Rapport de Stage: Méthodes MCMC: amélioration d'un algorithme d'adaptation régionale et applications à la climatologie

Nicolas Grenon-Godbout
Département de mathématiques et statistique
Université de Montréal
Sous la direction de Mylène Bédard

7 novembre 2015

1 Introduction

Les méthodes de Monte-Carlo par chaînes de Markov sont une classe d'algorithmes permettant de simuler des variables aléatoires provenant de distributions complexes, d'abord utilisées en physique statistique [9] puis développées subséquemment de façon plus générale par [7] . Elles sont désormais largement utilisées dans de nombreux domaines d'application de la statistique et leurs aspects théorique et pratique sont l'objet de nombreuses études. Notamment, plusieurs recherches ont tenté de déterminer une calibration optimale de ces algorithmes pour un problème donné, ce qui a mené au cours des quinze dernières années au développement d'algorithmes dits adaptatifs. Plus récemment encore, de nouveaux algorithmes, avec adaptation régionale, ont été introduits dans le but de simuler efficacement des distributions que l'on pourrait qualifier de plus exigeantes, pensons par exemple à des lois bimodales ou asymétriques. Ces algorithmes de dernière génération ont comme particularité de baser leurs calculs sur l'emplacement dans l'espace des variables déjà simulées, d'où leur nom. Nous verrons deux algorithmes populaires de ce genre : le premier, relativement efficace, est toutefois sensible à un mauvais choix de paramètres initiaux. Le second, conçu entre autres choses pour pallier à ce problème, a comme défaut d'être plus

demandant au niveau informatique, donc plus lent. Nous proposons dans ce rapport un troisième algorithme qui se veut un compromis entre ces deux options.

Dans un second temps, nous verrons un exemple d'application des méthodes MCMC à un problème de paramétrisation dans les modèles climatiques. Ces paramétrisations servent à estimer l'effet sur le climat de phénomènes dont le comportement se déroule à petite échelle, ce qui les rend impossibles à simuler numériquement en un temps acceptable. Il n'y a pour l'instant pas de consensus scientifique sur le choix de ces paramètres et le comportement chaotique des variables climatiques rend impraticables les méthodes d'ajustement de modèle standards. Ainsi, de grands efforts sont déployés afin de trouver des méthodes statistiques efficaces pour l'estimation de ces paramètres. Nous présenterons donc pour finir une tentative de simulation, sur un modèle très simple, de la distribution de ces paramètres à l'aide des méthodes MCMC.

Le rapport sera divisé ainsi : nous ferons d'abord un survol rapide de quelques notions préliminaires sur les chaînes de Markov, les méthodes de Monte-Carlo et la statistique bayésienne, qui nous permettront de justifier les développements qui vont suivre. Nous introduirons ensuite les MCMC par l'entremise de l'algorithme Metropolis-Hastings, puis les méthodes adaptatives et leurs particularités. Cela nous conduira finalement au coeur de notre étude, soit les algorithmes avec adaptation régionale. Nous présenterons d'abord brièvement les algorithmes RAPT et RAPTOR, pour ensuite expliquer les particularités de notre propre algorithme. Suivront une illustration des tests de performance et de robustesse permettant de confirmer la validité de celui-ci, puis une discussion des résultats obtenus. Finalement, nous verrons en détails la formulation du problème de paramétrisation évoqué précédemment, la définition du modèle climatique de Lorenz sur lequel nous avons travaillé et une tentative de résolution du problème.

2 Notions préliminaires

2.1 Chaînes de Markov

Nous résumons ici quelques propriétés des chaînes de Markov qui seront utiles pour la suite.

2.1.1 Définition

On appelle chaîne de Markov à temps discret toute suite de variables aléatoires $\{X_n\}_{n\geq 0}$, provenant d'un certain support S et respectant la propriété markovienne :

$$\mathbb{P}(X_n|X_{n-1},\ldots,X_0) = \mathbb{P}(X_n|X_{n-1}).$$

Autrement dit, la valeur d'une variable aléatoire de cette suite ne dépend que de celle qui la précède. Nous nous intéresserons ici aux chaînes dites homogènes, où les probabilités $\mathbb{P}(X_n|X_{n-1})$ sont indépendantes de la valeur de n. Ainsi, une chaîne de Markov homogène est complètement définie par la valeur ou la distribution de X_0 et les probabilités de transition en un pas $p_{ij} = \mathbb{P}(X_n = j|X_{n-1} = i), n \geq 1$, habituellement regroupées dans une matrice $P = \{p_{ij}\}$, définie sur l'ensemble des valeurs $i, j \in S$ (l'espace d'états). Finalement, on définit les probabilités de transition en k pas ainsi : $p_{ij}^{(k)} = \mathbb{P}(X_k = j|X_0 = i)$.

2.1.2 Quelques critères de classification

Pour qu'une chaîne ait des propriétés asymptotiques intéressantes, on souhaitera que tous ses états soient récurrents positifs et apériodiques. On dira alors que la chaîne est ergodique. Un état i est dit récurrent positif si

$$p_{ii}^{(n)} \nrightarrow 0, \text{ quand } n \to \infty.$$

Un état est apériodique si sa période vaut 1, la période étant définie comme étant le plus grand commun diviseur de $\{n \in \mathbb{N} : p_{ii}^{(n)} > 0\}$. Pour vérifier ces propriétés, il sera plus simple de passer par la notion de classe d'états, que nous définissons maintenant.

On dit qu'un état j est accessible à partir d'un état i, s'il est possible, en partant de i, d'atteindre l'état j en un nombre fini de transitions. Formellement,

$$i \to j \Leftrightarrow \exists\ n \geq 0, \text{ tel que } p_{ij}^{(n)} > 0.$$

Si l'on a à la fois $i \to j$ et $j \to i$, on dira que i et j communiquent. On montre alors facilement que les états communiquant entre eux forment une classe d'équivalence sur l'ensemble des états de la chaîne. Les états d'une même classe possèdent plusieurs propriétés communes. Entres autres, ils auront la même période et seront tous récurrents positifs si l'un d'entre eux l'est. Dans le cadre de ce projet, nous verrons uniquement des chaînes irréductibles, c'est-à-dire des chaînes ne possédant qu'une seule classe d'états.

2.1.3 Théorème ergodique et distribution stationnaire

La distribution stationnaire, lorsqu'elle existe est une distribution de probabilités prenant valeur sur S et respectant la condition :

$$\pi = \pi P \Leftrightarrow \forall j : \pi_j = \sum_i \pi_i p_{ij}$$

Dans le cas d'une chaîne ergodique, la distribution stationnaire existera toujours, et on aura alors, pour tous les états $i, j \in S$:

$$p_{ij}^{(n)} \xrightarrow[n\to\infty]{} \pi_j > 0.$$

Cette distribution et ses propriétés seront primordiales pour les développements qui vont suivre. Mentionnons seulement pour l'instant qu'une chaîne de Markov ergodique convergera en loi vers sa distribution stationnaire peu importe la valeur initiale X_0.

Pour un espace d'états continu, les transitions pourront être exprimées par des distributions continues conditionnelles de densité $P(x_n|x_{n-1}), n \geq 1$, ou encore sous forme de distributions conjointes de densité $P(x_{n-1}, x_n)$. Par souci de concision, nous ne reprendrons pas les notions précédentes dans le cas d'un espace continu. Mentionnons seulement que toutes les propriétés et résultats énoncés ont leur équivalent, à quelques considérations techniques près, dans cette nouvelle situation. Pour de plus amples informations, on pourra se référer à [10] ou [11].

2.2 Méthodes de Monte-Carlo

Les méthodes de Monte-Carlo sont des techniques d'échantillonnage aléatoire numérique visant à calculer des intégrales. Le problème classique est le suivant. Soit $h(x), x \in \mathbb{R}^m$, une fonction quelconque et $f(x)$, une fonction de densité de support $\mathscr{X} \subset \mathbb{R}^m$. On cherche à calculer :

$$I = \mathbb{E}_f(h(x)) = \int_{A\in\mathscr{X}} h(x)f(x)dx.$$

Pour ce faire, on génère un certain nombre n de variables $X_i, i = 1, \ldots, n$, i.i.d. de densité f, avec lesquelles on estime I par :

$$\overline{h}_n = \frac{1}{n}\sum_{j=1}^{n} h(x_i).$$

Propriétés

1. Par la loi des grands nombres, avec probabilité 1 :

$$\overline{h}_n \xrightarrow[n\to\infty]{} \mathbb{E}(h).$$

2. À la condition que $\mathbb{E}_f(h^2)$ soit finie :

$$\mathbb{V}(\overline{h}_n) = \frac{1}{n}\mathbb{V}(h(x)) = \frac{1}{n}\int h^2(x)f(x)dx - \mathbb{E}_f^2(h(x)).$$

3. Sous la même condition, par le théorème limite central :

$$\frac{\overline{h}_n - I}{\sqrt{\mathbb{V}(h(x))/n}} \xrightarrow[n\to\infty]{D} \mathcal{N}(0,1).$$

Ces techniques seront surtout utiles dans les problèmes en grande dimension, où les méthodes numériques traditionnelles perdent de leur efficacité. La difficulté sera de trouver une façon de générer efficacement un échantillon de variables i.i.d de densité f.

2.3 Analyse bayésienne

Dans le cadre du modèle Bayésien, nous supposons que les paramètres du modèle à l'étude, que nous dénoterons par le vecteur θ, sont des variables aléatoires suivant une distribution que l'on cherche à retrouver. Pour ce faire, à partir d'une information à priori sur les paramètres, qui prendra la forme d'une distribution de probabilité de densité $p(\theta)$, ainsi qu'un ensemble de données ou d'observations y provenant du modèle, on déduira une distribution à posteriori $p(\theta|y)$. Nous supposerons pour la suite que toutes ces lois sont continues. Par les identités probabilistes de base, on montre que

$$p(\theta|y) = \frac{p(\theta, y)}{p(y)} = \frac{p(y|\theta)p(\theta)}{\int_\Theta p(y|\theta)p(\theta)d\theta},$$

où Θ représente le support des paramètres. Souvent, ce calcul s'avérera problématique. En effet, dans des modèles complexes, il sera impossible d'obtenir une forme explicite ou suffisamment simple pour $p(\theta|y)$. Pour calculer des probabilités et des statistiques de cette distribution, il faudra pourtant trouver de bonnes estimations d'intégrales de la forme

$$\int h(\theta)p(\theta|y)d\theta.$$

Comme nous allons le voir, les méthodes de Monte-Carlo par chaîne de Markov sont une solution toute indiquée à ce genre de problèmes.

3 Méthodes MCMC

3.1 Introduction

Les méthodes de Monte-Carlo par chaînes de Markov permettent d'élargir grandement l'éventail des distributions pouvant être simulées numériquement. Elles sont relativement simples à implémenter et ne requièrent souvent que la connaissance de la fonction de densité cible à une constante près, ce qui les rend intéressantes dans de nombreuses situations. Cependant, une implémentation naïve peut mener à des temps de calcul très longs, puisque la convergence de ces méthodes est relativement lente lorsqu'elles ne sont pas bien calibrées à une situation donnée.

Nous verrons d'abord sommairement les justifications théoriques de ces méthodes, puis nous les illustrerons par la présentation de l'algorithme original de Metropolis-Hastings, pour ensuite voir les améliorations successives que l'on peut y apporter, leurs particularités et leur validité théorique.

L'idée de base est de simuler une distribution de densité f en utilisant une chaîne de Markov ergodique $\{X_t\}$ dont la distribution stationnaire est f. Le théorème ergodique garantit alors la convergence en loi de $\{X_t\}$ vers une variable de densité f et par conséquent, pour presque toute valeur initiale X_0 :

$$\frac{1}{n}\sum_{t=1}^{n} h(X_t) \xrightarrow[n\to\infty]{} \mathbb{E}_f(h).$$

Nous appellerons MCMC toute méthode permettant de simuler une distribution en utilisant une chaîne de Markov ergodique ayant celle-ci comme distribution stationnaire. Pour construire un tel algorithme, il faut donc déterminer un ensemble de probabilités de transition P approprié, c'est-à-dire irréductible, ergodique et ayant la bonne distribution stationnaire. Nous aurons besoin pour la suite du résultat suivant.

Proposition 3.1. *Soit une chaîne de Markov ayant comme probabilités de transition P et une distribution de probabilité $\pi(\cdot)$ définie sur le même espace d'états S. Si P possède la*

propriété de réversibilité par rapport à π :

$$\forall x, y \in S : \pi(x)P(x,y) = \pi(y)P(y,x),$$

alors la distribution stationnaire de la chaîne est π.

