High dimensional Bayesian computation

Alexander Buchholz

NNT : 2018SACLG004

THÈSE DE DOCTORAT

de

L'UNIVERSITÉ PARIS-SACLAY

École doctorale de mathématiques Hadamard (EDMH, ED 574)

Établissement d'inscription : École nationale de la statistique et de l'administration économique

Établissement d'accueil : Centre de recherche en économie et statistique

Laboratoire d'accueil : Laboratoire de statistique

Spécialité de doctorat : Mathématiques appliquées

Alexander BUCHHOLZ

High dimensional Bayesian computation

Date de soutenance : 22 Novembre 2018

Après avis des rapporteurs : SYLVIA RICHARDSON (University of Cambridge)
CHRISTIAN P. ROBERT (Université Paris Dauphine)

Jury de soutenance :

NICOLAS CHOPIN	(ENSAE) Directeur de thèse
ESTELLE KUHN	(INRA) Examinateur
KERRIE MENGERSEN	(Queensland University of Technology) Examinateur
SYLVIA RICHARDSON	(University of Cambridge) Rapporteur
CHRISTIAN P. ROBERT	(Université Paris Dauphine) Président, Rapporteur
ROBIN RYDER	(Université Paris Dauphine) Invité

ENSAE

DOCTORAL THESIS

High Dimensional Bayesian Computation

Author:
Alexander BUCHHOLZ

Supervisor:
Prof. Nicolas CHOPIN

A thesis submitted in fulfillment of the requirements
for the degree of Doctor of Philosophy in Applied Mathematics

prepared in the

Laboratoire de Statistique
ENSAE-CREST Paris

22 November 2018

This page intentionally left blue.

"Das akademische Leben ist also ein wilder Hazard. [...] Der Einfall ersetzt nicht die Arbeit. Und die Arbeit ihrerseits kann den Einfall nicht ersetzen oder erzwingen, so wenig wie die Leidenschaft es tut. Beide – vor allem: beide zusammen – locken ihn. Aber er kommt, wenn es ihm, nicht, wenn es uns beliebt. "

"Hence academic life is a mad hazard. [...] The idea is not a substitute for work; and work, in turn, cannot substitute for or compel an idea, just as little as enthusiasm can. Both, enthusiasm and work, and above all both of them jointly, can entice the idea. Ideas occur to us when they please, not when it pleases us. "

Max Weber - Wissenschaft als Beruf (1919)

Acknowledgements

First and foremost I would like to thank Nicolas. I am grateful for his patience, his advice, his invaluable training, his qualities as a mentor, as a teacher and as a true researcher. Without him this thesis would not have been possible.

I would like to thank the two referees Sylvia and Christian for having accepted to review this thesis. I also acknowledge the members of the committee: thank you Kerrie, Estelle and Robin. Then I would like to thank all the invaluable members of the CREST who made my time as a PhD student a most enjoyable one: I would like to thank Sascha, Arnak, Guillaume, Pierre and Marco. I also want to acknowledge my fellow PhD students, with whom I shared three important years of my life: Vincent, Mehdi, Edwin, Ti-tien, Lionel, Badr, Boris, Mohammed, Lena, Solènne, Philipp, Jeremie, Gauthier, Geoffrey.

Moreover I want to thank Pierre for welcoming me in Boston. My time there was extremely valuable, both from a scientific and personal perspective. In this line I would like to thank Alice and Jeremy for becoming my friends during my time in Boston. I would like to thank Florian and Stephan for our great collaboration and for all they taught me. The conferences I attended would not have been as much fun without Anna, Arthur, Robin, Elvis, Eugène, Julien.

I also would like to thank all the staff at the ENSAE and the CREST for making my research possible and providing the necessary logistic support. I acknowledge funding from the GENES throughout my PhD. I want to thank the KAS and the DAAD for a stimulating intelectual environment and financial support throughout my studies.

Without my family I would not be where I am now. Thank you for supporting me Christof, Delia, Ivonne, Marilena, Amina, Hilke, Simba. My friends were an important part of these sometimes difficult three years: thank you Georg, Bene, Leo, Sarah, Max, Andrea, Clemens, Guillaume, Sophie, Émilien, Clément, Chloé, Éléonore, Victor, Quentin, Toni, Mourad. And last I would like to thank Audrey for cheering me up and standing my capriciousness during these three years. Thank you for being there for me.

Contents

Denen, die vor mir waren, mit
mir sind und nach mir kommen.

A ceux qui étaient avant moi,
m'accompagnent et seront
ensuite.

To those, who were before me, are
with me and shall come after.

Chapter 1

Introduction (in French)

Ce chapitre donne une introduction aux travaux exposés dans la suite de cette thèse. Nous introduisons les concepts nécessaires à la compréhension des problèmes posés par le calcul bayésien. Nous développons d'abord les idées principales de la statistique bayésienne. Nous discutons ensuite les techniques d'échantillonnages Monte Carlo et quasi-Monte Carlo. Après cela, nous abordons l'optimisation stochastique et la construction des approximations variationnelles aux distributions *a posteriori*. Enfin, nous résumons les principales contributions des trois articles faisant l'objet des chapitres suivants de cette thèse.

1.1 Inférence bayésienne

La modélisation statistique a pour but de comprendre un phénomène à partir de données. Mathématiquement parlant, ce problème est décrit comme le tuple d'un espace d'observation $\mathcal{Y}$, son ensemble borélien $\mathcal{B}(\mathcal{Y})$ et une famille de mesures de probabilité $\mathbb{P}_x$, où $x \in \mathcal{X}$ et $\mathcal{X}$ est l'espace des paramètres. $(\mathcal{Y}, \mathcal{B}(\mathcal{Y}), \{\mathbb{P}_x, x \in \mathcal{X}\})$ constitue un modèle. Si $\mathcal{X} \subset \mathbb{R}^d$ et $d < \infty$ le modèle est dit paramétrique, c'est la cas que nous considérerons ici. Étant donné une suite de réalisations $y_1, \cdots, y_N$ de longueur N des variables aléatoires $Y_1, \cdots, Y_N \in \mathcal{Y}$, l'inférence statistique vise à identifier le paramètre x étant donné les $y_1, \cdots, y_N$ observées.

Dans ce qui suit, nous supposons que la mesure de probabilité $\mathbb{P}_x$ est dominée par une mesure de référence, ci-après désigné par $\mathrm{d}\, y$. La vraisemblance du modèle est donnée par :

$$p : x \times (y_1, \cdots, y_N) \mapsto p(y_1, \cdots, y_N | x).$$

Dans le cadre de la modélisation statistique, l'inférence bayésienne permet de prendre en compte explicitement les connaissances sur l'incertitude liée aux paramètres du modèle. L'incertitude sur les paramètres est modélisée en termes de distribution *a priori*, ce qui reflète, par exemple, les connaissances d'un expert sur le problème sous-jacent. Voir Robert (2007) pour plus de détails sur le fondement de la théorie de la décision des statistiques bayésiennes.

Le paramètre x lui-même est considéré comme une variable aléatoire définie sur l'espace mesuré $(\mathcal{X}, \mathcal{B}(\mathcal{X}), \mathrm{d}\, x)$ muni d'une densité *a priori* p_0 par rapport à $\mathrm{d}\, x$.

L'incertitude sur le paramètre x après observation des données est quantifiée par la densité *a posteriori*, qui est obtenue en utilisant la formule de Bayes :

$$\pi(x|y_1, \cdots, y_N) = \frac{p_0(x)p(y_1, \cdots, y_N|x)}{\int_{\mathcal{X}} p_0(x)p(y_1, \cdots, y_N|x)\,\mathrm{d}\,x}. \tag{1.1}$$

Tant que $\int_{\mathcal{X}} p_0(x)p(y_1, \cdots, y_N|x)\,\mathrm{d}\,x < \infty$ la distribution *a posteriori* est bien définie. L'inférence dans le cadre bayésien est effectuée en calculant des quantités par rapport à la distribution *a posteriori*. Les moments *a posteriori*, comme la moyenne, sont d'un intérêt essentiel :

$$\mathbb{E}\left[X\right] = \int_{\mathcal{X}} x\pi(x|y_1, \cdots, y_N)\,\mathrm{d}\,x.$$

Une autre quantité d'intérêt est le *maximum a posteriori* donné comme suit :

$$x^\star = \arg\max_{x\in\mathcal{X}} \pi(x|y_1, \cdots, y_N).$$

Le test d'hypothèse est effectué en calculant la probabilité sous la distribution *a posteriori*

$$\mathbb{P}_\pi(x \in \mathcal{X}_i) = \int_{\mathcal{X}_i} x\pi(x|y_1, \cdots, y_N)\,\mathrm{d}\,x,$$

où $\mathcal{X}_i$ pour $i = 1, 2$ correspondent aux ensembles caractérisant différentes hypothèses. Pour le choix du modèle, la probabilité marginale, également appelée évidence, définie comme

$$Z_\pi = \int_{\mathcal{X}} p_0(x)p(y_1, \cdots, y_N|x)\,\mathrm{d}\,x$$

est intéressante car elle permet de comparer deux modèles différents.

L'inférence bayésienne dépend donc fortement de la capacité du statisticien à calculer des intégrales par rapport à la distribution *a posteriori*. Cependant, il s'agit d'un problème difficile et en dehors des modèles conjugués, la forme explicite de la densité *a posteriori* est souvent disponible seulement à un facteur près tel que

$$\pi(x|y_1, \cdots, y_N) \propto p_0(x)p(y_1, \cdots, y_N|x).$$

Par conséquent, deux approches majeures sont apparues en statistique : (a) les approches basées sur la caractérisation de la distribution *a posteriori* par échantillonnage et (b) les approches basées sur une approximation de la loi *a posteriori* à travers une famille de distributions tractables, potentiellement différentes de la vraie loi *a posteriori*. Afin de simplifier la notation, nous notons y nos données observées $y_1, \cdots, y_N$ dans le reste de ce chapitre.

Nous présentons les approches d'échantillonnage dans les sections 1.2 et 1.3. La section 1.4 revoit les idées d'approximation stochastique. L'approximation via des familles tractables grâce à l'inférence variationnelle est exposée dans la section 1.5. La section 1.6 donne un résumé substantiel de la contribution de cette thèse au domaine des statistiques computationnelles, basé sur les concepts introduits ci-dessus.

1.2 Échantillonnage Monte Carlo

Le but de l'échantillonnage Monte Carlo (Metropolis and Ulam, 1949) est le calcul d'intégrales sur l'espace $\mathcal{X}$ de la forme

$$\int_{\mathcal{X}} f(x)\,\mathrm{d}\,x,$$

où $f : \mathcal{X} \to \mathbb{R}$, par exemple. Il est souvent possible de réécrire l'intégrale en tant que $f(x) = \psi(x)\pi(x)$, où $\pi(x)$ est la fonction de densité d'une variable aléatoire X définie sur l'espace $\mathcal{X}$. Le problème peut donc être reécrit comme le calcul de l'intégrale de $\psi(X)$ par rapport à la distribution de probabilité $\mathbb{P}$ de densité π :

$$\int_{\mathcal{X}} f(x)\,\mathrm{d}\,x = \int_{\mathcal{X}} \psi(x)\pi(x)\,\mathrm{d}\,x = \mathbb{E}\left[\psi(X)\right].$$

1.2.1 Echantillonnage indépendant

Le principe de la méthode de Monte Carlo est de remplacer le calcul explicite de l'intégrale de la variable aléatoire $\psi(X)$ par une approximation basée sur la moyenne empirique d'une série indépendante de N réalisations. La loi faible des grands nombres garantit la convergence si l'espérance et la variance de $\psi(X)$ existent et sont finies :

$$\frac{1}{N}\sum_{n=1}^{N} \psi(x_n) \xrightarrow[N\to\infty]{\mathbb{P}} \mathbb{E}\left[\psi(X)\right].$$

L'erreur quadratique moyenne de l'approximation

$$\mathbb{E}\left[\left\|\frac{1}{N}\sum_{n=1}^{N} \psi(x_n) - \mathbb{E}\left[\psi(X)\right]\right\|_2^2\right] = \frac{\operatorname{tr}\operatorname{Var}\left[\psi(X)\right]}{N},$$

tend vers 0 à une vitesse de $1/N$, indépendamment de la dimension de l'intégrale. Cette approche repose sur la capacité d'échantillonnage i.i.d. tiré de la distribution sous-jacente $\mathbb{P}_X$ de la variable aléatoire X. De plus, un théorème central limite de la forme suivante peut être établie :

$$\sqrt{N}\left(\frac{1}{N}\sum_{n=1}^{N} \psi(X_n) - \mathbb{E}\left[\psi(X)\right]\right) \xrightarrow[N\to\infty]{\mathcal{L}} \mathcal{N}(0, \operatorname{Var}\left[\psi(X)\right]).$$

Des intervalles de confiance pour quantifier l'incertitude de l'approximation $\mathbb{E}\left[\psi(X)\right]$ par $1/N\sum_{n=1}^{N}\psi(X_n)$ peuvent être obtenus par ce raisonnement asymptotique.

Génération de nombres aléatoires

Du point de vue computationnel, la méthode Monte Carlo repose sur la génération efficace de nombres aléatoires sur un ordinateur. Les générateurs de nombres aléatoires produisent des nombres pseudo-aléatoires sur l'hypercube $[0,1]^d$, qui sont ensuite transformés sur un espace d'intérêt plus compliqué. Un traitement approfondi du sujet est donné dans Devroye (1986).

Une approche simple en une seule dimension est l'utilisation de la fonction de répartition (cdf) inverse. Si $U \sim \mathcal{U}[0,1]$, alors $F^{-1}(U) = X \sim \mathbb{P}_X$, où F^{-1} est la fonction de répartition inverse de la variable aléatoire X. Les échantillons générés via l'inverse cdf peuvent ensuite être utilisés comme point de départ pour générer d'autres distributions en utilisant par exemple l'échantillonnage acceptation-rejet. Nous allons maintenant discuter l'échantillonnage préférentiel plus en détail puisqu'une grande partie du reste de cette thèse repose sur ce concept.

Echantillonnage préférentiel

L'échantillonnage préférentiel (IS), étudié en détail par Geweke (1989), est basé sur l'identité d'échantillonnage préférentiel :

$$\mathbb{E}_\pi\left[\psi(X)\right] = \int_{\mathcal{X}} \psi(x)\pi(x)\,\mathrm{d}\,x = \int_{\mathcal{X}} \psi(x)\,\underbrace{\frac{\pi(x)}{g(x)}}_{=:w(x)}\,g(x)\,\mathrm{d}\,x = \mathbb{E}_g\left[\psi(X)w(X)\right].$$

Nous avons alors exprimé l'intégrale initiale comme une espérance par rapport à une densité différente g, appelée densité de proposition. Pour que cette expression soit correcte, π doit être absolument continu par rapport à g. Le terme $w(x)$ est appelé poids d'importance.

Cette identité fournit une approche directe pour calculer des espérances par rapport à une distribution selon laquelle nous ne pouvons pas échantillonner directement mais que nous pouvons évaluer point par point. Pour un échantillon $x_1, \cdots, x_N \sim g(x)$ la loi des grands nombres garantit que

$$\frac{1}{N}\sum_{n=1}^{N} \psi(x_n)w(x_n) \xrightarrow[N\to\infty]{a.s.} \mathbb{E}_g\left[\psi(X)w(X)\right] = \mathbb{E}_\pi\left[\psi(X)\right], \tag{1.2}$$

si $\mathbb{E}_g\left[|\psi(X)w(X)|\right] < \infty$. Dans la plupart des cas d'intérêt, la densité π n'est connue que proportionnellement à une constante de normalisation. Notons cette densité $\tilde{\pi}$, alors $\pi(x) = \tilde{\pi}(x)/Z$, où $Z = \int_{\mathcal{X}} \tilde{\pi}(x)\,\mathrm{d}\,x$. Dans ce cas, IS nous fournit un mécanisme pour estimer la constante de normalisation :

$$\frac{1}{N}\sum_{n=1}^{N} w(x_n) \xrightarrow[N\to\infty]{a.s.} \mathbb{E}_g\left[w(X)\right] = \int_{\mathcal{X}} \tilde{\pi}(x)\,\mathrm{d}\,x = Z. \tag{1.3}$$

Une estimation de $\mathbb{E}_\pi[\psi(X)]$ si la constante de normalisation n'est pas disponible est basée sur l'estimateur auto-normalisé d'échantillonnage préférentiel donné par :

$$\frac{\frac{1}{N}\sum_{n=1}^{N}\psi(x_n)w(x_n)}{\frac{1}{N}\sum_{n=1}^{N}w(x_n)} \xrightarrow[N\to\infty]{a.s.} \mathbb{E}_\pi[\psi(X)]\,. \tag{1.4}$$

L'estimateur IS auto-normalisé est en général biaisé, mais consistent. Un théorème central limite pour les estimateurs dans (1.2) et (1.4) peut être établi en utilisant par exemple la delta-méthode ou le lemme de Slutsky, si les variances asymptotiques correspondantes existent. Notez que ce n'est pas le cas, par exemple, si les queues de la loi de proposition sont plus légères que les queues de la densité cible.

La performance des approches d'échantillonnage préférentiel dépend de la proximité entre la distribution cible π et la distribution de proposition g (Chatterjee et al., 2018). En outre, IS a tendance à échouer si la dimension de l'espace cible $\mathcal{X}$ augmente. C'est pourquoi différentes stratégies d'adaptation ont été développées, voir par exemple Cappé et al. (2004); Cornuet et al. (2012).

Outre l'utilisation de IS pour l'évaluation des intégrales, il est intéressant de visualiser des distributions. Pour cela, l'ensemble de l'échantillon pondéré $\{x_n, w(x_n)\}_{n=1,\cdots,N}$ peut être ré-échantillonné en fonction des poids afin d'obtenir une approximation non pondérée de la distribution cible et peut ensuite être utilisée, par exemple, pour construire des histogrammes. Une approche largement utilisée est basée sur le ré-échantillonnage multinomial, où les observations sont ré-échantillonnées selon une distribution multinomiale basée sur leurs poids. Cette étape de ré-échantillonnage introduit de la variance. Par conséquent, si l'objectif est le calcul d'une espérance, l'approximation pondérée doit être préférée. D'autres approches telles que le ré-échantillonnage systématique et le ré-échantillonnage stratifié peuvent également être utilisées. Ces approches ont l'avantage d'introduire moins de variance que le ré-échantillonnage multinomial. Pour une analyse théorique récente de ces approches voir Gerber et al. (2017).

Échantillonnage pseudo-marginal

IS s'appuie sur la capacité d'évaluer la densité non normalisée $\tilde{\pi}(\cdot)$. Cependant, il y a nombreux problèmes d'intérêt, où $\tilde{\pi}(\cdot)$ ne peut pas être évalué point par point mais un estimateur sans biais $\widehat{\tilde{\pi}}(x)$ de $\tilde{\pi}(x)$ peut être calculé pour tout x. Une approche naturelle consiste à remplacer $\tilde{\pi}(x)$ par $\widehat{\tilde{\pi}}(x)$ et utiliser la même approche de calcul que si les poids corrects étaient disponibles. Cette idée a été introduite par Beaumont (2003) et analysé plus tard en détail par Andrieu and Roberts (2009). Cette situation se produit par exemple en présence d'une variable aléatoire latente Z avec densité p :

$$\pi(x|y) \propto p_0(x)\int_{\mathcal{Z}} h(x,y|z)p(z)\,\mathrm{d}z,$$

où la probabilité du modèle est une espérance par rapport à la variable aléatoire Z. En simulant à partir de p, nous pouvons approcher la vraisemblance par

$$p(y|x) = \mathbb{E}_p\left[h(x,y|z)\right] \approx \frac{1}{M}\sum_{j=1}^{M} h(x,y|z_j).$$

Par conséquent, la densité *a posteriori* est approximée par une quantité aléatoire définie comme

$$\widehat{\pi}(x|y) = p_0(x)\frac{1}{M}\sum_{j=1}^{M} h(x,y|z_j).$$

Cette quantité peut ensuite être utilisée pour générer des échantillons à partir de la distribution *a posteriori*, soit en utilisant l'échantillonnage préférentiel, soit en utilisant des approches Monte Carlo par chaîne de Markov (voir plus loin dans ce chapitre). Ce problème se pose, par exemple, dans les modèles de chaînes de Markov cachées (Cappé et al., 2005). Le travail de Andrieu et al. (2010) étudie cette approche dans le contexte du filtrage particulaire. Un autre domaine dans lequel cette idée présente un intérêt majeur est le calcul bayésien approximatif (ABC), que nous présentons maintenant.

Calcul bayésien approximatif

ABC est un concept couramment utilisé dans les modèles où la vraisemblance n'est pas tractable mais où la simulation à partir du modèle étant donné une valeur du paramètre est possible. Cette idée remonte au travail de Tavaré et al. (1997). Ce problème se pose par exemple dans certains sous-champs de biologie statistique telle que la philogénétique et l'épidémiologie.

Pour rendre cela plus concret, supposons que nous avons des données observées $y^\star$ et un modèle donné par sa vraisemblance $p(y|x)$ ainsi qu'une distribution *a priori* $p_0(x)$. Pour chaque valeur de x échantillonnée à partir de la loi *a priori*, il est possible de générer des pseudo données telles que $y|x \sim p(\cdot|x)$. Une façon naturelle de faire de l'inférence basée sur la simulation de paramètre est de comparer y et $y^\star$ selon une distance $\delta : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^+$ et de pondérer x selon cette distance. Après avoir simulé un certain nombre d'observations acceptées ou rejetées si la distance est inférieure à un seuil ϵ, nous simulons effectivement d'une distribution jointe telle que

$$x, y \sim \pi_\epsilon(x,y|y^\star) \propto p_0(x)p(y|x)\mathbb{1}_{\{\delta(y,y^\star)\leq\epsilon\}}.$$

Après avoir marginalisé y, nous obtenons la loi *a posteriori* approximative

$$\pi_\epsilon(x|y^\star) \propto p_0(x)\mathbb{P}_x(\delta(y,y^\star) \leq \epsilon). \tag{1.5}$$

Ainsi, la simulation selon la loi *a posteriori* ABC peut être vue comme une pondération des simulations selon la probabilité que les pseudo-observations associées tombent dans une boule de rayon ϵ autour des vraies observations. Pour la simulation pratique

la quantité $\mathbb{P}_x(\delta(y, y^\star) \leq \epsilon)$ est approximée par le poids aléatoire $\mathbb{1}_{\{\delta(y,y^\star)\leq\epsilon\}}$. Ce poids aléatoire est un estimateur non-biaisé et positif car

$$\mathbb{E}_p\left[\mathbb{1}_{\{\delta(y,y^\star)\leq\epsilon\}}\right] = \int_{\mathcal{Y}} \mathbb{1}_{\{\delta(y,y^\star)\leq\epsilon\}} p(y|x)\,\mathrm{d}\,y = \mathbb{P}_x(\delta(y, y^\star) \leq \epsilon).$$

Cela relie ABC à l'idée d'échantillonnage pseudo-marginal, introduite précédemment.

Dans la plupart des cas, la distance entre les données observées et les pseudo données est exprimée par $\delta(y, y^\star) = \|s(y) - s(y^\star)\|_2$ où $s(\cdot)$ est une statistique de résumé qui extrait les caractéristiques les plus importantes des données. Si la statistique de résumé est suffisante et $\epsilon \to 0$, nous récupérons en fait la vraie $\pi(x|y^\star)$.

Dans sa forme la plus basique, l'échantillonnage ABC procède par échantillonnage d'acceptation-rejet. Cette approche peut être très inefficace car la génération d'échantillons à partir du modèle $p(\cdot|x)$ est souvent coûteuse en calcul. De plus, si la valeur de x est loin de la région de densité *a posteriori* élevée, l'échantillon généré de y conduira probablement à un rejet. Ce problème peut être résolu avec l'échantillonnage préférentiel. Nous essayons de trouver une distribution de proposition $g(x)$ proche de la loi *a posteriori* approximative dans (1.5). Nous limitons ainsi les simulations aux régions de densité *a posteriori* élevée. Des approches itératives ont été développées, qui sont séquentielles et qui essayent de concentrer le calcul dans ces régions de probabilité *a posteriori* élevée (Sisson et al., 2009; Del Moral et al., 2012).

La construction de statistiques de résumé (Fearnhead and Prangle, 2012) ou autres mesures de distance (Bernton et al., 2017), ainsi que les propriétés asymptotiques (Frazier et al., 2018) ont été un domaine de recherche actif au cours des dernières années.

Récemment, l'idée de construire des modèles de substitution, qui contournent l'échantillonnage du modèle a attiré l'attention (Wilkinson, 2014). Une autre approche consiste à contourner le choix d'un seuil ϵ, voir par exemple Papamakarios and Murray (2016); Price et al. (2018). D'autres développements consistent à utiliser l'optimisation bayésienne (Gutmann and Corander, 2016) et des forêts aléatoires (Pudlo et al., 2016).

1.2.2 Échantillonnage dépendant

Les approches d'échantillonnage introduites reposent sur la capacité de générer des échantillons indépendants qui sont pondérés afin de se rapprocher de la distribution cible. Une autre classe d'algorithmes d'échantillonnage génériques est basée sur la génération d'échantillons *dépendants* tels que la distribution marginale d'un processus stochastique converge vers la distribution souhaitée. Cette idée remonte au travail de Metropolis et al. (1953) et s'applique également aux distributions cibles non normalisées.

Monte Carlo par chaîne de Markov

Les méthodes de Monte Carlo par chaîne de Markov (MCMC) reposent sur la construction d'une chaîne de Markov $(X_n)_{n\geq 1}$ tel que

$$\frac{1}{N}\sum_{n=1}^{N}\psi(x_n) \xrightarrow[N\to\infty]{a.s.} \mathbb{E}_\pi\left[\psi(X)\right],$$

pour presque tous les points de départ x_1. La chaîne est caractérisée par une distribution initiale et un noyau de transition $\mathcal{K}$. Afin d'obtenir une convergence vers la distribution souhaitée, le noyau de transition $\mathcal{K}$ doit laisser la distribution π invariante :

$$\forall x \in \mathcal{X} \int_{\mathcal{Y}} \mathcal{K}(y,x)\pi(y)\,\mathrm{d}\,y = \pi(x).$$

Cela signifie que si $y \sim \pi(y)$, la distribution marginale reste la même après l'application du noyau $\mathcal{K}$ à y.

Si la chaîne est apériodique et irréductible, la convergence $\left\|\mathcal{K}^N(x_1,\cdot) - \pi(\cdot)\right\|_{TV} \to 0$ peut être établie quand les itérations du noyau $N \to \infty$, où $\|\cdot\|_{TV}$ est la norme de variation totale. Par conséquent, pour tout point de départ x_1, la chaîne converge vers sa distribution stationnaire.

Sous l'hypothèse supplémentaire de récurrence de Harris sur la chaîne générée l'ergodicité géométrique peut être montrée. Cela signifie que

$$\left\|\mathcal{K}^n(x_1,\cdot) - \pi(\cdot)\right\|_{TV} \leq M(x_1)\rho^n,$$

où $\rho < 1$ est une constante et $M(x_1) > 0$ dépend du point de départ x_1. Si de plus, ψ est intégrable, un théorème central limite est obtenu :

$$\sqrt{N}\left(\frac{1}{N}\sum_{n=1}^{N}\psi(x_n) - \mathbb{E}_\pi\left[\psi(X)\right]\right) \xrightarrow[N\to\infty]{\mathcal{L}} \mathcal{N}(0,\sigma_\psi^2),$$

où

$$\sigma_\psi^2 = \mathrm{Var}\left[\psi(X_1)\right] + 2\sum_{n=2}^{\infty}\mathrm{Cov}[\psi(X_1),\psi(X_n)].$$

Voir Tierney (1994) pour plus d'informations sur les propriétés théoriques. La série de covariances $\mathrm{Cov}[\psi(X_1),\psi(X_n)]$ peut être liée à l'autocorrélation de la chaîne. Plus l'autocorrélation de la chaîne diminue vite, plus la variance asymptotique d'un estimateur basé sur les réalisations de la chaîne sera petite.

L'algorithme Metropolis-Hastings (Metropolis et al., 1953; Hastings, 1970) est un algorithme à usage général pour la construction d'une chaîne de Markov avec les propriétés souhaitées. Nous introduisons maintenant trois algorithmes différents basés sur l'algorithme Metropolis-Hasting : l'algorithme de marche aléatoire Metropolis-Hasting (RWMH), l'algorithme de Langevin ajusté par Metropolis (MALA) et algorithme de Monte Carlo hamiltonien (HMC). Une autre approche pour construire des

chaînes de Markov avec la distribution cible invariante souhaitée est, par exemple, l'algorithme de Gibbs (Geman and Geman, 1984).

Marche aléatoire Metropolis-Hasting

Une manière simple de construire un noyau invariant par rapport à la distribution π est d'utiliser une marche aléatoire locale. L'algorithme résultant est donné dans l'Algorithme 1 et peut être compris comme suit. A partir de x_{s-1}, un nouvel état de la chaîne $x^\star$ est proposé, où l'état actuel x_{s-1} est perturbé par un bruit gaussien. Si le nouvel état déplace la chaîne dans des régions de densité plus élevée, le nouvel état est toujours accepté. Si le nouvel état déplace la chaîne dans les régions de densité plus faible, le nouvel état est accepté avec une probabilité proportionnelle au rapport de vraisemblance de l'état suggéré à l'état précédent. Ainsi, les mouvements qui explorent la distribution sont acceptés occasionnellement. Le noyau de proposition a essentiellement deux paramètres à choisir : le paramètre d'échelle, appelé σ^2 ici, et le paramètre de covariance, Σ. Le paramètre d'échelle σ^2 peut être absorbé dans Σ.

Algorithm 1: Algorithme de marche aléatoire Metropolis-Hasting

Input: Point de départ x_0, densité cible π

Result: Ensemble $(x_s)_{s\in 1:S}$

for $s = 1$ *to* S **do**
- Simuler $x^\star \sim \mathcal{N}(x^\star | x_{s-1}, \sigma^2\Sigma)$
- Calculer $r(x^\star, x_{s-1})) = \frac{\pi(x^\star)}{\pi(x_{s-1})}$
- Simuler $u \sim \mathcal{U}[0,1]$
- **if** $u \leq r(x^\star, x_{s-1})$ **then**
  - Poser $x_s = x^\star$
- **else**
  - Poser $x_s = x_{s-1}$

L'algorithme RWMH repose seulement sur l'hypothèse qu'il est possible d'évaluer la densité cible non-normalisée point par point. Si en plus la densité cible est différentiable, cette information peut être utilisée pour guider la chaîne vers des régions de densité plus élevée.

Algorithme de Langevin ajusté par Metropolis

Un algorithme basé sur cette approche est l'algorithme de Langevin ajusté par Metropolis (MALA), qui est basé sur la discrétisation en temps fini de la diffusion de Langevin. Voir Roberts and Tweedie (1996) pour plus de détails. La diffusion de Langevin est définie comme une équation différentielle stochastique en temps continu t

$$\mathrm{d}\, X_t = \frac{\sigma^2}{2}\nabla \log\{\pi(X_t)\}\,\mathrm{d}\, t + \sigma\,\mathrm{d}\, B_t,$$

où B_t est un mouvement brownien multivarié. La distribution stationnaire de ce processus est π. Pour la simulation, la diffusion est discrétisée et une étape Metropolis-Hastings est introduite afin de corriger l'erreur de discrétisation. Cette approche donne une chaîne de Markov avec la distribution invariante souhaitée. Le noyau de proposition résultant n'est pas symétrique. Lors de l'utilisation d'une matrice dite de préconditionnement M, les paramètres de réglage de l'algorithme sont la matrice M et le paramètre de variance σ^2. L'algorithme se lit comme suit.

Algorithm 2: Algorithme de Langevin ajusté par Metropolis

Input: Point de départ x_0, densité π, gradient de la log densité $\nabla_x \log \pi(\cdot)$

Result: Ensemble $(x_s)_{s\in 1:S}$

for $s = 1$ *to* S **do**
- Simuler $x^\star \sim \mathcal{N}(x^\star | x_{s-1} + \sigma^2/2M\nabla_x \log \pi(x_{s-1}), \sigma^2 M) =: g(x^\star | x_{s-1})$
- Calculer $r(x^\star, x_{s-1})) = \frac{\pi(x^\star) g(x_{s-1}|x^\star)}{\pi(x_{s-1}) g(x^\star|x_{s-1})}$
- Simuler $u \sim \mathcal{U}[0,1]$
- **if** $u \leq r(x^\star, x_{s-1})$ **then**
  - Poser $x_s = x^\star$
- **else**
  - Poser $x_s = x_{s-1}$

Monte Carlo hamiltonien

S'appuyant sur l'idée d'utiliser l'information du gradient, l'algorithme de Monte Carlo hamiltonien (HMC) (Duane et al., 1987) a attiré beaucoup d'attention dans la dernière décennie. Voir Neal (2011) pour une introduction. HMC peut être compris comme une procédure d'échantillonnage sur la distribution cible étendue $\mu(x,z) = \pi(x) \times g(z)$, où $g(z) = \mathcal{N}(z|0_d, M)$ est une distribution gaussienne multivariée avec moyenne 0 et matrice de covariance M. L'échantillonnage est réalisé en empruntant un concept de la mécanique physique : l'état d'une particule avec la position x et le moment z qui est déterminé par le hamiltonien

$$H(x,z) = -\log \mu(x,z)$$

représente des réalisations de la distribution jointe $\mu(x,z)$. Le mouvement de la particule est solution des équations de Hamilton

$$\begin{cases} \frac{dx}{d\tau} = \frac{\partial H}{\partial z} = M^{-1}z, \\ \frac{dz}{d\tau} = -\frac{\partial H}{\partial x} = \nabla_x \log \pi(x), \end{cases}$$

où les dérivées sont prises par rapport au temps fictif τ. La solution de ces équations différentielles induit un flux $\Phi_\kappa : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^d \times \mathbb{R}^d$, qui décrit l'évolution du système en fonction du temps κ. A partir de (x_0, z_0) le système évolue sur une période de temps κ menant à un nouvel état $\Phi_\kappa(x_0, z_0) = (x_\kappa, z_\kappa)$. En pratique,

la solution est approximée en utilisant une procédure d'intégration numérique, appelée l'intégrateur leapfrog. Celui-ci dépend d'une discrétisation ϵ et du nombre de pas L. Par conséquent, le temps d'intégration devient $\kappa = \epsilon \times L$ avec le flux numérique $\widehat{\Phi}_{\epsilon,L} : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^d \times \mathbb{R}^d$. L'intégration numérique introduit une erreur dans l'énergie totale du système. La conservation de l'énergie du système est violée : $\Delta E \neq 0$, où $\Delta E = H(\widehat{\Phi}_{\epsilon,L}(x_0, z_0)) - H(x_0, z_0)$. Une étape Metropolis-Hastings est utilisée pour corriger cette erreur, et potentiellement un état proposé est rejeté. La valeur de z est rafraîchie en échantillonnant $z \sim \mathcal{N}(z|0_d, M)$ et différents niveaux d'énergie sont explorés. Une chaîne de Markov avec indice de temps s et distribution invariante $\mu(x, z)$ est construite en itérant les étapes suivantes :

1. Rafraîchir $z_s \sim \mathcal{N}(z|0_d, M)$
2. Générer $(\hat{x}, \hat{z}) = \widehat{\Phi}_{\epsilon,L}(x_s, z_s)$ sur la base des solutions numériques des équations des mouvements à partir de (x_s, z_s).
3. Accepter $(\hat{x}, \hat{z})$ comme nouvel état (x_{s+1}, z_{s+1}) selon une étape Metropolis Hasting basée sur ΔE_s.

Pour plus de détails sur l'algorithme, voir le chapitre 5. L'algorithme résultant, avec des paramètres ϵ, L, M bien choisis, donne généralement une chaîne de Markov avec une faible autocorrélation et un déplacement plus éloigné dans l'espace cible. Cependant, le réglage de l'algorithme est une tâche difficile et constitue un domaine de recherche actif (Wang et al., 2013; Hoffman and Gelman, 2014; Levy et al., 2018).

Réglage et développements récents

Tous les algorithmes MCMC introduits dépendent d'un certain nombre de paramètres de réglage, qui sont cruciaux pour leur performance. En particulier, un compromis doit être trouvé entre l'exploration de l'espace cible, caractérisé par la distance que la chaîne peut traverser en une seule étape, et le taux d'acceptation de la chaîne. Par conséquent, des stratégies diverses d'adaptation ont émergé qui essayent d'améliorer la mélangeance de la chaîne en équilibrant les deux.

Concernant l'algorithme RWMH, le travail de Roberts et al. (1997) a montré que le réglage du noyau de proposition de telle sorte qu'une acceptation avec probabilité de 0.234 est atteinte, est optimal quand la dimension de l'espace cible tend vers l'infini. Un résultat similaire a été obtenu pour MALA : le noyau doit être réglé de telle sorte qu'une probabilité d'acceptation de 0.574 est atteinte (Roberts and Rosenthal, 1998). Ces résultats ont été étendus dans Roberts and Rosenthal (2001). Pour HMC, un résultat similaire a été établi par Beskos et al. (2013) : l'utilisateur doit viser un taux d'acceptation de 0.651 lors de la sélection de la discrétisation ϵ. Comme le taux d'acceptation peut être calculé uniquement *ex post*, une série d'études explore un réglage adaptatif des paramètres pendant que la chaîne explore l'espace cible. Cette idée a été initiée par Atchadé and Rosenthal (2005) et largement étudié depuis. Au

lieu de viser un certain taux d'acceptation, il a été suggéré d'utiliser la distance de saut au carré attendue (ESJD) comme critère pour la mélangeance de la chaîne. Maximiser l'ESJD équivaut à minimiser l'autocorrélation de premier ordre de la chaîne. Voir Pasarica and Gelman (2010); Wang et al. (2013) pour deux applications de cette approche.

Un axe de recherche récent consiste à utiliser des chaînes de Markov déterministes par morceaux (PDMC). En construisant une chaîne de Markov à temps continu non réversible, une mélangeance empirique plus rapide de la chaîne peut être obtenu. Les articles de Bierkens et al. (2016); Vanetti et al. (2017); Sherlock and Thiery (2017); Bouchard-Côté et al. (2018) sont à la pointe de ce développement.

Outre le réglage et l'extensibilité avec la dimension des algorithmes MCMC, la question de la parallélisation est d'actualité. Une voie prometteuse consiste à supprimer le biais de burn-in des chaînes MCMC et la simulation simultanée de plusieurs chaînes courtes (Jacob et al., 2017; Heng and Jacob, 2017). Un problème différent d'intérêt en pratique est l'abondance des individus observés, qui rendent la vraisemblance coûteuse à évaluer. Des approches basées sur le sous-échantillonnage des observations ont été introduits par Welling and Teh (2011) pour les algorithmes de type Langevin, par Chen et al. (2014) pour les algorithmes HMC et Pakman et al. (2017) pour une application à PDMC. Voir Alquier et al. (2016a) ou Bardenet et al. (2017) pour des analyses théoriques récentes.

Monte Carlo séquentiel

Une autre approche pour simuler selon d'une distribution cible π consiste dans une seule approache qui combine les idées d'échantillonnage préférentiel et des noyaux de Markov invariant. Par conséquent, une séquence de distributions est construite, qui est approximée de manière itérative. Cette idée a son origine dans le filtrage particulaire (Pitt and Shephard, 1999) et a été introduite au calcul bayésien par Chopin (2002). Pour l'inférence bayésienne, par exemple, on procède en simulant un certain nombre N de points selon la loi *a priori*. Ces points, appelés *particules*, sont ensuite déplacés en passant par une séquence des distributions intermédiaires allant vers la loi *a posteriori*. Ceci est réalisé via une séquence d'étape de repondération, de ré-échantillonnage et de mutation. Une analyse théorique approfondie du filtrage particulaire basée sur le formalisme de Feynman-Kac est donnée dans Del Moral (2004).

Plus précisément, nous sommes intéressés par une séquence de distributions π_t, où l'utilisation de l'échantillonnage préférentiel itérative donne lieu à l'échantillonnage Monte Carlo séquentiel, voir Del Moral et al. (2006).

La simulation de la distribution intermédiaire est réalisée par la simulation selon la distribution jointe $\pi_{0:t}(x_{0:t}) = \tilde{\pi}_{0:t}(x_{0:t})/Z_{0:t}$ défini sur $\mathbb{R}^d \times \cdots \times \mathbb{R}^d$. L'introduction de noyaux *backward* artificiels $\mathcal{L}_s : \mathbb{R}^d \times \mathcal{B}(\mathbb{R}^d) \to [0,1]$ avec densité $\mathcal{L}_s(x_{s+1}, x_s)$ donne

$$\tilde{\pi}_{0:t}(x_{0:t}) = \tilde{\pi}_t(x_t) \prod_{s=1}^{t-1} \mathcal{L}_s(x_{s+1}, x_s),$$

qui admet $\pi_t(x_t)$ par construction. La distribution de proposition est donnée par la séquence des noyaux *forward* $\mathcal{K}_s : \mathbb{R}^d \times \mathcal{B}(\mathbb{R}^d) \to [0,1]$ avec densité $\mathcal{K}_s(x_{s-1}, x_s)$:

$$\tilde{g}_{0:t}(x_{0:t}) = \tilde{g}_0(x_0) \prod_{s=2}^{t} \mathcal{K}_s(x_{s-1}, x_s).$$

Par conséquent, le poids d'importance pour l'échantillonnage sur l'espace commun est

$$\omega_t(x_{0:t}) \quad = \quad \frac{\tilde{\pi}_{0:t}(x_{0:t})}{\tilde{g}_{0:t}(x_{0:t})}. \tag{1.6}$$

La dimension de l'espace cible est étendue de manière itérative de telle sorte que les poids sont donnés récursivement par

$$\omega_t(x_{0:t}) \quad = \quad \omega_{t-1}(x_{0:t-1})\tilde{\omega}_t(x_{t-1}, x_t), \tag{1.7}$$

où le poids incrémental est

$$\tilde{\omega}_t(x_{t-1}, x_t) = \frac{\tilde{\pi}_t(x_t)\mathcal{L}_t(x_t, x_{t-1})}{\tilde{\pi}_{t-1}(x_{t-1})\mathcal{K}_t(x_{t-1}, x_t)}. \tag{1.8}$$

Donc, si au temps $t-1$ une approximation de particule $\left\{x^i_{0:t-1}, w^i_{0:t-1}\right\}_{i\in 1:N}$ de $\tilde{\pi}_{t-1}(x_{0:t-1})$ est disponible, où $w^i_{0:t-1}$ correspond à la version normalisée de $\omega_{t-1}(x^i_{0:t-1})$, son chemin est étendu via le noyau de proposition $\mathcal{K}_t$ et le poids incrémental dans (1.8). Cette construction permet l'estimation non-biaisée du ratio des constantes de normalisation Z_t/Z_{t-1} via

$$\frac{\widehat{Z}_t}{Z_{t-1}} = \sum_{i=1}^{N} w^i_{0:t-1}\tilde{\omega}_t(x^i_{t-1}, x^i_t).$$

Si les particules ont été ré-échantillonnées au temps $t-1$, cette expression simplifie et nous obtenons

$$\frac{\widehat{Z}_t}{Z_{t-1}} = 1/N \sum_{i=1}^{N} \tilde{\omega}_t(\tilde{x}^i_{t-1}, x^i_t),$$

où $\left\{\tilde{x}^i_{t-1}\right\}_{i\in 1:N}$ est l'ensemble des particules ré-échantillonnées. Le ré-échantillonnage est utilisé dans le filtrage particulaire pour éliminer les particules qui ont un poids négligeable. Dans la plupart des applications pratiques, la première distribution π_0 est choisie de telle manière qu'il soit facile d'échantillonner selon cette distribution et par conséquent, nous posons $\tilde{\pi}_0 = \tilde{g}_0$. Une version générique de l'algorithme SMC

est donnée dans l'Algorithme 3.

Algorithm 3: Algorithme SMC génerique

Input: Séquence de distribution non-normalisée $\tilde{\pi}_0, \cdots, \tilde{\pi}_t, \cdots, \tilde{\pi}_T$ et noyau de propagation $\mathcal{K}_t$

Result: Ensemble $\left\{x_t^i, w_t^i\right\}_{i\in 1:N}$ et estimations $\widehat{\frac{Z_t}{Z_{t-1}}}$ pour $t \in 1:T$

Initialization : $t = 1$

for $i = 1$ *to* N **do**
- Simuler $x_0^i \sim \tilde{\pi}_0$
- Pondérer $w_0^i \propto \frac{\tilde{\pi}_1(x_0^i)}{\tilde{\pi}_0(x_0^i)}$

Calculer $\widehat{\frac{Z_1}{Z_0}} = 1/N \sum_{i=1}^N \frac{\tilde{\pi}_1(x_0^i)}{\tilde{\pi}_0(x_0^i)}$

Iteration :

for $t = 2$ *to* T **do**
- Ré-échantillonner $\left\{x_{t-1}^i, w_{t-1}^i\right\}_{i\in 1:N}$ et obtenir $\left\{\tilde{x}_{t-1}^i\right\}_{i\in 1:N}$;
- **for** $i = 1$ *to* N **do**
  - Déplacer $x_t^i \sim \mathcal{K}_t(\tilde{x}_{t-1}^i, dx)$
- **for** $i = 1$ *to* N **do**
  - Pondérer $w_t^i \propto \tilde{\omega}_t(x_{t-1}^i, x_t^i)$
- Calculer $\widehat{\frac{Z_t}{Z_{t-1}}} = 1/N \sum_{i=1}^N \omega_t(x_t^i)$

L'utilisation d'échantillonnage SMC pour le calcul bayésien présente plusieurs propriétés intéressantes par rapport à l'échantillonnage basé sur MCMC : (a) la constante de normalisation du modèle est estimée et donc disponible pour le choix du modèle (Zhou et al., 2016). (b) L'utilisation d'un grand nombre de particules apporte une certaine robustesse à la multimodalité (Schweizer, 2012a). (c) L'échantillonnage SMC est hautement parallélisable (Murray et al., 2016). (d) Le nuage de particules fournit des informations utiles pour adapter l'algorithme (Fearnhead and Taylor, 2013). Les développements récents dans le domaine de SMC consistent en l'étude des propriétés théoriques d'adaptation (Beskos et al., 2016), l'estimation de la variance de l'algorithme (Lee and Whiteley, 2018) ou l'utilisation de quasi-Monte Carlo (Gerber and Chopin, 2015), voir aussi la section suivante.

1.3 Quasi-Monte Carlo

Comme nous l'avons vu, la plupart des générateurs de nombres aléatoires commencent par générer une séquence uniforme. La séquence uniforme peut être utilisée pour calculer une approximation de l'intégrale de la fonction $\phi : [0,1]^d \to \mathbb{R}$ définie comme $I = \int_{[0,1]^d} \phi(u)\,\mathrm{d}\,u$ via

$$\widehat{I}_N = \frac{1}{N}\sum_{n=1}^{N} \phi(u_n), \tag{1.9}$$

où la fonction ϕ englobe potentiellement une transformation de la séquence uniforme vers un espace d'intérêt différent.

Une approche couramment utilisée pour réduire la variance de l'intégration est la stratification. La stratification divise un hypercube uniforme en un nombre de *strata* et procède ensuite en échantillonnant au sein de chaque *strata*. Cette approche couvre l'hypercube uniforme de manière plus uniforme et par conséquent conduit à une erreur réduite de l'intégration.

