs dizaines de milliers tout en prenant en compte le design d'une étude, et en particulier le fait que plusieurs jeux de données soient fusionnés. Le modèle de régression probit mixte utilisé fait partie d'un modèle bayésien hiérarchique plus large et le jeu de données est considéré comme un effet aléatoire. Cette méthode est une extension de la méthode de Lee et al. [131]. La première étape consiste à spécifier le modèle ainsi que les distributions a priori, avec notamment l'utilisation de l'a priori conventionnel de Zellner (g-prior) pour le vecteur des coefficients associé aux effets fixes [238]. Dans une seconde étape, nous utilisons un algorithme Metropolis-within-Gibbs couplé à la grouping (ou blocking) technique de Liu [141] afin de surmonter certaines difficultés d'échantillonnage. Ce choix a des avantages théoriques et computationnels. La méthode développée est appliquée à des jeux de données microarray sur le cancer du sein. Cependant elle a une limite : la matrice de covariance utilisée dans le g-prior doit nécessairement être inversible. Or il y a deux cas pour lesquels cette matrice est singulière : lorsque le nombre de variables sélectionnées dépasse le nombre d'observations, ou lorsque des variables sont combinaisons linéaires d'autres variables. Nous proposons donc une modification de l'a priori de Zellner en y introduisant un paramètre de type ridge, ainsi qu'une manière de choisir les hyper-paramètres associés. L'a priori obtenu est un compromis entre le g-prior classique et l'a priori supposant l'indépendance des coefficients de régression, et se rapproche d'un a priori précédemment proposé par Gupta et Ibrahim [89].

Dans une seconde partie nous développons deux nouvelles méthodes MCMC basées sur des populations de chaînes. Dans le cas de modèles complexes ayant de nombreux paramètres, mais où la vraisemblance des données peut se calculer, l'algorithme Equi-Energy Sampler (EES) introduit par Kou et al. [118] est apparemment plus efficace que l'algorithme classique du Parallel Tempering (PT) introduit par Geyer [76]. Cependant, il est difficile d'utilisation lorsqu'il est couplé avec un échantillonneur de Gibbs, et nécessite un stockage important de valeurs. Nous proposons un algorithme combinant le PT avec le principe d'échanges entre chaînes ayant des niveaux d'énergie similaires dans le même esprit que l'EES. Cette adaptation appelée Parallel Tempering with Equi-Energy Moves (PTEEM) conserve l'idée originale qui fait la force de l'algorithme EES tout en assurant de bonnes propriétés théoriques et une utilisation facile avec un échantillonneur de Gibbs.

Enfin, dans certains cas complexes l'inférence peut être difficile car le calcul de la vraisemblance des données s'avère trop coûteux, voire impossible. De nombreuses méthodes sans vraisemblance ont été développées. Par analogie avec le Parallel Tempering, nous proposons une méthode appelée ABC-Parallel Tempering, basée sur la théorie des MCMC, utilisant une population de chaînes et permettant des échanges entre elles.

Mots-clés : *sélection bayésienne de variables, modèle probit mixte, a priori de Zellner, paramètre ridge, Monte Carlo Markov Chains, Parallel Tempering, Equi-Energy Sampler, Approximate Bayesian Computation, méthodes sans vraisemblance.*

Abstract

This thesis is divided into two main parts. In the first part, we propose a Bayesian variable selection method for probit mixed models. The objective is to select few relevant variables among tens of thousands while taking into account the design of a study, and in particular the fact that several datasets are merged together. The probit mixed model used is considered as part of a larger hierarchical Bayesian model, and the dataset is introduced as a random effect. The proposed method extends a work of Lee et al. [131]. The first step is to specify the model and prior distributions. In particular, we use the g-prior of Zellner [238] for the fixed regression coefficients. In a second step, we use a Metropolis-within-Gibbs algorithm combined with the grouping (or blocking) technique of Liu [141]. This choice has both theoritical and practical advantages. The method developed is applied to merged microarray datasets of patients with breast cancer. However, this method has a limit : the covariance matrix involved in the g-prior should not be singular. But there are two standard cases in which it is singular : if the number of observations is lower than the number of variables, or if some variables are linear combinations of others. In such situations we propose to modify the g-prior by introducing a ridge parameter, and a simple way to choose the associated hyper-parameters. The prior obtained is a compromise between the conditional independent case of the coefficient regressors and the automatic scaling advantage offered by the g-prior, and can be linked to the work of Gupta and Ibrahim [89].

In the second part, we develop two new population-based MCMC methods. In cases of complex models with several parameters, but whose likelihood can be computed, the Equi-Energy Sampler (EES) of Kou et al. [118] seems to be more efficient than the Parallel Tempering (PT) algorithm introduced by Geyer [76]. However it is difficult to use in combination with a Gibbs sampler, and it necessitates increased storage. We propose an algorithm combining the PT with the principle of exchange moves between chains with same levels of energy, in the spirit of the EES. This adaptation which we are calling Parallel Tempering with Equi-Energy Move (PTEEM) keeps the original idea of the EES method while ensuring good theoretical properties and a practical use in combination with a Gibbs sampler. Then, in some complex models whose likelihood is analytically or computationally intractable, the inference can be difficult. Several likelihood-free methods (or Approximate Bayesian Computational Methods) have been developed. We propose a new algorithm, the Likelihood Free-Parallel Tempering, based on the MCMC theory and on a population of chains, by using an analogy with the Parallel Tempering algorithm.

Keywords : *Bayesian variable selection, probit mixed model, Zellner g-prior, ridge parameter, Monte Carlo Markov Chains, Parallel Tempering, Equi-Energy Sampler, Approximate Bayesian Computation, Likelihood-Free methods.*