Démonstration. On aura stationnarité si, $\forall y \in S$:

$$\int_S \pi(dx)P(x,y) = \pi(y).$$

Or, sous l'hypothèse de réversibilité :

$$\forall y : \int_S \pi(dx)P(x,y) = \int_S \pi(y)P(y,dx) = \pi(y)\int_S P(y,dx) = \pi(y).$$

□

On utilisera cette propriété pour construire des probabilités de transition appropriées.

3.2 L'algorithme de Metropolis-Hastings

Voyons d'abord les détails de l'algorithme dans sa forme générale telle que décrite par [7] pour ensuite montrer sa validité. L'algorithme ne nécessite qu'une valeur de départ X_0 et le choix d'une distribution conditionnelle de densité $q(x,y) = q(y|x)$. À une étape donnée t, les manipulations suivantes sont effectuées.

3.2.1 Algorithme MH

1. À partir de la valeur $X_t = x$, on génère $Y_{t+1} = y$ selon la distribution de densité $q(y|x)$.
2. On pose $X_{t+1} = \begin{cases} Y_{t+1} & \text{avec probabilité} \quad \alpha(x,y), \\ X_t & \text{avec probabilité} \quad 1-\alpha(x,y); \end{cases}$

où les seuils α doivent avoir la forme générale

$$\alpha(x,y) = \frac{s(x,y)}{1+r(x,y)}.$$

$r(x,y)$ est le ratio $\frac{\pi(x)q(x,y)}{\pi(y)q(y,x)}$, et la fonction s est choisie de façon à ce que $s(x,y) = s(y,x)$ et $0 \leq \alpha(x,y) \leq 1$. Habituellement, on utilise exclusivement

$$\alpha(x,y) = \min\left\{1, \frac{\pi(y)q(x|y)}{\pi(x)q(y|x)}\right\},$$

qui correspond au choix $s(x,y) = \min\{1 + r(x,y), 1 + r(y,x)\}$. Si en plus la densité q est symétrique ($q(y|x) = q(x|y)$), le rapport devient tout simplement

$$\alpha(x,y) = \min\left\{1, \frac{\pi(y)}{\pi(x)}\right\}.$$

Nous nommerons q la densité de proposition des candidats, ou densité instrumentale, et α les probabilités d'acceptation de ces derniers. Voyons maintenant les propriétés théoriques de cette procédure.

3.2.2 Propriétés

L'algorithme tel que défini génère une chaîne de Markov dont les probabilités de transition sont données par :

$$\begin{aligned} P(x,y) &= q(x,y)\alpha(x,y), \text{ si } x \neq y, \\ P(x,x) &= 1 - \int P(x,y)dy, \text{ autrement.} \end{aligned}$$

Pour prouver que la distribution stationnaire de cette chaîne est π, il suffit de montrer qu'elle est réversible par rapport à π. Or,

$$\begin{aligned} \pi(x)P(x,y) &= \pi(x)q(x,y)\alpha(x,y) \\ &= \frac{\pi(x)q(x,y)s(x,y)}{1 + \frac{\pi(x)q(x,y)}{\pi(y)q(y,x)}} \\ &= \frac{\pi(x)\pi(y)q(x,y)q(y,x)s(x,y)}{\pi(y)q(y,x) + \pi(x)q(x,y)} \\ &= \frac{\pi(y)q(y,x)s(y,x)}{\frac{\pi(y)q(y,x)}{\pi(x)q(x,y)} + 1} \\ &= \pi(y)q(y,x)\alpha(y,x) = \pi(y)P(y,x). \end{aligned}$$

Maintenant, il faut s'assurer que la chaîne converge bien vers sa distribution station-

naire, c'est-à-dire qu'elle est ergodique. Or, ceci est facilement vérifié la plupart du temps. Par exemple, si $q(x, y)$ est positive pour toute paire (x, y) appartenant au support de π, $P(x, y)$ aussi sera toujours positive. Ainsi, à partir d'une valeur X_t donnée, toute valeur X_{t+1} sera atteignable en une seule étape avec une probabilité positive. La chaîne est donc irréductible. La chaîne sera aussi apériodique du moment qu'il existe au moins une paire (x, y) telle que $\alpha(x, y) < 1$, car on aura alors $P(x, x) > 0$. Cela sera pratiquement toujours vrai et on conclut donc que la chaîne est effectivement ergodique.

Remarques

1. C'est le choix de la distribution des candidats $q(x, y)$ qui influencera la qualité de l'algorithme. En effet, un choix judicieux favorisera une exploration rapide de l'espace d'états et accélérera en conséquence la convergence de la chaîne vers la distribution stationnaire. On veut donc une variabilité modérée dans les candidats possibles, de façon à favoriser les fluctuations tout en restreignant la probabilité de rejet. On verra dans les sections suivantes plusieurs façons d'ajuster la distribution instrumentale q pour une distribution cible donnée.

   Pour des raisons de performance, on choisira évidemment une distribution q facile à simuler. En pratique, sur le support $\mathbb{R}^d$, ce sera presque toujours une loi normale centrée à la valeur précédente de la chaîne : $X_{t+1} \sim \mathcal{N}(X_t, \Sigma)$. C'est donc par le choix de la matrice Σ que l'on pourra optimiser les performances de l'algorithme. Notons que ce choix rend q symétrique et positive sur S.

2. En pratique, on éliminera souvent de l'échantillon obtenu les premières valeurs, qui sont fortement influencées par le choix de la valeur initiale. De même, comme l'indépendance est asymptotique, pour éviter qu'il y ait corrélation entre des valeurs consécutives de l'échantillon, on pourrait choisir de conserver uniquement une valeur sur 10 par exemple.

3. Puisque les probabilités $\alpha(x, y)$ ne dépendent de la distribution cible que sous la forme du rapport $\pi(y)/\pi(x)$, il est souvent possible d'appliquer l'algorithme sans connaître la constante de normalisation de la densité de π.

3.3 Algorithmes adaptatifs

Comme nous l'avons déjà mentionné, la vitesse de convergence des MCMC est fortement influencée par le choix et la paramétrisation de la distribution des candidats. Il existe

certains critères permettant d'optimiser celle-ci, mais ils dépendent tous d'une certaine connaissance de la covariance de la distribution cible. Or, pour estimer cette information manquante, un échantillon de la distribution cible est nécessaire, qui devra lui aussi provenir d'un algorithme MCMC, forcément non calibré. Pour pallier à cette problématique, on a introduit des algorithmes utilisant une distribution candidate qui s'adapte au fil des itérations, en fonction de l'information contenue dans les valeurs X_t déjà générées. Il va de soi que la suite $\{X_t\}$ ainsi obtenue n'est pas une chaîne de Markov, puisque la valeur de X_{t+1} dépendra de toutes les valeurs $X_0, \ldots, X_t$. Cependant, sous certaines conditions que nous verrons plus loin, l'ergodicité est préservée. Présentons d'abord l'algorithme AM (*Adaptive Metropolis* de [4]), très utilisé, largement inspiré de celui de Metropolis-Hastings et respectant ces conditions.

3.3.1 Algorithme AM

On considère une distribution cible de densité $\pi(\cdot)$ avec support $S \subset \mathbb{R}^d$, S étant compact.

1. On choisit une valeur $X_0 \in S$, une matrice C_0 symétrique définie positive et un indice t_0.
2. À l'aide des valeurs $(X_0, \ldots, X_{t-1})$, on génère la valeur candidate Y_t à partir d'une distribution normale de moyenne X_{t-1} et de covariance C_t, définie ainsi :

$$C_t = \begin{cases} C_0 & \text{pour } t \leq t_0 \\ s_d \left(\mathrm{Cov}(X_0, \ldots, X_{t-1}) + \epsilon I_d \right) & \text{pour } t > t_0, \end{cases}$$

   où s_d est un paramètre constant dépendant uniquement de d, $\epsilon > 0$ est petit, et $\mathrm{Cov}(X_0, \ldots, X_{t-1})$ est la matrice de covariance empirique des vecteurs X_0 à X_{t-1} :

$$\mathrm{Cov}(x_0, \ldots, x_k) = \frac{1}{k} \left(\sum_{i=0}^{k} x_i x_i^T - (k+1) \overline{x}_k \overline{x}_k^T \right).$$

3. Comme d'habitude, on accepte pour X_t la nouvelle valeur Y_t avec probabilité

$$\min \left(1, \frac{\pi(Y_t)}{\pi(X_{t-1})} \right).$$

   Sinon, $X_t = X_{t-1}$.

3.3.2 Remarques

1. La raison d'être du second terme de C_t est essentiellement théorique. Il assure que la matrice de covariance restera définie positive et permet de justifier la convergence du processus. En pratique, on peut sans risque choisir $\epsilon = 0$.
2. Pour une distribution cible normale et une distribution instrumentale normale également, il a été montré que sous certains aspects $s_d \approx 2.4^2/d$ est le facteur de normalisation optimal (voir [3]).
3. La mise à jour de la matrice de covariance peut se faire à l'aide des formules de récurrence suivantes :

$$\begin{aligned} \overline{x}_t &= \frac{t}{t+1}\overline{x}_{t-1} + \frac{x_t}{t+1}, \\ C_{t+1} &= \frac{t-1}{t}C_t + \frac{s_d}{t}\left(t\overline{x}_{t-1}\overline{x}_{t-1}^T - (t+1)\overline{x}_t\overline{x}_t^T + x_t x_t^T + \epsilon I_d\right). \end{aligned}$$

4. La condition de compacité de S n'est pas vraiment contraignante car cela n'empêche pas S d'être arbitrairement grand, par exemple $[10^5, -10^5]^d$. Ainsi, on pourra remplacer une densité cible f^* définie sur $\mathbb{R}^d$ par :

$$f(x) := \begin{cases} f^*(x) & \text{si } x \in S, \\ 0 & \text{autrement.} \end{cases}$$

   Cette modification ne demande aucun calcul supplémentaire (la densité f n'est pas correctement normalisée, mais comme nous l'avons vu, cela n'a pas d'influence dans les MCMC symétriques) et en pratique ne changera en rien les résultats obtenus par l'algorithme. Pour la suite, nous supposerons toujours que S est compact.

Résultats théoriques

Tel que mentionné auparavant, l'ergodicité des algorithmes adaptatifs n'est pas automatique. Pour assurer celle-ci, il existe une paire de conditions suffisantes relativement faciles à évaluer. Pour les énoncer, nous utiliserons la notion de distance en variation totale entre deux distributions $P(\cdot)$ et $Q(\cdot)$, prenant valeur sur un support S, que nous noterons par $\|P - Q\|$ et qui se définit ainsi :

$$\|P - Q\| = \sup_A |P(A) - Q(A)|, A \subseteq S.$$

Autrement dit, il s'agit de la plus grande différence de probabilité possible que ces deux distributions peuvent assigner à un même événement. On peut vérifier que cette mesure possède bien toutes les propriétés habituelles de la distance, ainsi que plusieurs autres (voir [12]). On peut aussi définir l'ergodicité en utilisant cette notion : Un processus stochastique $\{X_n\}_{n\in\mathbb{N}}$ de distribution stationnaire $\pi(\cdot)$ sera ergodique si, pour toute valeur de X_0 :

$$\|\mathscr{L}(X_n) - \pi(\cdot)\| \xrightarrow[n\to\infty]{} 0.$$

Pour la suite, nous dénoterons par $\Gamma_n, n \in \mathbb{N}$ la variable aléatoire représentant les probabilités de transition entre X_n et X_{n+1}, par $\mathscr{Y}$ l'espace des distributions de probabilités de transition possibles (donc le support de Γ_n), et par y une réalisation quelconque de Γ_n. Nous utiliserons aussi la notation augmentée suivante pour les probabilités de transition.

$$P_y(x, B) = \mathbb{P}\left(X_{n+1} \in B | X_n = x, \Gamma_n = y\right).$$

Nous sommes maintenant prêts à énoncer quelques théorèmes, dont on pourra trouver les démonstrations dans [13].

Théorème 3.1 (Ergodicité des algorithmes adaptatifs)**.** *Soit un algorithme adaptatif sur l'espace d'états S compact, avec un espace de distributions de transitions $\mathscr{Y}$, et ayant une distribution stationnaire $\pi(\cdot)$, identique pour toute distribution de transition $P_y, y \in \mathscr{Y}$. Supposons que les deux conditions suivantes soient respectées :*

1. *Ergodicité uniforme simultanée : Pour tout $\epsilon > 0$, on peut trouver $N \in \mathbb{N}$ tel que*

$$\|P_y^{(N)}(x, \cdot) - \pi(\cdot)\| < \epsilon, \forall x \in S, y \in \mathscr{Y}.$$

2. *Adaptation décroissante : Après un certain temps, l'adaptation cesse ou devient négligeable. Formellement :*

$$D_n := \sup_x \|P_{\Gamma_n}(x, \cdot) - P_{\Gamma_{n-1}}(x, \cdot)\| \xrightarrow[n\to\infty]{} 0, \text{ en probabilité.}$$

Alors, la chaîne obtenue à partir de cet algorithme est ergodique.