1.3.1 Séquences Halton

Une approche plus sophistiquée consiste à construire des séquences déterministes, également appelées séquences à discrépance faible ou quasi-Monte Carlo. Nous illustrons cette approche avec la construction de séquences de Halton en suivant Dick et al. (2013). Soit $i \in \mathbb{N}$. Alors i peut être exprimé en base b comme

$$i = \sum_{a=1}^{\infty} i_a b^{a-1},$$

où $i_a \in \{0, 1, \cdots, b-1\}$. A titre d'exemple, nous représentons la séquence d'entiers $0, 1, 2, 3, 4, \cdots$ en base $b = 2$. Cela donne $0_2, 1_2, 10_2, 11_2, 100_2, \cdots$. Nous définissons la fonction inverse radicale $\nu_b(i)$ comme l'inversion de la représentation entière de i en base b. Elle est définie comme

$$\nu_b(i) := \sum_{a=1}^{\infty} \frac{i_a}{b^a}.$$

La fonction inverse radicale reflète cette représentation à la représentation décimale : $0, 0.1_2, 0.01_2, 0.11_2, 0.001_2, \cdots$. Si nous transformons cette séquence en base de représentation 10 nous obtenons $0, 0.5, 0.25, 0.75, 0.125 \cdots$. Continuer cette construction donne une séquence qui remplit l'intervalle $[0, 1]$. La séquence de Halton est basée sur cette idée. Soit $p_1, p_2, \cdots, p_d$ les d premiers nombres premiers. La séquence de Halton $u_0, u_1, \cdots$ en dimension d est donnés comme

$$u_i = (\nu_{p_1}(i), \nu_{p_2}(i), \cdots, \nu_{p_d}(i)).$$

Nous illustrons la séquence de Halton ainsi qu'une séquence pseudo-aléatoire sur $[0, 1]^2$ dans la Figure 1.1.

La séquence de Halton n'est qu'un moyen possible de construire des séquences qui couvrent $[0, 1]^d$ plus uniformément que l'échantillonnage aléatoire. D'autre séquences qui atteignent le même objectif sont, par exemple, la séquence Faure, la séquence Sobol ou les réseaux digitaux. La qualité de la couverture de la séquence déterministe peut être évaluée par la discrépance de la séquence, que nous discutons maintenant.

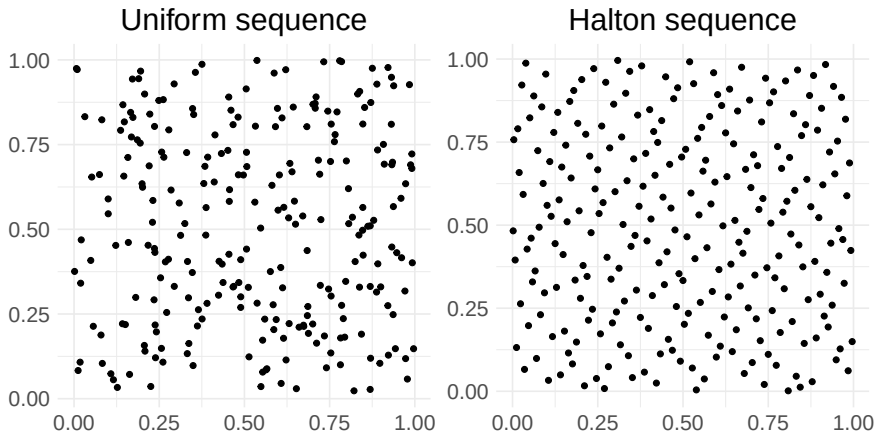


FIGURE 1.1: Séquence uniforme (à gauche) et séquence de Halton (à droite) de longueur $N = 256$ sur $[0,1]^2$.

1.3.2 Convergence de l'échantillonnage QMC

La notion générale de discrépance d'une séquence $u_1, \cdots, u_N$ est définie comme suit :

$$D(u_{1:N}, \mathcal{A}) := \sup_{A \in \mathcal{A}} \left| \frac{1}{N} \sum_{n=1}^{N} \mathbb{1}_{\{u_n \in A\}} - \lambda_d(A) \right|,$$

où $\lambda_d(A)$ est le volume (la mesure de Lebesgue sur $\mathbb{R}^d$) de A et $\mathcal{A}$ est un ensemble d'ensembles mesurables. Lorsque nous fixons les ensembles $A = [0, \mathbf{b}] = \prod_{i=1}^{d} [0, b_i]$ avec $0 \leq b_i \leq 1$ comme l'ensemble des produits d'intervalles ancrés en 0, nous obtenons la discrépance étoile

$$D^*(u_{1:N}) := \sup_{[0,\mathbf{b}]} \left| \frac{1}{N} \sum_{n=1}^{N} \mathbb{1}_{\{u_n \in [0,\mathbf{b}]\}} - \lambda_d([0, \mathbf{b}]) \right|.$$

La discrépance étoile peut être utilisée afin d'établir une limite supérieure de l'erreur d'approximation de l'intégrale. Nous définissons cette erreur comme

$$e(\phi, N) = \left| I - \widehat{I}_N \right|.$$

Nous avons vu dans la section 1.2.1 que pour l'utilisation d'une séquence aléatoire uniforme on a

$$\sqrt{\mathbb{E}\left[e(\psi, N)^2\right]} = \mathcal{O}(N^{-1/2}),$$

en supposant que la variable aléatoire $\phi(U)$ est de carré intégrable. Lors de l'utilisation d'une séquence à discrépance faible, l'erreur peut être quantifiée par l'inégalité Koksma-Hlawka :

$$e(\psi, N) \leq D^*(u_{1:N}) V(\phi), \tag{1.10}$$

où $V(\phi)$ est la variation de la fonction ϕ dans le sens de Hardy et Krause. Cette quantité est étroitement liée à la regularité de la fonction. La séquence de Halton présentée conduit à une discrépance en $\mathcal{O}((\log N)^d/N)$. Par conséquent, les séquences de Halton sont asymptotiquement plus efficaces que les séquences aléatoires pour l'intégration, à condition que $V(\phi) < \infty$.

1.3.3 Quasi-Monte Carlo randomisé

Un inconvénient majeur de l'échantillonnage QMC est le fait qu'il n'est pas facile d'évaluer l'erreur de l'approximation car $V(\phi)$ est difficile à calculer. En réintroduisant de l'aléa dans une séquence QMC tout en préservant sa structure de discrépance faible, l'incertitude peut être évaluée par échantillonnage répété. Cette idée est appelée quasi-Monte Carlo randomisé (RQMC). Ces séquences peuvent être construites, par exemple, en décalant au hasard la séquence entière (Cranley and Patterson, 1976) : Soit $v \sim \mathcal{U}[0,1]^d$ et soit $(u_n)_{1\leq n\leq N}$ une séquence QMC. Ensuite, la séquence

$$\tilde{u}_n := u_n + v \mod 1,$$

où $x \mapsto x \mod 1$ est la fonction modulo composants par composants, est une séquence QMC avec une probabilité de 1. De plus, la séquence est distribuée marginalement uniforme. Ainsi, lors de l'utilisation des points $\tilde{u}_n$ pour l'intégration, les estimations de (1.9) sont sans biais et nous pouvons utiliser notre boîte à outils probabilistes pour évaluer l'erreur.

Une approche plus complexe consiste en des réseaux brouillés, introduits par Owen (1997). En introduisant le hasard directement dans la construction de la séquence, il est possible d'obtenir des taux d'erreur en $\mathcal{O}((\log N)^{d-1}N^{-3})$. Ce résultat repose sur des hypothèses de régularité supplémentaires de la fonction ϕ.

Un résultat plus récent de Gerber (2015) obtient un taux en $\mathcal{O}(N^{-2})$ sous les mêmes les hypothèses de regularité comme Owen (1997). En assouplissant ces hypothèses, des taux en $o(N^{-1})$ sont réalisables. Ainsi, l'intégration RQMC est toujours au moins aussi efficace que l'intégration par le Monte Carlo classique.

1.3.4 Utilisation de séquences à discrépance faible en statistique

L'utilisation de QMC en statistique repose sur la capacité du statisticien à transformer une séquence sur $[0,1]^d$ vers la distribution d'intérêt tout en préservant la structure de discrépance faible des points initiaux. Cela peut être réalisé en réécrivant l'intégrale d'intérêt comme une espérance par rapport à la distribution uniforme :

$$\mathbb{E}_X\left[\psi(X)\right] = \mathbb{E}_U\left[\psi(\Gamma(U))\right].$$

Donc $\psi \circ \Gamma = \phi$, en utilisant notre fonction de test initiale, et il faut s'assurer que cette fonction satisfait les propriétés de regularité spécifiées ci-dessus. Une approche

générique est l'utilisation de la transformation inverse de Rosenblatt (Rosenblatt, 1952), qui généralise la cdf inverse au cas multivarié.

Jusqu'à aujourd'hui, l'utilisation de QMC a été largement étudiée dans les mathématiques financières (Lemieux and L'Ecuyer, 2001; L'Ecuyer, 2009; Glasserman, 2013), mais son utilisation dans les statistiques classiques reste plutôt limitée. En fait, QMC peut être utilisé en combinaison avec l'échantillonnage préférentiel. Cette approche peut conduire à des estimateurs avec une variance réduite, voir par exemple Gerber and Chopin (2015); Chopin and Ridgway (2017). Les recherches actuelles portent sur les applications QMC pour les méthodes MCMC, voir par exemple Owen and Tribble (2005); L'Ecuyer et al. (2008); L'Ecuyer and Sanvido (2010); Chen et al. (2011); Schwedes and Calderhead (2018).

1.3.5 Théorème centrale limite pour QMC

Dans diverses applications pratiques, il peut être impossible d'utiliser une séquence QMC pour toutes les dimensions du problème d'intérêt. Cela peut être dû au fait que la dimension du problème est trop grande, ou que pour certaines parties du problème, il n'est pas possible de transformer la séquence QMC vers l'espace d'intérêt. Dans un tel contexte, nous pouvons utiliser une séquence mixte composée en partie d'une séquence MC de dimension s, notée v_n, et en partie d'une séquence QMC de dimension $d-s$, notée u_n. La séquence jointe $(v_n, u_n) = r_n$ est appelée séquence mixte. La séquence mixte conduit à l'estimateur $\widehat{I}_N = 1/N \sum_{n=1}^{N} \phi(r_n)$.

Le taux de convergence d'une intégration basée sur une séquence mixte sera naturellement dominé par la partie la plus lente, à savoir la partie MC. Dans ce contexte, il est possible d'établir une réduction asymptotique de la variance basée sur un théorème centrale limite. En particulier pour ϕ borné et quelques hypothèses supplémentaires Ökten et al. (2006) obtiennent que

$$\sqrt{N}(\widehat{I}_N - I) \xrightarrow[N\to\infty]{\mathcal{L}} \mathcal{N}(0, \tilde{\sigma}^2),$$

où $\tilde{\sigma}^2$ est la variance asymptotique. En particulier $\tilde{\sigma}^2 \leq \sigma^2$, où σ^2 dénote la variance asymptotique du théorème centrale limite si r_n est une séquence MC pure.

1.4 Approximation stochastique

Dans sa version initiale l'approximation stochastique (Robbins and Monro, 1951) a été conçue comme une approche pour trouver les racines d'une fonction monotone $H : \lambda \mapsto H(\lambda)$ qui est corrompue par du bruit. Nous observons $F(\lambda, \xi)$, où $\lambda \in \Lambda \subset \mathbb{R}$ et ξ est le terme de bruit perturbant la fonction F telle que $\forall \lambda \ \mathbb{E}[F(\lambda, \xi)] = H(\lambda)$. $F(\lambda, \xi)$ est supposée avoir une limite supérieure et une limite inférieure pour tout λ donnée par une constante $\pm C$, ξ - presque sûrement. Nous voulons trouver $\lambda^\star$ tel que $\mathbb{E}[F(\lambda^\star, \xi)] - c = 0$ pour une valeur donnée de $c \in [-C, C]$. Ce problème

peut être résolu avec un algorithme itératif. Si après t itérations $F(\lambda_t, \xi) - c < 0$, cela suggère une augmentation de λ_{t+1} par rapport à l'itération précédente et *vice versa*. Par conséquent, on itère

$$\lambda_{t+1} = \lambda_t - \alpha_t(F(\lambda_t, \xi) - c),$$

avec un taux d'apprentissage décroissant α_t jusqu'à ce que $|F(\lambda_t, \xi) - c| < \epsilon$, pour un petit niveau de tolérance ϵ. Une possibilité pour garantir la convergence est d'exiger que les pas α_t soient tel que $\forall t, \alpha_t > 0$, $\sum_{t=1}^{T} \alpha_t = \infty$ et $\sum_{t=1}^{T} \alpha_t^2 < \infty$ quand $T \to \infty$. On obtient alors le résultat de convergence suivant :

$$\lambda_t \xrightarrow[t\to\infty]{\mathbb{P}} \lambda^\star.$$

Cette approche peut également être utilisée pour optimiser une fonction convexe $H(\lambda)$, c'est-à-dire trouver

$$\lambda^\star \in \arg\min_{\lambda \in \Lambda} H(\lambda).$$

Si H est différentiable, cela équivaut à trouver des points où la dérivée de la fonction devient 0 et donc à chercher des valeurs de λ telles que

$$\frac{\partial H(\lambda)}{\partial \lambda} = 0.$$

Sous des conditions appropriées et si la mesure de ξ ne dépend pas de λ, $\frac{\partial F(\lambda,\xi)}{\partial \lambda}$ est un estimateur sans biais de $\frac{\partial H(\lambda)}{\partial \lambda}$. Cela conduit à l'idée de la descente de gradient stochastique (SGD) donnée par la récursion de mise à jour :

$$\lambda_{t+1} = \lambda_t - \alpha_t \frac{\partial F(\lambda_t, \xi)}{\partial \lambda}.$$

Si un estimateur sans biais du gradient n'est pas disponible, le gradient peut être estimé en utilisant des différences finies (Kiefer and Wolfowitz, 1952).

Des hypothèses standards pour obtenir des garanties de convergence pour la descente de gradient stochastique sont la continuité Lipschitz des gradients et la convexité de la fonction H, voir par exemple Bottou et al. (2016). Sous des hypothèses supplémentaires comme une forte convexité de H, un taux de convergence linéaire en t de la différence $H(\lambda_t) - H(\lambda^\star)$ peut être établi.

Dans le domaine des statistiques et de l'apprentissage automatique, les méthodes basées sur l'approximation stochastique sont omniprésentes : elles fournissent l'outil algorithmique pour l'estimation du maximum de vraisemblance et la minimisation du risque empirique. Beaucoup de recherche a été faite pour accélérer la convergence dans les cas convexes et non convexes par rapport au schéma de base de descente de gradient stochastique. Certaines de ces approches sont, par exemple, l'introduction d'un terme de moment qui ajoute de l'inertie au gradient ou la réduction de la variance de l'estimateur de gradient, voir les travaux de Nesterov (1983); Johnson and Zhang (2013); Defazio et al. (2014). De plus, le choix automatique du taux d'apprentissage α_t

est un domaine de recherche actif (Duchi et al., 2011; Kingma and Ba, 2015). Récemment, le lien établi avec les équations différentielles en temps continu a fourni une meilleure compréhension de l'accélération de la descente de gradient stochastique (Wibisono et al., 2016).

1.5 Inférence variationnelle

Les sections précédentes ont exposé différentes méthodes pouvant être utilisées lors de l'approximation d'une distribution *a posteriori* par échantillonnage. Une autre approche, qui a suscité beaucoup d'intérêt ces dernières années, consiste à construire une approximation de la distribution *a posteriori* à travers une classe de distributions différente. Ceci est réalisé en résolvant un problème d'optimisation, souvent à l'aide des techniques introduites dans la section précédente. L'une de ces approches est appelée inférence variationnelle et consiste à minimiser la divergence Kullback-Leibler (KL) entre une famille de distributions (la famille variationnelle) et la distribution *a posteriori*. Comme précédemment, on note les données observées y, la variable latente x et la distribution *a posteriori* $\pi(x|y) \propto p(y|x)p_0(x)$. La distribution variationnelle est notée $q(x) \in \mathcal{Q}$, où $\mathcal{Q}$ désigne la famille variationnelle. Le problème est défini par :

$$q^\star(x) = \arg\min_{q \in \mathcal{Q}} \mathrm{KL}(q(x)||\pi(x|y)),$$

où $\mathrm{KL}(q(x)||\pi(x|y)) = \mathbb{E}_q\left[\log q(x) - \log \pi(x|y)\right]$ dénote la divergence de Kullback-Leibler. Choisir une classe de distributions $\mathcal{Q}$ revient donc à contrôler la complexité du problème et la qualité de l'approximation. La KL peut être réécrite en fonction de l'évidence du modèle Z_π et d'un deuxième terme nommé limite inférieure de l'évidence (ELBO) :

$$\mathrm{KL}(q(x)||\pi(x|y)) = \log Z_\pi - \mathcal{L}(q),$$

où $\mathcal{L}(q) = \mathbb{E}_q\left[\log p(y|x)p_0(x) - \log q(x)\right]$ est l'ELBO. Par conséquent, la maximisation de l'ELBO équivaut à la minimisation de la KL, car l'évidence ne dépend pas de la famille variationnelle.

1.5.1 Inférence variationnelle par champ moyen

Un choix courant de famille variationnelle $\mathcal{Q}$ est la classe de distributions avec des composants indépendants x_j. Cette approche est appelée inférence variationnelle par champ moyen et repose sur la factorisation

$$q(x) = \prod_{j=1}^{m} q_j(x_j),$$

où les facteurs m sont indépendants les uns des autres. L'avantage de cette approche est que les facteurs optimaux se trouvent sans supposer une forme spécifique de $q_j(x_j)$.

En particulier on obtient :

$$q_j^\star(x_j) \propto \exp\left(\mathbb{E}_{i\neq j}\log\{p(y|x)p_0(x)\}\right),$$

où $\mathbb{E}_{i\neq j}$ est l'ésperance par rapport à $q(x)$, en omettant le facteur $q_j(x_j)$, voir Jordan et al. (1999); Bishop (2006) pour plus de détails. Sous condition de conjugaison du modèle, les espérances peuvent être évaluées analytiquement (Hoffman et al., 2013). En parcourant toutes les espérances, l'ELBO est maximisé. Cet algorithme s'appelle l'inférence variationnelle par ascension de coordonnées (CAVI), voir Blei et al. (2017). Cependant, cette approche présente un inconvénient majeur : la classe de modèles à laquelle cette approche s'applique est plutôt restrictive.

1.5.2 Inférence variationnelle par Monte Carlo

Il est possible de restreindre $\mathcal{Q}$ à une classe paramétrique spécifique de distributions, par exemple la classe des distributions normales multivariées $\mathcal{N}(\mu, \Sigma)$, paramétrée par une moyenne $\mu \in \mathbb{R}^d$ et une matrice de covariance Σ, appartenant à l'espace des matrices symétriques positives sur $\mathbb{R}^{d\times d}$. Ensuite, l'approche variationnelle se résume à trouver le paramétrage de la gaussienne $\mathcal{N}(\mu^\star, \Sigma^\star)$ qui se rapproche le plus de $\pi(x|y)$ et le problème devient

$$\mu^\star, \Sigma^\star \in \arg\min_{\mu,\Sigma} \mathrm{KL}(\mathcal{N}(x|\mu,\Sigma)||\pi(x|y)).$$

Les espérances liées à la formulation de l'ELBO sont des espérances par rapport à la classe variationnelle $\mathcal{Q}$. Avec une grande classe de familles paramétriques potentielles choisies par l'utilisateur, il est souvent possible d'approximer les espérances par échantillonnage Monte Carlo lorsque des formules closes ne sont pas disponibles. Cela conduit à l'idée d'inférence variationnelle par Monte Carlo, qui a pris de l'ampleur dans les années récentes. Elle repose sur une approximation des espérances impliquées et de leurs gradients via l'échantillonnage. L'optimisation de l'ELBO se fait via un schéma d'ascente de gradient sur le paramètre $\lambda \in \Lambda$ de la famille variationnelle en utilisant les gradients estimés. Dans notre exemple gaussien, nous avons $\lambda = (\mu, \Sigma)$ et donc $q_\lambda(x) \widehat{=} \mathcal{N}(x|\mu, \Sigma)$.

Cela fait le lien avec la littérature sur l'optimisation stochastique, introduite dans la section précédente. La différenciation directe de l'ELBO en λ n'est pas possible, car la mesure de l'espérance dépend de ce paramètre. Les deux principales approches pour resourdre ce problème sont l'estimateur de la fonction du score (Ranganath et al., 2014) et l'estimateur par reparamétrisation (Kingma and Welling, 2014).

Le gradient de la fonction score (également appelé gradient REINFORCE (Williams, 1992)) exprime le gradient comme une espérance par rapport à $q_\lambda(x)$ et est donné par

$$\nabla_\lambda \mathcal{L}(\lambda) = \mathbb{E}_{q_\lambda(x)}\left[\nabla_\lambda \log q_\lambda(x)\left\{\log[p(y|x)p_0(x)] - \log q_\lambda(x)\right\}\right]. \tag{1.11}$$

Un estimateur de gradient est obtenu en approximant l'espérance avec des échantillons indépendants de la distribution variationnelle $q_\lambda(x)$. Cet estimateur est assez générique, et s'applique aux distributions variationnelles continues et discrètes.

Une autre approche est basée sur l'astuce de reparamétrisation, où la distribution sur x est exprimée comme une transformation déterministe d'une distribution différente par rapport à une variable de bruit ε, donc $x = g_\lambda(\varepsilon)$ où $\varepsilon \sim p(\varepsilon)$. En utilisant la reparamétrisation, l'ELBO est exprimé comme espérance par rapport à $p(\varepsilon)$ et la dérivée est déplacée à l'intérieur de l'intégrale :

$$\nabla_\lambda \mathcal{L}(\lambda) = \mathbb{E}_{p(\varepsilon)} \left[\nabla_\lambda \log\{p(y|g_\lambda(\varepsilon))p_0(g_\lambda(\varepsilon))\} - \nabla_\lambda \log q_\lambda(g_\lambda(\varepsilon)) \right]. \tag{1.12}$$

L'espérance est approximée en utilisant la moyenne de l'échantillon indépendant de la mesure de base $p(\varepsilon)$. Cet estimateur est limité aux distributions sur les variables continues qui permettent une reparamétrisation différentiable en λ.

En utilisant l'un des estimateurs de gradient désignés par $\hat{g}_N(\lambda) \approx \nabla_\lambda \mathcal{L}(\lambda)$, où N est la taille de l'échantillon, l'ELBO peut alors être optimisé par optimisation stochastique. Ceci est réalisé en itérant les mises à jour stochastiques du gradient avec un taux d'apprentissage décroissant α_t :

$$\lambda_{t+1} = \lambda_t + \alpha_t \hat{g}_N(\lambda_t). \tag{1.13}$$

La convergence du schéma d'ascente de gradient dans (1.13) tend à être lente lorsque les estimateurs de gradient ont une grande variance. Par conséquent, diverses approches pour réduire la variance des estimateurs de gradient existent; par exemple les variables de contrôle, la Rao-Blackwellisation et l'échantillonnage préférentiel. Ces schémas de réduction de la variance doivent souvent être adaptés au problème spécifique et la recherche de solutions plus générales reste donc un domaine de recherche actif.

D'autres développements récents consistent, par exemple, à appliquer l'inférence variationnelle aux équations différentielles stochastiques (Ryder et al., 2018), ou à des modèles implicites (Tran et al., 2017a,b). Un autre sujet d'intérêt actuel est la dérivation de garanties théoriques pour l'inférence variationnelle, qui manquent à ce jour malgré son succès pratique. Voir par exemple Alquier et al. (2016b); Germain et al. (2016); Wang and Blei (2018); Chérief-Abdellatif and Alquier (2018) pour des travaux récents.

1.6 Résumé substantiel

1.6.1 Résumé substantiel de notre travail sur le quasi-Monte Carlo et le calcul bayésien approximatif

Comme nous l'avons vu dans notre introduction au calcul Bayésian approximatif (ABC), il y a en général deux sources d'aléa lors de la génération d'échantillons x_n de la loi *a posteriori* approximative : (a) le caractère aléatoire qui vient de la génération des

échantillons à partir de la distribution *a priori*; (b) le hasard qui provient de la génération des pseudo-observations du modèle. Nous suggérons d'utiliser des points QMC pour la distribution *a priori*, car il est souvent possible de transformer une séquence uniforme de points vers la distribution *a priori* d'intérêt. Nous montrons que cette approche réduit efficacement la variance des estimateurs basée sur les échantillons *a posteriori* approximatifs. La partie restante et dominante de la variance provient du modèle.

Pour illustrer cela, supposons que nous nous intéressions à un estimateur de la constante de normalisation $\int_{\mathcal{X}} \pi_\epsilon(x|y^\star)\,\mathrm{d}\,x = Z_\epsilon$ approximée par

$$\hat{Z}_{N,M} = \frac{1}{NM}\sum_{i=1}^{N}\sum_{j=1}^{M} \mathbb{1}_{\{\delta(y_{i,j},y^\star)\leq\epsilon\}}.$$

Ici, pour chaque x_i simulé selon la loi *a priori*, nous générons M observations de $y_{i,j}$ pour $j = 1, \cdots, M$ du modèle afin d'estimer $\mathbb{P}_x(\delta(y, y^\star) \leq \epsilon)$. Lorsque nous calculons la variance de $\hat{Z}_{N,M}$, nous obtenons

$$\mathrm{Var}\left[\hat{Z}_{N,M}\right] = \underbrace{\mathrm{Var}\left[\mathbb{E}\left[\hat{Z}_{N,M}|x_{1:N}\right]\right]}_{=\mathcal{O}(\frac{1}{N})} + \underbrace{\mathbb{E}\left[\mathrm{Var}\left[\hat{Z}_{N,M}|x_{1:N}\right]\right]}_{=\mathcal{O}(\frac{1}{NM})},$$

en utilisant la formule de décomposition de la variance. Après ajustement par le coût de génération de M pseudo-observations pour chaque x_i, on a

$$M \times \mathrm{Var}\left[\hat{Z}_{N,M}\right] = M \times C_1/N + C_2/N.$$

La variance ajustée du coût augmente donc linéairement en M, conduisant à une valeur optimale de $M = 1$ (Bornn et al., 2015).

Supposons maintenant que $x_i = \Gamma(u_i)$, où u_i est une séquence RQMC et $\Gamma(\cdot)$ transforme la séquence uniforme vers la distribution *a priori*. Sous des conditions de régularité supplémentaires, nous obtenons

$$\mathrm{Var}\left[\hat{Z}_{N,M}\right] = \underbrace{\mathrm{Var}\left[\mathbb{E}\left[\hat{Z}_{N,M}|u_{1:N}\right]\right]}_{=o(\frac{1}{N})} + \underbrace{\mathbb{E}\left[\mathrm{Var}\left[\hat{Z}_{N,M}|u_{1:N}\right]\right]}_{=\mathcal{O}(\frac{1}{NM})}.$$

Le premier terme devient négligeable par rapport au second et on obtient une réduction de la variance en utilisant les séquences RQMC pour la loi *a priori*. Ce résultat implique que des valeurs $M > 1$ peuvent être utilisées car le premier terme devient négligeable, voir Proposition 1 pour les séquences RQMC et Proposition 2 pour les séquences QMC dans le chapitre 3.

De plus, nous établissons un théorème centrale limite pour les séquences QMC et RQMC pour la construction des échantillons selon la loi *a priori* et pour un M fixé, c'est-à-dire

$$\sqrt{N}\left(\hat{Z}_{N,M} - Z_\epsilon\right) \xrightarrow[N\to\infty]{\mathcal{L}} \mathcal{N}(0, \tilde{\sigma}^2),$$

suivant l'approche de Ökten et al. (2006). En particulier, $\tilde{\sigma}^2 \leq \sigma^2$, où σ^2 correspond à la variance d'une approche standard de Monte Carlo, voir Théorème 4. Nos résultats sont également valides pour un type plus général d'estimateurs par échantillonnage préférentiel de la forme auto-normalisée : quand l'objet est le calcul de l'espérance de $\psi(x)$ et on note le poids $w(x) = \mathbb{1}_{\{\delta(y,y^\star)\leq\epsilon\}} p_0(x)/g(x)$ alors

$$\mathbb{E}_{\pi_\epsilon(x|y)}\left[\psi(X)\right] \approx \frac{\sum_{i=1}^N w(x_i)\psi(x_i)}{\sum_{i=1}^N w(x_i)}$$

peut être utilisé comme approximation. Cet estimateur bénéficie également de la réduction de la variance QMC comme on peut le montrer en décomposant la variance ou en utilisant un théorème centrale limite .

Une autre contribution de notre travail est l'utilisation de QMC dans un contexte séquentiel. Par conséquent, nous procédons dans l'esprit de l'échantillonnage préférentiel adaptatif (Oh and Berger, 1992) et nous choisissons une distribution de proposition $g_t(x)$ basée sur les itérations précédentes tout en diminuant le seuil d'acceptation ϵ_t. Plus précisément, nous adaptons une distribution gaussienne g_t en fonction de l'ensemble pondéré $\{x_i^{t-1}, w(x_i^{t-1})\}$ de l'itération précédente. Ainsi, la distribution de proposition se concentre dans les régions de probabilité *a posteriori* lorsque ϵ_t est diminué. Nous choisissons une loi de proposition gaussienne adaptative car cette approche permet une transformation des points QMC vers l'espace d'intérêt et cela a bien fonctionné en pratique. Nous illustrons les avantages de cette approche à travers une étude de simulation approfondie. L'un des exemples que nous utilisons est l'inférence des paramètres dans un modèle Lotka-Volterra. Dans ce cadre, nous observons deux séries temporelles bruitées qui représentent l'interaction d'une population de prédateurs et de proies. L'intérêt réside dans la distribution *a posteriori* de 3 paramètres qui déterminent un système d'équations différentielles. Nous simulons à partir d'une loi uniforme *a priori* les paramètres, générons des séries temporelles bruitées donnant les observations y_i puis nous faisons varier le seuil d'acceptation pour la distance des données simulées aux valeurs observées.

Quand le seuil d'acceptation s'approche de zéro, la variance de l'estimateur de la moyenne cumulative augmente, comme le montre la Figure 1.2. L'utilisation de (R)QMC entraîne une réduction substantielle de la variance de l'estimateur, comme prédit par notre analyse théorique. Notre approche est facilement applicable aux approches ABC existantes et devrait donc être d'un grand intérêt pratique.

1.6.2 Résumé substantiel de notre travail sur le quasi-Monte Carlo et l'inférence variationnelle

Comme nous l'avons vu dans la section sur l'inférence variationnelle par Monte Carlo, un estimateur du gradient de l'ELBO peut être construit soit par échantillonnage selon la famille variationnelle et en utilisant l'estimateur de la fonction score, voir (1.11), ou en échantillonnant à partir de la reparamétrisation et en utilisant l'estimateur basé sur

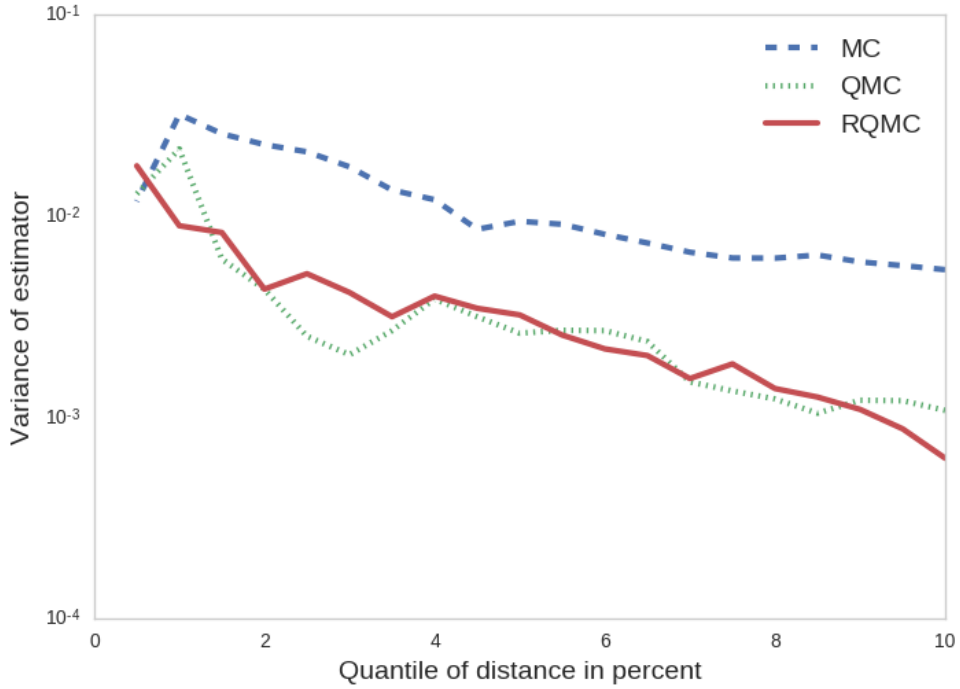


FIGURE 1.2: Variance de l'estimateur de la moyennne *a posteriori* pour le modèle Lotka-Volterra. Le graphique est basé sur 50 répétitions de 10^5 simulations selon le modèle. Les observations acceptées correspondent à des quantiles basés sur les plus petites distances $\delta(y_n, y^\star)$.

(1.12). Un problème majeur de cette implémentation est la grande variance de ces estimateurs lors de l'utilisation d'une taille d'échantillon trop petite. Par conséquent, diverses approches de réduction de la variance ont été proposées, voir par exemple Ranganath et al. (2014); Ruiz et al. (2016a); Miller et al. (2017); Roeder et al. (2017). Cependant, toutes ces approches n'améliorent pas le taux de convergence de $1/N$ de l'estimateur de Monte Carlo basé sur N simulations. Il s'est avéré que la majorité des familles variationnelles couramment utilisées pour l'estimateur définie en (1.11) et les reparamétrisations de l'estimateur en (1.12) peuvent être vue comme la transformation d'une séquence uniforme de variables aléatoires sur $\mathcal{U}[0,1]^d$. En utilisant une application régulière $\Gamma : [0,1]^d \to \mathbb{R}^d$ dans le même esprit que dans notre travail sur ABC, on peut obtenir une réduction de variance pour les deux estimateurs de gradient en utilisant une séquence RQMC au lieu d'une séquence aléatoire uniforme. L'estimateur, noté $\hat{g}_N(\lambda)$, où λ est le paramètre de la famille variationnelle, hérite du taux de convergence amélioré tel que $\text{Var}\,[\hat{g}_N(\lambda)] \leq \mathcal{O}(N^{-2})$, sous certaines hypothèses de régularité.

Dans notre travail, nous montrons que cette approche est avantageuse à plusieurs points de vue. Premièrement, l'approche conduit à une amélioration de la méthode du gradient stochastique avec un facteur de $1/N$ dans les limites standards lors de l'utilisation d'un taux d'apprentissage fixe. Dans le cas des gradients continus au sens

de Lipschitz, nous obtenons, quand $T \to \infty$,

$$\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}\|\nabla\mathcal{L}(\lambda_t)\|^2 \leq \mathcal{O}(N^{-2}),$$

où la constante omise dans la borne supérieure dépend de la constante de Lipschitz L, du taux d'apprentissage fixe α et d'une majorante universelle de la variance de $\hat{g}_N(\lambda)$ pour tous λ. Voir le Théorème 5 pour plus de détails. Cela se compare favorablement au résultat de l'approche Monte Carlo, où la dépendance à la taille de l'échantillon est de $1/N$.

Dans le cas d'une fonction fortement convexe $\mathcal{L}$, nous pouvons énoncer un résultat en termes d'écart entre la valeur de la fonction de l'itération en cours et le vrai optimiseur $\lambda^\star$. Nous obtenons quand $T \to \infty$

$$|\mathcal{L}(\lambda^\star) - \mathbb{E}\mathcal{L}(\lambda_T)| \leq \mathcal{O}(N^{-2}),$$

où la constante dépend maintenant du paramètre de forte convexité. Là encore, nous avons gagné un facteur de $1/N$ par rapport au cas de l'échantillonnage Monte Carlo, voir Théorème 6 pour plus de détails.

Deuxièmement, il est possible de profiter du gain de $1/N$ et d'augmenter la taille de l'échantillon à chaque itération et d'obtenir ainsi un taux de convergence plus rapide. Supposons que l'estimateur $\hat{g}_{N_t}(\lambda_t)$ est maintenant basé sur une taille d'échantillon croissante $N_t = 1/\lceil\tau^t\rceil$, où $\tau =: 1/\xi > 1$ est un taux géométrique. Sous ces hypothèses, nous obtenons que

$$|\mathcal{L}(\lambda^\star) - \mathbb{E}\mathcal{L}(\lambda_T)| \leq \omega\xi^{2t},$$

où ω est une constante. Encore une fois, ce résultat est plus satisfaisant que celui obtenu avec l'approche Monte Carlo où nous obtenons un taux de convergence plus lent ξ^t. Voir le Théorème 7 pour plus de détails.

Troisièmement, d'un point de vue expérimental, la réduction de la variance permet d'utiliser un plus grand taux d'apprentissage comme suggéré par un algorithme de gradient stochastique adaptatif comme Adagrad (Duchi et al., 2011). Par conséquent, l'algorithme fait des pas plus importants dans l'espace des paramètres et converge plus vite. Ici, nous illustrons ceci avec une régression logistique bayésienne en dimension 31, basée sur l'ensemble de données *breast cancer* disponible dans la bibliothèque python scikit-learn. Voir également le travail de Jaakkola and Jordan (2000) pour un autre approche computationnelle pour l'inférence variationnelle dans ce modèle. Nous utilisons l'approche d'inférence variationnelle dite boîte noire et le paramétrage de la distribution variationnelle par sa moyenne $\mu \in \mathbb{R}^d$ et sa matrice de covariance diagonale $\operatorname{diag}\Sigma \in \mathbb{R}^{d+}$.

L'optimisation Adagrad commence avec un taux d'apprentissage initial de 0.1. Nous comparons notre approche RQMC basée sur des échantillons de taille 30, pour

estimer le gradient, avec deux approches MC basées sur des échantillons de taille 30 et de 900, respectivement. Les estimateurs de gradient MC utilisent comme variable de contrôle la moyenne et la variance du paramétrage actuel. L'approche basée sur RQMC converge plus vite en termes d'ELBO comme l'illustre la Figure 1.3. Dans le corps de notre article, nous illustrons les avantages de notre approche par trois exemples numériques supplémentaires. Notre approche est plutôt générale et ne se limite pas à l'inférence variationnelle. Elle peut être utilisée dans différents contextes dans lesquels un estimateur de gradient peut être construit en utilisant un échantillonnage QMC.

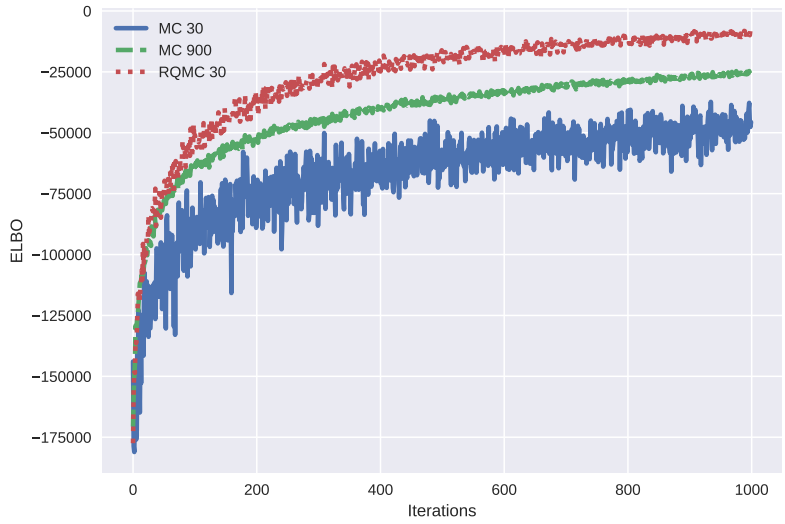


FIGURE 1.3: Borne inférieure en fonction des itérations pour l'approximation variationnelle d'une régression logistique bayésienne. Les deux estimateurs par Monte Carlo du gradient sont basés sur une taille d'échantillon de $N = 30$ et $N = 30^2$. L'estimateur de gradient RQMC est basé sur $N = 30$ simulations. L'ELBO est estimé à l'aide d'un échantillon de taille 1000.

1.6.3 Résumé substantiel de notre travail sur le réglage adaptatif du Monte Carlo hamiltonien dans le Monte Carlo séquentiel

Le dernier chapitre de cette thèse présente notre travail sur le réglage adaptatif de l'algorithme HMC dans SMC. Nous considérons le problème de l'échantillonnage selon la loi *a posteriori* $\pi_T(x) = \pi(x|y)$ en commençant avec un nuage de particules de taille N générées par la distribution *a priori* $\pi_0(x) = p_0(x)$. Une série de distributions intermédiaires $\pi_t(x)$ est construite en utilisant un pont géométrique :

$$\pi_t(x) \propto p_0(x)^{1-\beta_t} \times \pi(x|y)^{\beta_t} = p_0(x)p(y|x)^{\beta_t}$$

avec des températures croissantes $0 = \beta_0 < \cdots < \beta_t < \cdots < \beta_T = 1$. Les particules sont déplacées suivant la séquence des distributions intermédiaires en utilisant une

alternance d'étapes de repondération, de ré-échantillonnage et de propagation.

Notre idée d'utiliser HMC dans SMC est motivée par le fait que les noyaux HMC sont mieux adaptés à une dimension croissante que d'autres noyaux de type Metropolis-Hasting (Beskos et al., 2013; Mangoubi and Smith, 2017). La performance du noyau de propagation se répercute sur la performance de l'algorithme SMC. En même temps, le nuage de particules fournit des informations qui permettent d'apprendre des paramètres raisonnables pour le noyau de propagation. Ainsi, la combinaison de HMC et de SMC est mutuellement avantageuse.

Dans l'algorithme SMC bayésien statique, il existe généralement trois choix de réglages à prendre en compte : (a) le choix de la séquence des distributions intermédiaires ou de manière équivalente le choix des températures β_t; (b) la répétition du nombre d'étapes de propagation pour assurer une exploration correcte des distributions d'intérêt; (c) le choix des paramètres de réglage du noyau de propagation. Dans notre travail, nous illustrons l'importance d'approcher soigneusement (a), (b) et (c). Notre approche pour (a) est basée sur le travail de Zhou et al. (2016). Concernant (b), nous suggérons une heuristique pour choisir le nombre d'étapes de propagation basé sur l'autocorrélation du noyau de propagation, fournissant ainsi une ligne directrice pour des applications pratiques. En ce qui concerne (c) nous adaptons l'idée de Fearnhead and Taylor (2013) aux noyaux de propagation HMC dans SMC et développons une nouvelle méthode plus coûteuse en calcul mais plus robuste.

Notre approche repose sur un essai où nous testons différentes combinaisons des paramètres ϵ, L et nous évaluons leurs performances. L'algorithme que nous avons développé est donné dans l'Algorithme 4. Nous commençons avec le nuage de particules ré-échantillonnées, obtenu lors de l'itération précédente. Grâce à notre algorithme, nous obtenons un ensemble de valeurs (ϵ, L) qui donne une distance de saut au carré, qui doit être élevée pour le noyau de propagation invariant à la distribution actuelle π_{t-1}.

L'algorithme procède comme suit : nous échantillonnons d'abord N différentes valeurs des paramètres du noyau et nous les assignons aux différentes particules. Puis nous laissons évoluer le flux numérique HMC et on enregistre la distance de saut aussi bien que les fluctuations de l'énergie pour chaque particule i. Basé sur la performance obtenue $\tilde{\Lambda}(\cdot, \cdot)$ nous pondérons les paramètres et nous les ré-échantillonnons selon une distribution catégorique. Les paramètres ré-échantillonnés sont ensuite renvoyés en sortie. En tant que résultat supplémentaire, nous apprenons la dépendance des fluctuations d'énergie de la discrétisation ϵ. Cette information est utilisée pour déterminer une limite supérieure $\epsilon^\star$ de l'intervalle sur lequel ϵ sera tiré au cours de la prochaine itération.

Le coût supplémentaire provient de l'exécution supplémentaire du flux HMC nécessaire à chaque étape de température. D'une part ce calcul supplémentaire a l'avantage d'accorder le noyau de propagation précisément au pas de temps actuel. D'autre part notre adaptation de Fearnhead and Taylor (2013) repose sur l'hypothèse que les performances des paramètres du noyau ne changent pas beaucoup d'un pas de temps au

suivant. Dans nos simulations, nous montrons que les deux approches devraient être prises en compte pour des distributions cibles difficiles.

Algorithm 4: Réglage de l'algorithme HMC avec essai préliminaire.

Input: Particules $\tilde{x}_{t-1}^i$, $i \in 1:N$, flux HMC $\widehat{\Phi}_{\cdot,\cdot}$ avec loi invariante π_{t-1}, $\epsilon_{t-1}^\star$

Result: Echantillon (ϵ_t^i, L_t^i), $i \in 1:N$, borne supérieure $\epsilon_t^\star$

foreach $i \in 1:N$ **do**

- Simuler $\hat{\epsilon}_t^i \sim \mathcal{U}[0, \epsilon_{t-1}^\star]$ et $\hat{L}_t^i \sim \mathcal{U}\{1 : L_{max}\}$;
- Simuler $z_t^i \sim \mathcal{N}(0_d, M_{t-1})$;
- Utiliser l'integration leapfrog : $(\hat{x}_t^i, \hat{z}_t^i) \leftarrow \widehat{\Phi}_{\hat{\epsilon}_t^i, \hat{L}_t^i}(z_t^i, \tilde{x}_{t-1}^i)$;
- Calculer ΔE_t^i et $\tilde{\Lambda}(\hat{x}_{t-1}^i, \hat{x}_t^i)$

Calculer $\epsilon_t^\star$ basé sur la régression quantile de ΔE_t^i sur $\hat{\epsilon}_t^i$ $\forall i \in 1:N$;

Simuler $(\epsilon_t^i, L_t^i) \sim \mathcal{C}at\left(w_t^i, \{\hat{\epsilon}_t^i, \hat{L}_t^i\}\right)$, où $w_t^i \propto \tilde{\Lambda}(\hat{x}_{t-1}^i, \hat{x}_t^i)$ $\forall i \in 1:N$;

En utilisant un noyau HMC correctement réglé, nous montrons que les algorithmes SMC peuvent être utilisés dans des dimensions élevées pour l'inférence par simulation selon la loi *a posteriori*. Ceci contredit l'opinion largement répandue que SMC ne peut pas être utilisé pour des dimensions élevées. De plus le calcul de la constante de normalisation donne un moyen efficace de comparaison de modèle, encore disponible au fur et à mesure que la dimension augmente. Nous mettons en évidence ces arguments sur différentes applications.

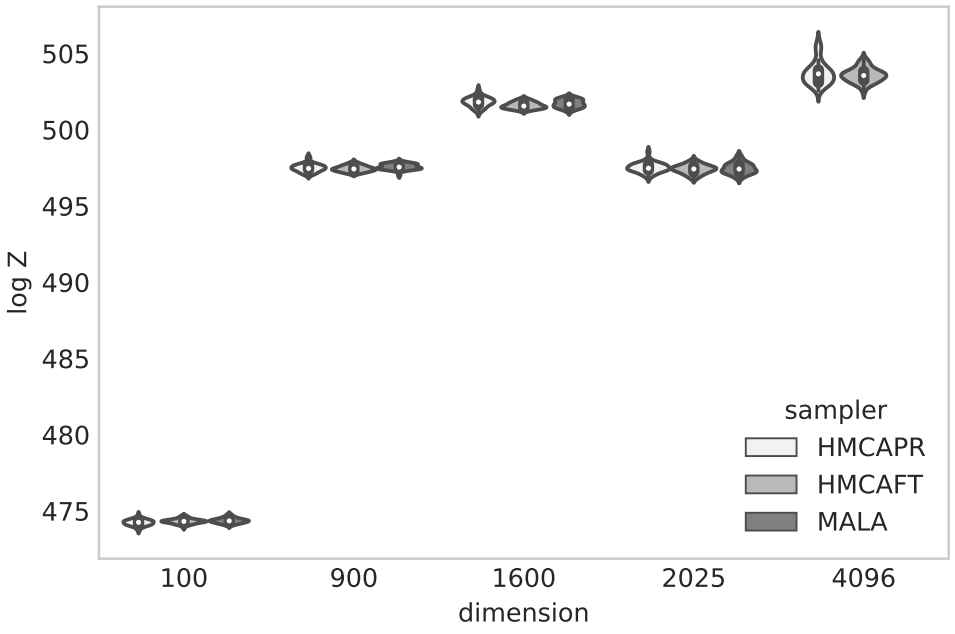


FIGURE 1.4: Temperer d'une loi normale *a priori* vers la loi *a posteriori* d'un processus de Cox log gaussien sur différentes dimensions. Nous illustrons les estimations des constantes de normalisation pour trois algorithmes différents. Tous les algorithmes donnent une estimation assez précise de la constante de normalisation.