En d'autres mots, la première condition exige que pour toutes les distributions candidates pouvant être obtenues par l'algorithme, une chaîne homogène utilisant cette distribution sera uniformément ergodique. La seconde demande qu'après un certain temps,

la probabilité que deux distributions candidates successives soient significativement différentes tende vers zéro.

Théorème 3.2. *L'algorithme AM tel que présenté respecte les 2 conditions du théorème 3.1 et produit donc une chaîne ergodique.*

Théorème 3.3 (Loi faible des grands nombres)**.** *Soit $g : S \to \mathbb{R}$, une fonction bornée et mesurable, et X_i, une chaîne de Markov générée à partir d'un algorithme adaptatif. Alors, sous les mêmes conditions d'ergodicité uniforme simultanée et d'adaptation décroissante et pour toutes valeurs initiales, nous avons :*

$$\frac{\sum_{i=1}^{n} g(X_i)}{n} \xrightarrow[n\to\infty]{} \pi(g).$$

Les algorithmes adaptatifs sont des outils à la fois puissants et versatiles, mais ils ont aussi leurs limites. Nous présentons maintenant une technique pouvant être intégrée à n'importe quel algorithme déjà existant et qui permet de repousser celles-ci.

3.4 Chaînes parallèles

Dans leur forme originale, les MCMC, de nature essentiellement séquentielle, ne bénéficient pas particulièrement d'un traitement informatique en parallèle. En effet, à cause de leur ergodicité, il n'est pas avantageux de générer plusieurs courtes chaînes au lieu d'une seule plus grande. Toutefois, un tel ajustement pourrait être souhaitable, voire nécessaire, dans l'une ou l'autre des situations suivantes.

1. Lorsque la distribution cible est multimodale. Dans ce cas, il est fort probable qu'une unique chaîne reste "coincée" dans un seul de ces modes et qu'elle représente donc mal l'ensemble de la distribution. L'utilisation de plusieurs chaînes indépendantes avec des points de départ variés augmentera la probabilité que tous les modes soient suffisamment visités. Il existe aussi des façons d'intégrer un certain degré d'adaptation locale au sein même de l'algorithme, comme nous le verrons plus loin.
2. Dans le cadre d'un algorithme adaptatif : on utilisera alors la covariance des données de toutes les chaînes pour mettre à jour la distribution des candidats (qui sera donc identique dans chaque chaîne). Cela permettra d'obtenir plus rapidement une distribution q de qualité. À titre d'exemple, voici une adaptation de l'algorithme AM présenté plus haut.

Algorithme : adaptation interchaîne

1. On génère un certain nombre m de chaînes suivant l'algorithme AM, ayant chacune sa propre valeur de départ et fonctionnant indépendamment des autres pour les t_0 premières itérations.
2. On construit alors la matrice C_t en tenant compte de la covariance entre toutes les valeurs de toutes les chaînes :

   $$C_t = s_d \text{Cov}(X_{1,0}, X_{1,1}, \ldots, X_{1,t-1}, X_{2,0}, \ldots, X_{m,t-1}), \text{ pour } t > t_0.$$

   On poursuit de la même façon en mettant à jour C_t après chaque itération. Encore une fois, des formules de récurrences permettent d'alléger la charge de calcul.

   $$\begin{aligned} \overline{\theta}_t &= \frac{t}{t+1}\overline{\theta}_{t-1} + \frac{\theta_t}{t+1}, \\ C_{t+1} &= \frac{t-1}{t}C_t + \frac{s_d}{t}(\theta_{t+1} - \overline{\theta}_t)(\theta_{t+1} - \overline{\theta}_t)^T, \end{aligned}$$

   où θ_t est un vecteur de $\mathbb{R}^m$ contenant l'ensemble des valeurs de l'itération t, et $\overline{\theta}_t = \sum_{i=0}^{t} \frac{\theta_i}{t+1}$.
3. On considérera alors l'ensemble des valeurs de toutes les chaînes pour notre échantillon.

4 Présentation et amélioration des algorithmes avec adaptation régionale

Comme nous l'avons mentionné précédemment, même les algorithmes adaptatifs les plus performants perdent de leur efficacité lorsque la distribution cible est multimodale, asymétrique ou de façon générale nettement différente de la distribution de proposition. En effet, dans ces cas problématiques, une unique distribution pour les candidats, même optimale, ne peut explorer l'espace de façon satisfaisante. Nous pouvons alors considérer une distribution de proposition pouvant varier selon les diverses régions de l'espace d'états S. Autrement dit, les spécifications de la distribution du candidat au temps $t+1$ varieront selon l'emplacement de la valeur X_t. Pour appréhender cette situation, nous partirons de l'hypothèse que l'espace d'états S puisse être divisé en K régions $S_{01}, \ldots, S_{0K}$, dans lesquelles les densités de propositions optimales sont $Q_1, \ldots, Q_K$, respectivement. Ainsi,

de façon idéale, on générerait le candidat Y_{t+1} selon la distribution

$$\sum_{i=1}^{K} \mathbb{I}(X_t \in S_{0i}) Q_k(X_t, Y_{t+1}),$$

où $\mathbb{I}(\cdot)$ est la fonction indicatrice. En pratique, il est impossible de délimiter ces régions de façon exacte, mais on peut possiblement les estimer, disons par $S_1, \ldots, S_K$. À partir de ce point, deux approches adaptatives sont envisageables afin de compenser l'erreur d'estimation. Premièrement, il est possible minimiser l'impact de cette erreur en ayant recours à une distribution de proposition plus complexe dont les paramètres refléterons en quelque sorte la qualité d'ajustement de la distribution à sa région de définition. Au contraire, on pourrait tenter d'adapter continuellement les délimitations des régions jusqu'à obtenir un partage optimal. Le second choix sera souvent plus puissant en terme du nombre d'itérations nécessaires pour obtenir un résultat intéressant, mais chacune de ces itérations demandera davantage de calcul et il n'est pas clair laquelle des deux options est préférable en temps réel. Nous nous pencherons maintenant sur les spécifications de ces deux approches telles qu'utilisées dans les algorithmes RAPT (*Regional Adaptation*) et RAPTOR (*Regional Adaptation with Online Recursion*), introduits dans [2] et [1]. Nous proposerons ensuite un troisième algorithme, largement inspiré des précédents et présentant un compromis intéressant entre ces deux méthodes.

4.1 Algorithme RAPT

4.1.1 Présentation

Dans le cadre de cet algorithme, l'emplacement des régions $S_1, \ldots, S_k$ est fixé d'avance. Aucune des distributions Q_j n'étant parfaite pour une région donnée, on utilise plutôt comme proposition un mélange des Q_j, différent pour chaque région, et prenant la forme suivante

$$Q(x, y) = \sum_{i=1}^{K} \mathbb{I}(x \in S_i) \sum_{j=1}^{K} \lambda_{ij} Q_j(x, y),$$

où $\sum_j \lambda_{ij} = 1, \forall i$. Ainsi, des poids λ_{ij} idéaux indiqueraient à quel point la proposition Q_j est plus appropriée que les autres dans la région S_i. Il va de soi que ces paramètres seront difficilement choisis d'avance et devront donc être adaptés en cours d'algorithme pour garantir une bonne performance. Pour déterminer des valeurs appropriées, on peut

utiliser la distance de saut quadratique moyenne accumulée jusqu'au temps actuel t, définie ainsi :

$$d_{ij}(t) = \frac{\sum_{s \in W_{ij}(t)} \|X_{s+1} - X_s\|^2}{|W_{ij}(t)|},$$

où $W_{ij}(t)$ contient les indices de tous les éléments de la région i qui sont suivis d'une proposition de la densité Q_j, jusqu'au temps t. Après l'itération $t+1$, une seule des K^2 valeurs $d_{ij}(t)$ sera modifiée, celle correspondant au couple (i,j) tel que $X_{t+1} \in W_{ij}(t+1)$. On peut facilement mettre celle-ci à jour par la formule récursive suivante.

$$\begin{aligned} |W_{ij}(t+1)| &= |W_{ij}(t)| + 1, \\ d_{ij}(t+1) &= \frac{|W_{ij}(t)|}{|W_{ij}(t+1)|} d_{ij}(t) + \frac{||X_{t+1} - X_t||^2}{|W_{ij}(t+1)|}. \end{aligned}$$

On définit alors

$$\lambda_{ij}(t) = \begin{cases} \frac{d_{ij}(t)}{\sum_{j=1}^{K} d_{ij}(t)}, & \text{si le dénominateur est} > 0\,; \\ 1/2, & \text{autrement.} \end{cases}$$

Ainsi, la proposition engendrant la meilleure exploration de l'espace dans une région donnée se verra octroyer un poids de sélection plus grand.

Il sera aussi primordial d'adapter les distributions Q_j. Si elles sont normales, il suffira d'adapter leurs matrices de covariance respectives selon le même processus que dans l'algorithme AM, mais en utilisant pour le calcul de la covariance de Q_1 uniquement les valeurs provenant de la région 1, et ainsi de suite.

Si les délimitations sont relativement exactes et qu'un mélange de propositions normales est une bonne approximation de la densité cible, on s'attend à ce que l'algorithme ait de bonnes performances indépendamment dans chaque région. Pour assurer un meilleur équilibre inter-régional, il est possible d'ajouter une dernière composante adaptative globale au mélange. On obtiendrait alors la densité de proposition :

$$Q_{(\gamma)}(x,y) = (1-\beta) \sum_{i=1}^{K} \mathbb{I}(x \in S_i) \sum_{j=1}^{K} \lambda_{ij} Q_j(x,y) + \beta Q_S(x,y),$$

avec $0 < \beta < 1$ constant et la distribution Q_S s'adaptant en utilisant l'ensemble de l'échantillon obtenu jusqu'alors. Cet ajout sera surtout important lorsque les différents modes ou composantes de la distribution cible sont séparés par une région de faible densité,

donc difficile à "traverser".

Finalement, on a pour les seuils d'acceptation les simplifications suivantes, où q représente la densité de Q :

$$\alpha^*(x,y) = \begin{cases} \frac{\pi(y)}{\pi(x)}, & \text{si } x \text{ et } y \text{ sont dans la même région ;} \\ \frac{\pi(y)\left((1-\beta)\sum_{j=1}^K \lambda_{aj} q_j(y,x) + \beta q_S(y,x)\right)}{\pi(x)\left((1-\beta)\sum_{j=1}^K \lambda_{bj} q_j(x,y) + \beta q_S(x,y)\right)}, & \text{si } x \in S_b \text{ et } y \in S_a, a \neq b. \end{cases}$$

Souvent, on utilisera uniquement deux régions, mais il est à noter qu'un algorithme général avec K régions différentes, quoique plus ardu à mettre en place, n'est pas beaucoup plus coûteux au niveau informatique, puisque seulement une fraction de l'ensemble des paramètres doit être mis à jour à chaque étape. Il va de soi que le nombre de régions devrait tout de même rester très modéré.

Malgré la flexibilité de cet algorithme, il demeure un certain risque que l'un des modes ne soit pas localisé durant la période de pré-adaptation, avec pour conséquence une sous-estimation significative de la covariance de la distribution de proposition globale, ce qui rendra d'autant plus difficile l'exploration du mode sous-représenté. Or, cette problématique peut facilement être contournée en utilisant une version avec chaînes parallèles, dont les valeurs initiales seraient dispersées dans chaque région, en supposant que la partition adoptée soit assez bonne. Des résultats expérimentaux semblent montrer qu'une seule chaîne par région produit déjà une nette amélioration dans la performance, pour un même nombre d'itérations. La difficulté principale dans l'utilisation de l'algorithme est de déterminer avec peu d'information une partition aussi optimale que possible de l'espace. Pour clarifier le fonctionnement, voici maintenant une description détaillée de l'algorithme avec paramètres habituels.

4.1.2 Algorithme

Valeurs initiales : X_0, $\beta = 0.3$, $\lambda_{ij}(0) = 1/2,\ \forall (i,j)$, $C_1 = C_2 = I_d$, $C_S = MI_d$, avec M sélectionné de façon à ce que C_S couvre l'ensemble de l'espace d'intérêt.

À l'étape $t+1$:

1. $Y_{t+1} \sim (1-\beta)\sum_{i=1}^K \mathbb{I}(X_t \in S_i)\sum_{j=1}^K \lambda_{ij}(t) Q_j^{(t)}(X_t, Y_{t+1}) + \beta Q_S^{(t)}(X_t, Y_{t+1})$,
   avec $Q_j^{(t)}(Y_{t+1}|X_t) \sim \mathcal{N}(X_t, C_j(t))$.
2. $X_{t+1} = \begin{cases} Y_{t+1} & \text{avec probabilité} \quad \alpha^*(x,y) \\ X_t & \text{avec probabilité} \quad 1-\alpha^*(x,y) \end{cases}$

3. Soit i tel que $X_t \in S_i$ et j tel que $Y_{t+1} \sim Q_j$ (si $Y_{t+1} \sim Q_S$, sauter cette étape), alors $t \in W_{ij}(t+1)$et donc :

$$\begin{aligned} |W_{ij}(t+1)| &= |W_{ij}(t)| + 1, \\ d_{ij}(t+1) &= \frac{|W_{ij}(t)|}{|W_{ij}(t+1)|} d_{ij}(t) + \frac{||X_{t+1} - X_t||^2}{|W_{ij}(t+1)|}. \end{aligned}$$

Pour toute paire $(a, b) \neq (i, j)$:

$$\begin{aligned} W_{ab}(t+1) &= W_{ab}(t), \\ d_{ab}(t+1) &= d_{ab}(t). \end{aligned}$$

Finalement, pour $i = 1, \ldots, K$:

$$\lambda_{ij}(t+1) = \frac{d_{ij}(t+1)}{\sum_{j=1}^{K} d_{ij}(t+1)},$$

et pour toute paire (a, b) telle que $a \neq i$:

$$\lambda_{ab}(t+1) = \lambda_{ab}(t).$$

4. Adaptation de $C_S(t)$ et de $C_i(t)$ (voir algorithme AM).

Finalement, il est possible de montrer que cette procédure respecte bien les 2 conditions suffisantes d'ergodicité des algorithmes adaptatifs lorsque les distributions cible et instrumentale sont positives et continues sur S compact (voir [2]).

4.2 Algorithme RAPTOR

Une façon de contourner le problème de la partition des régions est de rendre celle-ci adaptative. Pour justifier la motivation de l'algorithme RAPTOR, nous supposerons que la distribution cible est un mélange de lois à K composantes. Si de plus, chacune de ces composantes est relativement similaire à une loi normale, il serait possible de l'estimer ainsi :

$$\tilde{\pi}(x) = \sum_{k=1}^{K} \lambda^{(k)} \mathcal{N}_d(x, \mu^{(k)}, \Sigma^{(k)}) := \sum_{k=1}^{K} \tilde{\pi}^{(k)}(x),$$

avec $\sum_{k=1}^{K} \lambda^{(k)} = 1$, $\lambda^{(k)} > 0$ et $1 \leq k \leq K$, où $\mathcal{N}_d$ est la distribution normale à d dimensions, $\mu^{(k)}$ et $\Sigma^{(k)}$ respectivement la moyenne et la matrice de covariance de la composante k. On imagine facilement que pour des distributions de candidats normales, les régions de la partition idéale seraient très proches de :

$$S_k = \{x \in S | \max_{1 \leq k \leq K} \tilde{\pi}^{(k)}(x) = k\}.$$

L'idée derrière l'algorithme est donc d'estimer de façon adaptative la moyenne, la covariance et le poids de chacune des composantes, puis de comparer les densités normales utilisant ces estimés afin de décider à quelle région appartiendrait un point donné. On peut montrer théoriquement que les régions spécifiées sans les poids $\lambda^{(k)}$ sont aussi valables, on aurait alors :

$$S_k = \{x \in S | \max_{1 \leq k \leq K} \mathcal{N}_d(x, \mu^{(k)}, \Sigma^{(k)}) = k\}.$$

Pour démarrer, l'algorithme aura donc besoin de valeurs initiales hypothétiques pour la moyenne $\mu^{(k)}$, la variance $\Sigma^{(k)}$ et les poids $\lambda^{(k)}$ de chaque composante k, desquels on peut déduire une première partition de l'espace. Le processus étant adaptatif, après une période de pré-adaptation, ces mêmes paramètres seront mis à jour à chaque étape. La distribution du candidat au temps $t+1$ sera alors donnée par :

$$Q_t(x, y) = (1 - \beta) \sum_{k=1}^{K} \mathbb{I}\{x \in S_t^{(k)}\} \mathcal{N}_d(y, x, \Sigma_t^{(k)}) + \beta \mathcal{N}_d(y, x, \Sigma_t^{(g)}),$$

où $0 \leq \beta \leq 1$ et $\Sigma_t^{(g)}$ est calculé à partir de la covariance globale, comme dans l'algorithme RAPT.

Nous ne nous attarderons pas à détailler les étapes de cet algorithme, mentionnons seulement que si l'on compare celui-ci au précédent, on réalise que la mise à jour adaptative des paramètres requiert un nombre d'opérations élémentaires comparable pour les deux, mais que dans le second, chaque itération nécessite en plus le calcul d'au moins K densités normales supplémentaires.

4.3 Algorithme Road Runner

L'algorithme que nous présentons ici est une modification de l'algorithme RAPT, applicable surtout dans les cas ou l'on souhaite partitionner l'espace en deux régions uniquement, ce qui de toute façon s'avère suffisant dans la plupart des situations. L'idée de base est

d'utiliser le même processus adaptatif que dans RAPT, de permettre aussi l'adaptation de la partition de l'espace, mais que cette délimitation soit aussi simple que possible à redéfinir. La solution la plus simple serait bien sûr que la frontière soit un hyperplan.

Nous cherchons donc à implanter une composante de complexité minimale qui déterminera de façon adaptative le plan séparant le mieux possible les deux régions théoriquement optimales. Ce choix est motivé par le fait que dans bien des cas, même si la séparation idéale entre deux modes est loin d'être linéaire, en considérant uniquement l'ensemble des points de l'espace ayant une densité moindrement significative, nous pouvons aisément faire passer un plan qui sépare les deux modes de façon presque parfaite. De plus, l'utilisation d'une distribution de proposition multimodale avec poids adaptatifs confère une certaine protection contre une partition sous-optimale et devrait aussi favoriser la diffusion de la chaîne même avant que l'adaptation du plan ne soit terminée. Il sera plus difficile de trouver une bonne frontière linéaire lorsque les deux modes sont si près l'un de l'autre que les distributions se confondent, mais dans une telle situation, une adaptation régionale n'est probablement pas nécessaire, car un algorithme adaptatif simple aurait alors une performance similaire.

Tout cela mène à croire que la plupart du temps, ce nouvel algorithme devrait être aussi précis que le RAPTOR pour un même nombre d'itérations, mais plus rapide en temps réel. Nous verrons d'abord les détails du calcul du plan adaptatif, nous montrerons ensuite que l'ergodicité de la chaîne est préservée par cette nouvelle composante, puis nous présenterons une série de tests mesurant la qualité de l'algorithme pour clore avec une courte discussion.

4.3.1 Description de l'algorithme

Pour construire notre plan optimal à une étape donnée, nous aurons besoin des moyennes échantillonnales dans chaque région empirique. Définissons $W_i(t) = \{0 \leq s \leq t : x_s \in S_i^{(t)}\}$ pour $i = 1, 2$, des ensembles contenant les indices des valeurs appartenant aux régions $S_1^{(t)}, S_2^{(t)}$. Alors,

$$\hat{\mu}_i^{(t)} = \sum_{s \in W_i(t)} \frac{x_s}{|W_i(t)|} \ , \quad i = 1, 2.$$

On détermine alors les paramètres du plan d'équation $a_t^T X = b_t$.

$$\begin{aligned} a_t &= \hat{\mu}_1^{(t)} - \hat{\mu}_2^{(t)}, \\ b_t &= a_t^T \left(\frac{\hat{\mu}_1^{(t)} + \hat{\mu}_2^{(t)}}{2} \right). \end{aligned}$$

Ce plan sera la frontière entre les nouvelles régions $S_1^{(t)}$ et $S_2^{(t)}$. On déterminera l'emplacement d'une nouvelle valeur ainsi :

$$X \in \begin{cases} S_1^{(t)} & \text{si} \quad a_t^T X \geq b_t, \\ S_2^{(t)} & \text{si} \quad a_t^T X < b_t. \end{cases}$$

On définit donc le plan de façon perpendiculaire à la droite joignant les moyennes estimées $\hat{\mu}_1^{(t)}$ et $\hat{\mu}_2^{(t)}$ de chaque région, et passant par le point milieu de cette droite. Si les deux modes de la distribution cible sont de forme et d'échelle relativement similaires, ce choix sera clairement optimal. S'il arrivait que l'une des deux régions ne soit pas du tout visitée durant la période de pré-adaptation (ce qui suggère que la partition initiale était très mal choisie, ou la pré-adaptation trop courte), alors l'une des moyennes empiriques ne sera pas définie (supposons $\hat{\mu}_2$) et on pourrait alors choisir par exemple un plan passant par l'autre moyenne et d'orientation arbitraire : $a = (1, \ldots, 1)$, $b = a^T \hat{\mu}_1$. En pratique, on évitera cette situation en utilisant des chaînes parallèles ayant des valeurs initiales dispersées de part et d'autre de la délimitation initiale.

Du point de vue informatique, les valeurs initiales nécessaires sont les mêmes que pour l'algorithme RAPT et la mise à jour des données se fera de la même façon, mais avec l'ajout de l'étape suivante entre les étapes 2 et 3 de l'algorithme présenté plus haut.

Étape 2.5 : En supposant que $X_{t+1} \in S_1^{(t)}$, l'autre possibilité étant analogue :

$$\begin{aligned} \hat{\mu}_1^{(t+1)} &= \hat{\mu}_1^{(t)} + \frac{x_{t+1} - \hat{\mu}_1^{(t)}}{|W_i(t+1)|}, \\ \hat{\mu}_2^{(t+1)} &= \hat{\mu}_2^{(t)}, \\ a_{t+1} &= \hat{\mu}_1^{(t+1)} - \hat{\mu}_2^{(t+1)}, \\ b_{t+1} &= a_{t+1}^T \left(\frac{\hat{\mu}_1^{(t+1)} + \hat{\mu}_2^{(t+1)}}{2} \right). \end{aligned}$$

Avant de passer à la suite, plusieurs remarques s'imposent. Premièrement, la formule de mise à jour précédente ne reflète pas tout à fait le raisonnement théorique présenté auparavant, puisqu'à une itération donnée, on ne calcule pas le nouvel emplacement de l'ensemble des points de la chaîne, mais uniquement celui du plus récent. Or, il est clair que certains points risquent de changer de région en cours d'algorithme. Cependant, on peut s'attendre à ce qu'à long terme, ces valeurs erronées aient une influence négligeable dans le calcul des valeurs adaptatives, et qu'une fois la frontière stabilisée il serait ineffi-

cace de recalculer à chaque étape l'ensemble des emplacements, qui risquent de très peu changer. Les résultats que nous présenterons plus loin semblent confirmer cette intuition. Cela dit, ce calcul supplémentaire pourrait peut-être s'avérer utile lors de la toute première adaptation, au temps t_0. On pourra alors déterminer les emplacements une première fois pour le calcul des moyennes $\hat{\mu}$ et des paramètres a et b, puis les évaluer une seconde fois selon le nouveau découpage de $S_1^{(t_0)}$ et $S_2^{(t_0)}$ pour la suite de l'adaptation. Nous comparerons ces deux approches dans certains des tests qui vont suivre.

D'autre part, on pourrait se demander à quel point les poids λ demeurent utiles dans ce nouvel algorithme. D'après les expérimentations effectuées jusqu'à présent, leurs conservation accélère grandement la découverte du plan optimal, probablement parce qu'ils favorisent une meilleure exploration de l'espace avant que l'adaptation du plan ne soit complétée.

Puisque l'ensemble des nouveaux calculs est basé sur des moyennes de points provenant d'un ensemble compact, on s'attend à ce que les paramètres a_t et b_t convergent, donc que le plan se stabilise éventuellement. Intuitivement, notre algorithme respecterait donc bien les conditions suffisantes d'ergodicité, sachant que le RAPT les respecte aussi. Nous montrons de façon détaillée en annexe que c'est bien le cas.

4.4 Simulations

4.4.1 Comparaison

Nous débutons par un test comparatif entre les multiples algorithmes d'adaptation régionale. Ce test est le même que celui utilisé dans [1]. Il y a 10 densités cible à simuler, toutes de la forme suivante :

$$\pi(x, m, s) = 0.5\mathcal{N}_d(x, -m\mathbf{1}, I_d) + 0.5\mathcal{N}_d(x, m\mathbf{1}, sI_d),$$

et dont les paramètres précis sont donnés dans le tableau suivant :

	1	2	3	4	5	6	7	8	9	10
d	2	2	2	2	2	5	5	5	5	5
m	1	1	0	0	2	$\frac{1}{2}$	$\frac{1}{2}$	0	0	1
s	1	4	1	4	1	1	4	1	4	1

Les paramètres initiaux pour notre algorithme et le RAPT sont $C_1 = I_d/10$, $C_2 =$

$sI_d/10$, $C_S = 50I_d$ lorsque $d = 2$ et $C_S = 10I_d$ lorsque $d = 5$, $X_0 = 0$, et la partition initiale en deux régions est délimitée par le plan $x_1 = 0$, qui s'avère une séparation assez bonne quand $d = 2$, mais beaucoup moins lorsque $d = 5$. On s'attendrait donc à ce que l'algorithme RAPT classique soit moins efficace dans ce dernier cas.