Pour illustrer nos résultats expérimentaux, nous montrons ici la constante de normalisation estimée de la loi *a posteriori* d'un modèle de processus Cox log gaussien dans la Figure 1.4. Dans ce modèle, les observations suivent un processus de Poisson conditionnel à un processus gaussien. Le but de l'inférence *a posteriori* est de récupérer le processus latent compte tenu des observations. Les observations sont les lieux de 126 pins. La dimension du modèle dépend de la discrétisation des observations spatiales. Une caractéristique intéressante de ce modèle est le fait que la dimension peut être facilement augmentée en utilisant une discrétisation plus fine. Nous comparons trois algorithmes SMC différents. Tous les paramètres des trois algorithmes sont réglés de manière adaptative. Le premier algorithme, noté MALA, est basé sur un noyau de propagation MALA. Pour régler les paramètres du noyau, nous utilisons l'approche de Fearnhead and Taylor (2013). Les deux autres algorithmes utilisent un noyau de propagation HMC, en utilisant soit notre adaptation de Fearnhead and Taylor (2013), abrégé par FT, soit une approche que nous suggérons basée sur l'essai préliminaire, abrégée par PR. Quand la dimension dépasse $\approx$ 2000, l'algorithme MALA dépasse son budget de calcul. Comme mesure globale de la performance, nous utilisons la variance des quantités estimées et ajustons par le coût de calcul. L'algorithme MALA reste compétitif jusqu'à une dimension $1,600$. L'algorithme basé sur la procédure de réglage FT fonctionne mieux pour ce modèle. Dans nos autres expériences, nous montrons que l'algorithme basé sur l'essai préliminaire fonctionne mieux lorsque la distribution *a posteriori* est fortement corrélée. Dans ce cas, le coût supplémentaire de l'essai préliminaire est associé à une meilleure performance.

Chapter 2

Introduction

This chapter provides an introduction to the work in the rest of this thesis. We review the concepts necessary for understanding the problems arising in Bayesian computation. We introduce first the main ideas of Bayesian statistics. Then, we discuss Monte Carlo and quasi-Monte Carlo sampling. After that we turn to stochastic optimization and the construction of variational approximations to the posterior distributions. Finally, we summarize the main contributions of the three articles that constitute the remaining chapters of this thesis.

2.1 Bayesian inference

The purpose of statistical modeling is the understanding of a phenomenon given data. Mathematically speaking, this problem is described as the tuple of an observational space $\mathcal{Y}$, its Borel set $\mathcal{B}(\mathcal{Y})$ and a family of probability measures $\mathbb{P}_x$, where $x \in \mathcal{X}$ and $\mathcal{X}$ is the parameter space. $(\mathcal{Y}, \mathcal{B}(\mathcal{Y}), \{\mathbb{P}_x, x \in \mathcal{X}\})$ forms a model. If $\mathcal{X} \subset \mathbb{R}^d$ and $d < \infty$ the model is said to be parametric, what we shall consider from here onwards. Given a sequence of realizations $y_1, \cdots, y_N$ of length N of the random variables $Y_1, \cdots, Y_N \in \mathcal{Y}$, statistical inference aims at identifying the parameter x given the observed $y_1, \cdots, y_N$.

In what follows we shall consider that the probability measure $\mathbb{P}_x$ is dominated by a reference measure, hereafter denoted by $\mathrm{d}\, y$. The likelihood of the model is given as

$$p : x \times (y_1, \cdots, y_N) \mapsto p(y_1, \cdots, y_N | x).$$

In the context of statistical modeling Bayesian inference allows to take the potential knowledge of the uncertainty related to the parameters of the model explicitly into account. The uncertainty about the parameters is modeled in terms of the prior distribution, that might reflect the knowledge of an expert about the underlying problem. See Robert (2007) for more details on the decision-theoretic foundation of Bayesian statistics.

The parameter x itself is considered to be a random variable defined on the measured space $(\mathcal{X}, \mathcal{B}(\mathcal{X}), \mathrm{d}\, x)$ endowed with a prior density p_0 with respect to $\mathrm{d}\, x$.

The uncertainty of the parameter x after observing the data is quantified by the posterior density, which is obtained using Bayes' formula:

$$\pi(x|y_1,\cdots,y_N) = \frac{p_0(x)p(y_1,\cdots,y_N|x)}{\int_{\mathcal{X}} p_0(x)p(y_1,\cdots,y_N|x)\,\mathrm{d}\,x}. \tag{2.1}$$

As long as $\int_{\mathcal{X}} p_0(x)p(y_1,\cdots,y_N|x)\,\mathrm{d}\,x < \infty$ the posterior distribution is well defined. Inference in the Bayesian framework is carried out by calculating quantities with respect to the posterior distribution. Of common interest are for example posterior moments like the mean:

$$\mathbb{E}\left[X\right] = \int_{\mathcal{X}} x\pi(x|y_1,\cdots,y_N)\,\mathrm{d}\,x.$$

Another quantity of interest is the *maximum a posteriori* given as:

$$x^\star = \arg\max_{x\in\mathcal{X}} \pi(x|y_1,\cdots,y_N).$$

Hypothesis testing is carried out by calculating the probability under the posterior distribution

$$\mathbb{P}_\pi(x \in \mathcal{X}_i) = \int_{\mathcal{X}_i} x\pi(x|y_1,\cdots,y_N)\,\mathrm{d}\,x,$$

where $\mathcal{X}_i$ for $i = 1,2$ correspond to the sets characterizing different hypothesis. For the purpose of model choice the marginal likelihood, also called evidence,

$$Z_\pi = \int_{\mathcal{X}} p_0(x)p(y_1,\cdots,y_N|x)\,\mathrm{d}\,x$$

is of interest as it allows to compare two different models.

Bayesian inference thus strongly depends on the statistician's ability to calculate expectations with respect to the posterior distribution. However, this is a difficult problem and apart from conjugate models the explicit form of the posterior density is oftentimes only available up to a proportional factor such that

$$\pi(x|y_1,\cdots,y_N) \propto p_0(x)p(y_1,\cdots,y_N|x).$$

Therefore, two major approaches have emerged in statistics: (a) approaches that are based on characterizing the posterior distribution through sampling and (b) approaches that are based on an approximation of the posterior through a tractable family of distributions, potentially different from the true posterior. In order to simplify notation, we will refer to y as our observed data instead of $y_1,\cdots,y_N$ for the rest of this chapter.

We will now discuss sampling approaches in Section 2.2 and 2.3. Section 2.4 reviews the ideas of stochastic approximation. The approximation via tractable families through variational inference is exhibited in Section 2.5. Section 2.6 summarizes the contribution of this thesis to the field of computational statistics, based on the concepts introduced before.

2.2 Monte Carlo Sampling

The aim of Monte Carlo sampling (Metropolis and Ulam, 1949) is the calculation of integrals over the space $\mathcal{X}$ that are of the form

$$\int_{\mathcal{X}} f(x)\,\mathrm{d}\,x,$$

where $f : \mathcal{X} \to \mathbb{R}$, for example. It is often possible to rewrite the integrand as $f(x) = \psi(x)\pi(x)$, where $\pi(x)$ is the density function of a random variable X defined on the space $\mathcal{X}$. The problem can thus be reframed as the calculation of the expectation of $\psi(X)$ with respect to the probability distribution $\mathbb{P}$ with density π:

$$\int_{\mathcal{X}} f(x)\,\mathrm{d}\,x = \int_{\mathcal{X}} \psi(x)\pi(x)\,\mathrm{d}\,x = \mathbb{E}\left[\psi(X)\right].$$

2.2.1 Independent Sampling

The rationale behind the Monte Carlo method is to replace the explicit calculation of the expectation of the random variable $\psi(X)$ by an approximation that is based on the empirical mean of a series of N independent realizations. The weak law of large number guarantees the convergence:

$$\frac{1}{N}\sum_{n=1}^{N}\psi(x_n) \xrightarrow[N\to\infty]{\mathbb{P}} \mathbb{E}\left[\psi(X)\right],$$

given that the expectation and variance of $\psi(X)$ exist and are finite. The expected squared error of the approximation

$$\mathbb{E}\left[\left\|\frac{1}{N}\sum_{n=1}^{N}\psi(x_n) - \mathbb{E}\left[\psi(X)\right]\right\|_2^2\right] = \frac{\operatorname{tr}\operatorname{Var}\left[\psi(X)\right]}{N},$$

goes to 0 at rate of $1/N$, independent of the dimension of the integral. This approach relies on the ability of sampling i.i.d. draws from the underlying distribution $\mathbb{P}_X$ of the random variable X. Moreover, a central limit theorem of the following form may be established:

$$\sqrt{N}\left(\frac{1}{N}\sum_{n=1}^{N}\psi(X_n) - \mathbb{E}\left[\psi(X)\right]\right) \xrightarrow[N\to\infty]{\mathcal{L}} \mathcal{N}(0, \operatorname{Var}\left[\psi(X)\right]).$$

Confidence intervals for assessing the uncertainty of approximating $\mathbb{E}\left[\psi(X)\right]$ by $1/N\sum_{n=1}^{N}\psi(X_n)$ may be obtained through this asymptotic reasoning.

Random Number Generation

From a computational point of view the Monte Carlo method hinges on the efficient generation of random numbers on a computer. Random number generators proceed

by producing pseudo-random numbers on the $[0,1]^d$ hypercube, that are then transformed to a more complicated space of interest. An extensive treatment of the topic is given in Devroye (1986).

A simple approach in one dimension is the use of the inverse distribution function. If $U \sim \mathcal{U}[0,1]$, then $F^{-1}(U) = X \sim \mathbb{P}_X$, where F^{-1} is the inverse cumulative distribution function (cdf) of the random variable X. Samples generated via the inverse cdf may then be used as a starting point for generating other distributions using for example accept-reject based sampling. We are now going to discuss importance sampling in more detail as a large part of the rest of this thesis in based on this concept.

Importance Sampling

Importance sampling (IS), extensively studied by Geweke (1989), is based on the importance sampling identity:

$$\mathbb{E}_\pi\left[\psi(X)\right] = \int_{\mathcal{X}} \psi(x)\pi(x)\,\mathrm{d}\,x = \int_{\mathcal{X}} \psi(x) \underbrace{\frac{\pi(x)}{g(x)}}_{=:w(x)} g(x)\,\mathrm{d}\,x = \mathbb{E}_g\left[\psi(X)w(X)\right].$$

We have expressed the initial expectation as an expectation with respect to a different density g, called the proposal density. For this expression to be correct, π has to be absolutely continuous with respect to g. The term $w(x)$ is called importance weight.

This identity yields a direct approach for calculating expectations with respect to a distribution that we cannot sample from directly but that we can evaluate pointwise. For a sample $x_1, \cdots, x_N \sim g(x)$ the law of large number guarantees

$$\frac{1}{N}\sum_{n=1}^{N} \psi(x_n)w(x_n) \xrightarrow[N\to\infty]{a.s.} \mathbb{E}_g\left[\psi(X)w(X)\right] = \mathbb{E}_\pi\left[\psi(X)\right], \tag{2.2}$$

if $\mathbb{E}_g\left[|\psi(X)w(X)|\right] < \infty$. In a majority of the cases of interest the density π is known only up to a normalizing constant. Denote this density $\tilde{\pi}$, thus $\pi(x) = \tilde{\pi}(x)/Z$, where $Z = \int_{\mathcal{X}} \tilde{\pi}(x)\,\mathrm{d}\,x$. In this case IS provides us with a mechanism to estimate the normalizing constant as

$$\frac{1}{N}\sum_{n=1}^{N} w(x_n) \xrightarrow[N\to\infty]{a.s.} \mathbb{E}_g\left[w(X)\right] = \int_{\mathcal{X}} \tilde{\pi}(x)\,\mathrm{d}\,x = Z. \tag{2.3}$$

An estimation of $\mathbb{E}_\pi\left[\psi(X)\right]$ if the normalizing constant is not available is based on the autonormalized importance sampling estimator given as

$$\frac{\frac{1}{N}\sum_{n=1}^{N} \psi(x_n)w(x_n)}{\frac{1}{N}\sum_{n=1}^{N} w(x_n)} \xrightarrow[N\to\infty]{a.s.} \mathbb{E}_\pi\left[\psi(X)\right]. \tag{2.4}$$

The autonormalized IS estimator is in general biased, but consistent. A central limit theorem for the estimators in (2.2) and (2.4) may be established using for example the

δ-method or Slutsky's lemma, if the corresponding asymptotic variances exist. Note that this is, for instance, not the case, if the tails of the proposal are lighter than the tails of the target density.

The performance of importance sampling approaches depends on the closeness of the target distribution π and the proposal distribution g (Chatterjee et al., 2018). Moreover, IS tends to break down when the dimension of the target space $\mathcal{X}$ increases. Therefore various adaptation strategies have been developed, see for example Cappé et al. (2004); Cornuet et al. (2012).

Apart from using IS for evaluating integrals, interest may lie in the visualization of the approximated distribution. For this purpose, the set of weighted samples $\{x_n, w(x_n)\}_{n=1,\cdots,N}$ may be resampled according to the weights in order to obtain an unweighted approximation of the target distribution and may then be used, e.g. for constructing histograms. A widely used approach is based on multinomial resampling, where the observations are resampled according to a multinomial distribution based on their weights. This resampling step introduces variance. Hence, if interest lies in the calculation of an expectation, the weighted approximation should be preferred. Other approaches such as systematic resampling and stratified resampling might also be used. These approaches have the advantage of introducing less variance than multinomial resampling. For a recent theoretical analysis of these approaches see Gerber et al. (2017).

Pseudo-Marginal Sampling

IS relies on the ability to evaluate the unnormalized density $\tilde{\pi}(\cdot)$. However, there is a variety of problems of interest, where $\tilde{\pi}(\cdot)$ cannot be evaluated pointwise but an unbiased estimator $\widehat{\tilde{\pi}}(x)$ of $\tilde{\pi}(x)$ can be calculated for all x. A natural approach is to replace $\tilde{\pi}(x)$ by $\widehat{\tilde{\pi}}(x)$ and to use the same computational approach as if the correct weights were available. This idea has been introduced by Beaumont (2003) and later analyzed in detail by Andrieu and Roberts (2009). This situation occurs for example in the presence of a latent random variable Z with density p:

$$\pi(x|y) \propto p_0(x) \int_{\mathcal{Z}} h(x, y|z) p(z) \, \mathrm{d}\, z,$$

where the likelihood of the model is an expectation with respect to the random variable Z. By simulating from p we may approximate the likelihood as

$$p(y|x) = \mathbb{E}_p \left[h(x, y|z) \right] \approx \frac{1}{M} \sum_{j=1}^{M} h(x, y|z_j).$$

Consequently, the posterior density is approximated by a random quantity given as

$$\widehat{\tilde{\pi}}(x|y) = p_0(x) \frac{1}{M} \sum_{j=1}^{M} h(x, y|z_j).$$

This quantity can then be used for generating samples from the posterior distribution, either in an importance sampling fashion or in a Markov Chain Monte Carlo approach (see later in this chapter). This problem arises, for example, in hidden Markov models (Cappé et al., 2005). The work of Andrieu et al. (2010) studies this approach in the context of particle filtering. Another area where this idea is of major interest is approximate Bayesian computation (ABC), that we introduce now.

Approximate Bayesian computation

ABC is a concept commonly used in models where the likelihood is not tractable but simulating from the model given a parameter value is feasible. This idea goes back to the seminal work of Tavaré et al. (1997). This problem arises for instance in subfields of statistical Biology such as Philogenetics and Epidemiology.

To make this more concrete, suppose that we have some observed data $y^\star$ and a model given by its likelihood $p(y|x)$ as well as a prior distribution $p_0(x)$. For every value x sampled from the prior it is possible to generate pseudo data such that $y|x \sim p(\cdot|x)$. A natural way of doing simulation based inference in this setting is to compare a distance $\delta : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^+$ between y and $y^\star$ and to weight x according to this distance. When simulating a number of samples that are either accepted or rejected if the distance is smaller than a threshold ϵ we simulate effectively from a joint distribution such that

$$x, y \sim \pi_\epsilon(x, y|y^\star) \propto p_0(x)p(y|x)\mathbb{1}_{\{\delta(y,y^\star)\leq\epsilon\}}.$$

After marginalizing out y we get the approximate posterior

$$\pi_\epsilon(x|y^\star) \propto p_0(x)\mathbb{P}_x(\delta(y, y^\star) \leq \epsilon). \tag{2.5}$$

Thus the simulation of the ABC posterior can be seen as a weighting of simulations from the prior according to the probability that the associated pseudo observations fall inside an ϵ-ball around the true observations. For the purpose of simulation $\mathbb{P}_x(\delta(y, y^\star) \leq \epsilon)$ is approximated by the random weight $\mathbb{1}_{\{\delta(y,y^\star)\leq\epsilon\}}$. This random weight is an unbiased, positive estimator as

$$\mathbb{E}_p\left[\mathbb{1}_{\{\delta(y,y^\star)\leq\epsilon\}}\right] = \int_{\mathcal{Y}} \mathbb{1}_{\{\delta(y,y^\star)\leq\epsilon\}} p(y|x)\,\mathrm{d}\,y = \mathbb{P}_x(\delta(y, y^\star) \leq \epsilon).$$

This links ABC to the previously introduced idea of pseudo-marginal sampling.

In most cases the distance between the observed data and the pseudo data is expressed as $\delta(y, y^\star) = \|s(y) - s(y^\star)\|_2$, where $s(\cdot)$ is a summary statistic extracting the most important features of the data. If the summary statistic is sufficient and $\epsilon \to 0$, we recover in fact the true posterior $\pi(x|y^\star)$.

In its most basic form, ABC sampling proceeds as rejection sampling. This approach may be very inefficient, since generating samples from the model $p(\cdot|x)$ is

often computationally expensive and if the value of x is far from the region of high posterior probability, the generated sample of y will probably lead to a rejection. This problem can be tackled with importance sampling. We try to find a proposal distribution $g(x)$ close to the approximate posterior in (2.5). Thereby we restrict simulations to regions of high posterior probability. Iterative approaches have been developed, that are sequential in nature and try to concentrate the computation in these regions of high posterior probability (Sisson et al., 2009; Del Moral et al., 2012).

The construction of summary statistics (Fearnhead and Prangle, 2012) or other distance measures (Bernton et al., 2017), as well as the asymptotic properties (Frazier et al., 2018) have been an active area of research in the last years.

Recently, the idea of constructing surrogate models, that bypass extensive sampling from the model gained attention (Wilkinson, 2014). Another approach is bypassing the setting of a threshold ϵ, see for example Papamakarios and Murray (2016); Price et al. (2018). Other developments consist in the use of Bayesian optimization (Gutmann and Corander, 2016) and random forests (Pudlo et al., 2016).

2.2.2 Dependent Sampling

The approaches for sampling introduced sofar rely on the ability of generating independent samples that are weighted in order to approximate the target distribution. Another class of generic sampling algorithms are based on the generation of *dependent* samples, such that the marginal distribution of a stochastic process converges to the desired distribution. This idea goes back to the seminal work of Metropolis et al. (1953) and applies also to unnormalized target distributions.

Markov Chain Monte Carlo

Markov Chain Monte Carlo (MCMC) methods construct a Markov Chain $(X_n)_{n\geq 1}$ such that

$$\frac{1}{N}\sum_{n=1}^{N}\psi(x_n) \xrightarrow[N\to\infty]{a.s.} \mathbb{E}_\pi\left[\psi(X)\right],$$

for almost every starting point x_1. The chain is characterized by an initial distribution and a transition kernel $\mathcal{K}$. In order to obtain a convergence to the desired distribution, the transition kernel $\mathcal{K}$ must leave the distribution π invariant:

$$\forall x \in \mathcal{X} \int_{\mathcal{Y}} \mathcal{K}(y,x)\pi(y)\,\mathrm{d}\,y = \pi(x).$$

Broadly speaking this means that if $y \sim \pi(y)$, the marginal distribution stays the same after applying the kernel $\mathcal{K}$ to y. If the chain is aperiodic and irreducible the convergence $\left\|\mathcal{K}^N(x_1,\cdot) - \pi(\cdot)\right\|_{TV} \to 0$ can be established as the iterations of the kernel $N \to \infty$, where $\|\cdot\|_{TV}$ is the total variation norm. Consequently, for any starting point x_1 the chain converges to its stationary distribution.

Under the additional assumption of Harris-recurrence on the generated chain geometric ergodicity can be shown. This means that

$$\|\mathcal{K}^n(x_1, \cdot) - \pi(\cdot)\|_{TV} \leq M(x_1)\rho^n,$$

where $\rho < 1$ is a constant and $M(x_1) > 0$ depends on the starting point x_1. If, moreover, ψ is square-integrable, a central limit theorem holds:

$$\sqrt{N}\left(\frac{1}{N}\sum_{n=1}^{N}\psi(x_n) - \mathbb{E}_\pi\left[\psi(X)\right]\right) \xrightarrow[N\to\infty]{\mathcal{L}} \mathcal{N}(0, \sigma_\psi^2),$$

where

$$\sigma_\psi^2 = \mathrm{Var}\left[\psi(X_1)\right] + 2\sum_{n=2}^{\infty}\mathrm{Cov}[\psi(X_1), \psi(X_n)].$$

See Tierney (1994) for additional information on the theoretical properties. The series of covariances $\mathrm{Cov}[\psi(X_1), \psi(X_n)]$ can be linked to the autocorrelation of the chain. The faster the autocorrelation of the chain decreases, the lower is the asymptotic variance of an estimator based on realizations of the chain.

The Metropolis-Hastings algorithm (Metropolis et al., 1953; Hastings, 1970) is a general purpose algorithm for constructing a Markov chain with the desired properties. We introduce now three different algorithms that are based on the Metropolis-Hasting algorithm: the random walk Metropolis-Hasting algorithm (RWMH), the Metropolis adjusted Langevin algorithm (MALA) and the Hamiltonian Monte Carlo algorithm (HMC). Another approache for constructing Markov Chains with the desired invariant target distribution is, for instance, the Gibbs sampler (Geman and Geman, 1984).

Random walk Metropolis-Hasting

A simple way of constructing an invariant kernel with respect to the distribution π is to use a local random walk proposal. The resulting algorithm is given in Algorithm 5 and can be understood as follows. Starting from x_{s-1} a new state of the chain $x^\star$ is proposed, where the current state x_{s-1} is perturbed by a Gaussian noise. If the new state moves the chain into regions of higher density, the new state is always accepted. If the new state moves the chain in regions of lower density, the new state is accepted with probability proportional to the likelihood ratio of the suggested state to the previous state. Thus, moves that explore the distribution are accepted occasionally. The proposal kernel has essentially two parameters to choose: the scale parameter, called σ^2 here, and the covariance parameters, Σ. The scale parameter σ^2 may be absorbed

in Σ.

Algorithm 5: Random walk Metropolis Hastings algorithm

Input: Initial x_0, target density π

Result: Set of samples $(x_s)_{s\in 1:S}$

for $s = 1$ *to* S **do**
- Sample $x^\star \sim \mathcal{N}(x^\star | x_{s-1}, \sigma^2\Sigma)$
- Calculate $r(x^\star, x_{s-1})) = \frac{\pi(x^\star)}{\pi(x_{s-1})}$
- Sample $u \sim \mathcal{U}[0,1]$
- **if** $u \leq r(x^\star, x_{s-1})$ **then**
  - Set $x_s = x^\star$
- **else**
  - Set $x_s = x_{s-1}$

RWMH relies only on the assumption that it is possible to evaluate the unnormalized target density point wise. If additionally the target density is differentiable, this information may be used to guide the chain towards regions of higher density.

Metropolis adjusted Langevin algorithm

An algorithm based on this approach is the Metropolis adjusted Langevin algorithm (MALA), that is based on the finite time discretization of the Langevin diffusion. See Roberts and Tweedie (1996) for more details. The overdamped Langevin diffusion is given as the stochastic differential equation in continuous time t

$$\mathrm{d}\, X_t = \frac{\sigma^2}{2} \nabla \log\{\pi(X_t)\}\, \mathrm{d}\, t + \sigma\, \mathrm{d}\, B_t,$$

where B_t is a multivariate Brownian motion. The stationary distribution of this process is π. For the purpose of simulation the diffusion is discretized and a Metropolis-Hastings step is introduced in order to correct for the discretization error. This approach leads a Markov Chain with the desired invariant distribution. The resulting proposal kernel is not symmetric. When using a so called preconditioning matrix M, the tuning parameters of the algorithm are the matrix M and the variance parameter

σ^2. The algorithm reads as follows.

Algorithm 6: Metropolis adjusted Langevin Algorithm

Input: Initial x_0, density π, gradient of log density $\nabla_x \log \pi(\cdot)$

Result: Set of samples $(x_s)_{s\in 1:S}$

for $s = 1$ *to* S **do**
- Sample $x^\star \sim \mathcal{N}(x^\star | x_{s-1} + \sigma^2/2M\nabla_x \log \pi(x_{s-1}), \sigma^2 M) =: g(x^\star|x_{s-1})$
- Calculate $r(x^\star, x_{s-1})) = \frac{\pi(x^\star)g(x_{s-1}|x^\star)}{\pi(x_{s-1})g(x^\star|x_{s-1})}$
- Sample $u \sim \mathcal{U}[0,1]$
- **if** $u \leq r(x^\star, x_{s-1})$ **then**
  - Set $x_s = x^\star$
- **else**
  - Set $x_s = x_{s-1}$

Hamiltonian Monte Carlo

Building upon the idea of using gradient information, the Hamiltonian Monte Carlo (HMC) algorithm (Duane et al., 1987) has gained a lot of attention in the last decade. See Neal (2011) for a review. HMC can be understood as a procedure for sampling on the extended target distribution $\mu(x,z) = \pi(x) \times g(z)$, where $g(z) = \mathcal{N}(z|0_d, M)$ is set to a multivariate normal distribution with mean 0 and covariance matrix M. The sampling is achieved by borrowing a concept from physical mechanics: The state of a particle with position x and momentum z that is governed by the Hamiltonian

$$H(x,z) = -\log \mu(x,z)$$

represents realizations of the joint distribution $\mu(x,z)$. The movement of the particle is driven by its Hamilton equations

$$\begin{cases} \frac{dx}{d\tau} = \frac{\partial H}{\partial z} = M^{-1}z, \\ \frac{dz}{d\tau} = -\frac{\partial H}{\partial x} = \nabla_x \log \pi(x), \end{cases}$$

where the derivatives are taken with respect to the fictional time τ. The solution of these differential equations induces a flow $\Phi_\kappa : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^d \times \mathbb{R}^d$, that describes the evolution of the system according to the time κ. Starting from (x_0, z_0) the system is evolved over a time span κ leading to a new state $\Phi_\kappa(x_0, z_0) = (x_\kappa, z_\kappa)$. In practice, the solution is approximated using a numerical integration procedure, called the leapfrog integrator. It depends on a discretization ϵ and the number of steps L. Hence, the integration time becomes $\kappa = \epsilon \times L$ with the numerical flow $\widehat{\Phi}_{\epsilon,L} : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^d \times \mathbb{R}^d$. The numerical integration introduces an error in the total energy of the system. The energy conservation of the system is violated: $\Delta E \neq 0$, where $\Delta E = H(\widehat{\Phi}_{\epsilon,L}(x_0, z_0)) - H(x_0, z_0)$. A Metropolis-Hastings step is used for correcting this error, potentially rejecting a proposed state. The value of z is refreshed by

sampling $z \sim \mathcal{N}(z|0_d, M)$ and thus different energy levels are explored. A Markov Chain with time index s and invariant distribution $\mu(x, z)$ is constructed by iterating over the following steps:

1. Refresh $z_s \sim \mathcal{N}(z|0_d, M)$
2. Generate $(\hat{x}, \hat{z}) = \widehat{\Phi}_{\epsilon,L}(x_s, z_s)$ based on the numerical solutions of the equations of motions starting from (x_s, z_s).
3. Accept $(\hat{x}, \hat{z})$ as the new state (x_{s+1}, z_{s+1}) according to a Metropolis Hasting step based on ΔE_s.

For more details on the algorithm see Chapter 5. The resulting algorithm, with properly tuned parameters ϵ, L, M, typically yields a Markov chain with a low autocorrelation and makes larger moves in the target space. However, tuning of the algorithm is a difficult task and is an active area of research (Wang et al., 2013; Hoffman and Gelman, 2014; Levy et al., 2018).

Tuning and recent developments

All introduced MCMC algorithms hinge on a number of tuning parameters, that are crucial for their performance. In particular, there is a tradeoff between the exploration of the target space, characterized by the distance the chain may travel in one step, and the acceptance rate of the chain. Therefore various adaptation strategies have emerged that try to improve the mixing of the chain by balancing the two.

For RWMH the work of Roberts et al. (1997) showed that tuning the proposal kernel such that an acceptance probability of 0.234 is attained, is optimal as the dimension of the target space goes to infinity. A similar result has been obtained for MALA: the kernel should be tuned such that an acceptance probability of 0.574 is reached (Roberts and Rosenthal, 1998). These results have been extended in Roberts and Rosenthal (2001). For HMC a similar result has been established by Beskos et al. (2013): the user should aim for an acceptance rate of 0.651 when selecting the discretization ϵ.

As the acceptance rate may be only calculated *ex post*, a series of studies investigated an adaptive tuning of the parameters while the chain started exploring the target space. This idea has been initiated by Atchadé and Rosenthal (2005) and extensively studied since. As an alternative to aiming for a certain acceptance rate, it has been suggested to use the expected squared jumping distance (ESJD) as a criterion for the mixing of the chain. Maximizing the ESJD is equivalent to minimizing the first order autocorrelation of the chain. See Pasarica and Gelman (2010); Wang et al. (2013) for two applications of this approach.

A recent line of research consists in so called piecewise deterministic Markov Chains (PDMC). By constructing a continuous time Markov chain that is non-reversible, a faster empirical mixing of the chain may be obtained. The articles by Bierkens et al. (2016); Vanetti et al. (2017); Sherlock and Thiery (2017); Bouchard-Côté et al. (2018) are on the forefront of this development.

Besides the tuning and scalability with dimension of MCMC algorithms, the question of parallelization is of current interest. A promising avenue consists in removing the burnin bias of MCMC chains and the simultaneous run of several short chains (Jacob et al., 2017; Heng and Jacob, 2017). A different problem of practical relevance is the abundance of observed individuals, that make the likelihood costly to evaluate. Computational approaches based on subsampling the observations have been introduced by Welling and Teh (2011) for Langevin type algorithms, by Chen et al. (2014) for HMC algorithms and Pakman et al. (2017) for an application to PDMC. See Alquier et al. (2016a) or Bardenet et al. (2017) for recent theoretical analysis.

Sequential Monte Carlo

Another approach for simulating from a target distribution π consists in combining the ideas of importance sampling and invariant Markov kernels in one single approach that takes advantage of the two. Therefore a sequence of distributions is constructed, that is approached iteratively. This idea has its origins in particle filtering (Pitt and Shephard, 1999) and has been introduced to Bayesian computation by Chopin (2002). For the purpose of Bayesian inference, one might simulate a number of N points from the prior. These points, denoted *particles*, are then moved over a sequence of intermediate distributions towards the posterior. This is achieved via a sequence of reweighting, resampling and propagation steps. A thorough theoretical analysis of particle filters based on the Feynman-Kac formalism is given in Del Moral (2004).

More precisely, we are interested in an evolving sequence of distributions π_t, where the use of iterative importance sampling gives rise to so-called sequential Monte Carlo samplers, see Del Moral et al. (2006).

Simulation form the intermediate distribution is achieved by reframing the simulation problem to the joint distribution $\pi_{0:t}(x_{0:t}) = \tilde{\pi}_{0:t}(x_{0:t})/Z_{0:t}$ defined on $\mathbb{R}^d \times \cdots \times \mathbb{R}^d$. The introduction of artificial backward kernels $\mathcal{L}_s : \mathbb{R}^d \times \mathcal{B}(\mathbb{R}^d) \to [0,1]$ with density $\mathcal{L}_s(x_{s+1}, x_s)$ gives

$$\tilde{\pi}_{0:t}(x_{0:t}) = \tilde{\pi}_t(x_t) \prod_{s=1}^{t-1} \mathcal{L}_s(x_{s+1}, x_s),$$

that admits $\pi_t(x_t)$ by construction. The proposal distribution is given by the sequence of forward kernels $\mathcal{K}_s : \mathbb{R}^d \times \mathcal{B}(\mathbb{R}^d) \to [0,1]$ with density $\mathcal{K}_s(x_{s-1}, x_s)$:

$$\tilde{g}_{0:t}(x_{0:t}) = \tilde{g}_0(x_0) \prod_{s=2}^{t} \mathcal{K}_s(x_{s-1}, x_s).$$

Consequently, the importance weight for sampling on the joint space is

$$\omega_t(x_{0:t}) = \frac{\tilde{\pi}_{0:t}(x_{0:t})}{\tilde{g}_{0:t}(x_{0:t})}. \tag{2.6}$$

The dimension of the target space is extended iteratively such that the weights are given recursively as

$$\omega_t(x_{0:t}) \quad = \quad \omega_{t-1}(x_{0:t-1})\tilde{\omega}_t(x_{t-1}, x_t), \tag{2.7}$$

where the incremental weight is

$$\tilde{\omega}_t(x_{t-1}, x_t) = \frac{\tilde{\pi}_t(x_t)\mathcal{L}_t(x_t, x_{t-1})}{\tilde{\pi}_{t-1}(x_{t-1})\mathcal{K}_t(x_{t-1}, x_t)}. \tag{2.8}$$

Thus, if at time $t-1$ a particle approximation $\left\{x^i_{0:t-1}, w^i_{0:t-1}\right\}_{i\in 1:N}$ of $\tilde{\pi}_{t-1}(x_{0:t-1})$ is available, where $w^i_{0:t-1}$ corresponds to the normalized version of $\omega_{t-1}(x^i_{0:t-1})$, its path is extended via the proposal kernel $\mathcal{K}_t$ and the incremental weight in (2.8). This construction allows the unbiased estimation of the ratio of normalization constants Z_t/Z_{t-1} via

$$\frac{\widehat{Z}_t}{Z_{t-1}} = \sum_{i=1}^{N} w^i_{0:t-1}\tilde{\omega}_t(x^i_{t-1}, x^i_t).$$

If the particles have been resampled at time $t-1$, this expression simplifies and we get

$$\frac{\widehat{Z}_t}{Z_{t-1}} = 1/N \sum_{i=1}^{N} \tilde{\omega}_t(\tilde{x}^i_{t-1}, x^i_t),$$

where $\left\{\tilde{x}^i_{t-1}\right\}_{i\in 1:N}$ is the set of resampled particles. Resampling is used in particle filters in order to remove particles that have negligible weight. In most practical applications the first distribution π_0 is chosen in such a way that it is easy to sample from this distribution and consequently we set $\tilde{\pi}_0 = \tilde{g}_0$. A generic version of the SMC

sampler is given in Algorithm 7.

Algorithm 7: Generic SMC sampler algorithm

Input: Sequence of unnormalized distributions $\tilde{\pi}_0, \cdots, \tilde{\pi}_t, \cdots, \tilde{\pi}_T$ and propagation kernels $\mathcal{K}_t$

Result: Set of weighted samples $\left\{x_t^i, w_t^i\right\}_{i\in 1:N}$ and estimations $\widehat{\frac{Z_t}{Z_{t-1}}}$ for $t \in 1:T$

Initialization : $t = 1$

for $i = 1$ *to* N **do**
  Sample $x_0^i \sim \tilde{\pi}_0$
  Weight the particle $w_0^i \propto \frac{\tilde{\pi}_1(x_0^i)}{\tilde{\pi}_0(x_0^i)}$
Calculate $\widehat{\frac{Z_1}{Z_0}} = 1/N \sum_{i=1}^N \frac{\tilde{\pi}_1(x_0^i)}{\tilde{\pi}_0(x_0^i)}$

Iteration :

for $t = 2$ *to* T **do**
  Resample particles $\left\{x_{t-1}^i, w_{t-1}^i\right\}_{i\in 1:N}$ and get $\left\{\tilde{x}_{t-1}^i\right\}_{i\in 1:N}$;
  for $i = 1$ *to* N **do**
    Move the particle $x_t^i \sim \mathcal{K}_t(\tilde{x}_{t-1}^i, dx)$
  for $i = 1$ *to* N **do**
    Weight the particle $w_t^i \propto \tilde{\omega}_t(x_{t-1}^i, x_t^i)$
  Calculate $\widehat{\frac{Z_t}{Z_{t-1}}} = 1/N \sum_{i=1}^N \omega_t(x_t^i)$

The use of SMC samplers for Bayesian computation has several appealing properties compared to MCMC based samplers: (a) The normalization constant of the model is estimated on the fly and thus available for model choice (Zhou et al., 2016). (b) Using a large number of particles provides some robustness to multimodality (Schweizer, 2012a). (c) SMC samplers are highly parallelizable (Murray et al., 2016). (d) The cloud of particles provides information that may be helpful for adapting the sampler (Fearnhead and Taylor, 2013). Recent developments in the field of SMC samplers consist in the study of the theoretical properties of adaptation (Beskos et al., 2016), the estimation of the variance of the sampler (Lee and Whiteley, 2018) or the use of quasi-Monte Carlo (Gerber and Chopin, 2015), see also the following section.

2.3 Quasi Monte Carlo

As we have seen, most random number generators start from generating a uniform sequence. The uniform sequence may be used for calculating an approximation of the integral of the function $\phi : [0,1]^d \to \mathbb{R}$ defined as $I = \int_{[0,1]^d} \phi(u)\, \mathrm{d}\, u$ via

$$\widehat{I}_N = \frac{1}{N} \sum_{n=1}^{N} \phi(u_n), \tag{2.9}$$

where the function ϕ potentially encompasses a transformation of the uniform sequence to a different space of interest.

A commonly used approach for reducing the variance of the integration is through stratification. Stratification divides a uniform hypercube into a number of *strata* and then proceeds by sampling from the *strata*. This approach covers the uniform hypercube more evenly and consequently leads to a reduced error of the integration.

2.3.1 Halton sequences

A more sophisticated approach consists in the construction of deterministic sequences, also called low discrepancy or quasi-Monte Carlo sequences. We illustrate this approach with the construction of Halton sequences and follow the outline of Dick et al. (2013). Let $i \in \mathbb{N}$. Then i can be expressed in base b as

$$i = \sum_{a=1}^{\infty} i_a b^{a-1},$$

where $i_a \in \{0, 1, \cdots, b-1\}$. As an example we represent the sequence of integers $0, 1, 2, 3, 4, \cdots$ in base $b = 2$. This yields $0_2, 1_2, 10_2, 11_2, 100_2, \cdots$. We define the radical inverse function $\nu_b(i)$ as inversion of the integer representation of i in base b. It is defined as

$$\nu_b(i) := \sum_{a=1}^{\infty} \frac{i_a}{b^a}.$$

The radical inverse function reflects this representation to the decimal representation: $0, 0.1_2, 0.01_2, 0.11_2, 0.001_2, \cdots$. Transforming this sequence back to base 10 representation yields $0, 0.5, 0.25, 0.75, 0.125 \cdots$. Continuing this construction yields a sequence that fills up the interval $[0, 1]$. The Halton sequence is based on this idea. Let $p_1, p_2, \cdots, p_d$ be the first d prime numbers. The Halton sequence $u_0, u_1, \cdots$ in dimension d is given as

$$u_i = (\nu_{p_1}(i), \nu_{p_2}(i), \cdots, \nu_{p_d}(i)).$$

We illustrate the Halton sequence as well as a pseudo random sequence on $[0, 1]^2$ in Figure 2.1.

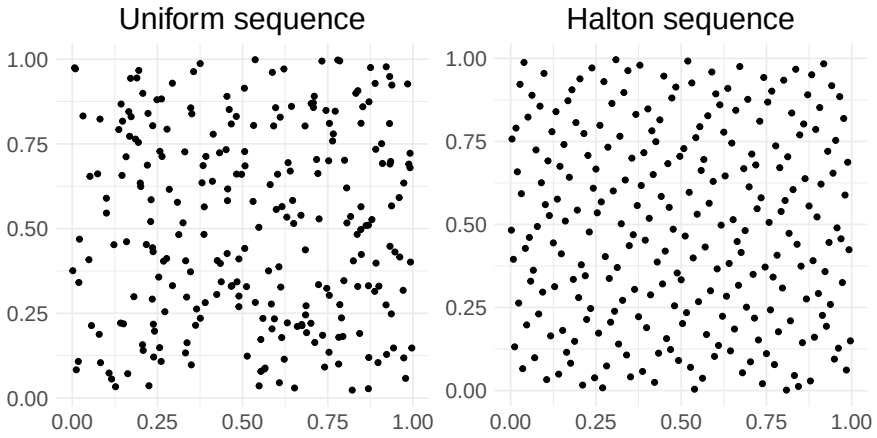


FIGURE 2.1: Uniform sequence (left) and Halton sequence (right) of length $N = 256$ on $[0, 1]^2$.

The Halton sequence is only one possible way of constructing sequences that cover $[0,1]^d$ more evenly than random sampling. Other sequences that achieve the same goal are, for example, the Faure sequence, the Sobol sequence or digital nets. The quality of the coverage of the deterministic sequence can be assessed by the discrepancy of the sequence, which we discuss now.

2.3.2 Convergence of QMC sampling

The general notion of discrepancy of a given sequence $u_1, \cdots, u_N$ is defined as follows:

$$D(u_{1:N}, \mathcal{A}) := \sup_{A \in \mathcal{A}} \left| \frac{1}{N} \sum_{n=1}^{N} \mathbb{1}_{\{u_n \in A\}} - \lambda_d(A) \right|,$$

where $\lambda_d(A)$ is the volume (Lebesgue measure on $\mathbb{R}^d$) of A and $\mathcal{A}$ is a set of measurable sets. When we fix the sets $A = [0, \mathbf{b}] = \prod_{i=1}^{d} [0, b_i]$ with $0 \leq b_i \leq 1$ as the products set of intervals anchored at 0, we obtain the star discrepancy

$$D^*(u_{1:N}) := \sup_{[0,\mathbf{b}]} \left| \frac{1}{N} \sum_{n=1}^{N} \mathbb{1}_{\{u_n \in [0,\mathbf{b}]\}} - \lambda_d([0, \mathbf{b}]) \right|.$$

The star discrepancy can be used in order to establish an upper bound on the error of the integral approximation. We define this error as

$$e(\phi, N) = \left| I - \widehat{I}_N \right|.$$

We have seen in Section 2.2.1 that using a uniform random sequence we get

$$\sqrt{\mathbb{E}\left[e(\psi, N)^2\right]} = \mathcal{O}(N^{-1/2}),$$

supposing that the random variable $\phi(U)$ is square integrable. When using a low discrepancy sequence, the error can be assessed through the Koksma-Hlawka inequality:

$$e(\psi, N) \leq D^*(u_{1:N}) V(\phi), \tag{2.10}$$

where $V(\phi)$ is the variation of the function ϕ in the sense of Hardy and Krause. This quantity is closely related to the smoothness of the function. The presented Halton sequence leads to a discrepancy of $\mathcal{O}((\log N)^d / N)$. Consequently, Halton sequences are asymptotically more efficient than uniformly random sequences for integration, under the condition that $V(\phi) < \infty$.

2.3.3 Randomized quasi-Monte Carlo

One major drawback of QMC sampling is the fact that it does not come with an easy way of assessing the error of the approximation as $V(\phi)$ is hard to compute. By reintroducing randomness in a QMC sequence while preserving the structure of the point

set the uncertainty may be assessed by repeated sampling. This idea is called randomized quasi-Monte Carlo (RQMC). These sequences may be constructed, for example, by randomly shifting the entire sequence (Cranley and Patterson, 1976): Let $v \sim \mathcal{U}[0,1]^d$ and let $(u_n)_{1 \leq n \leq N}$ be a QMC sequence. Then the sequence

$$\tilde{u}_n := u_n + v \mod 1,$$

where $x \mapsto x \mod 1$ is the componentwise modulo function, is a QMC sequence with probability 1. Moreover, the sequence is marginally uniformly distributed. Thus, when using the points $\tilde{u}_n$ for integration, the estimates of (2.9) are unbiased and we can use our probabilistic toolbox in order to asses the error.

A more involved approach consists in scrambled nets, introduced by Owen (1997). By introducing randomness directly in the construction of the sequence it is possible to obtain error rates of $\mathcal{O}((\log N)^{d-1} N^{-3})$. This result relies on additional smoothness assumptions of the function ϕ.

A more recent result by Gerber (2015) obtains a rate in $\mathcal{O}(N^{-2})$ under the same smoothness assumptions as Owen (1997). When relaxing these assumptions to square integrability rates of $o(N^{-1})$ are achievable. Thus, RQMC integration is always at least as efficient as Monte Carlo integration.

2.3.4 Using low discrepancy sequences in statistics

The use of QMC in statistics relies on the statisticians ability to transform a sequence on $[0,1]^d$ to the distribution of interest while preserving the low discrepancy structure of the initial points. This may be achieved by rewriting the integral of interest as an expectation with respect to the uniform distribution:

$$\mathbb{E}_X [\psi(X)] = \mathbb{E}_U [\psi(\Gamma(U))] .$$

Thus $\psi \circ \Gamma = \phi$, using our initial test function, and one has to make sure that this function satisfies the smoothness properties specified above. A generic approach is the use of the inverse Rosenblatt transform (Rosenblatt, 1952), that generalizes the inverse cdf to the multivariate setting.

Until today, the use of QMC has been extensively studied in financial mathematics (Lemieux and L'Ecuyer, 2001; L'Ecuyer, 2009; Glasserman, 2013), but its use in classical statistics remains rather limited. In fact, QMC may be used in combination with importance sampling. This approach can lead to estimators with a substantially reduced variance, see for example Gerber and Chopin (2015); Chopin and Ridgway (2017). Current research focuses on QMC applications for MCMC methods, see for example Owen and Tribble (2005); L'Ecuyer et al. (2008); L'Ecuyer and Sanvido (2010); Chen et al. (2011); Schwedes and Calderhead (2018).

2.3.5 Central limit theorems for QMC

In various practical applications it may not be possible to use a QMC sequence for all dimensions of the problem of interest. This might be due to the fact that the dimension of the problem is too large, or that for parts of the problem it is not possible to transform the QMC sequence to the space of interest. In such setting we may use a mixed sequence, that is composed in part of a MC sequence of dimension s, denoted by v_n and in part of a QMC sequence of dimension $d-s$, denoted by u_n. The joint sequence $(v_n, u_n) = r_n$ is termed mixed sequence. The mixed sequence leads the estimator $\widehat{I}_N = 1/N \sum_{n=1}^{N} \phi(r_n)$.