Pour le RAPTOR, on aura $\Sigma^{(i)} = C_i$, $\mu^{(1)} = (-2, 0, \ldots, 0)$, $\mu^{(2)} = (2, 0, \ldots, 0)$ et $\lambda^{(1)} = \lambda^{(2)} = 1/2$. Notons que ces choix favorisent légèrement l'algorithme RAPTOR, puisque les poids estimés sont déjà ajustés à la bonne valeur, et que l'initialisation inégale des covariances $\Sigma^{(1)}$ et $\Sigma^{(2)}$ permet de définir une frontière initiale avantageuse par rapport au plan $x_1 = 0$.

Pour chacune des distributions cibles, nous créons 1000 échantillons de taille $N = 1000$, et nous retirons 100 premières valeurs de chacun. Puis, nous estimons une moyenne de la distance au carré entre la moyenne théorique de la première composante (zéro dans tous les cas) et celle obtenue par MCMC. Voici maintenant les résultats obtenus par les deux versions de notre algorithme (la seconde effectuant un second calcul des régions à la première adaptation) et comparés aux autres méthodes (les résultats pour RAPTOR et RAPT sont tirés de [1]), desquels nous pouvons déjà tirer quelques conclusions.

(d, m, s)	Version 1	Version 2	RAPTOR	RAPT
$(2, 1, 1)$	21	24	21	22
$(2, 1, 4)$	46	42	43	46
$(2, 0, 1)$	10	10	10	11
$(2, 0, 4)$	26	28	25	28
$(2, 2, 1)$	180	192	170	136
$(5, 1/2, 1)$	31	28	30	41
$(5, 1/2, 4)$	88	81	72	108
$(5, 0, 1)$	20	22	23	29
$(5, 0, 4)$	48	51	51	62
$(5, 1, 1)$	111	125	126	142

1. Dans la plupart des cas, la performance de notre algorithme est similaire à celle du RAPTOR. Une exception notable est le cas 5 où les deux modes sont éloignés. En fait, il est curieux que l'algorithme RAPT soit autant supérieur dans cette situation, ce qui porte à croire que c'est encore le hasard qui domine les résultats pour un si petit nombre d'itérations. Dans tous les cas, une solution simple pour améliorer l'efficacité serait d'utiliser deux chaînes parallèles bien dispersées.

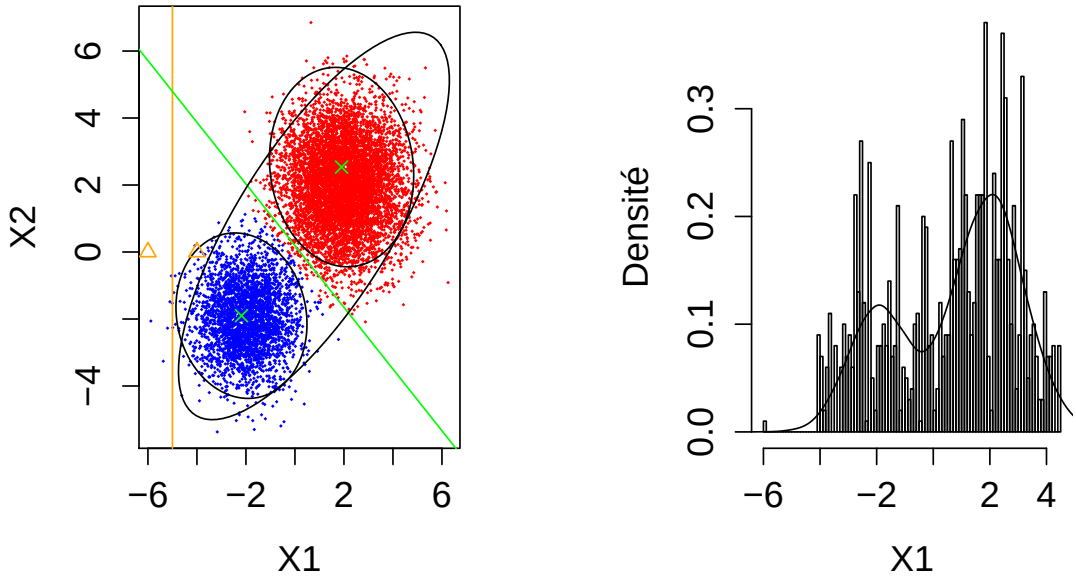

FIGURE 1 – Cible : $\lambda = 0.3$, $\mu_1 = (-2, -2)$, $\mu_2 = (2, 2)$, $\Sigma_1 = I_2$, $\Sigma_2 = 1.5I_2$, Algorithme : $N = 1000$, $t_0 = 100$, Concordance : 0.992, Indice de Proportion : 0.042.

2. Dans les situations où $d = 5$, notre algorithme semble meilleur que RAPT, qui rappelons-le souffre d'une d'une partition initiale sous-optimale. Ceci semble indiquer que l'adaptation de la frontière que nous avons implantée fonctionne et améliore effectivement les résultats.
3. La version avec un second calcul régional à la première adaptation ne semble pas apporter d'amélioration substantielle pour ce genre de distributions. De plus, d'autres tests semblent indiquer que mis à part une meilleure concordance par rapport aux régions optimales théoriques, cette seconde version n'apporte aucun gain notable, tout en étant marginalement plus coûteuse. Nous poursuivrons donc uniquement avec la première version de l'algorithme.

4.4.2 Robustesse

Nous présentons maintenant quelques tests conçus pour évaluer les limites de notre algorithme. Nous débutons en deux dimensions afin de pouvoir bien illustrer visuellement le comportement adaptatif. Dans tous les cas, nous utilisons une distribution cible de la

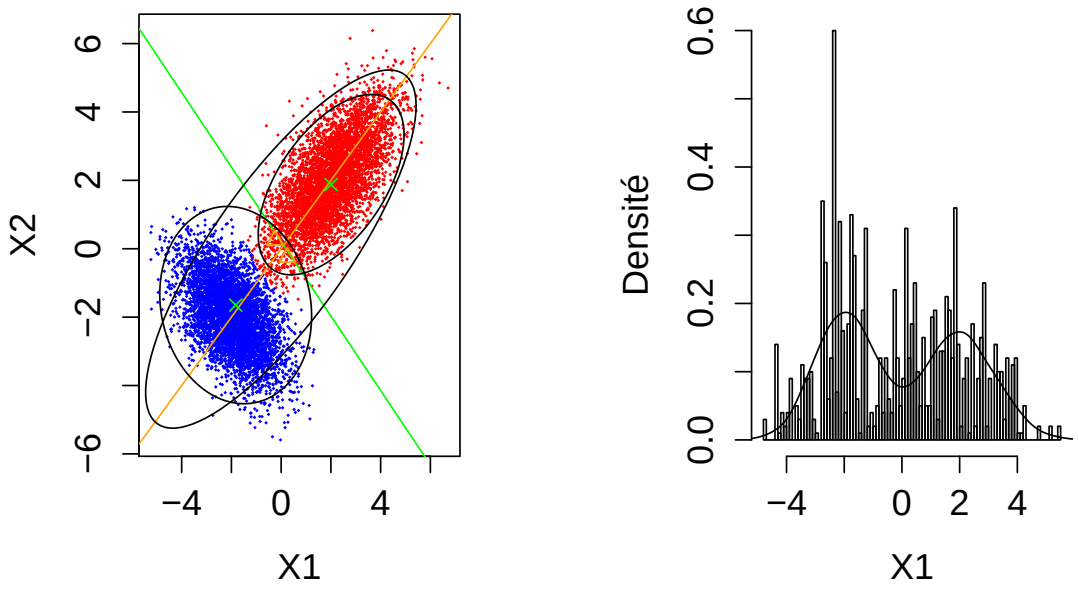

FIGURE 2 – Cible $\lambda = 0.5$, $\mu_1 = (-2, -2)$, $\mu_2 = (2, 2)$, $\Sigma_1 = 0.1(2, -1; -1, 2)$, $\Sigma_2 = 0.1(3, 2; 2, 3)$. Algorithme : $N = 1000$, $t_0 = 100$, Concordance : 0.939, Indice de Proportion : 0.002.

forme

$$\lambda \mathcal{N}_d(\mu_1, \Sigma_1) + (1 - \lambda)\mathcal{N}_d(\mu_2, \Sigma_2), \tag{1}$$

avec paramètres qui seront précisés à chaque fois. Nous aurons toujours deux chaînes parallèles démarrant de part et d'autre du plan formant délimitation initiale avec $C_1 = C_2 = 0.1I_d$ et $C_S = 25I_d$. Les résultats graphiques sont présentés de la façon suivante : on affiche un échantillon i.i.d de 10 000 valeurs tirées de la distribution cible, colorées en rouge ou en bleu en fonction de leur région d'appartenance théorique (les points où la densité de la première composante est supérieure à celle de la seconde sont en bleu, et vice-versa), avec en orangé la délimitation et les valeurs initiales de la chaîne, en vert la délimitation et les moyennes estimées $\hat{\mu}^{(1)}$ et $\hat{\mu}^{(2)}$ finales, et en noir les régions de confiances de niveau 95% estimées régionales et globales, d'après les matrices de covariance finales. Nous indiquons le taux de concordance de l'échantillon i.i.d, c'est-à-dire la proportion de points pour lesquels la région d'appartenance théorique et empirique coïncident, ainsi qu'un indice de l'équilibre

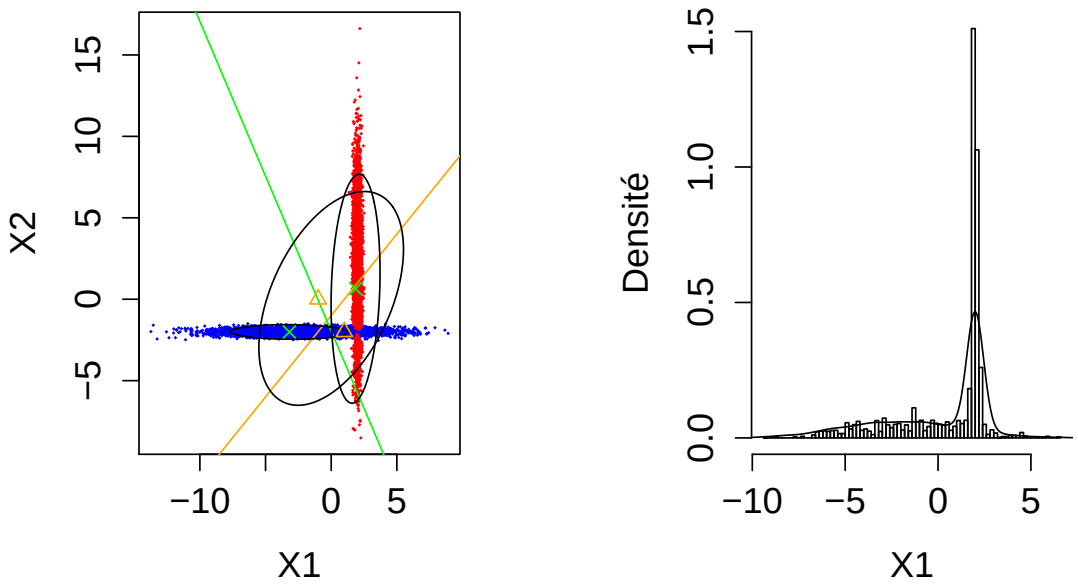

FIGURE 3 – $\lambda = 0.5$, $\mu_1 = (-2, -2)$, $\mu_2 = (2, 2)$, $\Sigma_1 = \text{diag}(10, 0.025)$, $\Sigma_2 = \text{diag}(0.025, 10)$. Algorithme : $N = 5000$, $t_0 = 500$, Concordance : 0.842, Indice de Proportion : 0.188.

entre les modes, donné par

$$\left| \left| \frac{|S_{01}| - |S_{02}|}{N} \right| - |\lambda - (1 - \lambda)| \right|,$$

qui s'avère souvent un bon indicateur de la qualité de l'échantillon (plus l'indice est petit, plus l'équilibre entre les modes devrait être bon). Dans tous les cas, nous avons pris un faible nombre d'itérations et considéré l'ensemble des valeurs obtenues dans notre échantillon, afin d'illustrer la rapidité d'adaptation du processus.