The rate of convergence of an integration based on a mixed sequence will naturally be dominated by the slowest part, i.e. the MC part. In this setting it is possible to establish an asymptotic variance reduction based on a central limit theorem. In particular for bounded ϕ and some additional assumptions Ökten et al. (2006) obtain that

$$\sqrt{N}(\widehat{I}_N - I) \xrightarrow[N\to\infty]{\mathcal{L}} \mathcal{N}(0, \tilde{\sigma}^2),$$

where $\tilde{\sigma}^2$ denotes the asymptotic variance. In particular $\tilde{\sigma}^2 \leq \sigma^2$, where σ^2 denotes the asymptotic variance of the central limit theorem using for r_n a pure MC sequence.

2.4 Stochastic approximation

In its initial version stochastic approximation (Robbins and Monro, 1951) was conceived as an approach for finding the roots of a monotone function $H : \lambda \mapsto H(\lambda)$ that is corrupted by noise. We observe $F(\lambda, \xi)$, where $\lambda \in \Lambda \subset \mathbb{R}$ and ξ is a noise term perturbing the function F such that $\forall\lambda\ \mathbb{E}[F(\lambda, \xi)] = H(\lambda)$. $F(\lambda, \xi)$ is assumed to be upper and lower bounded for all λ by a constant $\pm C$, ξ-almost surely. We want to find $\lambda^\star$ such that $\mathbb{E}[F(\lambda^\star, \xi)] - c = 0$ for a given value of $c \in [-C, C]$. This problem may be solved with an iterative algorithm. If after t iterations $F(\lambda_t, \xi) - c < 0$, this suggests increasing λ_{t+1} compared to the previous iteration and *vice versa*. Consequently, one iterates

$$\lambda_{t+1} = \lambda_t - \alpha_t(F(\lambda_t, \xi) - c),$$

with a decreasing step size schedule α_t until $|F(\lambda_t, \xi) - c| < \epsilon$, for a small tolerance level ϵ. An approach to guarantee convergence is to set the stepsizes α_t such that $\forall t, \alpha_t > 0$, $\sum_{t=1}^{T} \alpha_t = \infty$ and $\sum_{t=1}^{T} \alpha_t^2 < \infty$ when $T \to \infty$. Then we obtain the following convergence result:

$$\lambda_t \xrightarrow[t\to\infty]{\mathbb{P}} \lambda^\star.$$

This approach may also be used for optimizing a convex function $H(\lambda)$, i.e. finding

$$\lambda^\star \in \arg\min_{\lambda\in\Lambda} H(\lambda).$$

If H is differentiable this amounts to finding points where the derivative of the function becomes 0 and hence by searching for values of λ such that

$$\frac{\partial H(\lambda)}{\partial \lambda} = 0.$$

Under appropriate conditions and if the measure of ξ does not depend on λ, $\frac{\partial F(\lambda,\xi)}{\partial \lambda}$ is an unbiased estimator of $\frac{\partial H(\lambda)}{\partial \lambda}$. This leads to the idea of stochastic gradient descent (SGD) given by the update recursion

$$\lambda_{t+1} = \lambda_t - \alpha_t \frac{\partial F(\lambda_t, \xi)}{\partial \lambda}.$$

In case an unbiased estimator of the gradient is not available, the gradient may be estimated using finite differences (Kiefer and Wolfowitz, 1952).

Standard assumptions for deriving convergence guarantees for stochastic gradient descent are Lipschitz-continuity of the gradients and convexity of the function H, see for example Bottou et al. (2016). Under additional assumptions like strong convexity of H a linear rate of convergence in t of the gap $H(\lambda_t) - H(\lambda^\star)$ may be established.

In the field of statistics and machine learning methods based on stochastic approximation are ubiquitous: They provide the algorithmic tool for maximum likelihood estimation and empirical risk minimization. Much work has been done on speeding up the convergence in convex and non-convex settings compared to the basic gradient descent schemes. Some of these approaches are, for example, the introduction of a momentum term that adds inertia to the gradient or the reduction of variance of the gradient estimator, see the work of Nesterov (1983); Johnson and Zhang (2013); Defazio et al. (2014). Moreover, the automatic choice of the stepsizes α_t is an active area of research (Duchi et al., 2011; Kingma and Ba, 2015). Recently the link to continuous time differential equations yielded additional insight on the acceleration of stochastic gradient descent (Wibisono et al., 2016).

2.5 Variational Inference

The previous sections exposed different methods that can be used when approximating a posterior distribution through sampling. Another approach, that gained a lot of interest in recent years consists in building an approximation to the posterior distribution through a different class of distributions. This is achieved by solving an optimization problem, often using the techniques introduced in the previous section. One of such approaches is called variational inference and it consists in minimizing the Kullback-Leibler divergence (KL) between a family of distributions (the variational family) and the posterior distribution. As before we denote the observed data by y, the latent variable x and the posterior distribution $\pi(x|y) \propto p(y|x)p_0(x)$. The variational distribution is denoted $q(x) \in \mathcal{Q}$, where $\mathcal{Q}$ denotes the variational family. The

problem is defined as

$$q^\star(x) = \arg\min_{q\in\mathcal{Q}} \mathrm{KL}(q(x)||\pi(x|y)),$$

where $\mathrm{KL}(q(x)||\pi(x|y)) = \mathbb{E}_q\left[\log q(x) - \log \pi(x|y)\right]$ denotes the Kullback-Leibler divergence. Choosing a class of distributions $\mathcal{Q}$ thus amounts to controlling the complexity of the problem and the quality of the approximation. The KL may be rewritten in terms of the evidence of the model Z_π and a second term named evidence lower bound (ELBO):

$$\mathrm{KL}(q(x)||\pi(x|y)) = \log Z_\pi - \mathcal{L}(q),$$

where $\mathcal{L}(q) = \mathbb{E}_q\left[\log p(y|x)p_0(x) - \log q(x)\right]$ is the ELBO. Consequently, maximizing the ELBO is equivalent to minimizing the KL, as the evidence does not depend on the variational family.

2.5.1 Mean Field Variational Inference

A common choice of the variational family $\mathcal{Q}$ is the class of distributions with independent components x_j. This approach is called mean field variational inference and it relies on the factorization

$$q(x) = \prod_{j=1}^{m} q_j(x_j),$$

where the m factors are independent of each other. The advantage of this approach is that the optimal factors are found without supposing a specific form of $q_j(x_j)$. In particular one obtains

$$q_j^\star(x_j) \propto \exp\left(\mathbb{E}_{i\neq j}\log\{p(y|x)p_0(x)\}\right),$$

where $\mathbb{E}_{i\neq j}$ denotes the expectation with respect to $q(x)$, omitting the factor $q_j(x_j)$, see Jordan et al. (1999); Bishop (2006) for more details. In conditionally conjugate exponential family models the expectations may be evaluated analytically (Hoffman et al., 2013). By cycling through the expectations over all j the ELBO is maximized. This algorithm is called coordinate ascent variational inference (CAVI), see Blei et al. (2017). However, there is a major shortcoming of this approach: The class of models to which this approach applies is rather restrictive.

2.5.2 Monte Carlo Variational Inference

It is possible to restrict $\mathcal{Q}$ to a specific parametric class of distributions, for example the class of multivariate Normal distributions $\mathcal{N}(\mu, \Sigma)$, parameterized by a mean $\mu \in \mathbb{R}^d$ and a covariance matrix Σ, belonging to the space of positive symmetric matrices on $\mathbb{R}^{d\times d}$. Then, the variational approach boils down to finding the parameterization of

the Gaussian $\mathcal{N}(\mu^\star, \Sigma^\star)$ that approximates best $\pi(x|y)$ and the problem becomes

$$\mu^\star, \Sigma^\star \in \arg\min_{\mu,\Sigma} \mathrm{KL}(\mathcal{N}(x|\mu,\Sigma)||\pi(x|y)).$$

The expectations involved in the formulation of the ELBO are expectations with respect to the variational class $\mathcal{Q}$. With a large class of potential parametric families chosen by the user, it is often possible to approximate the expectations through Monte Carlo sampling when closed forms are not available. This leads to the idea of Monte Carlo variational inference, that has gained momentum in recent years. It relies on an approximation of the involved expectations and its gradients via sampling. The optimization of the ELBO is done via a gradient ascent scheme on the parameter $\lambda \in \Lambda$ of the variational family using the estimated gradients. In our Gaussian example we have $\lambda = (\mu, \Sigma)$ and thus $q_\lambda(x) \mathrel{\widehat{=}} \mathcal{N}(x|\mu,\Sigma)$.

This links the problem of finding good approximations to the literature of stochastic optimization, introduced in the previous section. The direct differentiation of the ELBO with respect to λ is not possible, as the measure of the expectation depends on this parameter. The two major approaches for overcoming this issue are the score function estimator (Ranganath et al., 2014) and the reparameterization estimator (Kingma and Welling, 2014).

The score function gradient (also called REINFORCE gradient (Williams, 1992)) expresses the gradient as expectation with respect to $q_\lambda(x)$ and is given by

$$\nabla_\lambda \mathcal{L}(\lambda) = \mathbb{E}_{q_\lambda(x)} \left[\nabla_\lambda \log q_\lambda(x) \left\{ \log[p(y|x)p_0(x)] - \log q_\lambda(x) \right\} \right]. \tag{2.11}$$

A gradient estimator is obtained by approximating the expectation with independent samples from the variational distribution $q_\lambda(x)$. This estimator is fairly generic, and applies to continuous and discrete variational distributions.

Another approach is based on the reparametrization trick, where the distribution over x is expressed as a deterministic transformation of a different distribution over a noise variable ε, hence $x = g_\lambda(\varepsilon)$ where $\varepsilon \sim p(\varepsilon)$. Using the reparameterization the ELBO is expressed as expectation with respect to $p(\varepsilon)$ and the derivative is moved inside the expectation:

$$\nabla_\lambda \mathcal{L}(\lambda) = \mathbb{E}_{p(\varepsilon)} \left[\nabla_\lambda \log\{ p(y|g_\lambda(\varepsilon)) p_0(g_\lambda(\varepsilon)) \} - \nabla_\lambda \log q_\lambda(g_\lambda(\varepsilon)) \right]. \tag{2.12}$$

The expectation is approximated using the average of independent samples from the base measure $p(\varepsilon)$. This estimator is restricted to distributions over continuous variables that allow for a reparameterization, differentiable in λ.

Using one of the gradient estimators denoted by $\hat{g}_N(\lambda) \approx \nabla_\lambda \mathcal{L}(\lambda)$, where N is the number of samples, the ELBO can then be optimized by stochastic optimization. This is achieved by iterating the stochastic gradient descent updates with decreasing step

size schedule α_t:

$$\lambda_{t+1} = \lambda_t + \alpha_t \hat{g}_N(\lambda_t). \tag{2.13}$$

The convergence of the gradient ascent scheme in (4.4) tends to be slow when gradient estimators have a high variance. Therefore, various approaches for reducing the variance of the gradient estimators exist; e.g. control variates, Rao-Blackwellization and importance sampling. These variance reduction schemes must often be tailored to the problem at hand and therefore seeking more general solutions remains an active area of research.

Other recent developments consists for example in the application of variational inference to stochastic differential equations (Ryder et al., 2018), or to implict models (Tran et al., 2017a,b). Another subject of current interest is the derivation of theoretical guarantees for variational inference, that have been lacking despite the practical success of variational inference. See for instance Alquier et al. (2016b); Germain et al. (2016); Wang and Blei (2018); Chérief-Abdellatif and Alquier (2018) for some recent work.

2.6 Summary

2.6.1 Summary of our work on quasi-Monte Carlo and approximate Bayesian computation

As we have seen in our introduction to ABC, there are in general two sources of randomness when generating samples x_n from the approximate posterior: (a) the randomness that comes from generating samples from the prior distribution; (b) the randomness that comes from generating pseudo-observations from the model. We suggest to use QMC points for the prior distribution, as it is often possible to transform an uniform sequence of points to the prior distribution of interest. We show that this approach reduces effectively the variance of estimators based on the approximate posterior samples. The remaining and dominating part of the variance comes form the model.

In order to illustrate this, suppose we are interested in an estimator of the normalizing constant $\int_{\mathcal{X}} \pi_\epsilon(x|y^\star)\,\mathrm{d}\,x = Z_\epsilon$ approximated by

$$\hat{Z}_{N,M} = \frac{1}{NM} \sum_{i=1}^{N} \sum_{j=1}^{M} \mathbb{1}_{\{\delta(y_{i,j}, y^\star) \leq \epsilon\}}.$$

Here, for every x_i simulated from the prior we generate M observations of $y_{i,j}$ for $j = 1, \cdots, M$ from the model in order to estimate $\mathbb{P}_x(\delta(y, y^\star) \leq \epsilon)$. When calculating the variance of $\hat{Z}_{N,M}$ we get

$$\mathrm{Var}\left[\hat{Z}_{N,M}\right] = \underbrace{\mathrm{Var}\left[\mathbb{E}\left[\hat{Z}_{N,M}|x_{1:N}\right]\right]}_{=\mathcal{O}(\frac{1}{N})} + \underbrace{\mathbb{E}\left[\mathrm{Var}\left[\hat{Z}_{N,M}|x_{1:N}\right]\right]}_{=\mathcal{O}(\frac{1}{NM})},$$

using the decomposition of variance formula. After adjusting for the cost of generating M samples for every x_i we get

$$M \times \text{Var}\left[\hat{Z}_{N,M}\right] = M \times C_1/N + C_2/N,$$

and thus the cost adjusted variance increases linearly in M, leading to an optimal value of $M = 1$ (Bornn et al., 2015).

Lets assume now, that $x_i = \Gamma(u_i)$, where u_i is an RQMC sequence and $\Gamma(\cdot)$ transforms the uniform sequence to the prior distribution. Under additional regularity conditions we obtain

$$\text{Var}\left[\hat{Z}_{N,M}\right] = \underbrace{\text{Var}\left[\mathbb{E}\left[\hat{Z}_{N,M}|u_{1:N}\right]\right]}_{=o(\frac{1}{N})} + \underbrace{\mathbb{E}\left[\text{Var}\left[\hat{Z}_{N,M}|u_{1:N}\right]\right]}_{=\mathcal{O}(\frac{1}{NM})}.$$

The first part becomes negligible compared to the second term and we obtain a reduced variance using RQMC sequences for the prior. This result implies that values $M > 1$ may be used as the first term becomes negligible, see Proposition 1 for RQMC sequences and Proposition 2 for QMC sequences in Chapter 3.

Moreover, we establish that a central limit theorem holds for both QMC and RQMC sequences for the construction of prior samples and for M fixed, i.e.

$$\sqrt{N}\left(\hat{Z}_{N,M} - Z_\epsilon\right) \xrightarrow[N\to\infty]{\mathcal{L}} \mathcal{N}(0, \tilde{\sigma}^2),$$

following the approach of Ökten et al. (2006). In particular, $\tilde{\sigma}^2 \leq \sigma^2$, where σ^2 corresponds to the variance of a standard Monte Carlo approach, see Theorem 4. Our results also hold for more general type of importance sampling estimators of the autonormalized form: When interest lies in calculating expectations of $\psi(x)$ and we denote the weight by $w(x) = \mathbb{1}_{\{\delta(y,y^\star)\leq\epsilon\}} p_0(x)/g(x)$ then

$$\mathbb{E}_{\pi_\epsilon(x|y)}\left[\psi(X)\right] \approx \frac{\sum_{i=1}^N w(x_i)\psi(x_i)}{\sum_{i=1}^N w(x_i)}$$

may be used as approximation. This estimator also benefits from the QMC variance reduction as can be shown by either decomposing the variance or using a central limit theorem.

Another contribution of our work is the use of QMC in a sequential setting. Therefore we proceed in the spirit of adaptive importance sampling (Oh and Berger, 1992) and adapt a proposal distribution $g_t(x)$ over iterations while decreasing the acceptance threshold ϵ_t. More precisely, we fit a Gaussian distribution g_t to the set of weighted $\{x_i^{t-1}, w(x_i^{t-1})\}$ of the previous iteration. Thereby, the proposal distribution concentrates in regions of high posterior probability as ϵ_t is decreased. We choose an adaptive Gaussian proposal as this approach allows a transformation of QMC points to the space of interest and it worked well practice. We illustrate the advantages of this

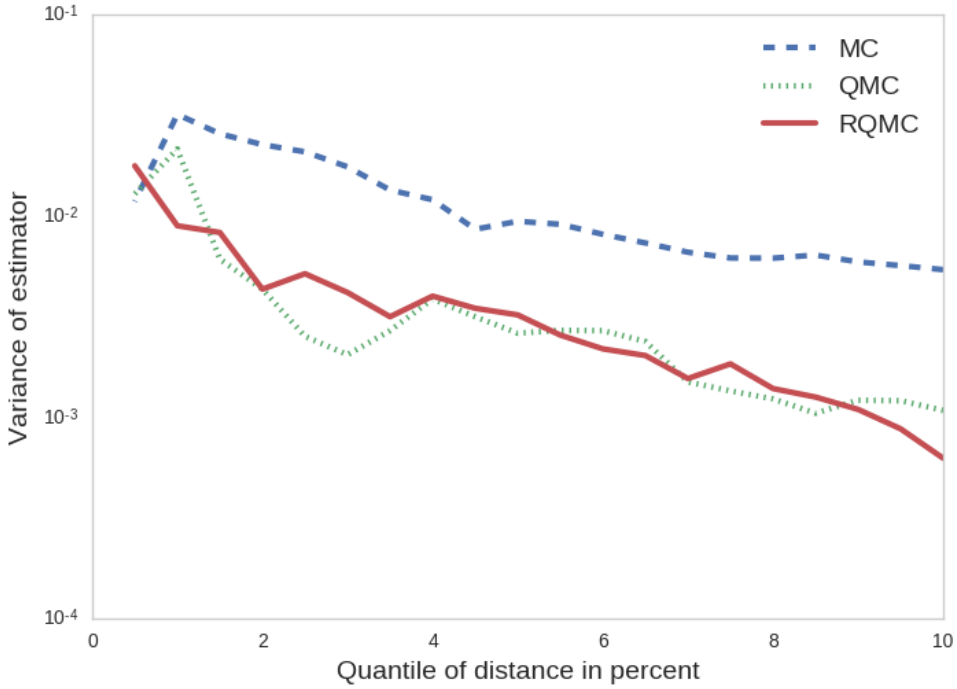


FIGURE 2.2: Variance of the posterior mean estimator for the Lotka–Volterra model. The plots are based on 50 repetitions of 10^5 simulations from the prior and the model. The accepted observations correspond to quantiles based on the smallest distances $\delta(y_n, y^\star)$.

approach through an extensive simulation study. One of the examples that we use is the inference of the parameters in a Lotka-Volterra model. In this setting we observe two noisy time series that represent the interaction of a predator and prey population. Interest lies in the posterior distribution of the 3 dimensional parameters that govern the rate equation. We simulate from a uniform prior over the parameters, generate noisy time series yielding the observations y_i and then vary the acceptance threshold for the distance of simulated data to observed values. As the acceptance threshold approaches zero, the variance of the estimator of the cumulative mean increases, as shows Figure 2.2. Using (R)QMC leads to a substantial variance reduction of the estimator, as predicted by our theoretical analysis. Our approach is easily implementable in existing ABC approaches and therefore should be of high practical interest.

2.6.2 Summary of our work on quasi-Monte Carlo and variational inference

As we have seen in the section on Monte Carlo variational inference, an estimator for the gradient of the ELBO may be constructed by either sampling from the variational family and using the score function estimator, see (4.2), or by sampling from the reparameterization and using the estimator based on (4.3). A major problem of this

implementation is the high variance of these estimators when using too small a sample size for their construction. Therefore, various variance reduction schemes have been suggested, see for example Ranganath et al. (2014); Ruiz et al. (2016a); Miller et al. (2017); Roeder et al. (2017). However, all these approaches do not improve the $1/N$ rate of convergence of the Monte Carlo estimator based on N samples. As it turns out, the majority of commonly used variational families for the estimator of (4.2) and reparameterizations for the estimator of (4.3) can be reframed to the transformation of a uniform sequence of random variables on $\mathcal{U}[0,1]^d$. By using a smooth mapping $\Gamma : [0,1]^d \to \mathbb{R}^d$ in the same spirit as in our work on ABC, we can obtain a variance reduction for both gradient estimators by using an RQMC sequence in place of an initial random uniform sequence. The estimator, denoted by $\hat{g}_N(\lambda)$, where λ is the parameter of the variational family, inherits an improved rate of convergence such that $\text{Var}\,[\hat{g}_N(\lambda)] \leq \mathcal{O}(N^{-2})$, under some smoothness assumptions.

In our work we show that this approach is beneficial from several points of view. First, the approach leads to an improved stochastic gradient ascent as we gain a factor of $1/N$ in the standard bounds when using a fixed learning rate. In the case of Lipschitz continuous gradients we get, as $T \to \infty$,

$$\frac{1}{T}\sum_{t=1}^{T} \mathbb{E}\|\nabla \mathcal{L}(\lambda_t)\|^2 \leq \mathcal{O}(N^{-2}),$$

where the omitted constant in the upper bound depends on the Lipschitz constant L, a fixed learning rate α and universal upperbound on the variance of $\hat{g}_N(\lambda)$ for all λ. See Theorem 5 for more details. This compares favorably with the result in the Monte Carlo setting, where the dependence on the sample size is $1/N$.

In the case of a strongly convex function $\mathcal{L}$ we can state a result in terms of the gap between the function value of the current iteration and the true optimizer $\lambda^\star$. We get as $T \to \infty$

$$|\mathcal{L}(\lambda^\star) - \mathbb{E}\mathcal{L}(\lambda_T)| \leq \mathcal{O}(N^{-2}),$$

where the constant now depends additionally on the strong convexity parameter. Here again we gained a factor of $1/N$ compared to the case of Monte Carlo sampling, see Theorem 6 for more details.

Second, it is possible to take advantage of the $1/N$ speed up by increasing the sample size over iterations and obtain a faster rate of convergence. Assume that the estimator $\hat{g}_{N_t}(\lambda_t)$ is now based on an increasing sample size $N_t = 1/\lceil \tau^t \rceil$, where $\tau =: 1/\xi > 1$ is a geometric rate. Under these assumptions we obtain that

$$|\mathcal{L}(\lambda^\star) - \mathbb{E}\mathcal{L}(\lambda_T)| \leq \omega \xi^{2t},$$

where ω is a constant. Once more, this result compares favorably with the Monte Carlo setting where we would otherwise obtain ξ^t and hence a slower rate. See Theorem 7 for more details.

Third, from an experimental point of view the variance reduction allows to use a larger learning rate as suggested by an adaptive stochastic gradient algorihm like Adagrad (Duchi et al., 2011). Hence, the algorithm makes larger steps in the parameter space and converges faster. Here we illustrate this on a Bayesian logistic regression in dimension 31, based on the breast cancer dataset available in the python library scikit-learn. See also the work by Jaakkola and Jordan (2000) for a different computational approach for variational inference in this model. We use the black box variational inference approach and parametrize the variational distribution in terms of its mean $\mu \in \mathbb{R}^d$ and a diagonal covariance matrix $\operatorname{diag} \Sigma \in \mathbb{R}^{d+}$.

The Adagrad optimization starts with an initial learning rate of 0.1. We compare our RQMC approach based on 30 samples for estimating the gradient with two MC approaches based on 30 and 900 samples respectively. The MC gradient estimators use as a control variate the mean and variance of the current parameterization. The approach based on RQMC converges faster in terms of the ELBO as illustrates Figure 2.3. In the main body of our paper we illustrate the advantages of our approach with three additional numerical examples. Our approach is rather general and not restricted to variational inference. It can be used in different settings where a gradient estimator may be built using QMC sampling.

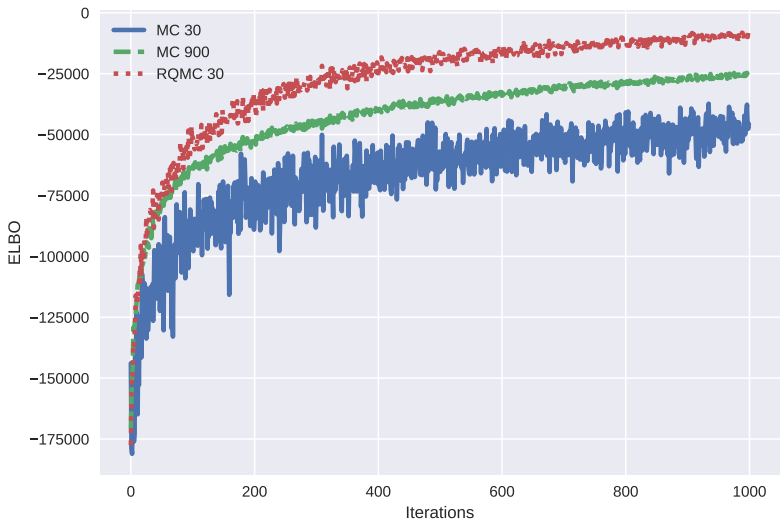


FIGURE 2.3: Evidence lower bound as a function of iterations for the variational approximation of a Bayesian logistic regression. The two Monte Carlo gradient estimators are based on $N = 30$ and $N = 30^2$ samples. The RQMC gradient estimator is based on $N = 30$ samples. The ELBO is estimated using 1000 samples.

2.6.3 Summary of our work on adaptive tuning of Hamiltonian Monte Carlo within sequential Monte Carlo

The last Chapter of this thesis presents our work on the adaptive tuning of HMC within an SMC sampler. We consider the problem of sampling from the posterior distribution $\pi_T(x) = \pi(x|y)$ while starting with a cloud of N generated particles from the prior distribution $\pi_0(x) = p_0(x)$. A series of intermediate distributions $\pi_t(x)$ is constructed using a geometric bridge:

$$\pi_t(x) \propto p_0(x)^{1-\beta_t} \times \pi(x|y)^{\beta_t} = p_0(x)p(y|x)^{\beta_t}$$

with an increasing temperature schedule $0 = \beta_0 < \cdots < \beta_t < \cdots < \beta_T = 1$. The particles are moved over the sequence of intermediate distributions using an alternation of reweighting, resampling, and propagation steps.

The rationale behind our idea to use HMC within SMC samplers is the fact that HMC kernels scale better with the dimension than other Metropolis-Hasting type kernels (Beskos et al., 2013; Mangoubi and Smith, 2017). The scalability of the propagation kernel impacts to a large extent the scalability of SMC samplers. At the same time the cloud of particles provides information that allows to learn reasonable parameters for the proposal kernel. In this sense the combination of HMC and SMC is mutual beneficial.

In this static Bayesian SMC sampler there are typically three design choices to make: (a) the choice of the sequence of intermediate distributions or equivalently the choice of the temperatures β_t; (b) the repetition of the number of propagation steps in order to assure a proper exploration of the distributions of interest; (c) the choice of the tuning parameters of the propagation kernel. In our work we illustrate the importance of carefully approaching (a), (b) and (c). Our approach of handling (a) is based on the work of Zhou et al. (2016). Concerning (b) we suggest a heuristic for choosing the number of propagation steps based on the autocorrelation of the propagation kernel, thereby providing a guideline for practical applications. With respect to (c) we adapt the idea of Fearnhead and Taylor (2013) to the tuning of HMC propagation kernels within SMC and develop a new method that is computational more expensive but also more robust.

Our approach is based on a trial run, where we test different combinations of the parameters ϵ, L and evaluate their performance. Our developed algorithm is given in Algorithm 8. We start with the cloud of resampled particles, obtained at the previous iteration. As a result of our algorithm we obtain a set of values (ϵ, L) that yield a high expected squared jumping distance for the propagation kernel that is invariant to the current distribution π_{t-1}. The algorithm proceeds as follows: We first sample N different values of the kernel parameters and assign them to the different particles. Then we let the numerical HMC flow evolve and record the jumping distance as well as the fluctuations in the energy for every particle i. Based on the achieved performance $\tilde{\Lambda}(\cdot, \cdot)$ we weight the parameters and resample them according to a categorical

distribution. The resampled parameters are then returned as output. As a byproduct we learn the dependence of the energy fluctuations on the discretization steps ϵ. This information is used in order to determine an upper bound $\epsilon^\star$ of the interval over that ϵ will be drawn during the next time step.

The additional cost comes from the extra run of the HMC flow that is necessary at every temperature step. On the one hand this additional run has the advantage of tuning the propagation kernel precisely for the current time step. On the other hand our adaptation of Fearnhead and Taylor (2013) relies on the assumption that the performance of the kernel parameters does not change much from one time step to the next. In our simulations we show that both approaches should be considered when facing difficult target distributions.

Algorithm 8: Tuning of the HMC algorithm with pretuning

Input: Resampled particles $\tilde{x}_{t-1}^i$, $i \in 1:N$, HMC flow $\widehat{\Phi}_{\cdot,\cdot}$ targeting π_{t-1}, $\epsilon_{t-1}^\star$
Result: Sample of (ϵ_t^i, L_t^i), $i \in 1:N$, upper bound $\epsilon_t^\star$

foreach $i \in 1:N$ **do**
  Sample $\hat{\epsilon}_t^i \sim \mathcal{U}[0, \epsilon_{t-1}^\star]$ and $\hat{L}_t^i \sim \mathcal{U}\{1:L_{max}\}$;
  Sample $z_t^i \sim \mathcal{N}(0_d, M_{t-1})$;
  Apply the leapfrog integration: $(\hat{x}_t^i, \hat{z}_t^i) \leftarrow \widehat{\Phi}_{\hat{\epsilon}_t^i, \hat{L}_t^i}(z_t^i, \tilde{x}_{t-1}^i)$;
  Calculate ΔE_t^i and $\tilde{\Lambda}(\hat{x}_{t-1}^i, \hat{x}_t^i)$
Calculate $\epsilon_t^\star$ based on the quantile regression of ΔE_t^i on $\hat{\epsilon}_t^i$ $\forall i \in 1:N$;
Sample $(\epsilon_t^i, L_t^i) \sim \mathcal{C}at\left(w_t^i, \{\hat{\epsilon}_t^i, \hat{L}_t^i\}\right)$, where $w_t^i \propto \tilde{\Lambda}(\hat{x}_{t-1}^i, \hat{x}_t^i)$ $\forall i \in 1:N$;

By using a properly tuned HMC kernel we show that SMC samplers may be used in high dimensions for simulation based posterior inference. This is contrary to the widespread opinion that SMC might not be used in high dimensions. Additionally the calculation of the normalizing constant enables an effective way of model comparison, still available as the dimension increases. We highlight our points over a number of different applications.

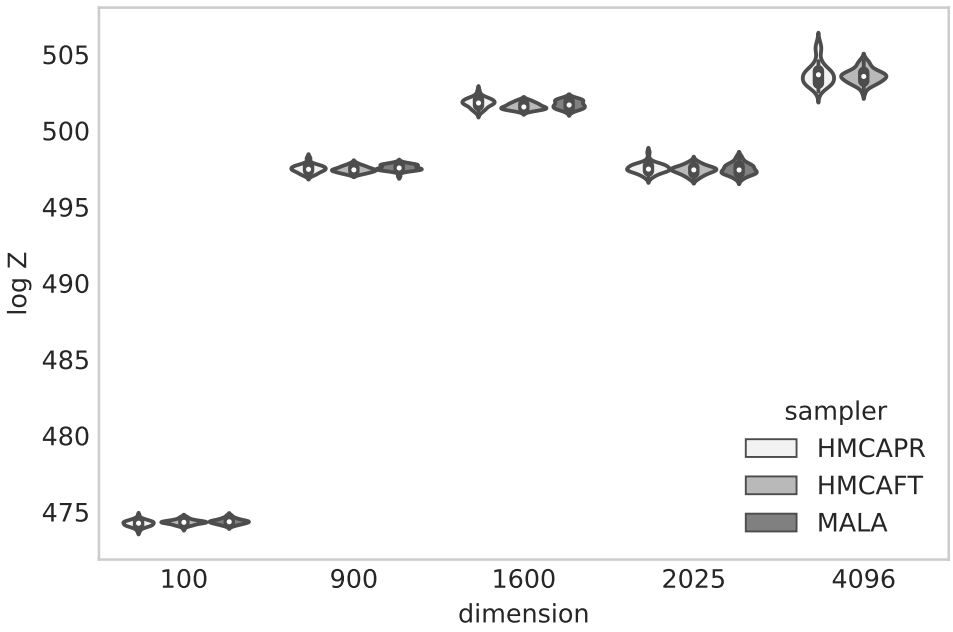


FIGURE 2.4: Tempering from a normal prior to the posterior of a log Gaussian Cox process over various dimensions. We illustrate the estimations of the normalizing constants for three different samplers. All samplers yield a quite precise estimation of the normalizing constant.

As an extract of our experimental results we show the estimated normalizing constant of the posterior of a log Gaussian Cox process model in Figure 2.4. In this model the observations follow a Poisson process conditional on a Gaussian process. The aim of the posterior inference is to recover the latent process given the observations. The observations are the locations of 126 pine trees. The dimension of the model depends on the discretization of the spatial observations. An interesting feature in this model is the fact that the dimension can be increased easily by using a finer discretization. We compare three different SMC samplers. All parameters of the three algorithms are tuned adaptively. The first sampler, denoted by MALA, is based on a MALA propagation kernel. For the tuning of the kernel parameters we use the approach of Fearnhead and Taylor (2013). The two other samplers use an HMC propagation kernel, either tuned using our adaptation of Fearnhead and Taylor (2013), abbreviated by FT, or our suggested approach based on pretuning, abbreviated by PR. As the dimension gets larger than $\approx 2,000$ the MALA sampler exceeds its computational budget. As a total measure of performance we use the variance of the estimated quantities and adjust for computation. The MALA based sampler stays competitive until dimension $1,600$. The sampler based on the tuning procedure FT works best for this model. In our other experiments we show that the sampler based on pretuning works better when the posterior distribution is highly correlated. In this cases incurring the additional cost of pretuning provides eventually a better performance.

Chapter 3

Improving approximate Bayesian computation via quasi-Monte Carlo

Joint work with Nicolas Chopin, to appear in *Journal of Computational and Graphical Statistics*

Abstract: ABC (approximate Bayesian computation) is a general approach for dealing with models with an intractable likelihood. In this work, we derive ABC algorithms based on QMC (quasi-Monte Carlo) sequences. We show that the resulting ABC estimates have a lower variance than their Monte Carlo counterparts. We also develop QMC variants of sequential ABC algorithms, which progressively adapt the proposal distribution and the acceptance threshold. We illustrate our QMC approach through several examples taken from the ABC literature.

Keywords: Approximate Bayesian computation, Likelihood-free inference, Quasi-Monte Carlo, Randomized Quasi-Monte Carlo, Adaptive importance sampling

3.1 Introduction

Since its introduction by Tavaré et al. (1997) approximate Bayesian computation (ABC) has received growing attention and has become today a major tool for Bayesian inference in settings where the likelihood of a statistical model is intractable but simulations from the model for a given parameter value can be generated. The approach of ABC is as convincing as intuitive: We first sample a value from the prior distribution, conditional on this prior simulation an observation from the model is generated. If the simulated observation is sufficiently close to the observation that has been observed in nature, we retain the simulation from the prior distribution and assign it to the set of posterior simulations. Otherwise the simulation is discarded. We repeat this procedure until enough samples have been obtained.

Since then several computational extensions related to ABC have been proposed. For instance the use of MCMC as by Marjoram et al. (2003) has improved the simulation of ABC posterior samples over the simple accept–reject algorithm. The use of sequential approaches by Beaumont et al. (2009), Sisson et al. (2009), Del Moral

et al. (2012) and Sedki et al. (2012) made it possible to exploit the information from previous iterations and eventually to choose adaptively the schedule of thresholds ϵ. Besides the question of an efficient simulation of high posterior probability regions, the choice of summary statistics, summarizing the information contained in the observation and the simulated observation, has been investigated (Fearnhead and Prangle, 2012). See Marin et al. (2012) and Lintusaari et al. (2017) for two recent reviews. Moreover, the introduction of more machine learning driven approaches like random forests (Marin et al., 2016), Gaussian processes (Wilkinson, 2014), Bayesian optimization (Gutmann and Corander, 2016), expectation propagation (Barthelmé and Chopin, 2014) and neural networks (Papamakarios and Murray, 2016) have been proposed. A post-processing approach was studied in Blum (2010).

In this paper we take a different perspective and approach the problem of reducing the variance of ABC estimators. We achieve this by introducing so called low discrepancy sequences in the simulation of the proposal distribution. We show that this allows to reduce significantly the variance of posterior estimates.

The rest of the paper is organized as follows. Section 3.2 reviews the basic ideas of approximate Bayesian computation and sets the notation. Section 3.3 introduces the concept of low discrepancy sequences. Section 3.4 brings the introduced concepts together and provides the theory that underpins the proposed idea. Section 3.5 presents a first set of numerical examples. Section 3.6 explains how to use our ideas in a sequential procedure which adapts progressively the proposal distribution and the value of ϵ. Section 3.6 illustrates the resulting sequential ABC procedure. Section 5.5 concludes.

3.2 Approximate Bayesian computation

3.2.1 Reject-ABC

Approximate Bayesian computation is motivated by models such that (a) the likelihood function is difficult or expensive to compute; (b) simulating from the model (for a given parameter θ) is feasible.

The most basic ABC algorithm is called reject-ABC. It consists in simulating pairs (θ, y), from the prior $p(\theta)$ and the likelihood $p(y|\theta)$, and keeping those pairs such that $\delta(y, y^\star) \leq \epsilon$, where $y^\star$ is the actual data, and $\delta : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^+$ is some distance (e.g. Euclidean). The sampling is continued until a number of N samples have been proposed. The target density of this rejection algorithm is:

$$p_\epsilon(\theta, y) = \frac{1}{Z_\epsilon} p(\theta) p(y|\theta) \mathbb{1}\left\{\delta(y, y^\star) \leq \epsilon\right\},$$

and its marginal density with respect to θ is:

$$p_\epsilon(\theta) = \frac{1}{Z_\epsilon} p(\theta) \mathbb{P}_\theta\left(\delta(y, y^\star) \leq \epsilon\right) \tag{3.1}$$

where $\mathbb{P}_\theta$ denotes a probability with respect to $y \sim p(y|\theta)$, and $Z_\epsilon = \int_\Theta p(\theta)\mathbb{P}_\theta\left(\delta(y, y^\star) \leq \epsilon\right) \mathrm{d}\theta$ is the normalising constant.

As $\epsilon \to 0$, (3.1) converges to the true posterior density. Actually, δ is often not a distance but a pseudo-distance of the form: $\delta(y, y^\star) = \|s(y) - s(y^\star)\|_2$, where $\|\cdot\|_2$ is the Euclidean norm, and $s(y)$ is a low-dimensional, imperfect summary of y. In that case, $p_\epsilon(\theta) \to p(\theta|s(y^\star))$. This introduces an extra level of approximation, which is hard to assess theoretically and practically. However, in this paper we focus on how to approximate well (3.1) for a given δ (and ϵ), and we refer to e.g. Fearnhead and Prangle (2012) for more discussion on the choice of δ or s.

3.2.2 Pseudo-marginal importance sampling

A simple generalisation of reject-ABC is described in Algorithm 9. For $n = 1, \ldots, N$, we sample the parameter $\theta_n \sim q(\theta)$, the latent variable $x_n \sim q_{\theta_n}(x)$, and reweight (θ_n, x_n) according to

$$w_n = \frac{p(\theta_n)}{q(\theta_n)} \times \hat{L}_\epsilon(x_n)$$

where, for $x \sim q_\theta$, $\hat{L}_\epsilon(x)$ is an unbiased estimate of the probability $\mathbb{P}_\theta\left(\delta(y, y^\star) \leq \epsilon\right)$:

$$\int q_\theta(x)\hat{L}_\epsilon(x)\,\mathrm{d}\,x = \mathbb{P}_\theta\left(\delta(y, y^\star) \leq \epsilon\right).$$

Algorithm 9: ABC importance sampling algorithm

Input : Observed $y^\star$, prior distribution $p(\theta)$, proposal distribution $q(\theta)$, simulator $q_{\theta_n}(x)$, distance function $\delta(\cdot,\cdot)$, target threshold ϵ, number of simulations N

Result: Set of weighted samples $(\theta_n, x_n, w_n)_{n\in 1:N}$

1 **for** $n = 1$ *to* N **do**
2 Sample $\theta_n \sim q(\theta)$
3 Sample $x_n \sim q_{\theta_n}(x)$
4 Set $w_n = p(\theta_n)\hat{L}_\epsilon(x_n)/q(\theta_n)$

The marginal density (with respect to θ) of the target density of this importance sampling scheme is again (3.1). In particular, the quantity

$$\hat{\phi}_N = \frac{\sum_{n=1}^N w_n \phi(\theta_n)}{\sum_{n=1}^N w_n}, \tag{3.2}$$

is a consistent (as $N \to \infty$ and under appropriate conditions) estimate of expectation $\mathbb{E}_{p_\epsilon(\theta)}[\phi(\theta)]$, for $\phi : \Theta \to \mathbb{R}$. Since the importance weight involves an unbiased estimator, the whole procedure may be viewed as a pseudo-marginal sampler, in the spirit of Andrieu and Roberts (2009).

A a special case, take the proposal $q(\theta)$ to be equal to the prior, $p(\theta)$, and take $x = y$, $\hat{L}_\epsilon(x) = \mathbb{1}\left\{\delta(y, y^\star) \leq \epsilon\right\}$; then we recover essentially the same procedure as reject-ABC (except that N stands for the number of proposed points, rather than the

number of accepted points). However, the generalized scheme allows us (a) to sample θ_n from a distribution $q(\theta)$ which may be more likely (than the prior) to generate high values for the probability $\mathbb{P}_\theta\left(\delta(y, y^\star) \leq \epsilon\right)$; and (b) to use a more sophisticated unbiased estimate for $\mathbb{P}_\theta\left(\delta(y, y^\star) \leq \epsilon\right)$.

Regarding (b), we consider two unbiased schemes in this work. In the first part, we focus on:

$$x = y_{1:M}, \quad q_\theta(x) = \prod_{m=1}^{M} p(y_m|\theta), \quad \hat{L}_\epsilon(x) = \frac{1}{M}\sum_{m=1}^{M} \mathbb{1}\{\delta(y_m, y^\star) \leq \epsilon\}. \tag{3.3}$$

for a certain $M \geq 1$. The possibility to associate more than one datapoints to each parameter θ_n was considered in e.g. Del Moral et al. (2012). Bornn et al. (2015) showed that $M = 1$ usually represents the best variance vs CPU time trade-off when using Monte Carlo sampling, however we shall see that this result does not hold when using QMC.

Later on in the paper, we shall consider an alternative unbiased estimator, based on properties of the negative binomial distribution. More precisely, assume that, for a given θ, we sample sequentially $y_1, y_2, \ldots \sim p(y|\theta)$, until we reach the time k where $r \geq 2$ datapoints are such that $\delta(y_n, y^\star) \leq \epsilon$; then k is distributed according to a negative binomial distribution with parameters r and $p = \mathbb{P}_\theta\left(\delta(y, y^\star) \leq \epsilon\right)$, and the minimum-variance unbiased estimator of $\mathbb{P}_\theta\left(\delta(y, y^\star) \leq \epsilon\right)$ is (Johnson et al., 2005, Chap. 8):

$$\hat{L}_\epsilon(x) = \frac{r-1}{k-1}$$

where $x = y_{1:k}$.

The second unbiased estimator is closely related, but not equivalent to, the r-hit kernel of Lee (2012); see also Lee and Łatuszyński (2014). Specifically, Lee (2012) proposed an MCMC kernel that generates *two* negative binomial variates (one for the current point, and one for the proposed point) at each iteration. The invariant distribution of this kernel is such that, marginally, θ is distributed according to (3.1).

In more practical terms, we shall use the latter estimator in situations where we would like to set ϵ beforehand to some value such that $\mathbb{P}_\theta\left(\delta(y, y^\star) \leq \epsilon\right)$ may be small. In that case, this estimator automatically adjusts the CPU budget (i.e. the number of simulations from the likelihood) so as to ensure that the number of simulated $y-$values is non-zero. But we shall return to this point in Section 3.6.

3.3 Quasi-Monte Carlo

Low discrepancy sequences (also called quasi-Monte Carlo sequences, see Niederreiter (1978); Dick et al. (2013); Dick and Pillichshammer (2010)), are used to approximate integrals over the $[0,1]^d$ hypercube:

$$\mathbb{E}\left[\psi(U)\right] = \int_{[0,1]^d} \psi(u)du,$$

that is the expectation of the random variable $\psi(U)$, where $U \sim \mathcal{U}\left([0,1]^d\right)$. The basic Monte Carlo approximation of the integral is $\hat{I}_N := N^{-1}\sum_{n=1}^N \psi(u_n)$, where each $u_n \sim \mathcal{U}\left([0,1]^d\right)$. The error of this approximation is $\mathcal{O}_P(N^{-1/2})$, since $\mathrm{Var}[\hat{I}_N] = \mathrm{Var}[\psi(U)]/N$.

It is possible to improve on this basic approximation, by replacing the random variables u_n by a low-discrepency sequence; that is, informally, a deterministic sequence that covers $[0,1]^d$ more regularly. This idea is illustrated in Figure 3.1.

More formally, the general notion of discrepancy of a given sequence is defined as follows:

$$D(u_{1:N}, \mathcal{A}) := \sup_{A\in\mathcal{A}} \left| \frac{1}{N}\sum_{n=1}^N \mathbb{1}\left\{u_n \in A\right\} - \lambda_d(A) \right|,$$

where $\lambda_d(A)$ is the volume (Lesbegue measure on $\mathbb{R}^d$) of A and $\mathcal{A}$ is a set of measurable sets. When we fix the sets A to be intervals anchored at 0 we obtain the so called star discrepancy:

$$D^*(u_{1:N}) := \sup_{A\in[0,\mathbf{b}]} \left| \frac{1}{N}\sum_{n=1}^N \mathbb{1}\left\{u_n \in A\right\} - \lambda_d(A) \right|,$$

where $[0,\mathbf{b}] = \prod_{i=1}^d [0,b_i], 0 \leq b_i \leq 1$. The importance of the notion of discrepancy and in particular the star discrepancy is highlighted by the Koksma-Hlawka inequality (Hickernell, 2006), which relates the error of the integration to the coverage of the space and the variation of the function that is integrated:

$$\left| \int_{[0,1]^d} \psi(u)du - \frac{1}{N}\sum_{n=1}^N \psi(u_n) \right| \leq V(\psi)D^*(u_{1:N}),$$

where $V(\psi)$ is the variation in the sense of Hardy and Krause (Hardy, 1905). This quantity is closely linked to the smoothness of the function ψ; see Kuipers and Niederreiter (2012) and Leobacher and Pillichshammer (2014) for more details.

It is possible to construct sequences u_n such that, when N is fixed in advance, $D^*(u_{1:N})$ is $\mathcal{O}\left(N^{-1}(\log N)^{d-1}\right)$, and, when N is allowed to grow, i.e., the sequence can be generated iteratively, then $D^*(u_{1:N}) = \mathcal{O}\left(N^{-1}(\log N)^d\right)$. Then $\forall\tau > 0$ the error rate is $\mathcal{O}\left(N^{-1+\tau}\right)$. Consequently, QMC integration schemes are asymptotically more efficient than MC schemes. One observes in practice that QMC integration outperforms MC integration even for small N in most applications, see e.g. the examples in Chapter 5 of Glasserman (2013).

3.3.1 Randomized quasi-Monte Carlo

A drawback of QMC is that it does not come with an easy way to assess the approximation error.