FIGURE 1 : On voit que malgré un plan initial complètement erroné, l'algorithme parvient à rétablir rapidement la situation, comme en témoigne le taux de concordance élevé. Remarquons que les estimés de la moyenne et de la covariance des composantes sont déjà très intéressants.

FIGURE 2 : On voit ici que les covariances empiriques s'adaptent convenablement dans chaque région. Sans surprise, davantage d'itérations seront nécessaires pour bien explorer les queues de la distribution.

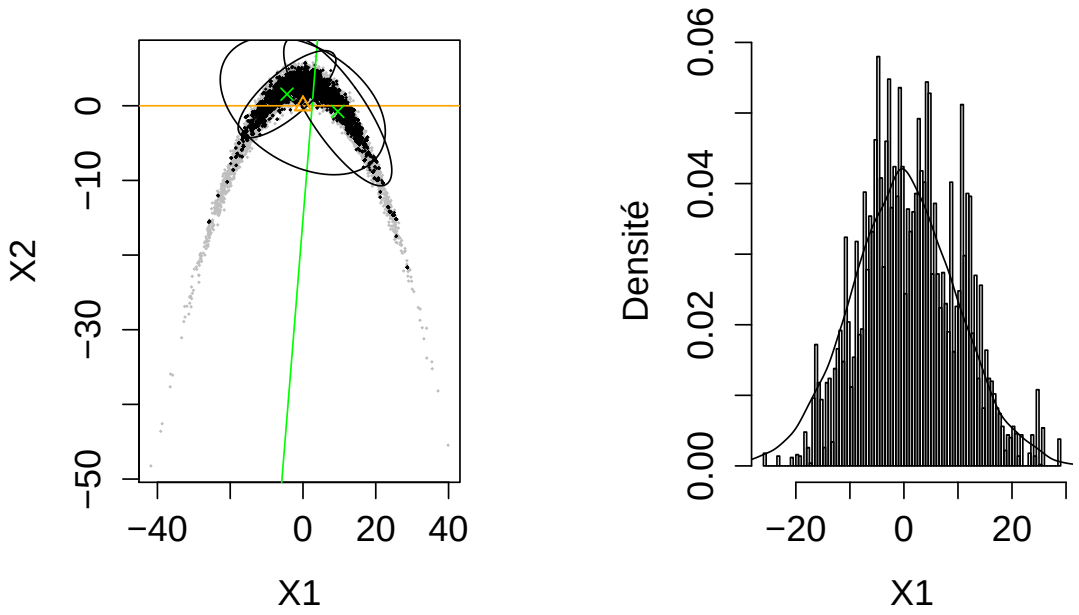

FIGURE 4 – Distribution "banane" de HST, avec $N = 10\ 000$, $t_0 = 1000$ et $b = 0.03$. Échantillon i.i.d. (en gris) avec celui de l'algorithme (en noir).

FIGURE 3 : Voici un exemple d'une distribution particulièrement difficile, où l'algorithme parvient tout de même rapidement à déterminer la forme générale de la cible, ainsi qu'un plan intéressant.

FIGURE 4 : Cette distribution irrégulière est suggérée par [4]. La densité d'un point donné dans $\mathbb{R}^d$ vaut $f(x_1, x_2 + bx_1^2 - 100b, x_3, \ldots, x_d)$, où f est une densité normale de moyenne 0_d et covariance $\mathrm{diag}(100, 1, \ldots, 1)$. Ici, on voit que même pour une distribution unimodale, un algorithme d'adaptation régionale peut engendrer des situations intéressantes, par exemple dans ce cas-ci une proposition adaptée à chaque branche de la distribution et une covariance globale permettant de passer d'une à l'autre et d'aller chercher des valeurs extrêmes.

4.4.3 Comportement asymptotique

Passons maintenant à des simulations de grande taille en plus grande dimension, donc semblables aux problèmes MCMC typiques. La forme de la densité cible sera identique à

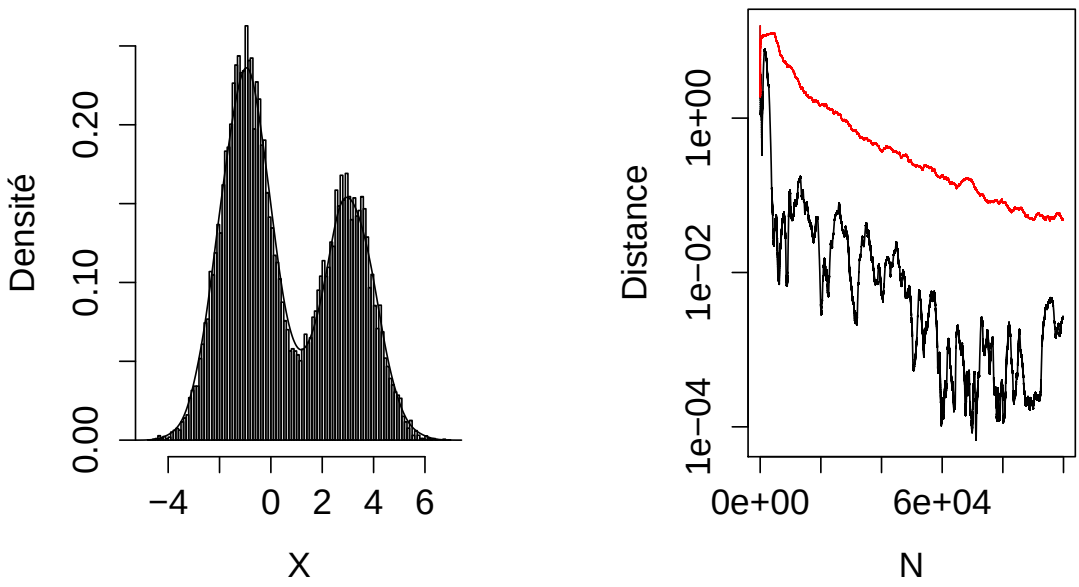

FIGURE 5 – Histogramme avec densité attendue et évolution de l'écart quadratique à la moyenne pour l'algorithme Road Runner (en noir) et RAPT (en rouge). $N = 100\ 000$, $t0 = 100$, $d = 5$, $\lambda = 3/5$, $\mu_1 = (-1, \ldots, -1), \mu_2 = (3, \ldots, 3)$ et $\Sigma_1 = \Sigma_2 = I$.

(1), mais maintenant en dimensions variées. Pour chaque exemple, nous affichons comme résultats graphiques l'histogramme d'une composante choisie aléatoirement ainsi que la progression de la distance quadratique moyenne cumulative entre la vraie moyenne et celle observée, afin de vérifier la convergence de l'algorithme. En guise de comparaison heuristique, nous affichons la même statistique telle qu'obtenue sur un échantillon généré par l'algorithme RAPT.

Dans tous les cas, nous utilisons 4 chaînes parallèles et les paramètres initiaux sont $C_1 = C_2 = I_d$, $C_S = 10I_d$, $a_0 = (-1, 1, 0, \ldots, 0, 0)$, $b_0 = 0$, et $X_0 = (i, \ldots, i)$, avec $i \in \{-2, -1, 1, 2\}$. Pour le RAPT, les paramètres du plan engendrent une division sous-optimale, mais pas catastrophique. Les spécificités de chaque exemple et leur illustration sont données dans les figures 5, 6 et 7.

Dans toutes les situations étudiées, on voit que notre algorithme semble converger éventuellement vers la distribution cible. Par contre, on remarque aussi une plus grande instabilité en début de chaîne comparativement au RAPT et une certaine variabilité dans

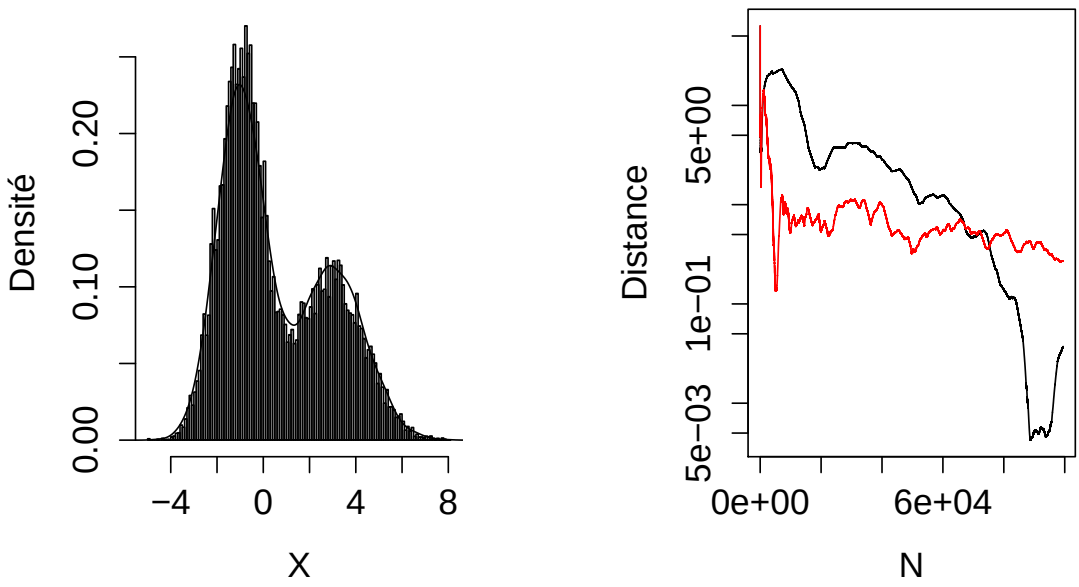

FIGURE 6 – Histogramme avec densité attendue et évolution de l'écart quadratique à la moyenne pour l'algorithme Road Runner (en noir) et RAPT (en rouge). $N = 100\ 000$, $t0 = 500$, $d = 10$, $\lambda = 3/5$, $\mu_1 = (-1, \ldots, -1), \mu_2 = (3, \ldots, 3)$, $\Sigma_1 = I$, $\Sigma_2 = 2I$.

l'évolution de l'erreur quadratique moyenne, ce qui nous empêche pour l'instant de départager nettement les deux méthodes. Un moyen permettant peut-être de stabiliser plus rapidement la partition serait de pondérer l'emplacement du plan entre les moyennes selon les covariances empiriques de chaque région, qui sont déjà stockées par l'algorithme. Finalement, bien que le gain en efficacité par rapport au RAPT soit parfois marginal, rappelons que la différence en terme de calculs nécessaires l'est également.

4.5 Conclusion

Pour résumer cette première partie, nous avons proposé une amélioration d'un algorithme d'adaptation régionale dans le cas $K = 2$ par une composante qui adapte de façon dynamique la partition de l'espace de la façon la moins coûteuse possible. Ce nouvel algorithme se compare souvent avantageusement à ses compétiteurs de complexité semblable et produit des résultats à peu près aussi intéressants que l'algorithme RAPTOR, tout en étant légèrement plus rapide. Mentionnons pour terminer qu'il serait possible de généra-

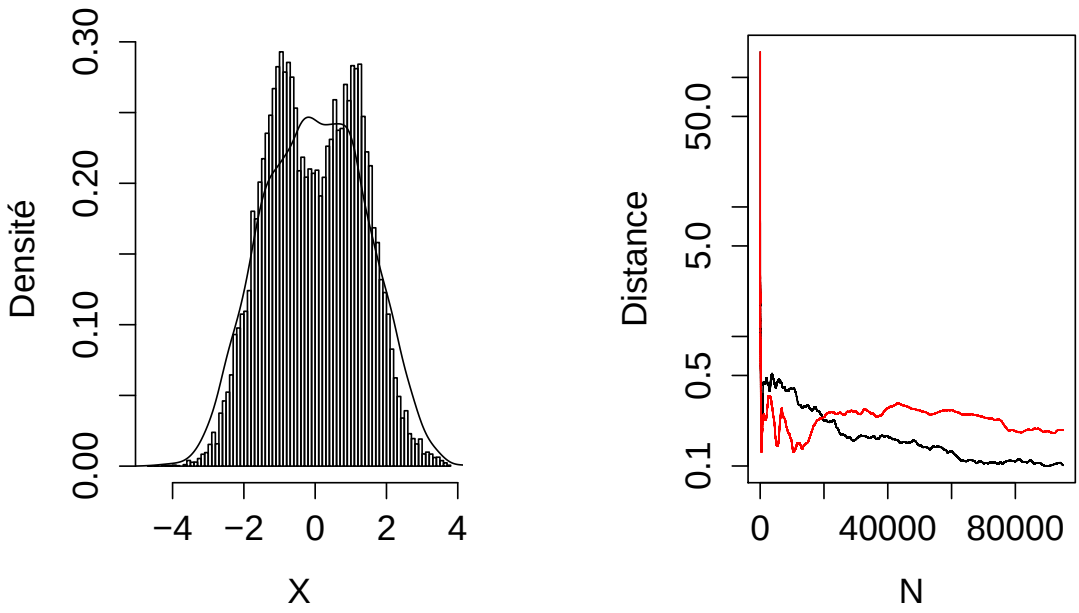

FIGURE 7 – Histogramme avec densité attendue et évolution de l'écart quadratique à la moyenne pour l'algorithme Road Runner (en noir) et RAPT (en rouge). $N = 100\ 000$, $t0 = 5000$, $d = 50$, $\lambda = 3/5$, $\mu_1 = (-1, \ldots, -1), \mu_2 = (1, \ldots, 1)$, $\Sigma_1 = \Sigma_2 = I$.

liser cet algorithme pour les situations où $K > 2$, ce qui pourra faire l'objet de travaux ultérieurs.