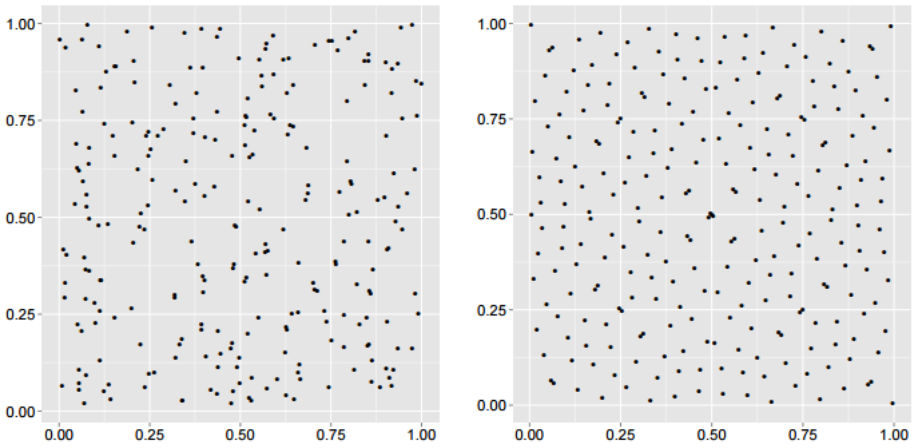


FIGURE 3.1: Uniform random (left) and Halton (right) point set of length 256 in $[0,1]^2$. The Halton sequence covers the target space more evenly than the random uniform sequence.

RQMC (randomized quasi-Monte Carlo) amounts to introduce randomness in a QMC sequence, in such a way that $u_n \sim \mathcal{U}\left([0,1]^d\right)$, marginally. The quantity $\hat{I}_N = N^{-1}\sum_{n=1}^{N}\psi(u_i)$ then becomes an unbiased estimate of the integral of interest. One may assess the approximation error by computing the empirical variance over repeated simulations.

The simplest way to obtain an RQMC sequence is to randomly shift a QMC sequence: Let $\mathbf{v} \sim \mathcal{U}\left([0,1]^d\right)$, and $u_{1:N}$ a QMC sequence; then

$$\hat{u}_n := u_n + \mathbf{v} \mod 1 \text{ (component wise)}$$

is an RQMC sequence.

A more sophisticated approach, called scrambled nets, was introduced by Owen (1997) and later refined in Owen (2008). The main advantage of this approach is that under the assumption of smoothness of the derivatives of the function, the speed of convergence can be even further improved, as stated in the following Theorem.

Theorem 1 *Owen (2008) Let $f : [0,1]^d \to \mathbb{R}$ be a function such that its cross partial derivatives up to order d exist and are continuous, and let $(u_n)_{n\in 1:N}$ be a relaxed scrambled (λ, t, m, d)-net in base b with dimension d with uniformly bounded gain coefficients. Then,*

$$\mathrm{Var}\left(\frac{1}{N}\sum_{n=1}^{N} f(u_n)\right) = \mathcal{O}\left(N^{-3}\log(N)^{(d-1)}\right),$$

where $N = \lambda b^m$.

In words, $\forall\tau > 0$ the RQMC error rate is $\mathcal{O}(N^{-3/2+\tau})$ when a scrambled (λ, t, m, d)-net is used. This result has the only inconvenience that the rate of convergence only holds for certain N. However, a more general result has recently been shown by Gerber (2015)[Corollary 1], where if $f \in L^2$ and $(u_n)_{n\in 1:N}$ is a scrambled (t, d)-sequence, then

$\forall N \in \mathbb{N}$,

$$\text{Var}\left(\frac{1}{N}\sum_{n=1}^{N} f(u_n)\right) = o\left(N^{-1}\right).$$

The construction of scrambled nets and sequences is quite involved. As the focus of our paper is the application and not the construction of these sequences, we refer the reader for more details to L'Ecuyer (2016) or Dick et al. (2013). In the following, when speaking about an RQMC sequence, we will assume that this sequence is a scrambled (t, d)-sequence.

3.3.2 Mixed sequences and a central limit theorem

One drawback of low discrepancy sequences is that the speed of convergence deteriorates with the dimension. In some situations, a small number of components contributes significantly to the variance of the target. One then might choose to use a low discrepancy sequence for those components and an ordinary Monte Carlo approach on the rest. This idea of using a *mixed sequence* is closely linked to the concept of effective dimension, see Owen (1998). Based on the randomness induced by the Monte Carlo part a central limit theorem (CLT) may be established:

Theorem 2 *Ökten et al. (2006) Let $u_k = (q_k^{1:d}, X_k^{d+1:s})$ be a mixed sequence of dimension s where $q_k^{1:d}$ denotes the deterministic QMC part and $X_k^{d+1:s}$ denotes the random independent MC part. Let $f : [0,1]^s \to \mathbb{R}^t, t \in \mathbb{N}$ a bounded, square integrable function, $Y_k = f(u_k)$, $\hat{I}_N = N^{-1}\sum_{k=1}^{N} Y_k$, and*

$$\begin{aligned}
\mu_k &:= \mathbb{E}[Y_k] = \int_{[0,1]^{s-d}} f(u_k)\,\mathrm{d}\,X^{d+1:s},\\
S_N &:= \frac{1}{N}\left(\sum_{k=1}^{N} Y_k - \sum_{k=1}^{N}\mu_k\right) = \left(\hat{I}_N - \frac{1}{N}\sum_{k=1}^{N}\mu_k\right),\\
\sigma_k^2 &:= \text{Var}[Y_k] = \int_{[0,1]^{s-d}} f(u_k)f(u_k)^T\,\mathrm{d}\,X^{d+1:s}\\
&\quad - \left(\int_{[0,1]^{s-d}} f(u_k)\,\mathrm{d}\,X^{d+1:s}\right)\left(\int_{[0,1]^{s-d}} f(u_k)\,\mathrm{d}\,X^{d+1:s}\right)^T,\\
C_N^2 &:= \text{Var}[N\hat{I}_N] = \sum_{k=1}^{N}\sigma_k^2.
\end{aligned}$$

Then, as $N \to +\infty$, $C_N^2/N \to C^2_{\text{qmc}-\text{mixed}}$ and

$$N^{1/2}S_N \xrightarrow{\mathcal{L}} \mathcal{N}\left(0, C^2_{\text{qmc}-\text{mixed}}\right),$$

where

$$C^2_{\text{qmc}-\text{mixed}} = \int_{[0,1]^s} f(x) f(x)^T \, \mathrm{d}\, x - \int_{[0,1]^d} \left(\int_{[0,1]^{s-d}} f(u) \, \mathrm{d}\, X^{d+1:s} \right) \left(\int_{[0,1]^{s-d}} f(u) \, \mathrm{d}\, X^{d+1:s} \right)^T \mathrm{d}\, q^{1:d}.$$

As a direct corollary of the previous Theorem we obtain that, provided f has a finite variation in the sense of Hardy and Krause, $N^{1/2}(\hat{I}_N - I) \xrightarrow{\mathcal{L}} \mathcal{N}(0, C^2_{\text{qmc}-\text{mixed}})$, where $I = \int f(u) \, \mathrm{d}\, u$. This is due to the fact that

$$N^{1/2} \left(\hat{I}_N - I \right) = N^{1/2} S_N + N^{1/2} \left(\frac{1}{N} \sum_{k=1}^{N} \mu_k - I \right)$$

and the second term on the right hand side converges deterministically to 0. In their work Ökten et al. (2006) present only a univariate version of their central limit theorem. Nevertheless, the multivariate extension is straightforward but lengthy.

Moreover, their work shows that the asymptotic variance of the mixed sequence estimator is smaller than for the same estimator based on Monte Carlo sequences in dimension one. We extend this result to the multivariate case.

Corollary 1 *Let $C^2_{\text{qmc}-\text{mixed}}$ be the asymptotic variance of an estimator based on a mixed sequence as defined in Theorem 2. Let C^2_{mc} be the variance of the same estimator based on a pure MC sequence, e.g., when $d = 0$. Then*

$$C^2_{\text{qmc}-\text{mixed}} \preceq C^2_{\text{mc}}$$

in the sense of positive definite matrices.

Moreover, we present a result here that allows us to apply the same technique to mixed sequences that combine Monte Carlo and randomized quasi-Monte Carlo sequences.

Theorem 3 *Let S_N^{RQMC} be the MC-RQMC equivalent of S_N under the same conditions as in Theorem 2. Then*

$$N^{1/2} S_N^{\text{RQMC}} \xrightarrow{\mathcal{L}} \mathcal{N}(0, C^2_{\text{rqmc}-\text{mixed}}),$$

where $C^2_{\text{rqmc}-\text{mixed}} = C^2_{\text{qmc}-\text{mixed}}$.

These results may be understood as follows. The randomness in the Monte Carlo sequence allows the construction of a central limit theorem. The part associated to the (R)QMC sequences converges faster to zero than the part associated to the Monte Carlo sequence. This leads to a reduced asymptotic variance for estimators based on mixed sequences.

3.4 Improved ABC via (R)QMC

Recall that we described our ABC importance sampler as an algorithm that samples pairs (θ_n, x_n) from $q(\theta)q_\theta(x)$, where x_n consists of datapoints generated from the model. In most ABC problems, using (R)QMC to generate the θ_n should be easy, but this should not be the case for the x_n's. Indeed, the simulator used to generate datapoints from the model may be a complex black box, which may require a very large, or random, number of uniform variates. Thus, we contemplate from now on generating the θ_n's using (R)QMC. That is, $\theta_n = \Gamma(u_n)$, where $u_{1:N}$ is a QMC or RQMC sequence, and Γ is a function such that $\Gamma(U)$, $U \sim \mathcal{U}\left([0,1]^d\right)$, is distributed according to the proposal $q(\theta)$; and $x_n|\theta_n \sim q_{\theta_n}$ is a random variate. In other words, (θ_n, x_n) is a mixed sequence.

We already know from the previous section that an estimate based on a mixed sequence converges at the Monte Carlo rate, $\mathcal{O}_P(N^{-1/2})$, but has a smaller asymptotic variance than the same estimate based on Monte Carlo. In fact, a similar result may be established directly for the actual variance. Let $\hat{I}_N := \sum_{n=1}^N \varphi(\theta_n, x_n)/N$ be an empirical average for some measurable function φ. For simplicity, we assume here that the θ_n's are either random variates, or RQMC variates. That is, in both cases, $\theta_n \sim q$ marginally. Then

$$\mathrm{Var}[\hat{I}_N] = \mathrm{Var}\left[\mathbb{E}\{\hat{I}_N|\theta_{1:N}\}\right] + \mathbb{E}\left[\mathrm{Var}\{\hat{I}_N|\theta_{1:N}\}\right] \tag{3.4}$$

The first term is $\mathcal{O}(N^{-1})$ when the θ_n's are generated using Monte Carlo, and should be $o(N^{-1})$ under appropriate conditions when the θ_n's are an RQMC sequence. On the other hand, the second term is $\mathcal{O}(N^{-1})$ in both cases. As a corrolary, the variance of $\hat{I}_N$ is smaller when using a mixed sequence, for N large enough.

The point of the following sections is to generalize this basic result to various ABC estimates of interest.

3.4.1 Improved estimation of the normalization constant

We first consider the approximation of the normalization constant of the ABC posterior:

$$Z_\epsilon = \int \mathbb{P}_\theta\left(\delta(y, y^\star) \leq \epsilon\right) p(\theta)\,\mathrm{d}\,\theta = \int \hat{L}_\epsilon(x) q_\theta(x) p(\theta)\,\mathrm{d}\,x\,\mathrm{d}\,\theta.$$

Recall that, for the moment, we take $x = y_{1:M}$, $q_\theta(x) = \prod_{m=1}^M p(y_m|\theta)$ and

$$\hat{L}_\epsilon(x) = \frac{1}{M}\sum_{m=1}^M \mathbb{1}\left\{\delta(y_m, y^\star) \leq \epsilon\right\}.$$

Thus, a natural estimator of Z_ϵ is

$$\hat{Z}_N := \frac{1}{N}\sum_{n=1}^{N}\frac{p(\theta_n)}{q(\theta_n)}\left[\frac{1}{M}\sum_{m=1}^{M}\mathbb{1}\left\{\delta(y_{n,m},y^\star)\leq\epsilon\right\}\right] \tag{3.5}$$

where the θ_n's are either a Monte Carlo or RQMC sample from the proposal $q(\theta)$, and $y_{n,m}\sim p(y|\theta_n)$ for $n=1,\ldots,N$, $m=1,\ldots,M$.

When the θ_n's are a Monte Carlo sample, it is always best to take $M=1$, as noted by Bornn et al. (2015). This may be seen by calculating both terms of decomposition (3.4) when applied to the estimator of the normalization constant $\hat{Z}_N$:

$$\mathrm{Var}\left[\mathbb{E}\{\hat{Z}_N|\theta_{1:N}\}\right] = \frac{1}{N}\times\mathrm{Var}_q\left[\frac{p(\theta)}{q(\theta)}\mathbb{P}_\theta\left(\delta(y,y^\star)\leq\epsilon\right)\right] \tag{3.6}$$

$$\mathbb{E}\left[\mathrm{Var}\{\hat{Z}_N|\theta_{1:N}\}\right] = \frac{1}{NM}\times\int_\Theta\frac{p(\theta)^2}{q(\theta)}\mathbb{P}_\theta\left(\delta(y,y^\star)\leq\epsilon\right)\left\{1-\mathbb{P}_\theta\left(\delta(y,y^\star)\leq\epsilon\right)\right\}\mathrm{d}\,\theta. \tag{3.7}$$

Increasing M increases the CPU cost and decreases the variance of $\hat{Z}_N$. To account for both simultaneously, we look at the adjusted variance, $M\times\mathrm{Var}\left[\hat{Z}_N\right]$. From (3.6) and (3.7), we see that the adjusted variance increases with M, hence the best CPU time vs error trade-off is obtained by taking $M=1$.

Now, consider the situation where the θ_n's form an RQMC sequence. As noted in the previous section, (3.7) still holds due to the unbiasedness property of RQMC sequences, however the first (3.6) term of the decomposition should converge faster.

Proposition 1 *Let $f(\theta)=\{p(\theta)/q(\theta)\}\mathbb{P}_\theta\left(\delta(y,y^\star)\leq\epsilon\right)$, assume that $\theta_n=\Gamma(u_n)$ where $u_{1:N}$ is a scrambled (λ,t,m,d)-net, and assume that $f\circ\Gamma\in L^2$. Then,*

$$\mathrm{Var}\left[\mathbb{E}\{\hat{Z}_N|\theta_{1:N}\}\right]=o\left(N^{-1}\right).$$

This result is a direct consequence of Corollary 1 of Gerber (2015) and the fact

$$\mathbb{E}\{\hat{Z}_N|\theta_{1:N}\}=\frac{1}{N}\sum_{n=1}^{N}f(\theta_n)=\frac{1}{N}\sum_{n=1}^{N}f\circ\Gamma(u_n).$$

It has two corollaries. First, the variance of $\hat{Z}_N$ is smaller when using a RQMC sequence for the θ_n's (for N large enough). Second, in that case, the adjusted variance is such that $M\,\mathrm{Var}[\hat{Z}_N]=\mathcal{O}(N^{-1})$, with a constant that does not depend on M. Thus taking $M>1$ (within a reasonable range) should have basically no impact on the CPU time vs error trade-off in the RQMC case.

Taking $M>1$ has the following advantage: it makes it possible to consistently estimate (3.7) with the quantity

$$\hat{\sigma}^2(Z_\epsilon):=\frac{1}{N^2(M-1)}\times\sum_{n=1}^{N}\frac{p(\theta_n)^2}{q(\theta_n)^2}\hat{L}_\epsilon(x_n)\{1-\hat{L}_\epsilon(x_n)\}. \tag{3.8}$$

where $\hat{L}_\epsilon(x_n) = M^{-1}\sum_{m=1}^{M}\mathbb{1}\{\delta(y_{n,m}, y^\star) \leq \epsilon\}$. As (3.7) corresponds to the non-negligible part of the variance of $\hat{Z}_N$, this allows us to obtain asymptotic confidence intervals for $\hat{Z}_N$.

We have focused on the RQMC case for now on, but a similar result holds for QMC sequences. Note, however, that we cannot use directly decomposition (3.4) when the θ_n's are deterministic. For the case of QMC sequences we obtain the following result.

Proposition 2 *Assume that $f \circ \Gamma$ (where f and Γ are defined as in Proposition 1) has a finite variation in the sense of Hardy and Krause, and that the ratio p/q is upper-bounded, $p(\theta)/q(\theta) \leq C$, then*

$$M \times \mathbb{E}\left[\left(\hat{Z}_N - Z_\epsilon\right)^2\right] = \mathcal{O}(N^{-1})$$

with a constant that does not depend on M. Furthermore, the mean square error above is smaller than in the Monte Carlo case, for N large enough.

3.4.2 Improved estimation of general importance sampling estimators

We now turn to the analysis of general importance sampling estimators of the form

$$\hat{\phi}_N = \frac{\sum_{n=1}^{N} w_n \phi(\theta_n)}{\sum_{n=1}^{N} w_n}. \tag{3.9}$$

As these estimators are ratios, we cannot apply decomposition (3.4) directly. However, we may apply the following inequality, due to Agapiou et al. (2015):

$$\mathbb{E}\left\{\hat{\phi}_N - \mathbb{E}_{p_\epsilon}\phi\right\}^2 \leq \frac{2}{Z_\epsilon^2}\left(\mathbb{E}\left\{\frac{1}{N}\sum_{n=1}^{N} w_n\phi(\theta_n) - Z_\epsilon \mathbb{E}_{p_\epsilon}\phi(\theta)\right\}^2 + \mathbb{E}\left\{\frac{1}{N}\sum_{n=1}^{N} w_n - Z_\epsilon\right\}^2\right)$$

provided $|\phi| \leq 1$. Both terms are mean squared errors of empirical averages, and hence may be bounded directly using a decomposition of variance and the results of the previous section. Thus, we see that, again, when the θ_n are generated with (R)QMC, the mean squared error of estimate $\hat{\phi}_N$ is $\mathcal{O}(M^{-1}N^{-1})$ as $N \to +\infty$. However, this inequality does not make it possible to compare the performance of our RQMC-ABC procedure with Monte Carlo-based ABC. For this, we now consider the asymptotic behavior of these estimators.

Theorem 4 *Let $\phi : \Theta \to \mathbb{R}$ be a bounded function, $\bar{\phi} = \phi - \mathbb{E}_{p_\epsilon}\phi$, $\hat{\phi}_N$ defined as (3.9), then, under the same conditions as Proposition 2, and assuming further that function $u \to \bar{\phi}(\Gamma(u))f(\Gamma(u))$ has a finite variation (in the sense of Hardy and Krause), one has that*

$$N^{1/2}\left(\hat{\phi}_N - \mathbb{E}_{p_\epsilon}\phi\right) \xrightarrow{\mathcal{L}} \mathcal{N}\left(0, \sigma^2_{mixed}(\phi)\right),$$

where, using the short-hand $b(\theta) = \mathbb{P}_\theta\left(\delta(y, y^\star) \leq \epsilon\right)$,

$$\sigma^2_{\text{mixed}}(\phi) = \frac{1}{MZ_\epsilon^2}\int_\Theta \frac{p(\theta)^2}{q(\theta)}\bar{\phi}(\theta)^2 b(\theta)\{1 - b(\theta)\}\,\mathrm{d}\,\theta. \tag{3.10}$$

Alternatively, if the parameter values θ_n were generated through Monte Carlo sampling, one would obtain a similar central limit theorem, but with asymptotic variance

$$\sigma^2_{\mathrm{MC}}(\phi) = \frac{1}{Z_\epsilon^2} \int_\Theta \frac{p(\theta)^2}{q(\theta)} \bar{\phi}(\theta)^2 \left[\frac{b(\theta)\{1-b(\theta)\}}{M} + b(\theta)^2 \right] \mathrm{d}\,\theta$$

which is larger than or equal to $\sigma^2_{\mathrm{mixed}}(\phi)$.

It is possible to obtain a similar result for the use of RQMC sequences by using Theorem 3.

As for the normalising constant, we observe that the adjusted (asymptotic) variance, i.e. $M \times \sigma^2_{\mathrm{mixed}}(\phi)$, is constant with respect to M. Thus, taking $M > 1$ does not deteriorate the performance of the algorithm (in terms of variance relative to CPU time). And it makes it possible to estimate consistently the asymptotic variance (3.10) (and thus compute confidence intervals) using

$$\widehat{\sigma}^2_{\mathrm{mixed}}(\phi) = \frac{1}{(\hat{Z}_N)^2 N(M-1)} \sum_{n=1}^{N} \frac{p(\theta_n)^2}{q(\theta_n)} \{\phi(\theta_n) - \hat{\phi}_N\}^2 \hat{L}_\epsilon(x_n) \left\{1 - \hat{L}_\epsilon(x_n)\right\}.$$

3.5 Numerical examples

We illustrate in this section the improvement brought by (R)QMC through several numerical examples. Code for reproducing the results of this section and of Section 3.7 is available under `https://github.com/alexanderbuchholz/ABC`.

Thus we compare three different approaches, all corresponding to Algorithm 9, but with particles generated using either Monte Carlo (ABC-IS), Quasi-Monte Carlo (ABC-QMC), or randomised QMC (ABC-RQMC). For the generation of the (R)QMC sequences we use the `R` package `randtoolbox` (Christophe and Petr, 2015) and generate Sobol sequences (QMC), or Owen-type scrambled Sobol sequences (RQMC), see Owen (1998).

We take $q(\theta) = p(\theta)$, i.e. points are generated from the prior, and, unless explicitely stated, we take $M = 1$. (The problem of adaptively choosing q will be considered in the next section.)

In this case, weights w_n are either 0 or 1 (according to whether $\delta(y_n, y^\star) \leq \epsilon$), and we set ϵ so that the proportion of non-zero weights is close to some pre-specified value, e.g. 10^{-3}.

3.5.1 Toy model

The first model we consider is the toy model used in Marin et al. (2012) that tries to recover the mean of a superposition of two Gaussian distributions with identical mean

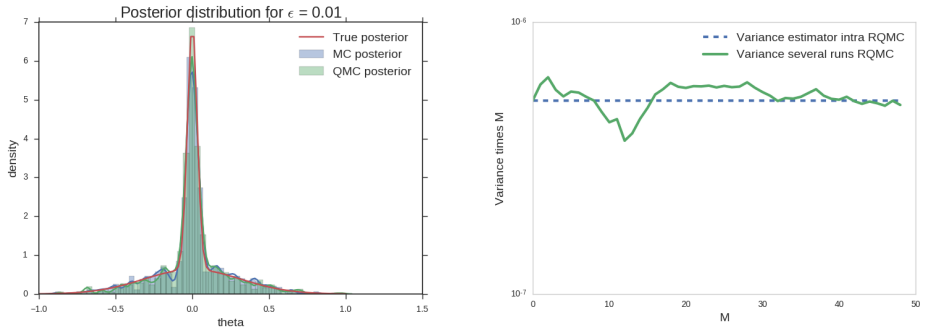
and different variances:

$$\begin{aligned} \theta &\sim \mathcal{U}\left([-10,10]^d\right), \\ y|\theta &\sim \frac{1}{2}\mathcal{N}(\theta; 0.1 I_d) + \frac{1}{2}\mathcal{N}(\theta; 0.001 I_d). \end{aligned}$$

The use of this model is motivated by the fact that the dimension of the model d can be scaled up easily. We set $y^\star = 0_d$ and $\delta(y, y^\star) = \|y - y^\star\|_2$. For a fixed value of ϵ the true approximate posterior of the form 3.1 can be calculated exactly. The resulting expression depends on the cdfs of non-central χ^2 distributions.

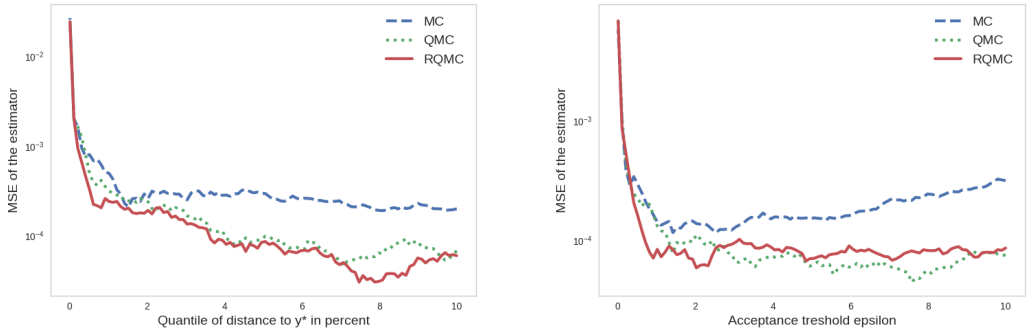
We run the three considered algorithms with $N = 10^6$. Figure 3.2a shows that the MC and QMC approximations match closely; for this plot, $\epsilon = 0.01$ (leading to a proportion of non-zero weights close to 10^{-3}), and $d = 1$.

Figure 3.3 compares the empirical variance (over 50 runs) obtained with the three considered approaches, as a function of ϵ, when estimating the expectation (left pane) and variance (right pane) of the ABC posterior. Here, $N = 10^6$, $d = 2$, and ϵ is chosen so as to generate a proportion of non-zero weights that vary from 0 to 10%.



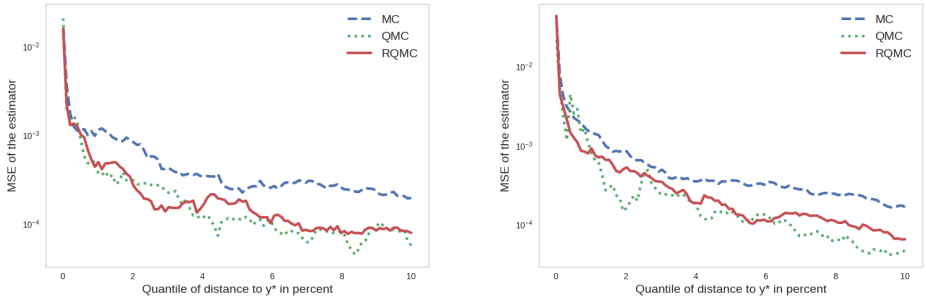
(A) (B)

FIGURE 3.2: Left: Kernel density estimation of the approximation of the posterior distribution based on $N = 10^6$ simulations and the threshold $\epsilon = 0.01$ for $d = 1$. The exact posterior can be calculated analytically. The approaches based on MC and QMC essentially recover the same distribution. Right: Adjusted variance (variance times M) of the normalization constant as a function of M: the dashed line corresponds to the variance estimator given by (3.8), the solid line corresponds to the empirical variance of the estimator based on 75 runs. The results are based on $N = 10^5$ simulations, $\epsilon = 1$, $d = 1$, and an RQMC sequence for the θ_n's. The adjusted variance stays roughly constant for $M > 1$.



(A) (B)

FIGURE 3.3: MSE of posterior estimates as ϵ varies (Left: ABC posterior mean; Right: ABC posterior variance). The plots are based on 50 runs, with $N = 10^6$ simulations and $d = 2$. The $x-$axis corresponds to a varying ϵ, which is set so that the proportion of non-zero weights (i.e. the proportion of simulated y_n such that $\delta(y_n, y^\star) \leq \epsilon$) varies from 0 to 10%. (R)QMC sequences lead to a reduced MSE. The effect vanishes as ϵ goes to 0.



(A) Left (B) Right

FIGURE 3.4: Same caption as for Figure 3.3b, except left (resp. right) panel corresponds to $d = 4$ (resp. $d = 8$); posterior estimate is the ABC posterior expectation in both cases.

We observe a variance reduction when using either QMC or RQMC and for not too small values of ϵ, but the variance reduction vanishes as $\epsilon \to 0$. However, interestingly, the variance reduction (again for not too small values of ϵ) remains significant when we increase the dimension, see Figures 3.4. (For $d > 1$, the considered estimated quantity is the expectation of the average of the d components of θ with respect to the ABC posterior.)

Finally, we consider increasing M, so as to be able to estimate the variance of a given ABC estimate from a single run of Algorithm 9, when using (R)QMC, as explained at the end of Section 3.4.1. The considered estimate is that of the normalising constant of the ABC posterior. We see that the variance estimate is fairly stable even for small values of M, and that it is close to the actual variance (over 75 runs) of the estimate as can be seen in Figure 3.2b.

Note that both quantities are multiplied by M in Figure 3.2b. This allows us to check that the adjusted variance (accounting for CPU time) remains constant, as expected. As already explained, this means that taking $M > 1$ is not sub-optimal (in terms of the variance vs CPU time trade-off), while it allows us to estimate the variance of any estimate obtained from the (R)QMC version of Algorithm 9.

3.5.2 Lotka-Volterra-Model

The Lotka-Volterra model, see Toni et al. (2009), is commonly used in population dynamics to study the interaction in predator-prey models, for example. The model is characterized by the respective size of the populations evolving over time and denoted by (X_1, X_2), taking values in $\mathbb{Z}^2$.

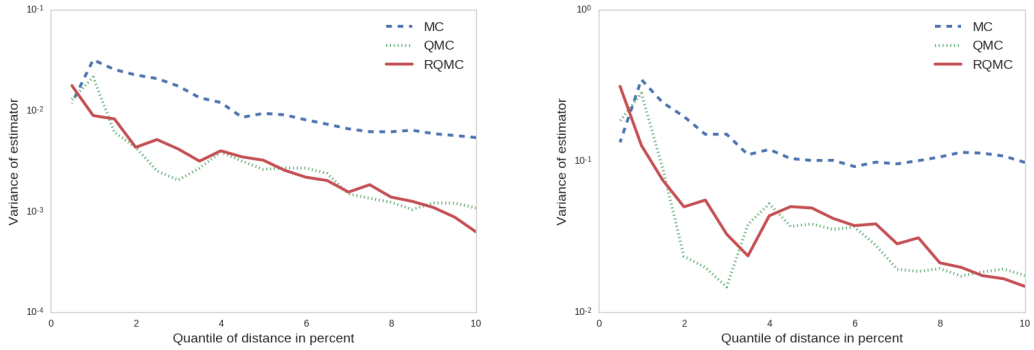
There are three possible transitions: the prey (denoted by X_1) may grow by one entity with rate α, a predation may happen with rate β, that reduces the prey by one unit and increases the predator population (denoted by X_2) by one unit, or the predator may die with rate γ. The system is summarized by the following rate equations:

$$\begin{aligned}(X_1, X_2) &\xrightarrow{\alpha} (X_1 + 1, X_2),\\ (X_1, X_2) &\xrightarrow{\beta} (X_1 - 1, X_2 + 1),\\ (X_1, X_2) &\xrightarrow{\gamma} (X_1, X_2 - 1),\end{aligned}$$

with the corresponding hazard rates αX_1, $\beta X_1 X_2$ and γX_2, respectively. The hazard rates characterize the instantaneous probability that the system changes to a new state. The parameter of the model is $\theta = (\alpha, \beta, \gamma)$. The initial population is fixed to $(50, 100)$.

We simulate from the model using Gillespie's algorithm, see Toni et al. (2009), for $T = 30$ time steps, and record the size of the population at times $t_i = 2i$, where $i = 0, \cdots, 15$. This gives two discrete time series of length 16. As a distance function for comparing our true observation and the pseudo-observations, we use the Euclidean norm $\|\cdot\|_2$ applied to the differences of the series. As a prior we use $u \sim \mathcal{U}[-6, 2]^3$, which is then transformed to $\theta = \exp(u)$.

As in the previous section, we compare the empirical variance over 50 runs of a given estimate obtained from the different approaches. The estimated quantity is the expectation of $(\alpha + \beta + \gamma)/3$ with respect to the ABC posterior.



(A) (B)

FIGURE 3.5: Variance of the mean and variance estimator for the Lotka–Volterra model. The plots are based on 50 repetitions of 10^5 simulations from the prior and the model. The accepted observations correspond to quantiles based on the smallest distances $\delta(y_n, y^\star)$. Left: Variance of the posterior mean estimator. Right: Variance of the posterior variance estimator

We observe the same phenomenon as in the previous example: the variance reduction brought by either QMC or RQMC is significant for not too small values of ϵ, but it vanishes as $\epsilon \to 0$.

3.5.3 Tuberculosis mutation

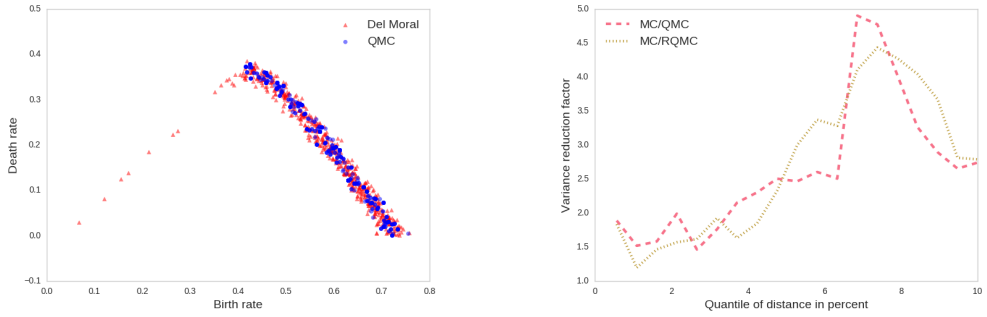
The following application is based on the estimation of tuberculosis reproduction rates as in Tanaka et al. (2006). The interest lies in recovering the posterior distribution of birth, death and mutation rates (α, β, γ) of a tuberculosis population that has been recorded in San Francisco over a period from 1991 to 1992.

The simulator of the model is based on an underlying continuous time Markov process where t denotes the time and $N(t)$ denotes the size of the population. Starting from one single bacterium the individual can either replicate itself with rate α, die with rate γ or mutate to a new genotype with rate β. The number of bacteria having the same genotype is recorded at every step and the simulation is run forward until a size of $N(t) = 10^4$ has been obtained. At every step in the simulation a bacterium is chosen uniformly at random and one of the three events (α, β, γ) is applied to it. After simulating a population of 10^4 bacteria, the simulation is stopped and a subpopulation of 473 bacteria is sampled. The ensuing population is characterized by the cluster size of bacteria that have the same genotype. The data is available in Table 3.1. For instance, there were 282 clusters with only one bacterium with the same genotype and there were 20 clusters that contained two bacteria with the same genotype.

Cluster size	1	2	3	4	5	8	10	15	23	30
Number of clusters	282	20	13	4	2	1	1	1	1	1

TABLE 3.1: Tuberculosis bacteria genotype data

The parameters must satisfy the conditions $\alpha + \beta + \gamma = 1$, $0 \leq \alpha, \beta, \gamma \leq 1$, and $\alpha > \gamma$. (The last constraint prevents the population from dying out.) Thus, we let $\beta = 1 - \alpha - \gamma$, and assign a uniform prior to (α, γ), subject to $\alpha > \gamma$. Tanaka et al. (2006) used as a summary statistic for the data the quantities $y = (g/473, 1 - \sum_i (n_i/473)^2)$, where g denotes the number of distinct clusters in the sample and n_i is the number of observed bacteria in the ith genotype cluster. The distance between a pseudo observation and the observed data is finally calculated as the Euclidean distance between y and $y^\star$. Figure 3.6a shows the recovered posterior distribution after application of a sequential sampling approach, that is described in Section 3.7. We see our method, denoted by QMC and the method of Del Moral et al. (2012), denoted by *Del Moral* recover the same posterior distribution. There remain some artifacts in the second method, due to a slightly higher acceptance threshold $\epsilon = 0.12$ compared to $\epsilon = 0.08$ as in the QMC approach. We estimate the ABC posterior expectation of $(\alpha + \gamma)/2$ and then compare the empirical variance of this estimator. The result of the repeated simulation of this estimator is shown in Figure 3.6b, where we show the value of Var_{MC} / $\text{Var}_{(R)QMC}$, where Var_{MC} is the variance of the posterior estimator based on a MC sequence. This quantity allows to assess the variance reduction factor as a function of the acceptance threshold. Again, we observe a declining variance reduction as $\epsilon \to 0$. Nevertheless, the variance reduction even for the smallest acceptance threshold is still of factor 1.5, which means that we need 33% fewer simulations in order to achieve the same precision of the estimator.



(A) (B)

FIGURE 3.6: Left: Posterior distribution of the tuberculosis mutation model. The x–axis corresponds to birth rate α, the y–axis corresponds to the death rate β, $N = 500$. Right: Variance reduction factors (computed from 50 runs based on $N = 10^4$) as a function of the proportion of non-zero weights.

3.5.4 Concluding remarks

As predicted by the theory, we observed that using QMC (or RQMC) to generate the parameter values (in Algorithm 9) always reduce the variance of ABC estimates. However, the variance reduction becomes small when $\epsilon \to 0$. But it should be noted that any static ABC algorithm, such as Algorithm 9 becomes very wasteful when ϵ

is small, as most simulated datapoints lies outside the ball defined by the constraint $\delta(y, y^\star) \leq \epsilon$ in such a case. In order to take ϵ smaller and smaller, it seems to make more sense to progressively refine the proposal distribution, based on past simulations. This is the point of sequential ABC algorithms, which we discuss in the next two sections.

3.6 Sequential ABC

3.6.1 Adaptive importance sampling

One major drawback of Algorithm 9 is that the quality of the approximation in (3.2) depends on how well the proposal distribution $q(\theta)$ matches the target distribution $p_\epsilon(\theta)$. If, for example, the proposal is very flat and the target is spiky due to a small value of ϵ, only a small number of particles will cover the region of interest. The idea of sequential ABC algorithms is therefore to sequentially decrease ϵ over a range of time steps $t \in 0 : T$ while adapting the proposal distribution $q_t(\theta)$ so as to make it closer and closer to the true posterior.

In the current setting we will use a flexible parametric approximation $q_t(\theta)$ of the ABC posterior $p_{\epsilon_t}(\theta)$, that is estimated from the the samples $(\theta_n^{t-1}, w_n^{t-1})_{n\in 1:N}$. This distribution $q_t(\theta)$ is then used to simulate new particles $(\theta_n^t)_{n\in 1:N}$. The corresponding algorithm is given as pseudo-code in Algorithm 10.

Algorithm 10: ABC adaptive importance sampling algorithm

Input : Observed $y^\star$, prior distribution $p(\theta)$, simulator $q_\theta(x)$, initial threshold ϵ_0, number of simulations N, weighting procedure $\hat{L}_\epsilon(x)$.

Result: Set of weighted samples $(\theta_n^t, x_n^t, w_n^t)_{n\in 1:N, t\in 0:T}$

```
for n = 1 to N do
    Sample θ_n^0 ~ p(θ) ;
    set w_n^0 = 1 ;
for t = 1 to T do
    Set ε_t and q_t(θ) based on (θ_n^{t-1}, x_n^{t-1}, w_n^{t-1})_{n∈1:N} ;
    for n = 1 to N do
        Sample θ_n^t ~ q_t(θ) ;
        Sample x_n^t ~ q_{θ_n^t}(x) ;
        Set w_n^t = p(θ_n^t) L̂_{ε_t}(x_n^t) / q_t(θ_n^t) ;
```

3.6.2 Adapting the proposal q_t

Gaussian proposal

The simplest strategy one may think of to adapt q_t is to set it to a Gaussian fit of the previous weighted sample. Although basic, we shall see that this approach tends to

work well in practice, unless of course the actual posterior is severely multimodal, strongly skewed or has heavy tails.

Mixture of N components

The sequential Monte Carlo sampler (SMC) of Sisson et al. (2009) may be viewed as a particular version of Algorithm 10, where q_t is set to a mixture of N Gaussian components centred on the N previous particles θ_n^{t-1}, with covariance matrix $\hat{\Sigma}^{t-1}$ set to twice the empirical covariance of these particles. The proposal distribution reads

$$q_t(\theta) = \frac{\sum_{n=1}^N w_n^{t-1} \mathcal{N}(\theta|\theta_n^{t-1}, 2\hat{\Sigma}^{t-1})}{\sum_{n=1}^N w_n^{t-1}}.$$

This results in an algorithm of complexity $\mathcal{O}(N^2)$ since for every proposed new particle θ_n^t, computing the corresponding weight involves a sum over N terms.

Mixture proposal with a small number of components

As an intermediate solution between a single Gaussian distribution and a mixture of N Gaussian distributions, we suggest to use a Gaussian mixture with a small number of components. We suggest to estimate the mixture via a Variational Bayesian procedure, see Blei et al. (2017), but other methods as Expectation Maximization could also be used. The proposal distribution reads

$$q_t(\theta) = \sum_{j=1}^J \alpha_j^{t-1} \mathcal{N}(\theta|\hat{\mu}_j^{t-1}, \lambda\hat{\Sigma}_j^{t-1}),$$

where α_j^{t-1}, $\hat{\mu}_j^{t-1}$, and $\hat{\Sigma}_j^{t-1}$ denote respectively the weight, mean, and covariance matrix of cluster j estimated at iteration $t-1$. Again, we artificially inflate the covariances with a factor $\lambda > 1$ in order to put more mass in the tails of the proposal distribution. In our practical applications we set $\lambda = 1.2$. Regarding J, we may either fix it arbitrarily or use the Variational Bayesian approach to choose it automatically.

In order to generate QMC or RQMC points from such a mixture distribution, we set the number of samples for each cluster j to $N_j^t = \left\lfloor \alpha_j^{t-1} N \right\rfloor$ and potentially adjust N_j^t as to make sure that $\sum_j N_j^t = N$ holds. For each cluster j, a (R)QMC sequence of length N_j^t is generated and transformed to the sample of a Gaussian distribution $\mathcal{N}(\theta|\hat{\mu}_j^{t-1}, \lambda\hat{\Sigma}_j^{t-1})$. This is achieved via the transformation of the (R)QMC sequence $(u_n)_{n\in 1:N_j^t}$ via the component-wise quantile function $\Phi^{-1}(\cdot)$: $\theta_n^t = \hat{\mu}_j^{t-1} + C_{t-1}\Phi^{-1}(u_n)$, where C_{t-1} is the Cholesky triangle of the covariance matrix: $C_{t-1}(C_{t-1})^T = \lambda\hat{\Sigma}_j^{t-1}$.

This approach has the following advantages. First, we maintain flexibility by allowing to cover several modes, as the posterior distribution might be multi–modal. Second, the use of a limited number of clusters makes sure that we can benefit from the better coverage of the space that comes from the use of (R)QMC sequences. Using

only a small number of clusters preserves the structure of the (R)QMC point set. Other approaches based on the inverse Rosenblatt transform (Gerber and Chopin, 2015) are computational more expensive. In contrast, using the approach of Sisson et al. (2009) would destroy the properties of the low discrepancy or scrambled net sequences and hence the variance reduction that comes from the (R)QMC sequence could vanish. (This has been found as a result of our simulation studies, not shown here.)

3.6.3 Adapting simultaneously ϵ_t and the number of simulations per parameter

As discussed in Section 3.2.2, the weights $\hat{L}_{\epsilon_t}(x_n^t)$ are unbiased estimators of the probabilities $P_{\theta_n^t}(\delta(y^\star, y) \leq \epsilon_t)$, which may be obtained in two ways: (a) as an average over a fixed number M of simulations; or (b) as a function of the number of simulations required so that k of them are at a ϵ distance of $y^\star$; that random number follows a negative binomial distribution.

So far, we have focused on (a), and even took $M = 1$ in our first set of numerical examples in Section 3.5. If we use this strategy, we may follow Del Moral et al. (2012) in adapting ϵ_t according to the ESS (effective sample size, Kong et al., 1994); i.e. at iteration t, once we have simulated the θ_n^t's and the x_n^t's, we solve numerically (using bisection) in ϵ_t the equation $\text{ESS} = \alpha N$, for $\alpha \in (0, 1)$, where

$$\text{ESS} = \frac{(\sum_{n=1}^N w_n^t)^2}{\sum_{n=1}^N (w_n^t)^2}$$

and $w_t^n = p(\theta_n^t)\hat{L}_\epsilon(x_n^t)/q(\theta_n^t)$, $\hat{L}_\epsilon(x_n^t) = M^{-1}\sum_{m=1}^M \mathbb{1}\{\delta(y^\star, y_{n,m}^t) \leq \epsilon_t\}$.

This approach usually works well during the first iterations of Algorithm 10, but it is bound to collapse as ϵ gets too small: as $\epsilon \to 0$, $P_\theta(\delta(y^\star, y) \leq \epsilon) \to 0$ whatever θ, and as a result most weights w_t^n become zero when ϵ_t is too small. One remedy is to set M to a much larger value, so that weights take much longer to collapse. However, this is expensive and wasteful, given that the first iterations would work well with a much smaller M.

In that sequential context, the negative binomial strategy for computing the weights becomes appealing, as it makes it possible to adapt automatically the CPU effort to a given ϵ: we may decrease ϵ_t at each iteration, while ensuring that the variance of the weights (as estimates of the probabilities $P_{\theta_n^t}(\delta(y^\star, y) \leq \epsilon_t)$ does not blow up. Of course, the price to pay is that iterations become more and more expensive.

In practice, we found that that this approach was unwieldy during the first iterations of the algorithm: during that time, a few simulated parameters θ_n^t are such that the corresponding probability that $\delta(y^\star, y) \leq \epsilon_t$ is much smaller than for the other particles. As a result, the negative binomial estimate requires generating a lot of observations for those particles, which typically gets discarded later.

Thus, in the end, we recommend the following hybrid strategy:

- At iterations $t = 0$ to $t = T_1$ (say $T_1 = 10$), use the 'fixed M' (say $M = 10$) strategy to compute the weights, and adapt ϵ_t using the ESS.
- At iterations $t > T_1$, switch to the negative binomial strategy for computing the weights, and adapt ϵ_t as follows: set it to the median of the distance values $\delta(y^\star, y_n)$ where the y_n's denote here all the artificial observations generating during the previous iteration such that $\delta(y^\star, y_n) \leq \epsilon_{t-1}$. Stop when ϵ_t gets below a certain target value $\epsilon^\star$.

3.7 Numerical illustration of the sequential procedure

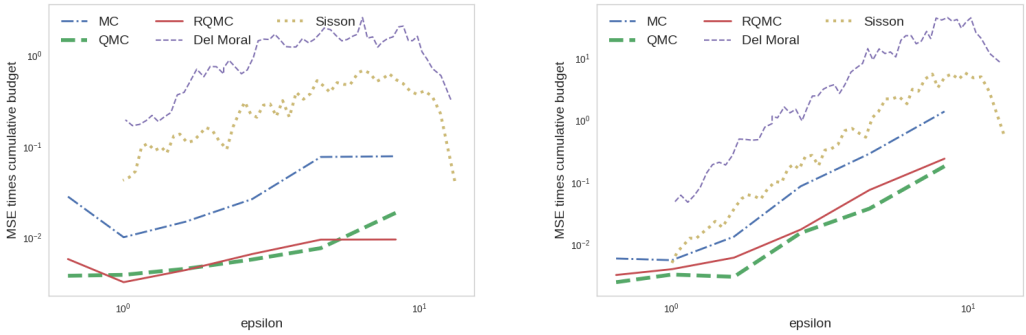
3.7.1 Toy model

We return to the toy model of Section 3.5.1, taking this time $d = 3$. We compare five algorithms: three versions of Algorithm 10 with the θ_n^t's generated using, respectively, Monte Carlo, Quasi-Monte Carlo, and RQMC; the sequential ABC algorithm of Sisson et al. (2009), which (as explained previously) is essentially Algorithm 10 with a mixture proposal with N components; and finally the algorithm of Del Moral et al. (2012). (The algorithm of Del Moral et al. (2012) generates the θ_n^t by evolving the particles resampled at the previous iteration through a Markov kernel; see the paper for more details.)

Regarding the adaptive choice of ϵ_t, we use the hybrid strategy outlined in the previous section for our MC, QMC and RQMC algorithms, we use the ESS-based strategy for Del Moral et al. (2012)'s algorithm, and we use the following strategy for Sisson et al. (2009)'s: ϵ_t is set to the median of the distances $\delta(y^\star, y_n)$ computed at the previous iteration. For all these algorithms, we set $M = 10$.