5 Le modèle climatique

5.1 Généralités

Nous aborderons ici brièvement le fonctionnement des modèles physiques du climat. Ceux-ci sont basés sur des équations décrivant des phénomènes et des relations physiques bien connus, tels que la mécanique des fluides, la conservation de la masse, les lois thermodynamiques, l'équilibre hydrostatique. Considérées conjointement sur l'ensemble de l'atmosphère et de la surface terrestre, ces équations permettent de décrire la distribution et les déplacements de l'énergie et de l'humidité par rapport au temps. Ainsi, la résolution de ce système d'équations différentielles, pour la plupart à dérivées partielles, muni d'un

certain ensemble de valeur initiales détaillées au temps zéro, devrait en principe permettre de déterminer les variables climatiques présentes dans les équations en tout point de la terre et en tout instant. En pratique, ces équations n'ont pas de solution analytique et leur nature chaotique rend inutilisables toutes simplifications de celles-ci. On a donc recours à des méthodes numériques afin de résoudre ces équations sur une représentation discrétisée de l'espace terrestre. On utilise habituellement un maillage uniforme pour les coordonnées horizontales et une représentation plus précise près de la surface pour les coordonnées verticales. Dans les modèles de dernière génération, l'ensemble des variables climatiques est calculé sur tous les points de cette grille tri-dimensionnelle à des intervalles de temps variant entre 5 et 30 minutes.

5.2 Paramétrisation

Certains processus physiques se déroulent sur une échelle trop petite pour être bien représentés par la résolution numérique de la grille. Il s'agit entre autres des phénomènes liés à la formation des nuages, à la condensation et aux précipitations. Ceux-ci ont un effet non-négligeable sur les phénomènes de plus grande échelle et doivent donc être modélisés d'une certaine façon. On utilise alors un paramétrage physique, c'est-à-dire un ensemble de relations permettant de modéliser leur effet en fonction de variables connues. Ces paramétrages prennent différentes formes selon le modèle et la résolution utilisés et sont en partie responsables des disparités entre les différents modèles climatiques. De plus, la plupart de ces relations dépendent de coefficients ajustables, qui ajoutent une incertitude supplémentaire aux résultats. La valeur de ces coefficients est habituellement décidée à partir de résultats empiriques et des connaissances acquises sur le phénomène en question, mais laisse une certaine place à la subjectivité. Depuis quelques années, des efforts sont faits afin de déterminer pour ces relations des paramètres optimaux au sens statistique. Parmi un ensemble de valeurs plausibles, on tente de déterminer celles engendrant, à l'aide d'un modèle climatique donné, les résultats les plus conformes aux données observées.

Considérons le cas simple suivant, tel que discuté dans [14]. Soient θ, un vecteur de paramètres dont on veut déterminer la valeur optimale, $F(\theta)$, une variable de sortie globale du modèle climatique après un certain temps T, utilisant θ et un ensemble de valeurs initiales observées comme arguments, F° la valeur réellement observée de cette variable après un temps T, et σ^2, la variance de cette variable en question, que l'on peut estimer assez précisément à l'aide d'une base de données climatiques par exemple. On peut alors

tenter de minimiser la fonction de coût suivante :

$$J_F(\theta) = \frac{F(\theta) - F^\circ}{\sigma^2}.$$

Or, un tel choix utilise très peu l'information à notre disposition. Une fonction plus intéressante serait celle-ci, où Y représente l'ensemble des régions (horizontales, verticales, ou les deux) de la grille :

$$J_F(\theta) = \frac{(F(\theta) - F^\circ)^2}{\sigma^2} + \sum_{t=1}^{T} \sum_{y \in Y} \frac{(F_{t,y}(\theta) - F^\circ_{t,y})^2}{\sigma^2_{t,y}}. \tag{2}$$

La fonction F nécessitant l'évaluation complète du modèle climatique, il n'existe pas de solution analytique à ce problème. Une avenue possible est d'aborder ce problème sous l'angle de l'inférence bayésienne. Supposons que nous voulions déterminer la probabilité qu'un choix de paramètre θ soit conforme, sous le modèle F, aux données observées F° :

$$\mathbb{P}(\theta | F^\circ, F) \propto \mathbb{P}(F - F^\circ | \theta)\mathbb{P}(\theta).$$

En supposant une erreur de loi normale sur les observations F° et une distribution à priori uniforme pour θ, nous obtenons la relation :

$$\mathbb{P}(\theta | F^\circ, F) \propto \exp(-J_F(\theta)/2),$$

où $J_F(\theta)$ est telle que définie dans (2). Nous pouvons alors utiliser les MCMC pour créer un échantillon approprié de la distribution à posteriori. Il va de soi que le choix de la variable étudiée, ainsi que la forme du critère J influencera grandement la qualité des résultats obtenus.

5.3 Applications au modèle de Lorenz

Le modèle Lorenz-96 énoncé dans [8] est un système unidimensionnel d'équations différentielles ordinaires de premier ordre. Il ne représente pas un phénomène naturel en particulier, mais ses variables ont un comportement analogue à certaines valeurs atmosphériques, notamment un certain caractère périodique et une évolution chaotique dans le temps (voir figure 8). Il s'agit donc d'un canevas intéressant permettant de tester nos méthodes sur un système de moindre envergure, mais relativement représentatif. Le système

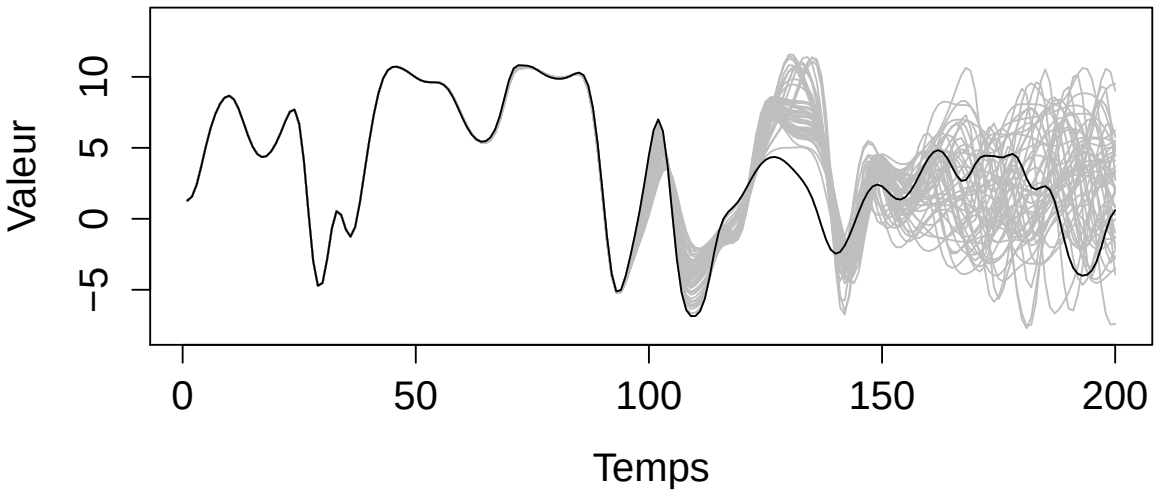

FIGURE 8 – Illustration du comportement chaotique du modèle de base (3) : trajectoire temporelle originale en noir d'une variable et 50 trajectoires avec valeur initiale très légèrement perturbée en gris.

contient les variables $X_1, \ldots, X_K$, avec $K \geq 4$ et est régi par les K équations suivantes

$$dX_k/dt = X_{k-1}(X_{k+1} - X_{k-2}) - X_k + G, \tag{3}$$

où les X_k sont en relation cyclique (donc $X_{K+1} = X_1$ et $X_0 = X_K$) et G est une constante indépendante de k. Ainsi, nous pourrions considérer que les X_k représentent la mesure d'une certaine quantité à différentes longitudes pour une latitude et une altitude fixées.

Afin d'introduire un problème de paramétrisation dans ce modèle, nous ajoutons une seconde famille de variables, également cycliques, mais beaucoup plus nombreuses :

$Y_{1,1}, Y_{2,1} \ldots, Y_{J,1}, Y_{1,2}, \ldots, Y_{J,K}$, pour une certaine valeur $J \geq 2$. Celles-ci sont liées entre elles et aux variables X_k par le système suivant, qui remplace le précédent :

$$dX_k/dt = X_{k-1}(X_{k+1} - X_{k-2}) - X_k + G + \frac{hc}{b}\sum_{j=1}^{J} Y_{j,k},$$

$$dY_{j,k}/dt = cbY_{j-1,k}(Y_{j-1,k} - Y_{j+2,k}) - cY_{j,k} + \frac{c}{b}G + \frac{hc}{b}X_k,$$

avec les équivalences suivantes : $Y_{j+J,k} = Y_{j,k+1}$, $Y_{j-J,k} = Y_{j,k-1}$. Les paramètres h, b et c

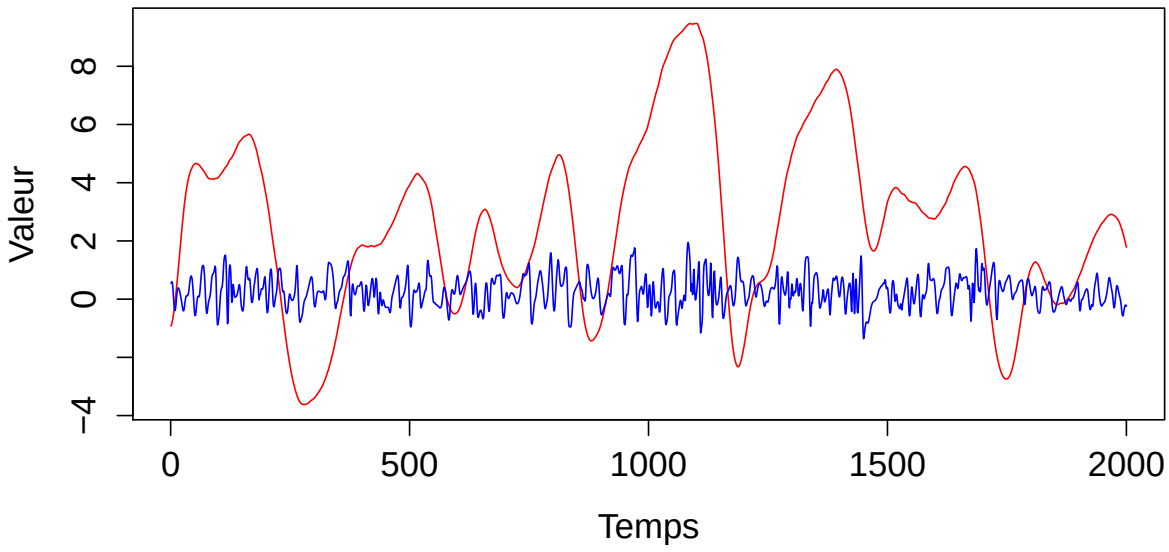

FIGURE 9 – Illustration du modèle complet avec $K = 36$ et $J = 10$. Comportement temporel d'une variable X (en rouge) et d'une variable Y (en bleu).

contrôlent respectivement le degré de couplage entre X et Y, le rapport inverse entre leur fréquence de fluctuation et le rapport de leur amplitude . Ils sont habituellement fixés à 1, 10 et 10, respectivement. Ainsi, les variables Y fluctueront 10 fois plus rapidement dans le temps et l'espace que les variables X et leur amplitude sera 10 fois moindre (voir figures 9 et 10).