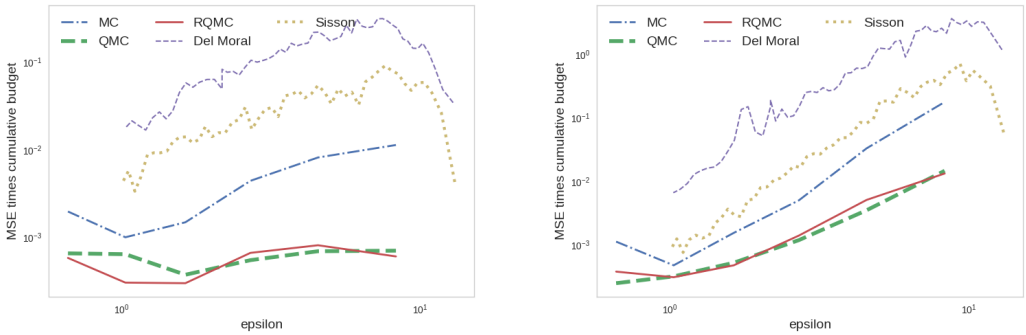
For this toy model, we simply consider the basic strategy for adapting q_t outlined in Section 3.6.2, i.e. q_t is a Gaussian fit to the previous set of particles. The five algorithms are run with either $N = 10^3$ (Figure 3.7) or $N = 10^4$ particles (Figure 3.8); in both cases the algorithms are stopped when $\epsilon_t \leq \epsilon^\star = 1$. In both figures, we plot the adjusted MSE at iteration t as a function of ϵ_t, where the adjusted MSE is the empirical MSE of a given estimate (over 50 runs) times the number of observations generated from the model up to time t. The adjusted MSE make it possible to account for the different running times of the algorithms. See also Table 3.2 for a direct comparison in terms of both CPU effort and MSE.

The considered estimates are the same as in Section 3.5.1, i.e. the ABC posterior expectation and variance of $\bar{\theta}$, the average of the components of vector θ. At least for posterior expectations, we see that the QMC and RQMC versions outperform the MC version of our algorithm, which in turn outperforms the sequential ABC algorithms of Sisson et al. (2009) and Del Moral et al. (2012).



(A) (B)

FIGURE 3.7: three-dimensional Gaussian toy example. Algorithms run with $N = 10^3$ particles. Adjusted MSE (as defined in the text) at iteration t, as function of ϵ_t, for the following posterior estimate: exceptation (left) and variance (right) of $\bar{\theta} = (\theta_1 + \theta_2 + \theta_3)/3$.



(A) (B)

FIGURE 3.8: Same as Figure 3.7, except algorithms are run with $N = 10^4$ particles.

Sampling method	MSE $\bar{\theta}$	MSE $\overline{\text{Var}\,\theta}$	number simulated datapoints	ϵ_T
AIS-MC	0.00162	0.00037	44,980	**0.65**
AIS-QMC	**0.00039**	0.00014	**32,919**	**0.65**
AIS-RQMC	0.00049	0.00013	42,088	**0.65**
Del Moral	0.00117	0.00018	580,000	1.0
Sisson	0.00117	**0.00010**	125,928	0.95
IS-MC	0.00128	0.00513	1,000,000	0.65

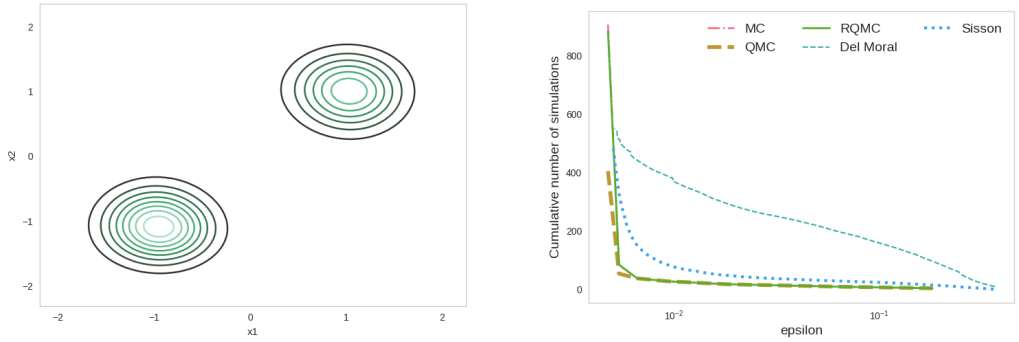
TABLE 3.2: Toy example, performance of the five considered sequential algorithms at the final iteration T, for $N = 10^3$ particles. IS-MC corresponds to the plain IS sampling without adaptation.

3.7.2 Bimodal Gaussian distribution

In order to illustrate the flexibility that comes from using a mixture of Gaussians for the proposal we consider a model that yields a multi-modal posterior:

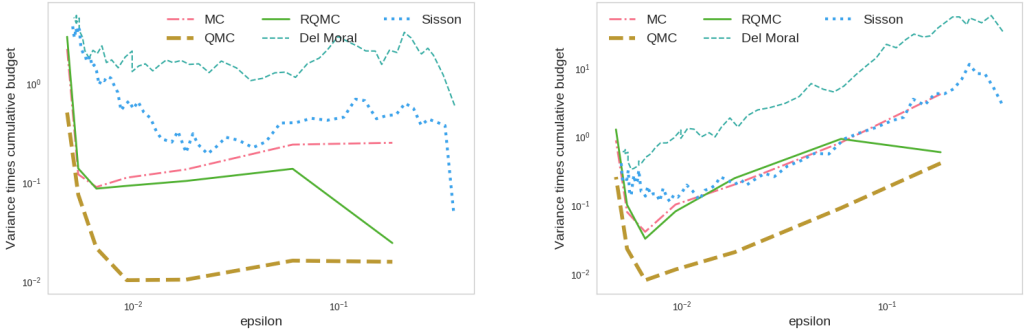
$$\begin{aligned} \theta &\sim \mathcal{U}\left([-10,10]^d\right), \\ y_i &\overset{iid}{\sim} \frac{1}{2}\mathcal{N}(\theta, I_d) + \frac{1}{2}\mathcal{N}(-\theta, I_d), \quad i = 1,\dots,100. \end{aligned}$$

We simulate $y^\star$ from the model. Throughout this application we set $d = 2$. The model is not identifiable and thus generates a bimodal posterior. Regarding the distance δ, we follow the idea of Bernton et al. (2017) and use the optimal transport distance between $\boldsymbol{y}$ and $\boldsymbol{y}^*$, more specifically the earth-movers-distance.



(A) (B)

FIGURE 3.9: Simulation for the bimodal distribution. Left: recovered posterior distribution. Right: average (over 50 runs) of cumulative number of simulations from the simulator across particles according to acceptance threshold; algorithms were run with $N = 10^3$ particles.



(A) (B)

FIGURE 3.10: Same plot as in Figure 3.7 for the bimodal example and $N = 10^3$.

We set $\epsilon^\star = 5 \times 10^{-3}$. This value has been chosen as before as a small quantile of the realized distances after 10^6 simulations from the prior and the simulator. The

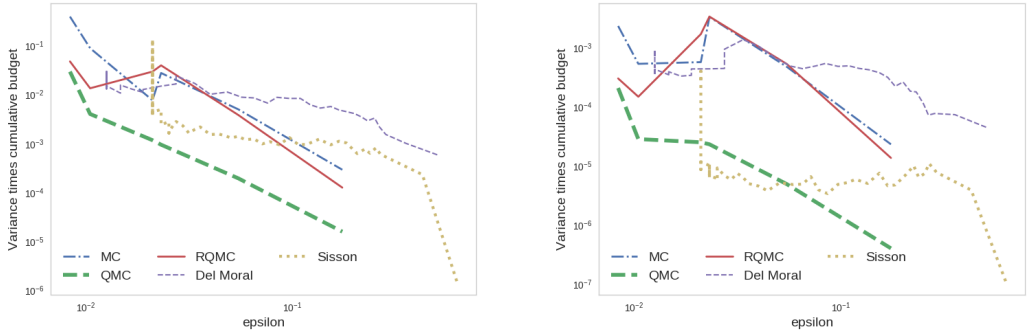
recovered posterior is shown in Figure 3.9a. Figure 3.9b illustrates the adaptivity in the simulation from the simulator achieved via the negative binomial approach. As the threshold becomes smaller and smaller, the number of necessary simulations start to increase severely. In the end, the number of necessary simulations of the different methods catch up with each other. Still, the approaches based on (R)QMC achieve a lower variance of the estimator as is illustrated in Figures 3.10a and 3.10b.

3.7.3 Tuberculosis mutation

We now return to the tuberculosis example presented in Section 3.5.3; we set the target value $\epsilon^\star = 0.01$, and restrict the CPU budget to 10^6 simulations from the model, as these simulations are computationally intensive. We see that again the QMC approach performs best in terms of number of simulations needed and also in terms of variance times computational budget; see Figures 3.11a and 3.11b, and Table 3.3. The approach of Sisson et al. (2009) exceeds the total computation budget and thus does not reach the fixed threshold. Figures 3.11a and 3.11b illustrate the effect of the hybrid strategy for adapting ϵ and the number of simulations per parameter value (Section 3.6.3). The kink in the lines for the adaptive importance sampling approaches corresponds to the moment when the weighting is obtained via the negative binomial distribution.

Sampling method	Variance $\overline{\theta}$	Variance $\overline{\mathrm{Var}\,\theta}$	number sim. datapoints	ϵ_T
AIS-MC	0.376	5.916×10^{-6}	419,353	0.008
AIS-QMC	0.380	1.156×10^{-6}	**212,183**	**0.008**
AIS-RQMC	0.378	1.001×10^{-6}	318,196	0.008
Del Moral	**0.375**	1.065×10^{-6}	495,000	0.010
Sisson	0.393	$\mathbf{1.834 \times 10^{-7}}$	1,367,949	0.021

TABLE 3.3: Tuberculosis example, performance of the five considered sequential algorithms at the final iteration T, for $N = 500$ particles



(A) (B)

FIGURE 3.11: Same plot as in Figure 3.7 for the tuberculosis example and $N = 500$ particles

3.8 Conclusion

In this paper we introduced the use of low discrepancy sequences in approximate Bayesian computation. We found that from both a theoretical and practical perspective the use of (R)QMC in ABC can yield substantial variance reduction of estimators based on the approximate posterior distribution. However, care must be taken when using (R)QMC sequences. First, the transformation of uniform sequences to the distribution of interest must preserve the low discrepancy properties of the point set. This is of major importance for a sequential version of the ABC algorithm that is based on adaptive importance sampling. Second, the advantage of using (R)QMC tends to diminish with increasing dimension. In the setting of ABC, however, the parameter space is often of low dimension such that the gains from using (R)QMC are visible in most applications, as shown by our simulations. From a practical perspective we recommend to use RQMC point sets instead of QMC as these allow the assessment of the error via repeated simulation and are unbiased.

Another contribution of this paper is the use of the negative binomial distribution for the sampling from the likelihood. We found that this approach reduces computation time significantly by increasing the computational load when the acceptance threshold decreases. If the user suspects a multimodal posterior, we recommend to estimate a mixture distribution based on the accepted samples and generate RQMC samples based on the mixture.

Acknowledgments

The research of the first author is funded by a GENES doctoral scholarship. The research of the second author is partially supported by a grant from the French National Research Agency (ANR) as part of the Investissements d'Avenir program (ANR-11-LABEX-0047). We are thankful to Mathieu Gerber, two anonymous referees and the editors who made comments that helped us to improve the paper.

3.9 Appendix

3.9.1 Proofs of main results

Proof of Corollary 1

For the mixed sequences we have

$$\lim_{N\to\infty} \frac{1}{N} \sum_{k=1}^{N} \sigma^2_{k,\text{qmc-mixed}} = C^2_{\text{qmc-mixed}}$$

and for the Monte Carlo estimate we have

$$\lim_{N\to\infty} \frac{1}{N} \sum_{k=1}^{N} \sigma_{k,\text{mc}}^2 = C_{\text{mc}}^2$$

where

$$C_{\text{qmc-mixed}}^2 = \int_{[0,1]^s} f(x)f(x)^T \,\mathrm{d}\, x - \int_{[0,1]^d} \left(\int_{[0,1]^{s-d}} f(u) \,\mathrm{d}\, X^{d+1:s} \right) \left(\int_{[0,1]^{s-d}} f(u) \,\mathrm{d}\, X^{d+1:s} \right)^T \mathrm{d}\, q^{1:d}$$

and

$$C_{\text{mc}}^2 = \int_{[0,1]^s} f(x)f(x)^T \,\mathrm{d}\, x - \left(\int_{[0,1]^s} f(x) \,\mathrm{d}\, x \right) \left(\int_{[0,1]^s} f(x) \,\mathrm{d}\, x \right)^T.$$

We must show that

$$\left(\int_{[0,1]^s} f(x) \,\mathrm{d}\, x \right) \left(\int_{[0,1]^s} f(x) \,\mathrm{d}\, x \right)^T \preceq \int_{[0,1]^d} \left(\int_{[0,1]^{s-d}} f(u) \,\mathrm{d}\, X^{d+1:s} \right) \left(\int_{[0,1]^{s-d}} f(u) \,\mathrm{d}\, X^{d+1:s} \right)^T \mathrm{d}\, q^{1:d},$$

in the sense of positive definite matrices. This inequality holds in the univariate case due to the Cauchy-Schwartz inequality. In the multivariate case, let $\int_{[0,1]^{s-d}} f(u^k) \,\mathrm{d}\, X^{d+1:s} = A\left(q^{1:d}\right)$. We rewrite:

$$\int_{[0,1]^d} A\left(q^{1:d}\right) \mathrm{d}\, q^{1:d} \int_{[0,1]^d} A\left(q^{1:d}\right)^T \mathrm{d}\, q^{1:d} \preceq \int_{[0,1]^d} A\left(q^{1:d}\right) A\left(q^{1:d}\right)^T \mathrm{d}\, q^{1:d}.$$

In order to check the positive definiteness let $v \in \mathbb{R}^s$. We check

$$\begin{aligned} v^T \int_{[0,1]^d} A\left(q^{1:d}\right) \mathrm{d}\, q^{1:d} \int_{[0,1]^d} A\left(q^{1:d}\right)^T \mathrm{d}\, q^{1:d} v &\leq v^T \int_{[0,1]^d} A\left(q^{1:d}\right) A\left(q^{1:d}\right)^T \mathrm{d}\, q^{1:d} v, \\ \int_{[0,1]^d} v^T A\left(q^{1:d}\right) \mathrm{d}\, q^{1:d} \int_{[0,1]^d} A\left(q^{1:d}\right)^T v \,\mathrm{d}\, q^{1:d} &\leq \int_{[0,1]^d} v^T A\left(q^{1:d}\right) A\left(q^{1:d}\right)^T v \,\mathrm{d}\, q^{1:d}. \end{aligned}$$

While noting that $v^T A\left(q^{1:d}\right) \in \mathbb{R}$ and $A\left(q^{1:d}\right)^T v \in \mathbb{R}, \forall v \in \mathbb{R}^s$ we are back in the univariate case and the inequality holds. □

Proof of Theorem 3

The statement of the theorem is equivalent to $\lim_{N\to\infty} |\mathbb{P}(T_N \leq t) - \mathbb{P}(Z \leq t)| = 0$ for all $t \in \mathbb{R}^s$, $T_N = N^{1/2} S_N^{RQMC}$, and Z a random variable distributed according to the Gaussian limit.

When conditioning on the random element V in the RQMC sequence, we have that

$$\lim_{N\to\infty} \mathbb{P}(T_N \leq t | V = v) = \mathbb{P}(Z \leq t)$$

for almost all v, by Theorem 2, as a RQMC sequence is a QMC sequence with probability one. Furthermore, $|\mathbb{P}(T_N \leq t | V = v)| \leq 1$, thus the function is dominated. For

all N we have

$$|\mathbb{P}(T_N \leq t) - \mathbb{P}(Z \leq t)| = \left| \int_{\mathcal{B}} \{\mathbb{P}(T_N \leq t|V = v) - \mathbb{P}(Z \leq t)\} \, \mathrm{d}\, \mathbb{P}(v) \right|,$$
$$\leq \int_{\mathcal{B}} |\mathbb{P}(T_N \leq t|b) - \mathbb{P}(Z \leq t)| \, \mathrm{d}\, \mathbb{P}(v).$$

And

$$\lim_{N\to\infty} \int_{\mathcal{B}} |\mathbb{P}(T_N \leq t|V = v) - \mathbb{P}(Z \leq t)| \, d\mathbb{P}(v) = 0,$$

due to the dominated convergence theorem. Therefore

$$\lim_{N\to\infty} |\mathbb{P}(T_N \leq t) - \mathbb{P}(Z \leq t)| = 0.$$

□

Proof of Proposition 2

Since the θ_n's are deterministic,

$$\mathbb{E}\left[\hat{Z}_N\right] = \frac{1}{N}\sum_{n=1}^{N} \frac{p(\theta_n)}{q(\theta_n)} \mathbb{P}_{\theta_n}\left(\delta(y, y^\star) \leq \epsilon\right) = \frac{1}{N}\sum_{n=1}^{N} f(\theta_n)$$
$$\mathrm{Var}[\hat{Z}_N] = \frac{1}{MN^2}\sum_{n=1}^{N} \left\{\frac{p(\theta_n)}{q(\theta_n)}\right\}^2 \mathbb{P}_{\theta_n}\left(\delta(y, y^\star) \leq \epsilon\right)\left\{1 - \mathbb{P}_{\theta_n}\left(\delta(y, y^\star) \leq \epsilon\right)\right\}$$

and $\left|\mathbb{E}\left[\hat{Z}_N\right] - Z_\epsilon\right| = \mathcal{O}(N^{\tau-1})$ for any $\tau > 0$, by Koksma-Hlawka inequality. By the standard decomposition of the mean square error:

$$\mathbb{E}\left[(\hat{Z}_N - Z_\epsilon)^2\right] = \left(\mathbb{E}\left[\hat{Z}_N\right] - Z_\epsilon\right)^2 + \mathrm{Var}\left[\hat{Z}_N\right]$$

and since $p(\theta_n)/q(\theta_n) \leq C$, we see that that the MSE times M is $\mathcal{O}(N^{-1})$.

Proof of Theorem 4

One has:

$$(\hat{\phi}_N - \mathbb{E}_{p_\epsilon}\phi) = \left(\frac{\sum_{n=1}^{N} w_n \phi(\theta_n)}{\sum_{n=1}^{N} w_n} - \mathbb{E}_{p_\epsilon}\phi\right)$$
$$= \frac{N^{-1}\sum_{n=1}^{N} w_n \bar{\phi}(\theta_n)}{N^{-1}\sum_{n=1}^{N} w_n}$$

where $\bar{\phi} = \phi - \mathbb{E}_{p_\epsilon}\phi$. Since the denominator converges almost surely to Z_ϵ, and the numerator (times $N^{1/2}$) converges to a Gaussian limit (per Theorem 2), we may apply Slutsky's theorem to obtain the desired result.

More precisely, the numerator has a null expectation, and is such that

$$N^{-1/2} \sum_{n=1}^{N} w_n \bar{\phi}(\theta_n) \xrightarrow{\mathcal{L}} \mathcal{N}\left(0, \tau^2(\phi)\right)$$

where

$$\tau^2(\phi) = \int_{\Theta} \frac{p(\theta)^2}{q(\theta)} \bar{\phi}(\theta)^2 \frac{b(\theta_n)\{1 - b(\theta_n)\}}{M} \, \mathrm{d}\,\theta$$

again by direct application of Theorem 2, and using the fact that, for a fixed θ_n,

$$\mathrm{Var}_{x_n \sim q_{\theta_n}}[\hat{L}_\epsilon(x_n)] = \frac{b(\theta_n)\{1 - b(\theta_n)\}}{M}$$

with $b(\theta) = \mathbb{P}_\theta\left(\delta(y, y^\star) \leq \epsilon\right)$. □

Chapter 4

Quasi-Monte Carlo Variational Inference

Joint work with Florian Wenzel and Stephan Mandt, appeared in *Proceedings of the International Conference on Machine Learning, 2018*

Abstract: Many machine learning problems involve Monte Carlo gradient estimators. As a prominent example, we focus on Monte Carlo variational inference (MCVI) in this paper. The performance of MCVI crucially depends on the variance of its stochastic gradients. We propose variance reduction by means of Quasi-Monte Carlo (QMC) sampling. QMC replaces N i.i.d. samples from a uniform probability distribution by a deterministic sequence of samples of length N. This sequence covers the underlying random variable space more evenly than i.i.d. draws, reducing the variance of the gradient estimator. With our novel approach, both the score function and the reparameterization gradient estimators lead to much faster convergence. We also propose a new algorithm for Monte Carlo objectives, where we operate with a constant learning rate and increase the number of QMC samples per iteration. We prove that this way, our algorithm can converge asymptotically at a faster rate than SGD. We furthermore provide theoretical guarantees on QMC for Monte Carlo objectives that go beyond MCVI, and support our findings by several experiments on large-scale data sets from various domains.

Keywords: Quasi-Monte Carlo, Black Box Variational Inference, Bayesian Approximate Inference

4.1 Introduction

In many situations in machine learning and statistics, we encounter objective functions which are expectations over continuous distributions. Among other examples, this situation occurs in reinforcement learning (Sutton and Barto, 1998) and variational inference (Jordan et al., 1999). If the expectation cannot be computed in closed form, an approximation can often be obtained via Monte Carlo (MC) sampling from the underlying distribution. As most optimization procedures rely on the gradient of the objective, a MC gradient estimator has to be built by sampling from this distribution.

The finite number of MC samples per gradient step introduces noise. When averaging over multiple samples, the error in approximating the gradient can be decreased, and thus its variance reduced. This guarantees stability and fast convergence of stochastic gradient descent (SGD).

Certain objective functions require a large number of MC samples per stochastic gradient step. As a consequence, the algorithm gets slow. It is therefore desirable to obtain the same degree of variance reduction with fewer samples. This paper proposes the idea of using Quasi-Monte Carlo (QMC) samples instead of i.i.d. samples to achieve this goal.

A QMC sequence is a deterministic sequence which covers a hypercube $[0,1]^d$ more regularly than random samples. When using a QMC sequence for Monte Carlo integration, the mean squared error (MSE) decreases asymptotically with the number of samples N as $\mathcal{O}(N^{-2}(\log N)^{2d-2})$ (Leobacher and Pillichshammer, 2014). In contrast, the naive MC integration error decreases as $\mathcal{O}(N^{-1})$. Since the cost of generating N QMC samples is $\mathcal{O}(N \log N)$, this implies that a much smaller number of operations per gradient step is required in order to achieve the same precision (provided that N is large enough). Alternatively, we can achieve a larger variance reduction with the same number of samples, allowing for larger gradient steps and therefore also faster convergence. This paper investigates the benefits of this approach both experimentally and theoretically.

Our ideas apply in the context of Monte Carlo variational inference (MCVI), a set of methods which make approximate Bayesian inference scalable and easy to use. Variance reduction is an active area of research in this field. Our algorithm has the advantage of being very general; it can be easily implemented in existing software packages such as STAN and Edward Carpenter et al. (2017); Tran et al. (2016). In Appendix 4.9 we show how our approach can be easily implemented in your existing code.

The main contributions are as follows:

- We investigate the idea of using QMC sequences for Monte Carlo variational inference. While the usage of QMC for VI has been suggested in the outlook section of Ranganath et al. (2014), to our knowledge, we are the first to actually investigate this approach both theoretically and experimentally.

- We show that when using a randomized version of QMC (RQMC), the resulting stochastic gradient is unbiased and its variance is asymptotically reduced. We also show that when operating SGD with a constant learning rate, the stationary variance of the iterates is reduced by a factor of N, allowing us to get closer to the optimum.

- We propose an algorithm which operates at a constant learning rate, but increases the number of RQMC samples over iterations. We prove that this algorithm has a better asymptotic convergence rate than SGD.

- Based on three different experiments and for two popular types of gradient estimators we illustrate that our method allows us to train complex models several orders of magnitude faster than with standard MCVI.

Our paper is structured as follows. Section 4.2 reviews related work. Section 4.3 explains our method and exhibits our theoretical results. In Section 5.4 we describe our experiments and show comparisons to other existing approaches. Finally, Section 4.5 concludes and lays out future research directions.

4.2 Related Work

Monte Carlo Variational Inference (MCVI) Since the introduction of the score function (or REINFORCE) gradient estimator for variational inference Paisley et al. (2012); Ranganath et al. (2014), Monte Carlo variational inference has received an ever-growing attention, see Zhang et al. (2017a) for a recent review. The introduction of the gradient estimator made VI applicable to non-conjugate models but highly depends on the variance of the gradient estimator. Therefore various variance reduction techniques have been introduced; for example Rao-Blackwellization and control variates, see Ranganath et al. (2014) and importance sampling, see Ruiz et al. (2016a); Burda et al. (2016).

At the same time the work of Kingma and Welling (2014); Rezende et al. (2014) introduced reparameterization gradients for MCVI, which typically exhibits lower variance but are restricted to models where the variational family can be reparametrized via a differentiable mapping. In this sense MCVI based on score function gradient estimators is more general but training the algorithm is more difficult. A unifying view is provided by Ruiz et al. (2016b). Miller et al. (2017) introduce a modification of the reparametrized version, but relies itself on assumptions on the underlying variational family. Roeder et al. (2017) propose a lower variance gradient estimator by omitting a term of the ELBO. The idea of using QMC in order to reduce the variance has been suggested by Ranganath et al. (2014) and Ruiz et al. (2016a) and used for a specific model by Tran et al. (2017b), but without a focus on analyzing or benchmarking the method.

Quasi-Monte Carlo and Stochastic Optimization Besides the generation of random samples for approximating posterior distributions (Robert and Casella, 2013), Monte Carlo methods are used for calculating expectations of intractable integrals via the law of large numbers. The error of the integration with random samples goes to zero at a rate of $\mathcal{O}(N^{-1})$ in terms of the MSE. For practical application this rate can be too slow. Faster rates of convergence in reasonable dimensions can be obtained by replacing the randomness by a deterministic sequence, also called Quasi-Monte Carlo.

Compared to Monte Carlo and for sufficiently regular functions, QMC reaches a faster rate of convergence of the approximation error of an integral. Niederreiter

(1992); L'Ecuyer and Lemieux (2005); Leobacher and Pillichshammer (2014); Dick et al. (2013) provide excellent reviews on this topic. From a theoretical point of view, the benefits of QMC vanish in very high dimensions. Nevertheless, the error bounds are often too pessimistic and in practice, gains are observed up to dimension 150, see Glasserman (2013).

QMC has frequently been used in financial applications (Glasserman, 2013; Joy et al., 1996; Lemieux and L'Ecuyer, 2001). In statistics, some applications include particle filtering (Gerber and Chopin, 2015), approximate Bayesian computation (Buchholz and Chopin, 2017), control functionals (Oates and Girolami, 2016) and Bayesian optimal design (Drovandi and Tran, 2018). Yang et al. (2014) used QMC in the context of large scale kernel methods.

Stochastic optimization has been pioneered by the work of Robbins and Monro (1951). As stochastic gradient descent suffers from noisy gradients, various approaches for reducing the variance and adapting the step size have been introduced (Johnson and Zhang, 2013; Kingma and Ba, 2015; Defazio et al., 2014; Duchi et al., 2011; Zhang et al., 2017b). Extensive theoretical results on the convergence of stochastic gradients algorithms are provided by Moulines and Bach (2011). Mandt et al. (2017) interpreted stochastic gradient descent with constant learning rates as approximate Bayesian inference. Some recent reviews are for example Bottou et al. (2016); Nesterov (2013). Naturally, concepts from QMC can be beneficial to stochastic optimization. Contributions on exploiting this idea are e.g. Gerber and Bornn (2017) and Drew and Homem-de Mello (2006).

4.3 Quasi-Monte Carlo Variational Inference

In this Section, we introduce Quasi-Monte Carlo Variational Inference (QMCVI), using randomized QMC (RQMC) for variational inference. We review MCVI in Section 4.3.1. RQMC and the details of our algorithm are exposed in Section 4.3.2. Theoretical results are given in Section 4.3.3.

4.3.1 Background: Monte Carlo Variational Inference

Variational inference (VI) is key to modern probabilistic modeling and Bayesian deep learning (Jordan et al., 1999; Blei et al., 2017; Zhang et al., 2017a). In Bayesian inference, the object of interest is a posterior distribution of latent variables $\mathbf{z}$ given observations $\mathbf{x}$. VI approximates Bayesian inference by an optimization problem which we can solve by (stochastic) gradient ascent (Jordan et al., 1999; Hoffman et al., 2013).

In more detail, VI builds a tractable approximation of the posterior $p(\mathbf{z}|\mathbf{x})$ by minimizing the KL-divergence between a variational family $q(\mathbf{z}|\lambda)$, parametrized by free

parameters $\lambda \in \mathbb{R}^d$, and $p(\mathbf{z}|\mathbf{x})$. This is equivalent to maximizing the so-called evidence lower bound (ELBO):

$$\mathcal{L}(\lambda) = \mathbb{E}_{q(\mathbf{z}|\lambda)}[\log p(\mathbf{x}, \mathbf{z}) - \log q(\mathbf{z}|\lambda)]. \tag{4.1}$$

In classical variational inference, the expectations involved in (4.1) are carried out analytically (Jordan et al., 1999). However, this is only possible for the fairly restricted class of so-called conditionally conjugate exponential family models (Hoffman et al., 2013). More recently, black-box variational methods have gained momentum, which make the analytical evaluation of these expectation redundant, and which shall be considered in this paper.

Maximizing the objective (4.1) is often based on a gradient ascent scheme. However, a direct differentiation of the objective (4.1) with respect to λ is not possible, as the measure of the expectation depends on this parameter. The two major approaches for overcoming this issue are the score function estimator and the reparameterization estimator.

Score Function Gradient The score function gradient (also called REINFORCE gradient) (Ranganath et al., 2014) expresses the gradient as expectation with respect to $q(\mathbf{z}|\lambda)$ and is given by

$$\nabla_\lambda \mathcal{L}(\lambda) = \mathbb{E}_{q(\mathbf{z}|\lambda)}[\nabla_\lambda \log q(\mathbf{z}|\lambda) \, (\log p(\mathbf{x}, \mathbf{z}) - \log q(\mathbf{z}|\lambda))]. \tag{4.2}$$

The gradient estimator is obtained by approximating the expectation with independent samples from the variational distribution $q(\mathbf{z}|\lambda)$. This estimator applies to continuous and discrete variational distributions.

Reparameterization Gradient The second approach is based on the reparametrization trick (Kingma and Welling, 2014), where the distribution over $\mathbf{z}$ is expressed as a deterministic transformation of another distribution over a noise variable ε, hence $\mathbf{z} = g_\lambda(\varepsilon)$ where $\varepsilon \sim p(\varepsilon)$. Using the reparameterization trick, the ELBO is expressed as expectation with respect to $p(\varepsilon)$ and the derivative is moved inside the expectation:

$$\nabla_\lambda \mathcal{L}(\lambda) = \mathbb{E}_{p(\varepsilon)}[\nabla_\lambda \log p(\mathbf{x}, g_\lambda(\varepsilon)) - \nabla_\lambda \log q(g_\lambda(\varepsilon)|\lambda)]. \tag{4.3}$$

The expectation is approximated using a MC sum of independent samples from $p(\varepsilon)$. In its basic form, the estimator is restricted to distributions over continuous variables.

MCVI In the general setup of MCVI considered here, the gradient of the ELBO is represented as an expectation $\nabla_\lambda \mathcal{L}(\lambda)$
$= \mathbb{E}[g_{\tilde{\mathbf{z}}}(\lambda)]$ over a random variable $\tilde{\mathbf{z}}$. For the score function estimator we choose g according to Equation (4.2) with $\tilde{\mathbf{z}} = \mathbf{z}$ and for the reparameterization gradient according to Equation (4.3) with $\tilde{\mathbf{z}} = \varepsilon$, respectively. This allows us to obtain a stochastic

estimator of the gradient by an average over a finite sample $\{\tilde{\mathbf{z}}_1, \cdots, \tilde{\mathbf{z}}_N\}$ as $\hat{g}_N(\lambda_t) = (1/N)\sum_{i=1}^N g_{\tilde{\mathbf{z}}_i}(\lambda_t)$.This way, the ELBO can be optimized by stochastic optimization. This is achieved by iterating the SGD updates with decreasing step sizes α_t:

$$\lambda_{t+1} = \lambda_t + \alpha_t \hat{g}_N(\lambda_t). \tag{4.4}$$

The convergence of the gradient ascent scheme in (4.4) tends to be slow when gradient estimators have a high variance. Therefore, various approaches for reducing the variance of both gradient estimators exist; e.g. control variates (CV), Rao-Blackwellization and importance sampling. However these variance reduction techniques do not improve the $\mathcal{O}(N^{-1})$ rate of the MSE of the estimator, except under some restrictive conditions (Oates et al., 2017). Moreover, the variance reduction schemes must often be tailored to the problem at hand.

4.3.2 Quasi-Monte Carlo Variational Inference

Quasi Monte Carlo Low discrepancy sequences, also called QMC sequences, are used for integrating a function ψ over the $[0,1]^d$ hypercube. When using standard i.i.d. samples on $[0,1]^d$, the error of the approximation is $\mathcal{O}(N^{-1})$. QMC achieves a rate of convergence in terms of the MSE of $\mathcal{O}\left(N^{-2}(\log N)^{2d-2}\right)$ if ψ is sufficiently regular (Leobacher and Pillichshammer, 2014). This is achieved by a deterministic sequence that covers $[0,1]^d$ more evenly.

On a high level, QMC sequences are constructed such that the number of points that fall in a rectangular volume is proportional to the volume. This idea is closely linked to stratification. Halton sequences e.g. are constructed using coprime numbers (Halton, 1964). Sobol sequences are based on the reflected binary code (Antonov and Saleev, 1979). The exact construction of QMC sequences is quite involved and we refer to Niederreiter (1992); Leobacher and Pillichshammer (2014); Dick et al. (2013) for more details.

The approximation error of QMC increases with the dimension, and it is difficult to quantify. Carefully reintroducing randomness while preserving the structure of the sequence leads to randomized QMC. RQMC sequences are unbiased and the error can be assessed by repeated simulation. Moreover, under slightly stronger regularity conditions on F we can achieve rates of convergence of $\mathcal{O}(N^{-2})$ (Gerber, 2015). For illustration purposes, we show different sequences in Figure 4.1. In Appendix 4.6 we provide more technical details.

QMC or RQMC can be used for integration with respect to arbitrary distributions by transforming the initial sequence on $[0,1]^d$ via a transformation Γ to the distribution of interest. Constructing the sequence typically costs $\mathcal{O}(N \log N)$ (Gerber and Chopin, 2015).

QMC and VI We suggest to replace N independent MC samples for computing $\hat{g}_N(\lambda_t)$ by an RQMC sequence of the same length. With our approach, the variance

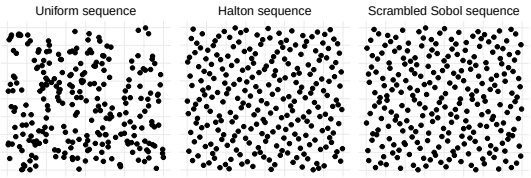


FIGURE 4.1: MC (left), QMC (center) and RQMC (right) sequences of length $N = 256$ on $[0,1]^2$. QMC and RQMC tend to cover the target space more evenly.

of the gradient estimators becomes $\mathcal{O}(N^{-2})$, and the costs for creating the sequence is $\mathcal{O}(N \log N)$. The incorporation of RQMC in VI is straightforward: instead of sampling $\tilde{\mathbf{z}}$ as independent MC samples, we generate a uniform RQMC sequence $u_1, \cdots, u_N$ and transform this sequence via a mapping Γ to the original random variable $\tilde{\mathbf{z}} = \Gamma(u)$. Using this transformation we obtain the RQMC gradient estimator

$$\hat{g}_N(\lambda_t) = (1/N) \sum_{i=1}^{N} g_{\Gamma(u_i)}(\lambda). \tag{4.5}$$

From a theoretical perspective, the function $u \mapsto g_{\Gamma(u)}(\lambda)$ has to be sufficiently smooth for all λ. For commonly used variational families this transformation is readily available. Although evaluating these transforms adds computational overhead, we found this cost negligible in practice. For example, in order to sample from a multivariate Gaussian $\mathbf{z}_n \sim \mathcal{N}(\mu, \Sigma)$, we generate an RQMC squence u_n and apply the transformation $\mathbf{z}_n = \Phi^{-1}(u_n)\Sigma^{1/2} + \mu$, where $\Sigma^{1/2}$ is the Cholesky decomposition of Σ and Φ^{-1} is the component-wise inverse cdf of a standard normal distribution. Similar procedures are easily obtained for exponential, Gamma, and other distributions that belong to the exponential family. Algorithm 11 summarizes the procedure.

Algorithm 11: Quasi-Monte Carlo Variational Inference

Input: Data $\mathbf{x}$, model $p(\mathbf{x}, \mathbf{z})$, variational family $q(\mathbf{z}|\lambda)$
Result: Variational parameters λ^*

while *not converged* **do**
- Generate uniform RQMC sequence $u_{1:N}$
- Transform the sequence via Γ
- Estimate the gradient $\hat{g}_N(\lambda_t) = \frac{1}{N} \sum_{i=1}^{N} g_{\Gamma(u_i)}(\lambda_t)$
- Update $\lambda_{t+1} = \lambda_t + \alpha_t \, \hat{g}_N(\lambda_t)$

RQMC samples can be generated via standard packages such as `randtoolbox` (Christophe and Petr, 2015), available in R. Existing MCVI algorithms are adapted by replacing the random variable sampler by an RQMC version. Our approach reduces the variance in MCVI and applies in particular to the reparametrization gradient estimator and the score function estimator. RQMC can in principle be combined with additional variance reduction techniques such as CV, but care must be taken as the optimal CV for RQMC are not the same as for MC (Hickernell et al., 2005).

4.3.3 Theoretical Properties of QMCVI

In what follows we give a theoretical analysis of using RQMC in stochastic optimization. Our results apply in particular to VI but are more general.

QMCVI leads to faster convergence in combination with Adam (Kingma and Ba, 2015) or Adagrad (Duchi et al., 2011), as we will show empirically in Section 5.4. Our analysis, presented in this section, underlines this statement for the simple case of SGD with fixed step size in the Lipschitz continuous (Theorem 5) and strongly convex case (Theorem 6). We show that for N sufficiently large, SGD with RQMC samples reaches regions closer to the true optimizer of the ELBO. Moreover, we obtain a faster convergence rate than SGD when using a fixed step size and increasing the sample size over iterations (Theorem 7).

RQMC for Optimizing Monte Carlo Objectives We step back from black box variational inference and consider the more general setup of optimizing Monte Carlo objectives. Our goal is to minimize a function $F(\lambda)$, where the optimizer has only access to a noisy, unbiased version $\hat{F}_N(\lambda)$, with $\mathbb{E}[\hat{F}_N(\lambda)] = F(\lambda)$ and access to an unbiased noisy estimator of the gradients $\hat{g}_N(\lambda)$, with $\mathbb{E}[\hat{g}_N(\lambda)] = \nabla F(\lambda)$. The optimum of $F(\lambda)$ is $\lambda^\star$.

We furthermore assume that the gradient estimator $\hat{g}_N(\lambda)$ has the form as in Eq. 4.5, where Γ is a reparameterization function that converts uniform samples from the hypercube into samples from the target distribution. In this paper, $u_1, \cdots, u_N$ is an RQMC sequence.

In the following theoretical analysis, we focus on SGD with a constant learning rate α. The optimal value $\lambda^\star$ is approximated by SGD using the update rule

$$\lambda_{t+1} = \lambda_t - \alpha \hat{g}_N(\lambda_t). \tag{4.6}$$

Starting from λ_1 the procedure is iterated until $|\hat{F}_N(\lambda_t) - \hat{F}_N(\lambda_{t+1})| \leq \epsilon$, for a small threshold ϵ. The quality of the approximation $\lambda_T \approx \lambda^\star$ crucially depends on the variance of the estimator $\hat{g}_N$ (Johnson and Zhang, 2013).

Intuitively, the variance of $\hat{g}_N(\lambda)$ based on an RQMC sequence will be $\mathcal{O}(N^{-2})$ and thus for N large enough, the variance will be smaller than for the MC counterpart, that is $\mathcal{O}(N^{-1})$. This will be beneficial to the optimization procedure defined in (4.6). Our following theoretical results are based on standard proof techniques for stochastic approximation, see e.g. Bottou et al. (2016).

Stochastic Gradient Descent with Fixed Step Size In the case of functions with Lipschitz continuous derivatives, we obtain the following upper bound on the norm of the gradients.

Theorem 5 *Let F be a function with Lipschitz continuous derivatives, i.e. there exists $L > 0$ s.t. $\forall \lambda, \overline{\lambda} \, \|\nabla F(\lambda) - \nabla F(\overline{\lambda})\|_2^2 \leq L\|\lambda - \overline{\lambda}\|_2^2$, let $U_N = \{u_1, \cdots, u_N\}$ be an* RQMC *sequence*

and let $\forall\lambda$, $G : u \mapsto g_{\Gamma(u)}(\lambda)$ *has cross partial derivatives of up to order* d. *Let the constant learning rate* $\alpha < 2/L$ *and let* $\mu = 1 - \alpha L/2$. *Then* $\forall\lambda, \operatorname{tr}\operatorname{Var}_{U_N}[\hat{g}_N(\lambda)] \leq M_V \times r(N)$, *where* $M_V < \infty$ *and* $r(N) = \mathcal{O}\left(N^{-2}\right)$ *and*

$$\frac{\sum_{t=1}^{T} \mathbb{E}\|\nabla F(\lambda_t)\|_2^2}{T} \leq \frac{1}{2\mu}\alpha L M_V r(N) + \frac{F(\lambda_1) - F(\lambda^\star)}{\alpha\mu T},$$

where λ_t *is iteratively defined in* (4.6). *Consequently,*

$$\lim_{T\to\infty} \frac{\sum_{t=1}^{T} \mathbb{E}\|\nabla F(\lambda_t)\|_2^2}{T} \leq \frac{1}{2\mu}\alpha L M_V r(N). \tag{4.7}$$

Equation (4.7) underlines the dependence of the sum of the norm of the gradients on the variance of the gradients. The better the gradients are estimated, the closer one gets to the optimum where the gradient vanishes. As the dependence on the sample size becomes $\mathcal{O}\left(N^{-2}\right)$ for an RQMC sequence instead of $1/N$ for a MC sequence, the gradient is more precisely estimated for N large enough.

We now study the impact of a reduced variance on SGD with a fixed step size and strongly convex functions. We obtain an improved upper bound on the optimality gap.

Theorem 6 *Let* F *have Lipschitz continuous derivatives and be a strongly convex function, i.e. there exists a constant* $c > 0$ *s.t.* $\forall\lambda, \overline{\lambda}\ F(\overline{\lambda}) \geq F(\lambda) + \nabla F(\lambda)^T(\lambda - \overline{\lambda}) + \frac{1}{2}c\|\lambda - \overline{\lambda}\|_2^2$, *let* $U_N = \{u_1, \cdots, u_N\}$ *be an* RQMC *sequence and let* $\forall\lambda, G : u \mapsto g_{\Gamma(u)}(\lambda)$ *be as in Theorem 5. Let the constant learning rate* $\alpha < \frac{1}{2c}$ *and* $\alpha < \frac{2}{L}$. *Then the expected optimality gap satisfies,* $\forall t \geq 0$,

$$\mathbb{E}[F(\lambda_{t+1}) - F(\lambda^\star)] \leq \left[\left(\frac{\alpha^2 L}{2} - \alpha\right) 2c + 1\right] \times \mathbb{E}[F_N(\lambda_t) - F(\lambda^\star)] + \frac{1}{2}L\alpha^2 \left[M_V r(N)\right].$$

Consequently,

$$\lim_{T\to\infty} \mathbb{E}[F(\lambda_T) - F(\lambda^\star)] \leq \frac{\alpha L}{4c - \alpha L c} \left[M_V r(N)\right].$$

The previous result has the following interpretation. The expected optimality gap between the last iteration λ_T and the true minimizer $\lambda^\star$ is upper bounded by the magnitude of the variance. The smaller this variance, the closer we get to $\lambda^\star$. Using RQMC we gain a factor $1/N$ in the bound.

Increasing Sample Size Over Iterations While SGD with a fixed step size and a fixed number of samples per gradient step does not converge, convergence can be achieved when increasing the number of samples used for estimating the gradient over iterations. As an extension of Theorem 6, we show that a linear convergence is obtained while increasing the sample size at a slower rate than for MC sampling.

Theorem 7 *Assume the conditions of Theorem 6 with the modification* $\alpha \leq \min\{1/c, 1/L\}$. *Let* $1 - \alpha c/2 < \xi^2 = \frac{1}{\tau^2} < 1$. *Use an increasing sample size* $N_t = \underline{N} + \lceil \tau^t \rceil$, *where* $\underline{N} < \infty$ *is defined in Appendix 4.7.3. Then* $\forall t \in \mathbb{N}, \exists \hat{M}_V < \infty$,

$$\operatorname{tr} \operatorname{Var}_{U_N}[\hat{g}_{N_t}(\lambda)] \leq \hat{M}_V \times \frac{1}{\tau^{2t}}$$

and

$$\mathbb{E}[F(\lambda_{t+1}) - F(\lambda^\star)] \leq \omega \xi^{2t},$$

where $\omega = \max\{\alpha L \hat{M}_V / c, F(\lambda_1) - F(\lambda^\star)\}$.

This result compares favorably with a standard result on the linear convergence of SGD with fixed step size and strongly convex functions (Bottou et al., 2016). For MC sampling one obtains a different constant $\tilde{\omega}$ and an upper bound with ξ^t and not ξ^{2t}. Thus, besides the constant factor, RQMC samples allow us to close the optimality gap faster for the same geometric increase in the sample size τ^t or to use $\tau^{t/2}$ to obtain the same linear rate of convergence as MC based estimators.

Other Remarks The reduced variance in the estimation of the gradients should allow us to make larger moves in the parameter space. This is for example achieved by using adaptive step size algorithms as Adam (Kingma and Ba, 2015), or Adagrad (Duchi et al., 2011). However, the theoretical analysis of these algorithms is beyond the scope of this paper.

Also, note that it is possible to relax the smoothness assumptions on G while supposing only square integrability. Then one obtains rates in $o(N^{-1})$. Thus, RQMC yields always a faster rate than MC, regardless of the smoothness. See Appendix 4.6 for more details.

In the previous analysis, we have assumed that the entire randomness in the gradient estimator comes from the sampling of the variational distribution. In practice, additional randomness is introduced in the gradient via mini batch sampling. This leads to a dominating term in the variance of $\mathcal{O}(K^{-1})$ for mini batches of size K. Still, the part of the variance related to the variational family itself is reduced and so is the variance of the gradient estimator as a whole.

4.4 Experiments

We study the effectiveness of our method in three different settings: a hierarchical linear regression, a multi-level Poisson generalized linear model (GLM) and a Bayesian neural network (BNN). Finally, we confirm the result of Theorem 7, which proposes to increase the sample size over iterations in QMCVI for faster asymptotic convergence.

Setup In the first three experiments we optimize the ELBO using the Adam optimizer (Kingma and Ba, 2015) with the initial step size set to 0.1, unless otherwise

stated. The RQMC sequences are generated through a `python` interface to the `R` package `randtoolbox` (Christophe and Petr, 2015). In particular we use scrambled Sobol sequences. The gradients are calculated using an automatic differentiation toolbox. The ELBO values are computed by using 10,000 MC samples, the variance of the gradient estimators is estimated by resampling the gradient 1000 times in each optimization step and computing the empirical variance.

Benchmarks The first benchmark is the vanilla MCVI algorithm based on ordinary MC sampling. Our method QMCVI replaces the MC samples by RQMC sequences and comes at almost no computational overhead (Section 4.3).

Our second benchmark in the second and third experiment is the control variate (CV) approach of Miller et al. (2017), where we use the code provided with the publication. In the first experiment, this comparison is omitted since the method of Miller et al. (2017) does not apply in this setting due to the non-Gaussian variational distribution.

Main Results We find that our approach generally leads to a faster convergence compared to our baselines due to a decreased gradient variance. For the multi-level Poisson GLM experiment, we also find that our RQMC algorithm converges to a better local optimum of the ELBO. As proposed in Theorem 7, we find that increasing the sample size over iteration in QMCVI leads to a better asymptotic convergence rate than in MCVI.

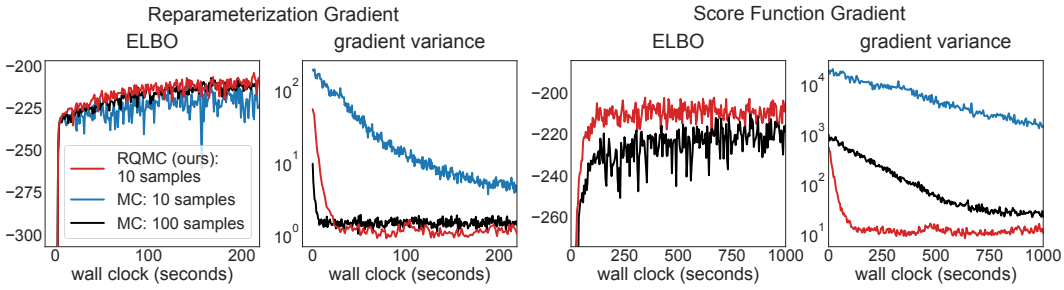


FIGURE 4.2: Toy: experiment 4.4.1. ELBO optimization path using Adam and variance of the stochastic gradient using the RQMC based gradient estimator using 10 samples (ours, in red) and the MC based estimator using 10 samples (blue) and 100 samples (black), respectively. The upper panel corresponds to the reparameterization gradient and the lower panel to the score function gradient estimator[1]. For both versions of MCVI, using RQMC samples (proposed) leads to variance reduction and faster convergence.

[1]Using only 10 samples for the MC based score function estimator leads to divergence and the ELBO values are out of the scope of the plot.

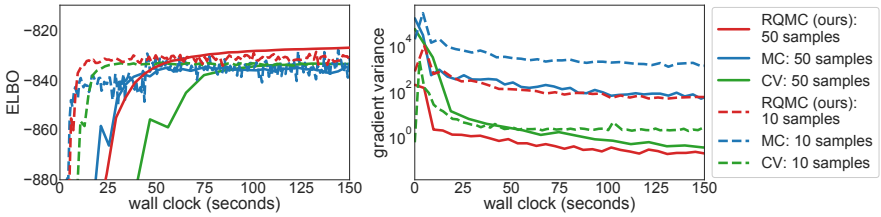


FIGURE 4.3: Frisk: experiment 4.4.2. Left, the optimization path of the ELBO is shown using Adam with the RQMC, MC and CV based reparameterization gradient estimator, respectively. Right, the gradient variances as function of time are reported. In the case of using 10 samples (dashed lines) RQMC (ours) outperforms the baselines in terms of speed while the CV based method exhibits lowest gradient variance. When increasing the sample size to 50 (solid lines) for all methods, RQMC converges closer to the optimum than the baselines while having lowest gradient variance.

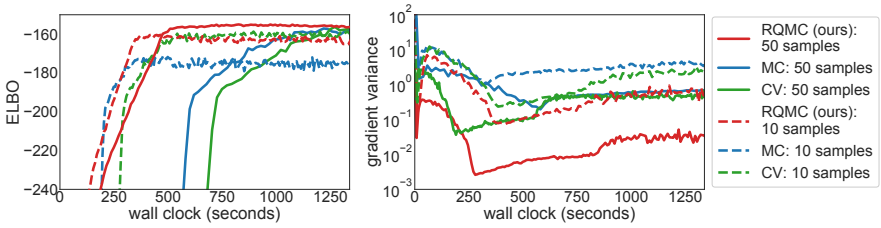


FIGURE 4.4: BNN: experiment 4.4.3. Left, the optimization path of the ELBO is shown using Adam with the RQMC, MC and CV based reparameterization gradient estimator, respectively. Right, the gradient variances as function of time are reported. RQMC (ours) based on 10 samples outperforms the baselines in terms of speed. RQMC with 50 samples is bit slower but converges closer to the optimum as its gradient variance is up to 3 orders of magnitude lower than for the baselines.

4.4.1 Hierarchical Linear Regression

We begin the experiments with a toy model of hierarchical linear regression with simulated data. The sampling process for the outputs y_i is $y_i \sim \mathcal{N}(\mathbf{x}_i^\top \mathbf{b}_i, \epsilon)$, $\mathbf{b}_i \sim \mathcal{N}(\mu_\beta, \sigma_\beta)$. We place lognormal hyper priors on the variance of the intercepts σ_β and on the noise ϵ; and a Gaussian hyper prior on μ_β. Details on the model are provided in Appendix 4.8.1. We set the dimension of the data points to be 10 and simulated 100 data points from the model. This results in a 1012-dimensional posterior, which we approximate by a variational distribution that mirrors the prior distributions.

We optimize the ELBO using Adam (Kingma and Ba, 2015) based on the score function as well as the reparameterization gradient estimator. We compare the standard MC based approach using 10 and 100 samples with our RQMC based approach using 10 samples, respectively. The CV based estimator cannot be used in this setting since it only supports Gaussian variational distributions and the variational family includes a lognormal distribution. For the score function estimator, we set the initial step size of Adam to 0.01.

The results are shown in Figure 4.2. We find that using RQMC samples decreases the variance of the gradient estimator substantially. This applies both to the score function and the reparameterization gradient estimator. Our approach substantially improves the standard score function estimator in terms of convergence speed and leads to a decreased gradient variance of up to three orders of magnitude. Our approach is also beneficial in the case of the reparameterization gradient estimator, as it allows for reducing the sample size from 100 MC samples to 10 RQMC samples, yielding a similar gradient variance and optimization speed.

4.4.2 Multi-level Poisson GLM

We use a multi-level Poisson generalized linear model (GLM), as introduced in (Gelman and Hill, 2006) as an example of multi-level modeling. This model has a 37-dim posterior, resulting from its hierarchical structure.

As in (Miller et al., 2017), we apply this model to the *frisk* data set Gelman et al. (2006) that contains information on the number of stop-and-frisk events within different ethnicity groups. The generative process of the model is described in Appendix 4.8.2. We approximate the posterior by a diagonal Gaussian variational distribution.

The results are shown in Figure 4.3. When using a small number of samples $(N = 10)$, all three methods have comparable convergence speed and attain a similar optimum. In this setting, the CV based method has lowest gradient variance. When increasing the sample size to 50, our proposed RQMC approach leads to substantially decreased gradient variance and allows Adam to convergence closer to the optimum than the baselines. This agrees with the fact that RQMC improves over MC for sufficiently large sample sizes.

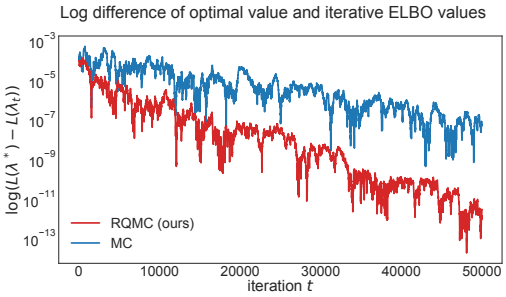


FIGURE 4.5: Constant SGD: experiment 4.4.4. We exemplify the consequences of Theorem 7 and optimize a simple concave ELBO using SGD with fixed learning rate $\alpha = 0.001$ while the number of samples are iteratively increased. We use an exponential sample size schedule (starting with one sample and 50.000 samples in the final iteration). The logarithmic difference of the ELBO to the optimum is plotted. We empirically confirm the result of Theorem 7 and observe a faster asymptotic convergence rate when using RQMC samples over MC samples.

4.4.3 Bayesian Neural Network

As a third example, we study QMCVI and its baselines in the context of a Bayesian neural network. The network consists of a 50-unit hidden layer with ReLU activations. We place a normal prior over each weight, and each weight prior has an inverse Gamma hyper prior. We also place an inverse Gamma prior over the observation variance. The model exhibits a posterior of dimension $d = 653$ and is applied to a 100-row subsample of the wine dataset from the UCI repository[2]. The generative process is described in Appendix 4.8.3. We approximate the posterior by a variational diagonal Gaussian.

The results are shown in Figure 4.4. For $N = 10$, both the RQMC and the CV version converge to a comparable value of the ELBO, whereas the ordinary MC approach converges to a lower value. For $N = 50$, all three algorithms reach approximately the same value of the ELBO, but our RQMC method converges much faster. In both settings, the variance of the RQMC gradient estimator is one to three orders of magnitude lower than the variance of the baselines.

4.4.4 Increasing the Sample Size Over Iterations

Along with our new Monte Carlo variational inference approach QMCVI, Theorem 7 gives rise to a new stochastic optimization algorithm for Monte Carlo objectives. Here, we investigate this algorithm empirically, using a constant learning rate and an (exponentially) increasing sample size schedule. We show that, for strongly convex objective functions and some mild regularity assumptions, our RQMC based gradient estimator leads to a faster asymptotic convergence rate than using the ordinary MC based gradient estimator.

[2] https://archive.ics.uci.edu/ml/datasets/Wine+Quality

In our experiment, we consider a two-dimensional factorizing normal target distribution with zero mean and standard deviation one. Our variational distribution is also a normal distribution with fixed standard deviation of 1, and with a variational mean parameter, i.e., we only optimize the mean parameter. In this simple setting, the ELBO is strongly convex and the variational family includes the target distribution. We optimize the ELBO with an increasing sample size, using the SGD algorithm described in Theorem 7. We initialize the variational parameter to $(0.1, 0.1)$. Results are shown in Figure 4.5.

We considered both RQMC (red) and MC (blue) based gradient estimators. We plot the difference between the optimal ELBO value and the optimization trace in logarithmic scale. The experiment confirms the theoretical result of Theorem 7 as our RQMC based method attains a faster asymptotic convergence rate than the ordinary MC based approach. This means that, in the absence of additional noise due to data subsampling, optimizing Monte Carlo objectives with RQMC can drastically outperform SGD.

4.5 Conclusion

We investigated randomized Quasi-Monte Carlo (RQMC) for stochastic optimization of Monte Carlo objectives. We termed our method Quasi-Monte Carlo Variational Inference (QMCVI), currently focusing on variational inference applications. Using our method, we showed that we can achieve faster convergence due to variance reduction.

QMCVI has strong theoretical guarantees and provably gets us closer to the optimum of the stochastic objective. Furthermore, in absence of additional sources of noise such as data subsampling noise, QMCVI converges at a faster rate than SGD when increasing the sample size over iterations.

QMCVI can be easily integrated into automated inference packages. All one needs to do is replace a sequence of uniform random numbers over the hypercube by an RQMC sequence, and perform the necessary reparameterizations to sample from the target distributions.

An open question remains as to which degree QMCVI can be combined with control variates, as RQMC may introduce additional unwanted correlations between the gradient and the CV. We will leave this aspect for future studies. We see particular potential for QMCVI in the context of reinforcement learning, which we consider to investigate.

Acknowledgments

We would like to thank Pierre E. Jacob, Nicolas Chopin, Rajesh Ranganath, Jaan Altosaar and Marius Kloft for their valuable feedback on our manuscript. This work was partly funded by the German Research Foundation (DFG) award KL 2698/2-1 and a GENES doctoral research scholarship.

4.6 Additional Information on QMC

We provide some background on QMC sequences that we estimate necessary for the understanding of our algorithm and our theoretical results.

Quasi-Monte Carlo (QMC) Low discrepancy sequences (also called Quasi Monte Carlo sequences), are used to approximate integrals over the $[0,1]^d$ hyper-cube: $\mathbb{E}\psi(U) = \int_{[0,1]^d} \psi(u)du$, that is the expectation of the random variable $\psi(U)$, where $U \sim \mathcal{U}[0,1]^d$, is a uniform distribution on $[0,1]^d$. The basic Monte Carlo approximation of the integral is $\hat{I}_N := \frac{1}{N}\sum_{n=1}^N \psi(u_n)$, where each $u_n \sim \mathcal{U}[0,1]^d$, independently. The error of this approximation is $\mathcal{O}(N^{-1})$, since $\mathrm{Var}[\hat{I}_N] = \mathrm{Var}[\psi(U)]/N$.

This basic approximation may be improved by replacing the random variables u_n by a low-discrepancy sequence; that is, informally, a deterministic sequence that covers $[0,1]^d$ more regularly. The error of this approximation is assessed by the Koksma-Hlawka inequality (Hickernell, 2006):

$$\left| \int_{[0,1]^d} \psi(u)du - \frac{1}{N}\sum_{n=1}^N \psi(u_n) \right| \leq V(\psi)D^*(u_{1:N}), \tag{4.8}$$

where $V(\psi)$ is the total variation in the sense of Hardy and Krause (Hardy, 1905). This quantity is closely linked to the smoothness of the function ψ. $D^*(u_{1:N})$ is called the star discrepancy, that measures how well the sequence covers the target space.

The general notion of discrepancy of a given sequence $u_1, \cdots, u_N$ is defined as follows:

$$D(u_{1:N}, \mathcal{A}) := \sup_{A \in \mathcal{A}} \left| \frac{1}{N}\sum_{n=1}^N \mathbb{1}\{u_n \in A\} - \lambda_d(A) \right|,$$

where $\lambda_d(A)$ is the volume (Lebesgue measure on $\mathbb{R}^d$) of A and $\mathcal{A}$ is a set of measurable sets. When we fix the sets $A = [0, \mathbf{b}] = \prod_{i=1}^d [0, b_i]$ with $0 \leq b_i \leq 1$ as a products set of intervals anchored at 0, the star discrepancy is then defined as follows

$$D^*(u_{1:N}) := \sup_{[0,\mathbf{b}]} \left| \frac{1}{N}\sum_{n=1}^N \mathbb{1}\{u_n \in [0,\mathbf{b}]\} - \lambda_d([0,\mathbf{b}]) \right|.$$

It is possible to construct sequences such that $D^*(u_{1:N}) = \mathcal{O}((\log N)^{2d-2}/N^2)$. See also Kuipers and Niederreiter (2012) and Leobacher and Pillichshammer (2014) for more details.

Thus, QMC integration schemes are asymptotically more efficient than MC schemes. However, if the dimension d gets too large, the number of necessary samples N in order to reach the asymptotic regime becomes prohibitive. As the upper bound is rather pessimistic, in practice QMC integration outperforms MC integration even for small N in most applications, see e.g. the examples in Chapter 5 of Glasserman (2013). Popular QMC sequences are for example the Halton sequence or the Sobol sequence. See

MC	QMC	RQMC
N^{-1}	$N^{-2}(\log N)^{2d-2}$	N^{-2}

TABLE 4.1: Best achievable rates for MC, QMC and RQMC in terms of the MSE of the approximation.

e.g. Dick et al. (2013) for details on the construction. A drawback of QMC is that it is difficult to assess the error and that the deterministic approximation is inherently biased.

Ramdomized Quasi Monte Carlo (RQMC). The reintroduction of randomness in a low discrepancy sequence while preserving the low discrepancy properties enables the construction of confidence intervals by repeated simulation. Moreover, the randomization makes the approximation unbiased. The simplest method for this purpose is a randomly shifted sequence. Let $v \sim \mathcal{U}[0,1]^d$. Then the sequence based on $\hat{u}_n := u_n + v \mod 1$ preserves the properties of the QMC sequence with probability 1 and is marginally uniformly distributed.

Scrambled nets (Owen, 1997) represent a more sophisticated approach. Assuming smoothness of the derivatives of the function, Gerber (2015) showed recently, that rates of $\mathcal{O}(N^{-2})$ are achievable. We summarize the best rates in table 4.1.

Transforming QMC and RQMC sequences A generic recipe for using QMC / RQMC for integration is given by transforming a sequence with the inverse Rosenblatt transformation $\Gamma : u \in [0,1]^d \mapsto \mathbf{z} \in \mathbb{R}^d$, see Rosenblatt (1952) and Gerber and Chopin (2015), such that

$$\int \psi(\Gamma(u))du = \int \psi(\mathbf{z})p(\mathbf{z})d\mathbf{z},$$

where $p(\mathbf{z})$ is the respective measure of integration. The inverse Rosenblatt transformation can be understood as the multivariate extension of the inverse cdf transform. For the procedure to be correct we have to make sure that $\psi \circ \Gamma$ is sufficiently regular.

Theoretical Results on RQMC In our analysis we mainly use the following result.

Theorem 8 *Owen (2008) Let $\psi : [0,1]^d \to \mathbb{R}$ be a function such that its cross partial derivatives up to order d exist and are continuous, and let $(u_n)_{n\in 1:N}$ be a relaxed scrambled (α, s, m, d)-net in base b with dimension d with uniformly bounded gain coefficients. Then,*

$$\text{Var}\left(\frac{1}{N}\sum_{n=1}^{N}\psi(u_n)\right) = \mathcal{O}\left(N^{-3}\log(N)^{(d-1)}\right),$$

where $N = \alpha b^m$.

In words, $\forall\tau > 0$ the RQMC error rate is $\mathcal{O}(N^{-3+\tau})$ when a scrambled (α, s, m, d)-net is used. However, a more general result has recently been shown by Gerber

(2015)[Corollary 1], where if ψ is square integrable and $(u_n)_{n\in 1:N}$ is a scrambled (s,d)-sequence, then

$$\text{Var}\left(\frac{1}{N}\sum_{n=1}^{N}\psi(u_n)\right) = o\left(N^{-1}\right).$$

This result shows that RQMC integration is always better than MC integration. Moreover, Gerber (2015)[Proposition 1] shows that rates $\mathcal{O}(N^{-2})$ can be obtained when the function ψ is regular in the sense of Theorem 8 . In particular one gets

$$\text{Var}\left(\frac{1}{N}\sum_{n=1}^{N}\psi(u_n)\right) = \mathcal{O}\left(N^{-2}\right).$$

4.7 Proofs

Our proof are deduced from standard results in the stochastic approximation literature, e.g. Bottou et al. (2016), when the variance of the gradient estimator is reduced due to RQMC sampling. Our proofs rely on scrambled (s,d)-sequences in order to use the result of Gerber (2015). The scrambled Sobol sequence, that we use in our simulations satisfies the required properties. We denote by $\mathbb{E}$ the total expectation and by $\mathbb{E}_{U_{N,t}}$ the expectation with respect to the RQMC sequence U_N generated at time t. Note, that λ_t is not a random variable w.r.t. $U_{N,t}$ as it only depends on all the previous $U_{N,1},\cdots,U_{N,t-1}$ due to the update equation in (4.6). However, $\hat{F}_N(\lambda_t)$ is a random variable depending on $U_{N,t}$.

4.7.1 Proof of Theorem 5

Let us first prove that $\text{tr}\,\text{Var}\left[\hat{g}_N(\lambda)\right] \leq M_V \times r(N)$ for all λ. By assumption we have that $g_z(\lambda)$ with $z=\Gamma(u)$ is a function $G: u \mapsto g_{\Gamma(u)}(\lambda)$ with continuous mixed partial derivatives of up to order d for all λ. Therefore, if $u_1,\cdots,u_N$ is a RQMC sequence, then the trace of the variance of the estimator $\hat{g}_N(\lambda)$ is upper bounded by Theorem 8 and its extension by Gerber (2015)[Proposition 1] by a uniform bound M_V and the quantity $r(N)=\mathcal{O}(N^{-2})$, that goes to 0 faster than the Monte Carlo rate $1/N$.

By the Lipschitz assumption we have that $F(\lambda) \leq F(\overline{\lambda}) + \nabla F(\overline{\lambda})^T(\lambda-\overline{\lambda}) + \frac{1}{2}L\|\lambda-\overline{\lambda}\|_2^2$, $\forall \lambda, \overline{\lambda}$, see for example Bottou et al. (2016). By using the fact that $\lambda_{t+1}-\lambda_t = -\alpha\hat{g}_N(\lambda_t)$ we obtain

$$\begin{aligned} & F(\lambda_{t+1}) - F(\lambda_t) \\ \leq\ & \nabla F(\lambda_t)^T(\lambda_{t+1}-\lambda_t) + \frac{1}{2}L\|\lambda_{t+1}-\lambda_t\|_2^2, \\ =\ & -\alpha\nabla F(\lambda_t)^T\hat{g}_N(\lambda_t) + \frac{\alpha^2 L}{2}\|\hat{g}_N(\lambda_t)\|_2^2. \end{aligned}$$

After taking expectations with respect to $U_{N,t}$ we obtain

$$\begin{aligned} & \mathbb{E}_{U_{N,t}} F(\lambda_{t+1}) - F(\lambda_t) \\ \leq \; & -\alpha \nabla F(\lambda_t) \mathbb{E}_{U_{N,t}} \hat{g}_N(\lambda_t) + \frac{\alpha^2 L}{2} \mathbb{E}_{U_{N,t}} \|\hat{g}_N(\lambda_t)\|_2^2. \end{aligned}$$

We now use the fact that $\mathbb{E}_{U_{N,t}} \|\hat{g}_N(\lambda_t)\|_2^2 = \operatorname{tr} \operatorname{Var}_{U_{N,t}} [\hat{g}_N(\lambda_t)] + \|\mathbb{E}_{U_{N,t}} \hat{g}_N(\lambda_t)\|_2^2$ and after exploiting the fact that $\mathbb{E}_{U_{N,t}} \hat{g}_N(\lambda_t) = \nabla F(\lambda_t)$ we obtain

$$\begin{aligned} & \mathbb{E}_{U_{N,t}} F(\lambda_{t+1}) - F(\lambda_t) \\ \leq \; & -\alpha \|\nabla F(\lambda_t)\|_2^2 + \frac{\alpha^2 L}{2} \left[\operatorname{tr} \operatorname{Var}_{U_{N,t}} [\hat{g}_N(\lambda_t)] + \|\nabla F(\lambda_t)\|_2^2 \right], \\ = \; & \frac{\alpha^2 L}{2} \operatorname{tr} \operatorname{Var}_{U_{N,t}} [\hat{g}_N(\lambda_t)] + \left(\frac{\alpha^2 L}{2} - \alpha \right) \|\nabla F(\lambda_t)\|_2^2. \end{aligned}$$

The inequality is now summed for $t = 1, \cdots, T$ and we take the total expectation:

$$\begin{aligned} & \mathbb{E} F(\lambda_T) - F(\lambda_1) \\ \leq \; & \frac{\alpha^2 L}{2} \sum_{t=1}^{T} \mathbb{E} \operatorname{tr} \operatorname{Var}_{U_{N,t}} [\hat{g}_N(\lambda_t)] \\ & + \left(\frac{\alpha^2 L}{2} - \alpha \right) \sum_{t=1}^{T} \mathbb{E} \|\nabla F(\lambda_t)\|_2^2. \end{aligned}$$

We use the fact that $F(\lambda^\star) - F(\lambda_1) \leq \mathbb{E} F(\lambda_T) - F(\lambda_1)$, where λ_1 is deterministic and $\lambda^\star$ is the true minimizer, and divide the inequality by T:

$$\begin{aligned} & \frac{1}{T} \left[F(\lambda^\star) - F(\lambda_1) \right] \\ \leq \; & \frac{\alpha^2 L}{2} \frac{1}{T} \sum_{t=1}^{T} \mathbb{E} \operatorname{tr} \operatorname{Var}_{U_{N,t}} [\hat{g}_N(\lambda_t)] \\ & + \left(\frac{\alpha^2 L}{2} - \alpha \right) \frac{1}{T} \sum_{t=1}^{T} \mathbb{E} \|\nabla F(\lambda_t)\|_2^2. \end{aligned}$$

By rearranging and using $\alpha < 2/L$ and $\mu = 1 - \alpha L/2$ we obtain

$$\begin{aligned} & \frac{1}{T} \sum_{t=1}^{T} \mathbb{E} \|\nabla F(\lambda_t)\|_2^2 \\ \leq \; & \frac{1}{T \alpha \mu} \left[F(\lambda_1) - F(\lambda^\star) \right] \\ & + \frac{\alpha L}{2\mu} \frac{1}{T} \sum_{t=1}^{T} \mathbb{E} \operatorname{tr} \operatorname{Var}_{U_{N,t}} [\hat{g}_N(\lambda_t)]. \end{aligned}$$

We now use $\operatorname{tr} \operatorname{Var}_{U_{N,t}} [\hat{g}_N(\lambda_t)] \leq M_V r(N)$ for all t. Equation (4.7) is obtained as $T \to \infty$.

4.7.2 Proof of Theorem 6

A direct consequence of strong convexity is the fact that the optimality gap can be upper bounded by the gradient in the current point λ, e.g. $2c(F(\lambda) - F(\lambda^\star)) \leq \|\nabla F(\lambda)\|_2^2, \forall \lambda$. The following proof uses this result. Based on the previous proof we get

$$\begin{aligned} & \mathbb{E}_{U_{N,t}} F(\lambda_{t+1}) - F(\lambda_t) \\ \leq \; & \frac{\alpha^2 L}{2} \operatorname{tr} \operatorname{Var}_{U_{N,t}} \left[\hat{g}_N(\lambda_t)\right] \\ & + \left(\frac{\alpha^2 L}{2} - \alpha\right) \|\nabla F(\lambda_t)\|_2^2 \\ \leq \; & \frac{\alpha^2 L}{2} \operatorname{tr} \operatorname{Var}_{U_{N,t}} \left[\hat{g}_N(\lambda_t)\right] \\ & + \left(\frac{\alpha^2 L}{2} - \alpha\right) 2c \left(F(\lambda_t) - F(\lambda^\star)\right), \end{aligned}$$

where we have used that $\left(\frac{\alpha L}{2} - 1\right) \leq 0$. By subtracting $F(\lambda^\star)$ from both sides, taking total expectations and rearranging we obtain:

$$\begin{aligned} & \mathbb{E} F(\lambda_{t+1}) - F(\lambda^\star) \\ \leq \; & \frac{\alpha^2 L}{2} \mathbb{E} \operatorname{tr} \operatorname{Var}_{U_{N,t}} \left[\hat{g}_N(\lambda_t)\right] \\ & + \left[\left(\frac{\alpha^2 L}{2} - \alpha\right) 2c + 1\right] \left(\mathbb{E} F(\lambda_t) - F(\lambda^\star)\right). \end{aligned}$$

Define $\beta = \left[\left(\frac{\alpha^2 L}{2} - \alpha\right) 2c + 1\right]$. We add

$$\frac{\alpha^2 L \mathbb{E} \operatorname{tr} \operatorname{Var}_{U_{N,t}} \left[\hat{g}_N(\lambda_t)\right]}{2(\beta - 1)}$$

to both sides of the equation. This yields

$$\begin{aligned} & \mathbb{E} F(\lambda_{t+1}) - F(\lambda^\star) + \frac{\alpha^2 L \mathbb{E} \operatorname{tr} \operatorname{Var}_{U_{N,t}} \left[\hat{g}_N(\lambda_t)\right]}{2(\beta - 1)} \\ \leq \; & \frac{\alpha^2 L}{2} \mathbb{E} \operatorname{tr} \operatorname{Var}_{U_{N,t}} \left[\hat{g}_N(\lambda_t)\right] \\ & + \beta \left(\mathbb{E} F(\lambda_t) - F(\lambda^\star)\right) + \frac{\alpha^2 L \mathbb{E} \operatorname{tr} \operatorname{Var}_{U_{N,t}} \left[\hat{g}_N(\lambda_t)\right]}{2(\beta - 1)} \\ \leq \; & \beta \left(\mathbb{E} F(\lambda_t) - F(\lambda^\star) + \frac{\alpha^2 L \mathbb{E} \operatorname{tr} \operatorname{Var}_{U_{N,t}} \left[\hat{g}_N(\lambda_t)\right]}{2(\beta - 1)}\right). \end{aligned}$$

Let us now show that $\beta < 1$:

$$\beta \leq \left[\left(\frac{\alpha L}{2} - 1\right) 2\alpha c + 1\right]$$

And as $\frac{\alpha L}{2} < 1$ we get $\beta < 1 - 2\alpha c$. Using $\alpha < 1/2c$ we obtain $\beta < 1$ and thus get a contracting equation when iterating over t:

$$\begin{aligned}
& \mathbb{E}F(\lambda_{t+1}) - F(\lambda^\star) \\
\leq\ & \beta^t \left(F(\lambda_1) - F(\lambda^\star) + \frac{\alpha^2 L \mathbb{E} \operatorname{tr} \operatorname{Var}_{U_{N,t}} [\hat{g}_N(\lambda_t)]}{2(\beta - 1)} \right) \\
& - \frac{\alpha^2 L \mathbb{E} \operatorname{tr} \operatorname{Var}_{U_{N,t}} [\hat{g}_N(\lambda_t)]}{2(\beta - 1)}, \\
\leq\ & \beta^t \left(F(\lambda_1) - F(\lambda^\star) + \frac{\alpha^2 L M_V r(N)}{2(\beta - 1)} \right) \\
& + \frac{\alpha^2 L M_V r(N)}{2(1 - \beta)}.
\end{aligned}$$

After simplification we get

$$\begin{aligned}
& \mathbb{E}F(\lambda_{t+1}) - F(\lambda^\star) \\
\leq\ & \beta^t \left(F(\lambda_1) - F(\lambda^\star) + \frac{\alpha L}{2\alpha L c - 4c} M_V r(N) \right) \\
& + \frac{\alpha L}{4c - 2\alpha L c} M_V r(N),
\end{aligned}$$

where the first term of the r.h.s. goes to 0 as $t \to \infty$.

4.7.3 Proof of Theorem 7

We require $\underline{N} \geq b^{s+d}$, due to a remark of Gerber (2015), where d is the dimension and b, s are integer parameters of the RQMC sequence. As $u \mapsto g_{\Gamma(u)}(\lambda)$ has continuous mixed partial derivatives of order d for all λ, $\operatorname{tr} \operatorname{Var} [\hat{g}_{N_t}(\lambda)] = \mathcal{O}(1/N_t^2)$ and consequently $\operatorname{tr} \operatorname{Var} [\hat{g}_{N_t}(\lambda)] \leq \hat{M}_V \times (1/N_t^2)$, where $\hat{M}_V$ is an universal upper bound on the variance. We recall that $N_t = \underline{N} + \lceil \tau^t \rceil$. Consequently

$$\operatorname{tr} \operatorname{Var} [\hat{g}_{N_t}(\lambda)] \leq \hat{M}_V \times \frac{1}{(\underline{N} + \lceil \tau^t \rceil)^2} \leq \hat{M}_V \times \frac{1}{\tau^{2t}}.$$

Now we take an intermediate result from the previous proof:

$$\begin{aligned}
& \mathbb{E}_{U_{N,t}} F(\lambda_{t+1}) - F(\lambda_t) \\
\leq\ & \frac{\alpha^2 L}{2} \operatorname{tr} \operatorname{Var}_{U_{N,t}} [\hat{g}_{N_t}(\lambda_t)] + \left(\frac{\alpha^2 L}{2} - \alpha \right) \|\nabla F(\lambda_t)\|_2^2 \\
\leq\ & \frac{\alpha^2 L}{2} \operatorname{tr} \operatorname{Var}_{U_{N,t}} [\hat{g}_{N_t}(\lambda_t)] - \frac{\alpha}{2} \|\nabla F(\lambda_t)\|_2^2 \\
\leq\ & \frac{\alpha^2 L}{2} \operatorname{tr} \operatorname{Var}_{U_{N,t}} [\hat{g}_{N_t}(\lambda_t)] - \alpha c \left(F(\lambda_t) - F(\lambda^\star) \right),
\end{aligned}$$

where we have used $\alpha \leq \min\{1/c, 1/L\}$ as well as strong convexity. Adding $F(\lambda^\star)$, rearranging and taking total expectations yields

$$\begin{aligned} & \mathbb{E}F(\lambda_{t+1}) - F(\lambda^\star) \\ \leq \; & \frac{\alpha^2 L}{2} \mathbb{E} \operatorname{tr} \operatorname{Var}_{U_{N,t}} \left[\hat{g}_{N_t}(\lambda_t)\right] \\ & + [1 - \alpha c] \left(\mathbb{E}F(\lambda_t) - F(\lambda^\star)\right). \end{aligned}$$

We now use $\operatorname{tr} \operatorname{Var}_{U_{N,t}} \left[\hat{g}_{N_t}(\lambda_t)\right] \leq \hat{M}_V \xi^{2t}$ and get

$$\begin{aligned} & \mathbb{E}F(\lambda_{t+1}) - F(\lambda^\star) \\ \leq \; & \frac{\alpha^2 L}{2} \hat{M}_V \xi^{2t} + [1 - \alpha c] \left(\mathbb{E}F(\lambda_t) - F(\lambda^\star)\right). \end{aligned}$$

We now use induction to prove the main result. The initialization for $t = 0$ holds true by the definition of ω. Then, for all $t \geq 1$,

$$\begin{aligned} & \mathbb{E}F(\lambda_{t+1}) - F(\lambda^\star) \\ \leq \; & [1 - \alpha c]\omega\xi^{2t} + \frac{\alpha^2 L}{2} \hat{M}_V \xi^{2t}, \\ \leq \; & \omega\xi^{2t} \left(1 - \alpha c + \frac{\alpha^2 L \hat{M}_V}{2\omega}\right), \\ \leq \; & \omega\xi^{2t} \left(1 - \alpha c + \frac{\alpha c}{2}\right), \\ \leq \; & \omega\xi^{2t} \left(1 - \frac{\alpha c}{2}\right) \\ \leq \; & \omega\xi^{2(t+1)}, \end{aligned}$$

where we have used the definition of ω and that $\left(1 - \frac{\alpha c}{2}\right) \leq \xi^2$.

4.8 Details for the Models Considered in the Experiments

4.8.1 Hierarchical Linear Regression

The generative process of the hierarchical linear regression model is as follows.

$\mu_\beta \sim \mathcal{N}(0, 10^2)$	intercept hyper prior
$\sigma_\beta \sim$ LogNormal(0.5)	intercept hyper prior
$\epsilon \sim$ LogNormal(0.5)	noise
$\mathbf{b}_i \sim \mathcal{N}(\mu_\beta, \sigma_\beta)$	intercepts
$y_i \sim \mathcal{N}(\mathbf{x}_i^\top \mathbf{b}_i, \epsilon)$	output

The dimension of the parameter space is $d = I \times k + k + 2$, where k denotes the dimension of the data points x_i and I their number. We set $I = 100$ and $k = 10$. The

dimension hence equals $d = 1012$.

4.8.2 Multi-level Poisson GLM

The generative process of the multi-level Poisson GLM is

$$\begin{aligned}
&\mu \sim \mathcal{N}(0, 10^2) && \text{mean offset}\\
&\log \sigma_\alpha^2,\ \log \sigma_\beta^2 \sim \mathcal{N}(0, 10^2) && \text{group variances}\\
&\alpha_e \sim \mathcal{N}(0, \sigma_\alpha^2) && \text{ethnicity effect}\\
&\beta_p \sim \mathcal{N}(0, \sigma_\beta^2) && \text{precinct effect}\\
&\log \lambda_{ep} = \mu + \alpha_e + \beta_p + \log N_{ep} && \text{log rate}\\
&Y_{ep} \sim \text{Poisson}(\lambda_{ep}) && \text{stop-and-frisk events}
\end{aligned}$$

Y_{ep} denotes the number of stop-and-frisk events within ethnicity group e and precinct p over some fixed period. N_{ep} represents the total number of arrests of group e in precinct p over the same period; α_e and β_p are the ethnicity and precinct effects.

4.8.3 Bayesian Neural Network

We study a Bayesian neural network which consists of a 50-unit hidden layer with ReLU activations.

The generative process is

$$\begin{aligned}
&\alpha \sim \text{InvGamma}(1, 0.1) && \text{weight hyper prior}\\
&\tau \sim \text{InvGamma}(1, 0.1) && \text{noise hyper prior}\\
&w_i \sim \mathcal{N}(0, 1/\alpha) && \text{weights}\\
&y \sim \mathcal{N}(\phi(\mathbf{x}, \mathbf{w}), 1/\tau) && \text{output distribution}
\end{aligned}$$

Above, w is the set of weights, and $\phi(\mathbf{x}, \mathbf{w})$ is a multi-layer perceptron that maps input $\mathbf{x}$ to output y as a function of parameters $\mathbf{w}$. We denote the set of parameters as $\grave{} := (\mathbf{w}, \alpha, \tau)$. The model exhibits a posterior of dimension $d = 653$.

4.9 Practical Advice for Implementing QMCVI in Your Code

It is easy to implement RQMC based stochastic optimization in your existing code. First, you have to look for all places where random samples are used. Then replace the ordinary random number generator by RQMC sampling. To replace an ordinarily sampled random variable $\mathbf{z}$ by an RQMC sample we need a mapping Γ from a uniformly distributed random variable u to $\mathbf{z} = \Gamma(u|\lambda)$, where λ are the parameters of the distribution (e.g. mean and covariance parameter for a Gaussian random variable). Fortunately, such a mapping can often be found (see Appendix 4.6).

In many recent machine learning models, such as variational auto encoders (Kingma and Ba, 2015) or generative adversarial networks (Goodfellow et al., 2014), the application of RQMC sampling is straightforward. In those models, all random variables are often expressed as transformations of Gaussian random variables via deep neural networks. To apply our proposed RQMC sampling approach we only have to replace the Gaussian random variables of the base distributions by RQMC sampled random variables.

In the following Python code snippet we show how to apply our proposed RQMC sampling approach in such settings.

CODE 4.1: Python code for RQMC sampling from a Gaussian distribution.

```
import numpy.random as nr
import numpy as np
import rpy2.robjects.packages as rpackages
import rpy2.robjects as robjects
from scipy.stats import norm


randtoolbox = rpackages.importr('randtoolbox')


def random_sequence_rqmc(dim, i=0, n=1, random_seed=0):
    """
    generate uniform RQMC random sequence
    """
    dim = np.int(dim)
    n = np.int(n)
    u = np.array(randtoolbox.sobol(n=n, dim=dim, init=(i==0),
            scrambling=1, seed=random_seed)).reshape((n,dim))
    # randtoolbox for sobol sequence
    return(u)

def random_sequence_mc(dim, n=1, random_seed=0):
    """
    generate uniform MC random sequence
    """
    dim = np.int(dim)
    n = np.int(n)
    np.random.seed(seed=random_seed)
    u = np.asarray(nr.uniform(size=dim*n).reshape((n,dim)))
    return(u)
```

```
def transfrom_uniform_to_normal(u, mu, sigma):
    """
    generat a multivariate normal based on
    a unifrom sequence
    """
    l_cholesky = np.linalg.cholesky(sigma)
    epsilon = norm.ppf(u).transpose()
    res = np.transpose(l_cholesky.dot(epsilon))+mu
    return res

if __name__ == '__main__':
    # example in dimension 2
    dim = 2
    n = 100
    mu = np.ones(dim)*2. # mean of the Gaussian
    sigma = np.array([[2.,1.],[1.,2.]]) # variance of the Gaussian

    # generate Gaussian random variables via RQMC
    u_random = random_sequence_rqmc(dim, i=0, n=n, random_seed=1)
    x_normal = transfrom_uniform_to_normal(u_random, mu, sigma)

    # here comes the existing code of your model
    deep_bayesian_model(x_normal)
```

Chapter 5

Adaptive Tuning Of Hamiltonian Monte Carlo Within Sequential Monte Carlo

Joint work with Pierre E. Jacob and Nicolas Chopin, submitted to *Bayesian Analysis*

Abstract: Sequential Monte Carlo (SMC) samplers form an attractive alternative to MCMC for Bayesian computation. However, their performance depends strongly on the Markov kernels used to rejuvenate particles. We discuss how to calibrate automatically (using the current particles) Hamiltonian Monte Carlo kernels within SMC. To do so, we build upon the adaptive SMC approach of Fearnhead and Taylor (2013), and we also suggest alternative methods. We illustrate the advantages of using HMC kernels within an SMC sampler via an extensive numerical study.

Keywords: Hamiltonian Monte Carlo, Sequential Monte Carlo

5.1 Introduction

Sequential Monte Carlo (SMC) samplers (Neal, 2001; Chopin, 2002; Del Moral et al., 2006) approximate a target distribution π by sampling *particles* from an initial distribution π_0, and moving them through a sequence of distributions π_t which ends at $\pi_T = \pi$. In Bayesian computation this approach has several advantages over Markov chain Monte Carlo (MCMC). In particular, it enables the estimation of normalizing constants and can thus be used for model choice (Zhou et al., 2016). Moreover, particles can be propagated mostly in parallel, with direct advantages on parallel computing devices (Murray et al., 2016). Finally, SMC samplers are more robust to multimodality, see Schweizer (2012b) and Jasra et al. (2015).

SMC samplers iterate over a sequence of resampling, propagation and reweighting steps. The propagation of the particles relies on MCMC kernels, that depend oftentimes on some tuning parameters. Choosing these parameters in a sensible manner

is challenging and has been of interest both from a theoretical and practical point of view; see Fearnhead and Taylor (2013); Schäfer and Chopin (2013); Beskos et al. (2016).

The appealing properties of Hamiltonian Monte Carlo (HMC) kernels in high dimension (Beskos et al., 2013; Mangoubi and Smith, 2017) motivate their use within SMC samplers. HMC has originally been developed in Physics (Duane et al., 1987), and introduced to the Statistics community by Neal (1993). It has become a standard MCMC tool for sampling distributions with continuously differentiable density functions on $\mathbb{R}^d$ (Neal, 2011). However, until recently (Gunawan et al., 2018), the use of HMC kernels within SMC samplers has received little attention.

Methods for tuning HMC (when run as a Markov chain) are proposed in Mohamed et al. (2013); Beskos et al. (2013); Betancourt et al. (2014); Betancourt (2016) and Hoffman and Gelman (2014), for example. In contrast with these papers, we consider the tuning of such kernels based on the set of current particles at each iteration of a SMC sampler. We base our approach on the work of Fearnhead and Taylor (2013), which concerned the tuning of generic MCMC kernels within SMC samplers, and on existing approaches to tuning HMC.

We apply the proposed SMC sampler with HMC kernels to five examples; three toy examples, a binary Bayesian regression of dimension up to 95 and a log Gaussian Cox model of dimension up to 4,096. Our numerical study illustrates the potential of SMC samplers for model choice in high dimensions and the improved performance of HMC propagation kernels within SMC samplers compared to random walk or Langevin based propagation kernels. We also investigate the importance of adapting the tempering ladder, and the number of move steps for diversifying the particles.

The paper is organized as follows. Section 5.2 reviews SMC samplers and HMC kernels. Section 5.3 discusses adaptive tuning procedures for SMC. Section 5.4 provides numerical experiments. Section 5.5 discusses our results.

5.2 Background

We consider the problem of calculating expectations of an integrable test function $\varphi : \mathbb{R}^d \mapsto \mathbb{R}$ with respect to a posterior distribution defined as $\pi(x) = p(x)l(y|x)/Z$. The random variable x with density $\pi(\cdot)$ is defined on the space $(\mathbb{R}^d, \mathcal{B}(\mathbb{R}^d))$. Here $p(x)$ denotes the prior distribution, $l(y|x)$ is the likelihood of the observed data y given the parameter $x \in \mathbb{R}^d$, and $Z = \int_{\mathbb{R}^d} l(y|x)p(x)dx$ denotes the normalizing constant, also called marginal likelihood or evidence. We next describe two algorithms widely used to approximate posterior distributions: Sequential Monte Carlo (SMC) and Hamiltonian Monte Carlo (HMC). These are building blocks for the adaptive SMC procedures discussed in this paper. The methods proposed in this article could apply to generic target distributions, but we will focus on posterior distributions and thus we will use the associated terminology.

5.2.1 Sequential Monte Carlo samplers

Introducing a sequence of targets: tempering from the prior to the posterior

Sequential Monte Carlo (SMC) approaches the problem of sampling from π by introducing a sequence of intermediate distributions $\pi_0, \cdots, \pi_t, \cdots, \pi_T$ defined on the common measurable target spaces $(\mathbb{R}^d, \mathcal{B}(\mathbb{R}^d))$ for all t, and such that π_0 is easy to sample from, and $\pi_T = \pi$.

We focus on tempering to construct intermediate distributions, that is $\pi_t(x) \propto p(x)l(y|x)^{\lambda_t}$, where the sequence of exponents λ_t is such that $0 = \lambda_0 < \cdots < \lambda_t < \cdots < \lambda_T = 1$. These exponents do not need to be pre-specified: they may be automatically selected during the run of a SMC sampler, as described later. Other choices of sequences of distributions are possible (see e.g. Chopin, 2002; Del Moral et al., 2006; Chopin et al., 2013). Note also that we assume throughout the article that the prior distribution $p(x)$ is a proper probability distribution, from which samples can be drawn.

Generic SMC samplers

We denote by $\gamma_t(x_t) = p(x)l(y|x)^{\lambda_t}$ the unnormalized density associated with $\pi_t(x_t)$, and by Z_t the normalizing constant: $Z_t = \int_{\mathbb{R}^d} \gamma_t(x_t)dx_t$. One way of constructing SMC samplers is as follows. Suppose that at time $t-1$ an equally weighted particle approximation $\{\tilde{x}_{t-1}^i\}_{i\in 1:N}$ of π_{t-1} is available, with possible duplicates among the particles. This cloud of particles is then moved with a Markov kernel $\mathcal{K}_t$, that leaves the distribution π_{t-1} invariant: for each i,

$$x_t^i \sim \mathcal{K}_t(\tilde{x}_{t-1}^i, dx).$$

Consequently a set of new samples $\{x_t^i\}_{i\in 1:N}$ is obtained. These particles are then weighted: particle x_t^i is assigned an importance weight $w_t^i = \gamma_t(x_t^i)/\gamma_{t-1}(x_t^i)$, so that the next distribution π_t is approximated by the set $\{x_t^i, w_t^i\}_{i\in 1:N}$. After resampling particles according to their weights, one obtains again an equally weighted set $\{\tilde{x}_t^i\}_{i\in 1:N}$ and the procedure is repeated for the next target distribution π_{t+1}. The resulting algorithm is given in Algorithm 12.

Upon completion, the algorithm returns weighted samples $\{x_t^i, w_t^i\}_{i\in 1:N}$, which may be used to approximate moments of the sequence of targets as follows:

$$\frac{\sum_{i=1}^N w_t^i \varphi(x_t^i)}{\sum_{i=1}^N w_t^i} \to \mathbb{E}_{\pi_t}[\varphi(x)]$$

as $N \to +\infty$. The algorithm also returns estimates of the ratios Z_t/Z_{t-1} (and thus of Z_T/Z_0); see line 13. (In the context of tempering, one may use the path sampling identity to derive an alternate estimate of Z_t/Z_{t-1}, as explained in Zhou et al. (2016). However, in our experiments, the two estimates were very close numerically, so we focus on the former estimate from now on.)

The kernels $\mathcal{K}_t$ may be constructed, for example, as Metropolis–Hastings kernels (see e.g. Chopin, 2002; Fearnhead and Taylor, 2013; Zhou et al., 2016; Jasra et al., 2011). More details on the general construction of kernels and on optimality can be found in Del Moral et al. (2006) or Del Moral et al. (2007). In general Markov kernels may depend on a set of tuning parameters h, and are hereafter denoted by $\mathcal{K}_t^h$ to make this dependence explicit, as in Algorithm 12.