Pour faire un parallèle avec le problème de paramétrisation présenté auparavant, supposons que les indices $1, \ldots, K$ représentent les points de discrétisation du modèle. Ainsi, dans le modèle de base (3), les X_k seraient bien représentés. Par contre, dans le second modèle, il faut tenir compte de l'effet connu et non-négligeable des variables Y, bien que les fluctuations de ces dernières se produisent à une trop petite échelle pour être représentées correctement (variables non-résolues). On tentera donc de modéliser approximativement leur comportement par une paramétrisation à l'échelle des indices $1, \ldots, K$, tel qu'expliqué dans [15]. Les équations précédentes nous permettent de déterminer la valeur véritable des variables X_k, mais nous supposerons qu'en pratique, il nous faudra utiliser le modèle

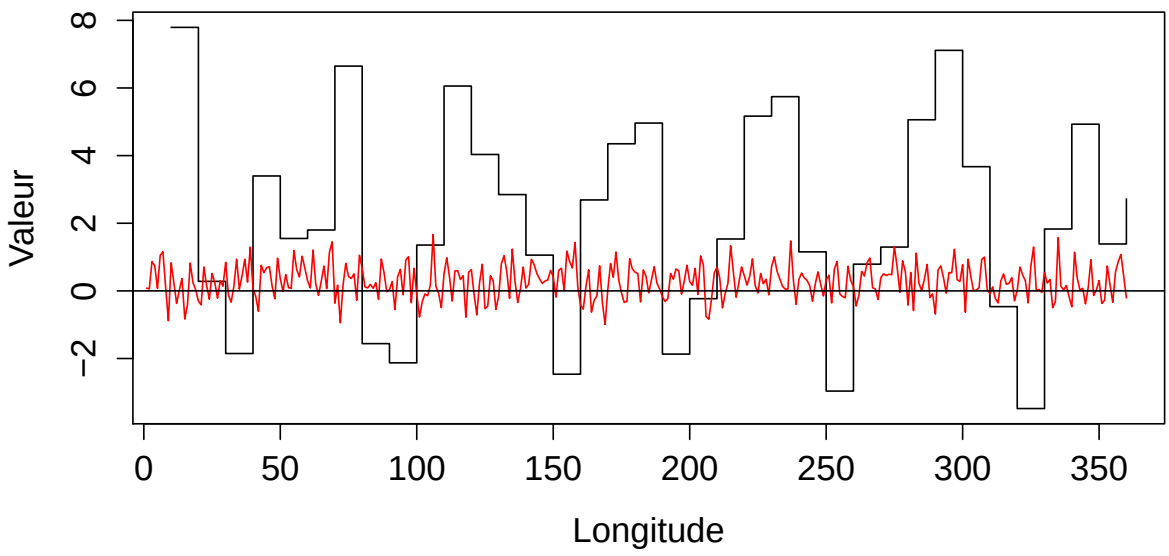

FIGURE 10 – Illustration du modèle complet avec $K = 36$ et $J = 10$. Valeurs simultanées à un temps donné de toutes les variables X (en noir) et Y (en rouge).

simplifié suivant :

$$dX_k/dt = X_{k-1}(X_{k+1} - X_{k-2}) - X_k + J_F(\theta)G - g(X_k) + \epsilon_k,$$

où ϵ_k est un terme d'erreur que nous supposerons normal et $g(X_k) = \theta_0 + \theta_1 X_k$ est une paramétrisation linéaire de l'apport des variables non-résolues. La tâche sera donc de créer un échantillon de la distribution à posteriori $\exp(-J(\theta_0, \theta_1)/2)$ en la considérant comme la densité cible d'une chaîne MCMC et de déterminer une fonction de coût J adaptée à la situation, c'est-à-dire qui nous permettra d'identifier les meilleurs choix de paramètres. Nous présentons maintenant une façon parmi d'autres d'aborder et de résoudre ce problème.

5.4 Concordance statistique

La définition du modèle complet nous permet de générer un ensemble de valeurs X_t, $t = 1, \ldots, T$ de références et une covariance empirique $\hat{\Sigma}$ déterminée à partir d'une longue simulation du modèle. Dans une situation plus simple, nous pourrions alors tout simplement générer, pour une paramétrisation θ donnée et à partir des mêmes valeurs initiales $X_{k,0}$,

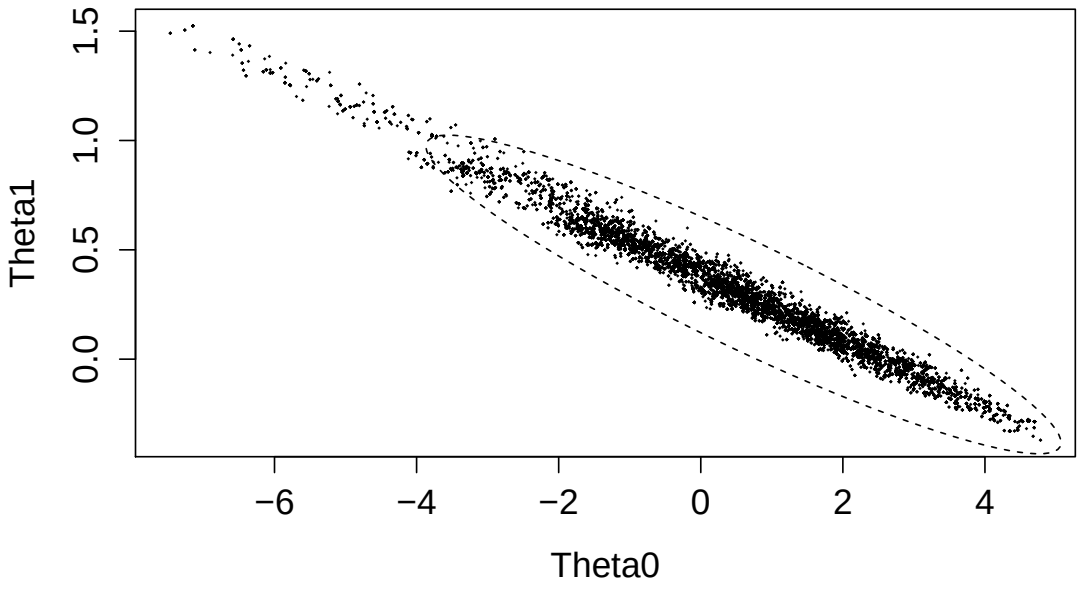

FIGURE 11 – Échantillon MCMC par l'algorithme AM avec $t_0 = 1000$, $N = 20000$, $C_0 = I/10$, et une distribution à priori pour θ uniforme sur $[-10, 15] \times [-1, 2]$. Affichage d'une région de confiance normale de 95%, d'après la matrice de covariance C_t finale.

un ensemble de valeurs estimées $X_t^{(\theta)}$ et utiliser un critère des moindres carrés normalisés pour définir la fonction J :

$$J(\theta) = \frac{1}{T} \sum_t (X_t - X_t^{(\theta)})^T \hat{\Sigma}^{-1} (X_t - X_t^{(\theta)}).$$

Malheureusement, dans le modèle à l'étude, ce critère ne sera pas suffisant. En effet, le comportement chaotique des variables provoquera toujours un écart marqué dans les comparaisons point par point, peu importe la qualité de la paramétrisation. Nous cherchons plutôt les modèles paramétrés dont le *comportement* est similaire à celui du vrai modèle. Pour ce faire, nous pourrions plutôt mesurer l'écart entre certaines statistiques calculées sur les valeurs X_t et $X_t^{(\theta)}$. Considérons donc un vecteur de mesures statistiques, $S = (S_1, \ldots, S_K)$, prises sur les vraies valeurs $X_{k,t}$ du modèle, sa matrice de covariance empirique C_S, basée sur un grand nombre d'échantillons S mesurés sur des modèles de même durée T, ainsi que son homologue $S(\theta)$, calculé sur les valeurs du modèle paramétré

par θ. Si les composantes de S reflètent bien les caractéristiques des variables, ainsi que leurs comportements temporel et spatial, alors la fonction de coût suivante devrait être une bonne mesure de la vraisemblance d'un paramètre θ donné

$$J(\theta) = (S - S(\theta))^T C_S^{-1} (S - S(\theta)).$$

Le défi est maintenant de déterminer un ensemble de statistiques traduisant bien le comportement du modèle, mais étant aussi restreint que possible. Dans [5], on suggère pour S les statistiques suivantes : moyenne, variance, covariance, auto-covariance, et covariances croisées avec décalage de 1 à gauche et à droite. On obtient alors à l'aide de l'algorithme AM un échantillon de la distribution $\mathbb{P}(\theta|X)$ (voir figure 11) à partir duquel on peut déterminer l'espérance des paramètres, en l'occurrence : $(\theta_0, \theta_1) \approx (0.60, 0.30)$. Or, on peut se demander à quel point le choix de statistiques résume bien le comportement du modèle et si un autre choix donnerait des valeurs semblables ou pas.

En fait, ce problème est présent aussi lorsque d'autres méthodes sont utilisées et il n'est pas rare que des techniques toutes théoriquement valables mènent à des résultats significativement différents (voir [6]). Nous conclurons simplement en mentionnant que l'estimation de paramètres dans des systèmes chaotiques pose un grand défi à la communauté scientifique, qui cherche encore une méthode générale permettant de déterminer des paramètres optimaux de façon précise et unique.

Références

[1] Y. Bai, R. V. Craiu, and A. F. Di Narzo, "Divide and conquer : a mixture-based approach to regional adaptation for MCMC," *J. Comput. Graph. Statist.*, vol. 20, no. 1, pp. 63–79, 2011, supplementary material available online. [Online]. Available : http ://dx.doi.org/10.1198/jcgs.2010.09035

[2] R. V. Craiu, J. Rosenthal, and C. Yang, "Learn from thy neighbor : parallel-chain and regional adaptive MCMC," *J. Amer. Statist. Assoc.*, vol. 104, no. 488, pp. 1454–1466, 2009. [Online]. Available : http ://dx.doi.org/10.1198/jasa.2009.tm08393

[3] A. Gelman, G. O. Roberts, and W. R. Gilks, "Efficient Metropolis jumping rules," in *Bayesian statistics, 5 (Alicante, 1994)*, ser. Oxford Sci. Publ. Oxford Univ. Press, New York, 1996, pp. 599–607.

[4] H. Haario, E. Saksman, and J. Tamminen, "An adaptive metropolis algorithm," *Bernoulli*, vol. 7, no. 2, pp. 223–242, 2001. [Online]. Available : http ://dx.doi.org/10.2307/3318737

[5] J. Hakkarainen, A. Ilin, A. Solonen, M. Laine, H. Haario, J. Tamminen, E. Oja, and H. Järvinen, "On closure parameter estimation in chaotic systems," *Nonlinear Processes in Geophysics*, vol. 19, no. 1, pp. 127–143, 2012.

[6] J. Hakkarainen, A. Solonen, A. Ilin, J. Susiluoto, M. Laine, H. Haario, and H. Järvinen, "A dilemma of the uniqueness of weather and climate model closure parameters," *Tellus A*, vol. 65, no. 0, 2013. [Online]. Available : http ://www.tellusa.net/index.php/tellusa/article/view/20147

[7] W. K. Hastings, "Monte carlo sampling methods using markov chains and their applications," *Biometrika*, vol. 57, no. 1, pp. pp. 97–109, 1970. [Online]. Available : http ://www.jstor.org/stable/2334940

[8] E. N. Lorenz, "Predictability : A problem partly solved," in *Proc. Seminar on predictability*, vol. 1, no. 1, 1996.

[9] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller, "Equation of state calculations by fast computing machines," *J. Chem. Phys.*, vol. 21, p. 1087, 1953.

[10] S. Meyn and R. L. Tweedie, *Markov chains and stochastic stability*, 2nd ed. Cambridge University Press, Cambridge, 2009, with a prologue by Peter W. Glynn. [Online]. Available : http ://dx.doi.org/10.1017/CBO9780511626630

[11] C. P. Robert and G. Casella, *Monte Carlo statistical methods*, 2nd ed., ser. Springer Texts in Statistics. Springer-Verlag, New York, 2004. [Online]. Available : http ://dx.doi.org/10.1007/978-1-4757-4145-2

[12] G. O. Roberts and J. S. Rosenthal, "General state space Markov chains and MCMC algorithms," *Probab. Surv.*, vol. 1, pp. 20–71, 2004. [Online]. Available : http ://dx.doi.org/10.1214/154957804100000024

[13] ——, "Coupling and ergodicity of adaptive Markov chain Monte Carlo algorithms," *J. Appl. Probab.*, vol. 44, no. 2, pp. 458–475, 2007. [Online]. Available : http ://dx.doi.org/10.1239/jap/1183667414

[14] A. Solonen, P. Ollinaho, M. Laine, H. Haario, J. Tamminen, and H. Järvinen, "Efficient MCMC for climate model parameter estimation : parallel adaptive chains and early rejection," *Bayesian Anal.*, vol. 7, no. 3, pp. 715–736, 2012. [Online]. Available : http ://dx.doi.org/10.1214/12-BA724

[15] D. S. Wilks, "Effects of stochastic parametrizations in the lorenz'96 system," *Quarterly Journal of the Royal Meteorological Society*, vol. 131, no. 606, pp. 389–407, 2005.