Algorithm 12: Tempering SMC sampler algorithm

Input: Initial distribution π_0, target distribution π_T and intermediate distributions $\pi_{t-1}(x) \propto p(x)l(y|x)^{\lambda_{t-1}}$, rule for constructing Markov kernels $\mathcal{K}_t^h$ that are π_{t-1} invariant.

Result: Set of weighted samples $\{x_t^i, w_t^i\}_{i\in 1:N}$ for $t \in 1:T$ and normalizing constant estimates $\widehat{Z_t/Z_{t-1}}$ for $t \in 1:T$.

Initialization : t = 1, $\lambda_0 = 0$;

Iteration :

1 **while** $\lambda_t < 1$ **do**
2 **if** $t = 1$ **then**
3 **foreach** $i \in 1:N$ **do**
4 Sample $x_1^i \sim \gamma_0$;
5 **else**
6 Tune Markov kernel parameters h using available particles; see Algorithm 16 or 17;
7 **foreach** $i \in 1:N$ **do**
8 Move particle $x_t^i \sim \mathcal{K}_t^h(\tilde{x}_{t-1}^i, dx)$;
9 Move step can be iterated for better mixing, see Algorithm 14 ;
10 Choose the next exponent $\lambda_t \in (\lambda_{t-1}, 1]$ based on available particles; see Algorithm 13;
11 **foreach** $i \in 1:N$ **do**
12 Weight particle $w_t^i = \frac{\gamma_t(x_t^i)}{\gamma_{t-1}(x_t^i)}$;
13 Calculate $\widehat{\frac{Z_t}{Z_{t-1}}} = N^{-1}\sum_{i=1}^N w_t^i$;
14 Resample particles $\{x_t^i, w_t^i\}_{i\in 1:N}$ to obtain $\{\tilde{x}_t^i\}_{i\in 1:N}$;
15 Set $t = t + 1$;

Tuning of the SMC sampler

Different design choices have to be made for the SMC sampler of Algorithm 12 to be operational.

(a) The choice of the next exponent λ_t, at line 10 of Algorithm 12, may be based on available particles; for instance on their effective sample size, as explained below.

(b) The number of move steps, at line 9 of Algorithm 12, may be based on the observed performance of the Markov kernels; see below.

(c) The tuning of the Markov kernel parameters h, at line 6 of Algorithm 12, may be based on the particles. The main contribution of this paper is to investigate this tuning in the case of HMC kernels, and is described in Section 5.3.

(a) Choice of the next exponent A common approach (Jasra et al., 2011; Schäfer and Chopin, 2013) to choose adaptively intermediate distributions within SMC is to rely on the ESS (effective sample size, Kong et al., 1994). This criterion is calculated as follows:

$$\mathrm{ESS}(\lambda_t) = \frac{\left(\sum_{i=1}^N w_t^i\right)^2}{\sum_{i=1}^N \left(w_t^i\right)^2}, \tag{5.1}$$

where $w_t^i = \gamma_t(x_t^i)/\gamma_{t-1}(x_t^i) = l(y|x_t^i)^{\lambda_t - \lambda_{t-1}}$ in the setting considered here. The ESS is linked to the χ^2-divergence between the distributions π_{t-1} and π_t or equivalently to the variance of the weights.

We may choose λ_t by solving (in λ) the equation $\mathrm{ESS}(\lambda) = \alpha N$, for some user-chosen value $\alpha \in (0,1)$. The corresponding algorithm is described in Algorithm 13. The validity of adaptive SMC samplers based on an ESS criterion is studied in e.g. Beskos et al. (2016), see also Huggins and Roy (2015).

Algorithm 13: Choice of the next exponent based on the effective sample size.

Input: Target value α, likelihood $l(y|x_t^i)$ for the N particles x_t^i, current temperature λ_{t-1}.

Result: Next temperature λ_t

1 Define $\beta^i(\lambda) := l(y|x_t^i)^{\lambda - \lambda_{t-1}}$ and

$$\mathrm{ESS}(\lambda) = \frac{\left(\sum_{i=1}^N \beta^i(\lambda)\right)^2}{\sum_{i=1}^N \left(\beta^i(\lambda)\right)^2};$$

2 Solve $\mathrm{ESS}(\lambda) = \alpha N$ in λ, using bisection;

(b) Number of move steps The mixing of MCMC kernels plays a crucial role in the performance and stability of SMC samplers (see e.g. Del Moral et al., 2006; Schweizer, 2012a; Ridgway, 2016).

For any MCMC kernel targeting a distribution π, mixing is improved by repeated application of the kernel, for a cost linear in the number of repetitions. We propose to monitor the product of componentwise first-order autocorrelations of the particles to decide how many repetitions to use. Autocorrelations are calculated w.r.t. $\{\tilde{x}_t^i\}_{i\in 1:N}$, the cloud of particles after reweighting and resampling at time t. After k move steps through the kernel $\mathcal{K}_t^h$ the cloud of particles is $\{x_{t,k}^i\}_{i\in 1:N}$. We then calculate the empirical correlation of the component-wise statistic $x_{t,k}^i(j) + x_{t,k}^i(j)^2$, where $x_{t,k}^i(j)$ denotes component j of the vector $x_{t,k}^i$, using the successive states of the chain $x_{t,k}^i(j)$ and

$x^i_{t,k-1}(j)$. This empirical correlation is denoted by $\hat{\rho}_k(j)$. This statistic is chosen to reflect the first two moments of the particles, but is otherwise arbitrary. We suggest to continue applying the Markov kernel until a large fraction (e.g. 90%) of the product of the first order autocorrelations drops below a threshold $\alpha' = 0.1$, for example. The resulting algorithm is described in Algorithm 14.

Algorithm 14: Adaptive move step based on autocorrelations.

Input: Particles $\{\tilde{x}^i_t\}_{i\in 1:N}$, proposal kernel $\mathcal{K}^h_t$
Result: Particles after k move steps $\{x^i_{t,k}\}_{i\in 1:N}$.
Initialization : $\{x^i_{t,0}\}_{i\in 1:N} \leftarrow \{\tilde{x}^i_t\}_{i\in 1:N}$, $k \leftarrow 0$.
while $\#\{j : \prod_{l=1}^k \hat{\rho}_l(j) > \alpha'\}/d \geq 10\%$ **do**
  Set $k \leftarrow k+1$;
  Move particle $x^i_{t,k} \sim \mathcal{K}^h_t(x^i_{t,k-1}, dx)$ for all i;
  Calculate the correlation $\hat{\rho}_k(j)$ for all j;

Instead of using component-wise autocorrelations, one could also draw on recent work on the performance evaluation of MCMC algorithms in multidimensional spaces (Vats et al., 2015) or approaches based on the Stein discrepancy (Gorham and Mackey, 2015).

5.2.2 Hamiltonian Monte Carlo

Hamiltonian Monte Carlo (HMC) consists in proposing moves by solving the equations of motion of a particle evolving in a potential.We first describe HMC, by following the exposition in Neal (2011), before turning to existing approaches to the tuning of its algorithmic parameters.

MCMC based on Hamiltonian dynamics

Let $\mathcal{L}(x) = \log \gamma(x)$ be the unnormalized log density of the random variable of interest x. We introduce an auxiliary random variable $p \in \mathbb{R}^d$ with distribution $\mathcal{N}(0, \mathbf{M})$, and hence unnormalized log density $\log f(p) = -1/2 p^T \mathbf{M}^{-1} p$. The joint unnormalized density of (p, x) is given as $\mu(p, x) = f(p)\gamma(x)$ and the negative joint log-density is denoted by

$$H(p, x) = -\log \mu(p, x) = -\mathcal{L}(x) + \frac{1}{2} p^T \mathbf{M}^{-1} p.$$

The physical analogy of this quantity is the Hamiltonian, where the first term denotes the potential energy at position x, and the second term denotes the kinetic energy of the momentum p with the mass matrix $\mathbf{M}$. The movement in time of a particle with position x and momentum p can be described via its Hamilton equations,

$$\begin{cases} \frac{dx}{d\tau} = \frac{\partial H}{\partial p} = \mathbf{M}^{-1} p, \\ \frac{dp}{d\tau} = -\frac{\partial H}{\partial x} = \nabla_x \mathcal{L}(x), \end{cases}$$

where $dx/d\tau, dp/d\tau$ denote the derivatives of the position and the momentum with respect to the continuous time τ. The solution of this differential equation induces a flow Φ_τ that describes the evolution of a system with initial momentum and position (p_0, x_0) such that $\Phi_\tau(p_0, x_0) = (p_\tau, x_\tau)$. The solution is (a) energy preserving, e.g. $H(p_\tau, x_\tau) = H(p_0, x_0)$; (b) volume preserving and consequently the determinant of the Jacobian of Φ_τ equals one; (c) the flow is reversible w.r.t. time.

In terms of probability distributions this means that if $(p_0, x_0) \sim \mu(p, x)$ then also $(p_\tau, x_\tau) \sim \mu(p, x)$. Thus, if an exact expression of the flow is available, at stationarity every trajectory induced by the flow represents a realization of the joint probability distribution.

In most cases an exact solution of the flow is not available and one has to use numerical integration methods instead. One widely used integrator is the Störmer-Verlet or leapfrog integrator (Hairer et al., 2003). The leapfrog integrator is volume preserving and reversible but not energy preserving. It iterates through the following updates:

$$\begin{aligned} p_{\tau+\epsilon/2} &= p_\tau + \epsilon/2 \nabla_x \mathcal{L}(x_\tau), \\ x_{\tau+\epsilon} &= x_\tau + \epsilon \mathbf{M}^{-1} p_{\tau+\epsilon/2}, \\ p_{\tau+\epsilon} &= p_{\tau+\epsilon/2} + \epsilon/2 \nabla_x \mathcal{L}(x_{\tau+\epsilon}), \end{aligned}$$

where ϵ denotes the step size of the leapfrog integrator. Thus, in order to let the system evolve from τ to $\tau + \kappa$ with $\kappa = L \times \epsilon$ we need to make L steps as described above. This induces a numerical flow $\widehat{\Phi}_{\epsilon,L}$ such that $\widehat{\Phi}_{\epsilon,L}(p_\tau, x_\tau) = (\hat{p}_{\tau+\kappa}, \hat{x}_{\tau+\kappa})$. In general we have $\Delta E_\kappa \neq 0$ where $\Delta E_\kappa = H(\hat{p}_{\tau+\kappa}, \hat{x}_{\tau+\kappa}) - H(p_\tau, x_\tau)$ is the variation of the Hamiltonian. The dynamics can be used to construct a Markov chain targeting μ on the joint space, with a Metropolis–Hastings step that corrects for the variation in the energy after sampling a random momentum and constructing a numerical flow. Algorithm 15 describes the Markov kernel of HMC.

Algorithm 15: Hamiltonian Monte Carlo algorithm

Input: Gradient function $\nabla_x \mathcal{L}(\cdot)$, initial state x_s, energy function ΔE

Result: Next state of the chain (p_{s+1}, x_{s+1})

Sample $p_s \sim \mathcal{N}(0_d, \mathbf{M})$

Apply the leapfrog integration: $(\hat{p}_{s+1}, \hat{x}_{s+1}) \leftarrow \widehat{\Phi}_{\epsilon,L}(p_s, x_s)$

Sample $u \sim \mathcal{U}[0, 1]$

if $\log(u) \leq \min(0, \Delta E_s)$ **then**

  Set $x_{s+1} \leftarrow \hat{x}_{s+1}$

else

  Set $x_{s+1} \leftarrow x_s$

The performance of the HMC algorithm crucially depends on the tuning parameters (ϵ, L) and the mass matrix $\mathbf{M}$, see Neal (1993, 2011) for additional details.

Existing approaches to tuning HMC

The error analysis of geometric integration gives insights that allow to find step sizes ϵ that yield stable trajectories. For the leapfrog integrator the error of the energy is

$$|H(\hat{p}_{\tau+\kappa}, \hat{x}_{\tau+\kappa}) - H(p_\tau, x_\tau)| \leq C_1 \epsilon^2, \tag{5.2}$$

and the error of the position and momentum is

$$\left\| \widehat{\Phi}_{\epsilon,L}(\hat{p}_\tau, \hat{x}_\tau) - \Phi(p_\tau, x_\tau) \right\|_2 \leq C_2 \epsilon^2, \tag{5.3}$$

see Leimkuhler and Matthews (2016); Bou-Rabee and Sanz-Serna (2018) for more details. It can be shown that the constant $C_1 > 0$ in (5.2) stays stable over exponential long time intervals ϵL (Hairer et al., 2006, Theorem 8.1), whereas the constant $C_2 > 0$ in the (5.3) typically grows in L. Hence, care must be taken when choosing (ϵ, L).

From a practical point of view the tuning of the HMC kernel requires the consideration of the following aspects. If ϵ is too large, the numerical integration of the HMC flow becomes unstable and results in large variations in the energy and thus a low acceptance rate, see (5.2). On the other hand if ϵ is too small, for a fixed number of steps L the trajectories tend to be short and high autocorrelations will be observed, see Neal (2011). To counterbalance this effect a large L would be needed and thus computation time would increase. If L gets too large, the trajectories might eventually double back on themselves (Hoffman and Gelman, 2014).

From a theoretical perspective Beskos et al. (2013) and later Betancourt et al. (2014) show that the integrator step size ϵ should be chosen such that acceptance rates between 0.651 and 0.9 are obtained, when the dimension of the target space goes to infinity. This idea has been exploited in Hoffman and Gelman (2014) where stochastic approximation is used in order to obtain reasonable values of ϵ.

A different approach is to choose the parameters of the kernel such that the expected squared jumping distance (ESJD) of the kernel is maximized, see Pasarica and Gelman (2010). The ESJD in one dimension is defined as

$$\text{ESJD} = \mathbb{E}\left[\|x_s - x_{s-1}\|_2^2 \right] = 2(1 - \rho_1) \operatorname{Var}_\pi[x],$$

assuming stationarity. In this sense maximizing the ESJD of a Markov chain is equivalent to minimizing the first order autocorrelation ρ_1. In d dimensions maximizing the ESJD in Euclidean norm amounts to minimizing the correlation of the d dimensional process in Euclidean norm. In the case of HMC this has been discussed by Mohamed et al. (2013) and Hoffman and Gelman (2014). Mohamed et al. (2013) tune the HMC sampler with Bayesian optimization and vanishing adaptation in the spirit of adaptive MCMC algorithms, see Andrieu and Thoms (2008). The ESJD is then maximized as a function of (ϵ, L). Hoffman and Gelman (2014) discuss the ESJD as a criterion for the choice of L. As a general idea the simulation of the trajectories should be stopped

when the ESJD starts to decrease. However, this idea has the inconvenience of impacting the reversibility of the chain. This problem is solved by adjusting the acceptance step in the algorithm. The resulting algorithm is called NUTS and is used in the probabilistic programming language STAN, see Carpenter et al. (2017).

Neal (2011) suggests to use preliminary runs in order to find reasonable values of (ϵ, L) and to randomize the values around the chosen values. The randomization avoids pathological behavior that might occur when (ϵ, L) are selected badly. Other approaches on identifying the optimal trajectory length are discussed in Betancourt (2016).

Another important tuning parameter is the mass matrix $\mathbf{M}$, that is used for sampling the momentum. When the target distribution is close to a Gaussian, rescaling the target by the Cholesky decomposition of the inverse covariance matrix eliminates the correlation of the target and can improve the performance of the sampler. Equivalently, the inverse covariance matrix can be set to the mass matrix of the momentum. This yields the same transformation, see Neal (2011). Recently, Girolami and Calderhead (2011) suggested to use a position dependent mass matrix that takes the local curvature of the target into account. However, the integrator of the Hamiltonian needs to be modified in consequence.

5.3 Tuning Of Hamiltonian Monte Carlo Within Sequential Monte Carlo

We now discuss the tuning of the Markov kernel in line 6 of Algorithm 12. The tuning of Markov kernels within SMC samplers can be linked to the tuning of MCMC kernels in general. One advantage of tuning the kernels in the framework of SMC is that information on the intermediate distributions is available in form of the particle approximations. Moreover, different kernel parameters can be assigned to different particles and hence a large number of parameters can be tested in parallel. This idea has been exploited by Fearnhead and Taylor (2013). We build upon their methodology and adjust their approach to the tuning of HMC kernels.

We first describe the tuning of the mass matrix. Second, we present our adaptation of the approach of Fearnhead and Taylor (2013) to the tuning of HMC kernels, abbreviated by FT. Then we present an alternative approach based on a pre-tuning phase, abbreviated by PR for preliminary run. Finally, we discuss the comparative advantages and drawbacks of the two approaches.

5.3.1 Tuning of the mass matrix of the kernels

The HMC kernels depend on the mass matrix $\mathbf{M}$, used for sampling the momentum. Calibrating this matrix based on the particles of the previous iteration allows to exploit the information generated by the particles. In the case of an independent MH proposal this has been used by Chopin (2002), for instance. For the HMC kernel we use the

following matrix at iteration t

$$\mathbf{M}_t = \text{diag}(\widehat{\text{Var}}_{\pi_t}[x_t])^{-1}, \tag{5.4}$$

where $\widehat{\text{Var}}_{\pi_t}[x_t]$ is the particle estimate of the covariance matrix of target π_t. The restriction to a diagonal matrix makes this approach applicable in high dimensions. Thus, the global structure of the target distribution π_{t-1} is taken into account and moves in directions of higher variance are proposed.

5.3.2 Adapting the tuning procedure of Fearnhead and Taylor (2013)

Suppose we are at line 6 during iteration t of Algorithm 12. We are now interested in choosing the parameters h of the propagation kernel $\mathcal{K}_t^h$.

Fearnhead and Taylor (2013) consider the following ESJD (expected squared jumping distance) criterion:

$$g_t(h) = \int \pi_{t-1}(x_{t-1})\mathcal{K}_t^h(x_{t-1}, x_t)\, \|x_{t-1} - x_t\|_M^2\, dx_{t-1}dx_t \tag{5.5}$$

where $\|\cdot - \cdot\|_M^2$ stands for the Mahalanobis distance with respect to matrix M; in our case we set $M = \mathbf{M}_{t-1}$, see (5.4). (Fearnhead and Taylor (2013) set M to the covariance matrix of the particles at time $t-1$, but, again, this requires inverting a full matrix, which may be too expensive in high-dimensional problems.)

By maximizing $g_t(h)$ we minimize the first-order autocorrelation of the chain. This leads to a reduced asymptotic variance of the chain and hopefully to a reduced asymptotic variance for estimates obtained from the SMC sampler.

FT has the following steps:

1. Assign different values of h_t^i according to their performance to the resampled particles $\tilde{x}_{t-1}^i$.
2. Propagate $x_t^i \sim \mathcal{K}_t^{h_t^i}(\tilde{x}_{t-1}^i, dx)$.
3. Evaluate the performance of h_t^i based on $x_t^i, \tilde{x}_{t-1}^i$.

We now turn to choice of the actual performance metric. As in Fearnhead and Taylor (2013), we use the following Rao-Blackwellized estimator of (5.5):

$$\tilde{\Lambda}(\tilde{x}_{t-1}^i, \hat{x}_t^i) = \frac{\|\tilde{x}_{t-1}^i - \hat{x}_t^i\|_M^2}{L} \times \max(1, \exp[\Delta E_t^i]). \tag{5.6}$$

Here $\hat{x}_t^i$ is the proposed position based on the Hamiltonian flow $\widehat{\Phi}_{\epsilon,L}(\tilde{x}_{t-1}^i, p_t^i)$ before the MH step. The acceptance rate $\max(1, \exp[\Delta E_t^i])$ of the MH step is based on the variation of the energy ΔE_t^i, and serves as weight.

Note that (5.6) is divided by L, in order to account for the computational cost of increasing L. Mohamed et al. (2013) normalise their criterion by $L^{1/2}$, claiming good

practical performance in the problems they study; of course other powers of L could be considered.

The pairs $h_t^i = (\epsilon_t^i, L_t^i)$ are weighted according to the performance metric $\tilde{\Lambda}(\tilde{x}_{t-1}^i, \hat{x}_t^i)$. The next set of parameters h_{t+1}^i are sampled from

$$\chi_{t+1}(h) \propto \sum_{i=1}^{N} \tilde{\Lambda}(\tilde{x}_{t-1}^i, \hat{x}_t^i) R(h; h_t^i),$$

where R is a perturbation kernel. We suggest to set R to

$$R(h; h_{t-1}^i) = \mathcal{TN}(\epsilon; \epsilon_{t-1}^i, 0.015^2) \otimes \left\{ \frac{1}{3}\mathbb{1}\left\{L_{t-1}^i - 1\right\}(L) + \frac{1}{3}\mathbb{1}\left\{L_{t-1}^i\right\}(L) + \frac{1}{3}\mathbb{1}\left\{L_{t-1}^i + 1\right\}(L) \right\},$$

where $\mathcal{TN}$ denotes a normal distribution truncated to $\mathbb{R}^+$. Thus ϵ_{t-1}^i is perturbed by a small (truncated) Gaussian noise, and L has an equal chance of increasing, decreasing or staying the same. The variance of the Gaussian noise is set to the value suggested by Fearnhead and Taylor (2013). The resulting algorithm is given in Algorithm 16.

Algorithm 16: (FT) Tuning of the HMC algorithm based on Fearnhead and Taylor (2013)

Input: Previous parameters h_{t-1}^i, estimator of associated utility $\tilde{\Lambda}(\tilde{x}_{t-2}^i, \hat{x}_{t-1}^i)$, $i \in 1:N$, perturbation kernel R

Result: Sample of $h_t^i = (\epsilon_t^i, L_t^i)$, $i \in 1:N$

1 **foreach** $i \in 1:N$ **do**

2 Sample $h_t^i \sim \chi_t(h) \propto \sum_{i=1}^{N} \tilde{\Lambda}(\tilde{x}_{t-2}^i, \hat{x}_{t-1}^i) R(h; h_{t-1}^i)$;

5.3.3 Pretuning of the kernel at every time step

The previous tuning algorithm relies on the assumption that parameters suited for the kernel used at time $t-1$ will also achieve good performance at time t.

We suggest as an alternative to the previous approach the following two-stage procedure:

1. We apply a HMC step (with respect to π_{t-1}) to the N current particles; for each particle the value of (ϵ, L) is chosen randomly from a certain uniform distribution (described in next section). We then construct a new random distribution for (ϵ, L) based on the performance of these N steps (Section 5.3.3). The HMC trajectories are then discarded.

2. We apply again a HMC step to the N current particles, except this time (L, ϵ) is generated from the distribution constructed in the previous step.

We now explain this approach in more detail.

Range of values for ϵ

In the first stage of our pre-tuning parameter, ϵ is generated from $\mathcal{U}[0, \epsilon^\star_{t-1}]$, and L from $\mathcal{U}[0, L_{\max}]$. We discuss in this section how to choose $\epsilon^\star_{t-1}$.

Our approach is motivated by (5.2). If for different step sizes $\hat{\epsilon}^i_t$ and different momenta and positions $(p^i_t, \tilde{x}^i_{t-1})$ for $i \in \{1:N\}$ we observe $\left|\Delta E^i_t\right| = \left|H(\hat{p}^i_t, \hat{x}^i_t) - H(p^i_t, \tilde{x}^i_{t-1})\right|$, this information may be used to fit a model of the form

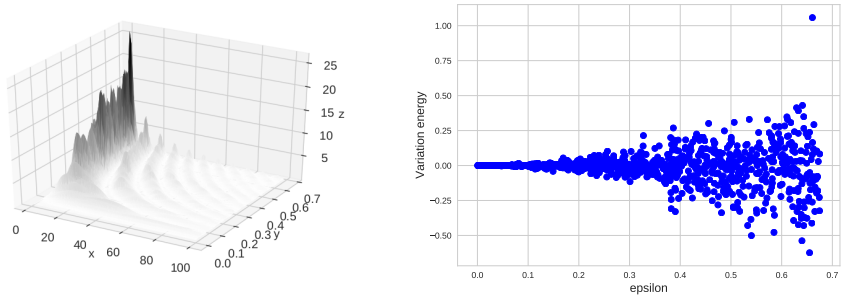
$$\left|\Delta E^i_t\right| = f(\hat{\epsilon}^i_t) + \xi^i_t,$$

where ξ^i_t is assumed to be such that $\forall i, \mathbb{E}\left[\xi^i_t\right] = 0, \mathrm{Var}\left[\xi^i_t\right] = \sigma^2 < \infty$ and $f : \mathbb{R}^+ \to \mathbb{R}^+$. We may then choose $\epsilon^\star$ so that $f(\epsilon^\star) = |\log 0.9|$. This ensures that the acceptance rate of the HMC kernel stays close to 90%, following the suggestions of Betancourt et al. (2014).

In particular we suggest to use the model

$$f(\hat{\epsilon}^i_t) = \alpha_0 + \alpha_1 (\hat{\epsilon}^i_t)^2,$$

and we minimize the sum of the absolute value errors $\sum_{i=1}^N \left|\xi^i_t\right|$ w.r.t. (α_0, α_1), which amounts to a median regression. Compared to least squares regression, this approach is more robust to high fluctuations in the energy which typically occur when ϵ approaches its stability limit. We illustrate this point in Figure 5.1b.



(A) (B)

FIGURE 5.1: Tempering of a normal distribution to a shifted and correlated normal distribution in dimension 10 (see the example in Section 5.4.1 for more details). Left: The normalized and weighted squared jumping distance (z-axis) as a function of ϵ (y-axis) and L (x-axis) for the temperature 0.008. Right: Variation of the difference in energy ΔE as a function of ϵ for the same temperature. The values of L are randomized. Based on an SMC sampler with an HMC kernel based on $N = 1,024$ particles.

Construction of a random distribution for (ϵ, L)

Algorithm 17 describes how we generate values for (ϵ, L) during the second stage of our pre-tuning procedure. In words, these values are sampled from the weighted

empirical distribution that correspond to the N values $(\hat{\epsilon}_t^i, \hat{L}_t^i)$ generated (uniformly) during the first stage, with weights given by performance metric (5.6).

Algorithm 17: (PR) Pre-tuning of the HMC kernel

Input: Resampled particles $\tilde{x}_{t-1}^i$, $i \in 1:N$, HMC flow $\widehat{\Phi}_{\cdot,\cdot}$ (targeting π_{t-1}), $\epsilon_{t-1}^\star$
Result: Sample of (ϵ_t^i, L_t^i), $i \in 1:N$, upper bound $\epsilon_t^\star$

foreach $i \in 1:N$ **do**
  Sample $\hat{\epsilon}_t^i \sim \mathcal{U}[0, \epsilon_{t-1}^\star]$ and $\hat{L}_t^i \sim \mathcal{U}\{1:L_{max}\}$;
  Sample $p_t^i \sim \mathcal{N}(0_d, \mathbf{M}_{t-1})$;
  Apply the leapfrog integration: $(\hat{p}_t^i, \hat{x}_t^i) \leftarrow \widehat{\Phi}_{\hat{\epsilon}_t^i, \hat{L}_t^i}(p_t^i, \tilde{x}_{t-1}^i)$;
  Calculate ΔE_t^i and $\tilde{\Lambda}(\tilde{x}_{t-1}^i, \hat{x}_t^i)$
Calculate $\epsilon_t^\star$ based on the quantile regression of ΔE_t^i on $\hat{\epsilon}_t^i$ $\forall i \in 1:N$;
Sample $(\epsilon_t^i, L_t^i) \sim \mathcal{C}at\left(w_t^i, \{\hat{\epsilon}_t^i, \hat{L}_t^i\}\right)$, where $w_t^i \propto \tilde{\Lambda}(\tilde{x}_{t-1}^i, \hat{x}_t^i)$ $\forall i \in 1:N$;

Range of values for L

During the first stage of our pre-tuning procedure, L is generated uniformly within range $[0, L_{\text{max}}]$. The quantity L_{max} is initialized to some user-chosen value (we took $L_{\text{max}} = 100$ in our simulations). Whenever a large proportion of the L_t^i generated by Algorithm 17 is close to L_{max}, we increase L_{max} by a small amount (5 in our simulations). Similarly, whenever a large proportion of these values are far away from L_{max}, we decrease L_{max} by the same small amount.

5.3.4 Discussion of the tuning procedures

Comparison of the two algorithms The main difference between the two procedures consists in the pre-tuning phase of the sampler. On one hand, pre-tuning makes the SMC sampler more costly. On the other hand this approach makes the sampler more robust to a sudden change in the sequence of distributions. We illustrate this point in our numerical experiments.

Other potential approaches to tuning HMC within SMC One could try to maximize the squared jumping distance as a function of the position of every particle, based on the associated values of (ϵ, L). However, this is not practical as it will be too expensive to learn optimal parameters for every position.

In line with Girolami and Calderhead (2011) one could use a position dependent mass matrix that would take the geometry of the target space into account.

Returning to the choice of (ϵ, L) one could use an approach based on Bayesian optimization (Snoek et al., 2012), based on the performance of the $(\epsilon_{t-1}^i, L_{t-1}^i)$ at the previous iteration. This idea would amount to a parallel version of Mohamed et al. (2013). However, it is not clear how this approach would behave if the underlying distributions evolve over time. Not using a pre-tuning step reduces the computational

load at the expense of making the sampler potentially less robust. Moreover, the approach of Fearnhead and Taylor (2013) already explores the hyperparameter space adaptively without requiring additional model specifications. If framed as a Bandit problem, fixing over time a grid of possible values (ϵ, L) could be problematic if the grid misses relevant parts of the hyperparameter space. This holds true in particular when using continuous Bayesian optimization, where one typically has to define some box constraints on the underlying space.

Both of our suggested tuning procedures are $\mathcal{O}(N)$, which makes theses approaches scalable.

Extensions The tuning procedure based on pre-tuning might also be used for tuning random walk (RW) Metropolis or MALA (Metropolis adjusted Langevin) kernels. In the first case we may use median regression to find an upper bound for the scale such that the acceptance rate is close to 23.4% (Roberts et al., 1997). In the second case one may target an acceptance rate of 57.4% (Roberts and Rosenthal, 1998). (MALA kernels may be viewed as HMC kernels with $L = 1$.) It recently came to our knowledge that the work of Salomone et al. (2018) also uses a pre-tuning approach for RW or MALA based kernels within SMC samplers.

5.4 Experiments

Our experiments highlight the importance of adapting SMC samplers, in particular the parameters of their Markov kernels. Specifically, we try to answer the following questions. How important is is to adapt (a) the number of temperature steps and (b) the number of move steps? (c) Does our tuning procedure of HMC kernels provide reasonable values of (ϵ, L) compared to other tuning procedures of HMC? (d) To what extent does HMC within an SMC sampler scale with the dimension and may be applied to real data applications? (e) How robust are SMC samplers to multimodality?

For this purpose we compare adaptive (A) and non-adaptive (N) versions of HMC-based SMC samplers, where the adaptation may be carried out using either the FT approach (our variant of the approach of Fearnhead and Taylor, 2013) or the PR (pre-tuning) approach. We also include in our comparison SMC samplers based on random walk (RW) and MALA kernels, and the FT adaptation procedure. We call our algorithms accordingly: i.e. HMCAFT stands for an SMC sampler using HMC kernels, which are adapted using the FT procedure.

In all the considered SMC samplers, the mass matrix $\mathbf{M}_t$ of the MCMC kernels is set the diagonal of the covariance matrix obtained at the previous iteration. Unless otherwise stated, the number of particles is $N = 1,024$ and the resampling is triggered when the ESS drops below $N/2$. The computational load of a given sampler is defined as the total number of gradient evaluations. Most of our comparisons are in terms of adjusted variance, or adjusted MSE (mean squared error), by which we mean: variance (or MSE) times computational load.

All HMC-based samplers are initialized with uniform random draws of ϵ on $[0, 0.1]$ and L on $\{1, 100\}$. The MALA and RW-based samplers are initialized with random draws of the scale parameter on $[0, 1]$. The initial mass matrix is set to the identity for all different samplers. All samplers choose adaptively the number of move steps based on Algorithm 14. Code for reproducing the results shown below is available under https://github.com/alexanderbuchholz/hsmc.

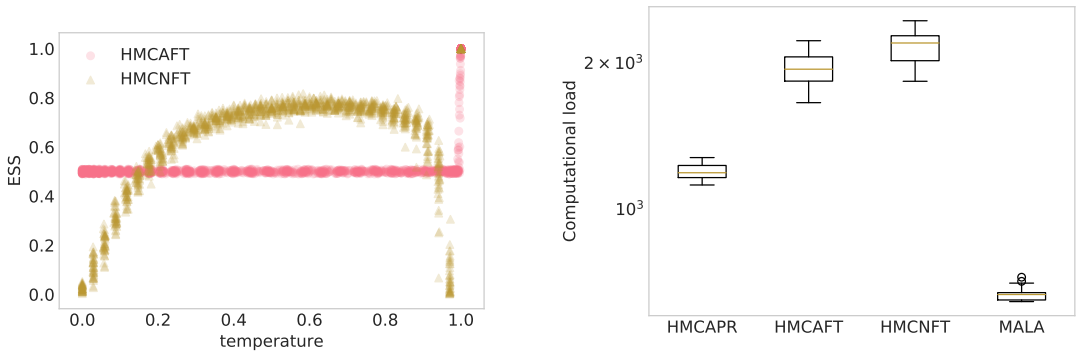
5.4.1 Tempering from an isotropic Gaussian to a shifted correlated Gaussian

As a first toy example we consider a tempering sequence that starts at an isotropic Gaussian $\pi_0 = \mathcal{N}(0_d, I_d)$, and finishes at a shifted and correlated Gaussian $\pi_T = \mathcal{N}(\mu, \Xi)$, where $\mu = 2 \times \mathbf{1}_d$, for different values of d. For the covariance we set the off-diagonal correlation to 0.7 and the variances to elements of the equally spaced sequence $\tilde{\Xi} = [0.1, \cdots, 10]$. We get the covariance $\Xi = \operatorname{diag} \tilde{\Xi}^{1/2} \operatorname{corr}(X) \operatorname{diag} \tilde{\Xi}^{1/2}$. This toy example is rather challenging due to the different length scales of the variance, the correlation and the shifted mean of the target. In this example the true mean, variance and normalizing constants are available. Therefore we report the mean squared error (MSE) of the estimators. We use normalized importance weights and thus $Z_T / Z_0 = 1$.

We compare the following SMC samplers: MALA, HMCAFT and HMCAPR (according to the denomination laid out in the previous section). We add to the comparison HMCNFT, an SMC sampler using adaptive (FT based) HMC steps, but where the sequence of temperatures is fixed a priori to a long equi-spaced sequence (the size of which is set according to the number of temperatures chosen adaptively during one run of HMCAFT).

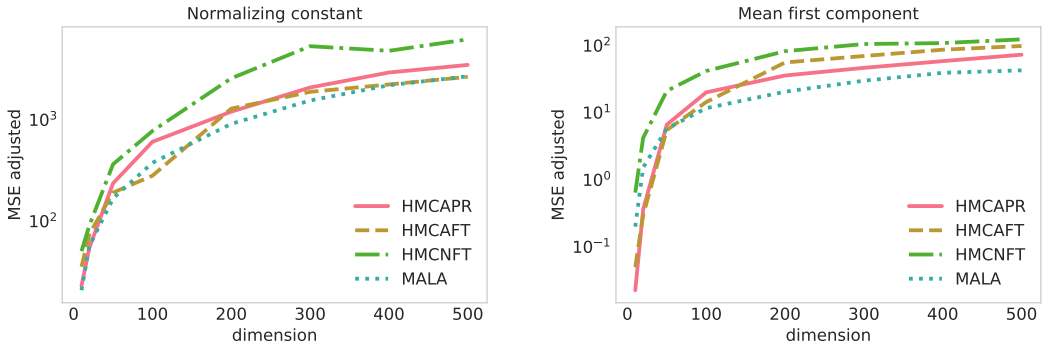
Figure 5.2a plots the ESS as a function of the temperature, for dimension $d = 500$, for algorithms HMCAFT and HMCNFT. Figures 2b and 3 compare the SMC samplers in terms of computational load (Figure 2b) and adjusted MSE (i.e. MSE times the computational load) for the normalizing constant and the expectation of the first component (with respect to the target).

A first observation is that it is really useful to adapt the sequence of temperatures: HMCNFT is strongly outperformed by all the other algorithms. A second observation is that the MALA approach is competitive on this simple example: it is far less expensive, and, as result, it produces better adjusted MSE, even for high dimensions. A third observation is that the FT approach turns out to be slightly more expensive, but on the other hand it outperforms slightly the PR approach in terms of adjusted MSE.



(A) (B)

FIGURE 5.2: ESS and temperature steps in dimension $d = 500$ (Figure 5.2a) and the computational load of the sampler in terms of the number of total gradient evaluations (Figure 5.2b).



(A) (B)

FIGURE 5.3: Mean squared error of the the normalization constant (Figure 5.3a) and the mean squared error of the first component of the mean (Figure 5.3b) multiplied by the computational cost over dimensions. Based on 40 repetitions of the samplers with $N = 1,024$ particles.

In order to better assess the performance of the two tuning procedures (FT and PR), we compare the tuning parameters obtained at the final stage of our SMC samplers (HMCAFT and HMCAPR) with those obtained from the following MCMC procedures: NUTS (no u-turn sampler, Hoffman and Gelman, 2014) and the adaptive MCMC algorithm of Mohamed et al. (2013). NUTS iteratively tunes the MASS matrix $\mathbf{M}$, the number of leapfrog steps L and the step size ϵ in order to achieve a high ESJD (expected squared jumping distance). The adaptive HMC sampler of Mohamed et al. (2013) uses Bayesian optimization in order to find values of (ϵ, L) that yield a high ESJD.

For this purpose we run our HMC-based SMC samplers and record the achieved ESJD of the HMC kernel at the final distribution π_T. NUTS and the adaptive HMC sampler are run directly on the final target distribution. For NUTS we use the implementation available in STAN; for the adaptive HMC sampler we reimplemented the procedure of Mohamed et al. (2013). After the convergence of the tuning procedures

we run the chain for 2,000 iterations and discard a burnin of 1,000 samples. The ESJD is calculated on the remaining 1,000 iterations of the chain. The results of this comparison are shown in Table 5.1. We see that both the HMC-based SMC samplers, NUTS and the adaptive HMC tuning achieve an ESJD of the same order of magnitude. Our tuning procedure gives values of (ϵ, L) that yield an ESJD comparable to other existing procedures, although being based on a different tuning paradigm.

Dimension	SMC HMCAPR	SMC HMCAFT	SMC MALA	NUTS	adaptive HMC
10	50.64	61.03	9.89	59.88	134.70
50	174.64	255.98	30.56	204.34	190.67
200	639.35	989.97	85.22	1,281.06	927.30
500	1,556.27	1,311.60	154.05	2,210.44	1,731.04

TABLE 5.1: Average squared jumping distance of different algorithms for the Gaussian target in the first example (based on 40 runs). The results are based on 1,024 samples for the SMC samplers. For the samplers based on a HMC chain we used a length of 2,000 states where the first 1,000 states are discarded as burn-in.

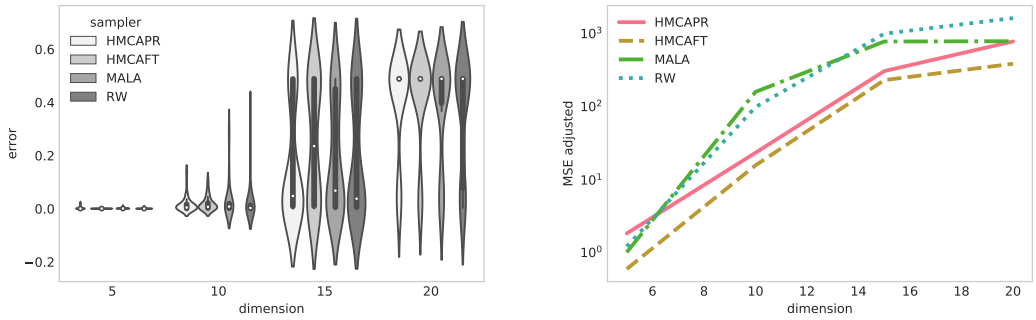
5.4.2 Tempering from a Gaussian to a mixture of two correlated Gaussians

The aim of our second example is to assess the robustness of SMC samplers with respect to multimodality. We temper from the prior $\pi_0 = \mathcal{N}(\mu_0, 5I_d)$ towards a mixture of shifted and correlated Gaussians $\pi_T = 0.3\mathcal{N}(\mu, \Xi_1) + 0.7\mathcal{N}(-\mu, \Xi_2)$, where $\mu = 4 \times \mathbf{1}_d$ and we set the off diagonal correlation to 0.7 for Ξ_1 and to 0.1 for Ξ_2. The variances are set to elements of the equally spaced sequence $\tilde{\Xi}_j = [1, \cdots, 2]$ for $j = 1, 2$. The covariances Ξ_j are based on the same formula as in the first example. In order to make the example more challenging we set $\mu_0 = \mathbf{1}_d$. Thus, the start of the sampler is slightly biased towards one of the two modes. We evaluate the performance of the samplers by evaluating the signs of the particles and use therefore the statistic $T_i := 1/d \sum_{j=1}^{d} \mathbb{1}_{\{\text{sign } X_{j,i}=+1\}}$, based on a single particle. We expect a proportion of 30% of the signs to be positive, i.e. $1/N \sum_{i=1}^{N} T_i \approx 0.3$, if the modes are correctly recovered. Our measure of error is based on the squared deviation from this value. We consider the following SMC samplers: MALA, RW, HMCAFT, and HMCAPR. All the samplers choose adaptively the number of move steps and the temperatures.

As shown by Figure 5.4b the two HMC-based samplers yield a lower error of the recovered modes adjusted for computation in moderate dimensions. In terms of the recovery of the modes all samplers behave comparable as illustrates Figure 5.4a. Nevertheless, as the dimension of the problem exceeds 20 all samplers tend to concentrate on one single mode. This problem may be solved by initializing the sampler with a wider distribution. However, this approach relies on the knowledge of the location of the modes.

Consequently, SMC samplers are robust to multimodality only in small dimensions and care must be taken when using SMC samplers in this setting. That said,

HMC-based samplers seem slightly more robust to multimodality; see e.g. results for $d = 10$ in Figure 5.4a. See also the recent work of Mangoubi et al. (2018) for the performance of HMC versus RWMH on multimodal targets.

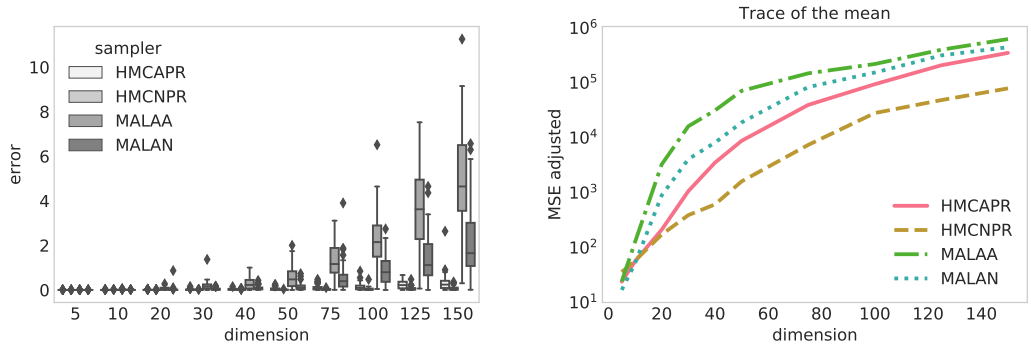


(A) (B)

FIGURE 5.4: Left: Violinplot of the error, based on the squared difference of $1/N \sum_{i=1}^{N} T_i - 0.3$. Right: Mean squared error of the proportion of recovered modes adjusted for the computation. Based on 40 runs of the samplers.

5.4.3 Tempering from an isotropic Student distribution to a shifted correlated Student distribution

As a different challenging toy example we study the tempering from an isotropic Student distribution $\pi_0 = \mathcal{T}_3(0_d, I_d)$ with 3 degrees of freedom towards a shifted and correlated Student distribution with $\nu = 10$ degrees of freedom, i.e. $\pi_T = \mathcal{T}_\nu(\mu, \Xi)$. The mean and scale matrix are set as in the unimodal Gaussian example in 5.4.1.



(A) (B)

FIGURE 5.5: Figure 5.5a shows the squared error of the estimator of the normalizing constant. Figure 5.5b shows the squared error of the trace of the mean over different dimensions adjust for computation. The results are based on 100 runs of the samplers with $N = 1,024$ particles.

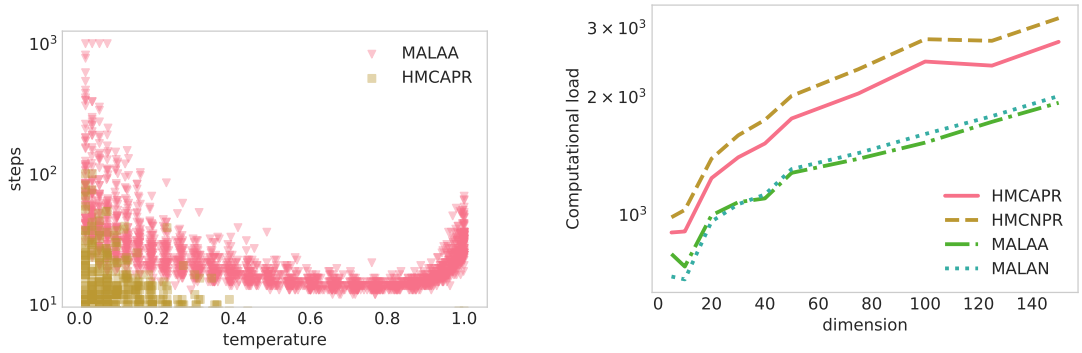
This example is more challenging as the target distribution π_T has heavy tails. We vary the dimension between $d = 5$ and $d = 150$. The temperature steps are chosen such that an ESS of 90% is attained. The adaptive samplers (A) adjust the number of

move steps according to Algorithm 14. For the non-adaptive samplers (N) the number of move steps is fixed to a constant number, equal to the average number of move steps over the complete run of an adaptive sampler. The temperatures are chosen adaptively. The MALA-based sampler uses FT tuning, the HMC-based sampler uses our pre-tuning (PR) approach.

The HMC-based samplers perform better when it comes to estimating the mean (see Figure 5.5b) and the normalizing constant (see Figure 5.5a). Both samplers based on a MALA kernel tend to work poorly in terms of the estimation of the normalizing constant when the dimension increases, see Figure 5.5a.

The adaptation of the number of move steps has some negative impact on the performance of the samplers. Setting the number of move steps to a fixed, large value may be beneficial as our simulation suggests. Our approach provides a guideline for finding this number.

In summary we suggest to use a fixed, large number of move steps found using our suggested adaptation and HMC kernels for heavy tailed distribution such as the Student-t distribution.



(A) (B)

FIGURE 5.6: Figure 5.6a shows the number of rejuvenation steps over the different temperature in dimension 150. Figure 5.6b shows the computational load over dimensions. The results are based on 100 runs of the samplers with $N = 1,024$ particles.

5.4.4 Binary regression posterior

We now consider a Bayesian binary regression model. We observe J vectors $z_j \in \mathbb{R}^d$ and J outcomes $y_j \in \{0,1\}$. We assume $y_j \sim \mathcal{B}er(r(z_j^T\beta))$ where $r : \mathbb{R} \mapsto [0,1]$ is a link function. The parameter of interest is $\beta \in \mathbb{R}^d$, endowed with a Gaussian prior, i.e. $\beta \sim \mathcal{N}(0_d, I_d)$.

When using the logit link function $r : x \to \exp(-x)/(1+\exp(-x))$ we obtain a Bayesian logistic regression where the unnormalized log posterior is given as

$$\gamma(\beta) = \sum_{j=1}^{J} \left[(y_j - 1) z_j^T \beta - \log(1 + \exp(-z_j^T \beta)) \right] - 1/2 \, \|\beta\|_2^2 .$$

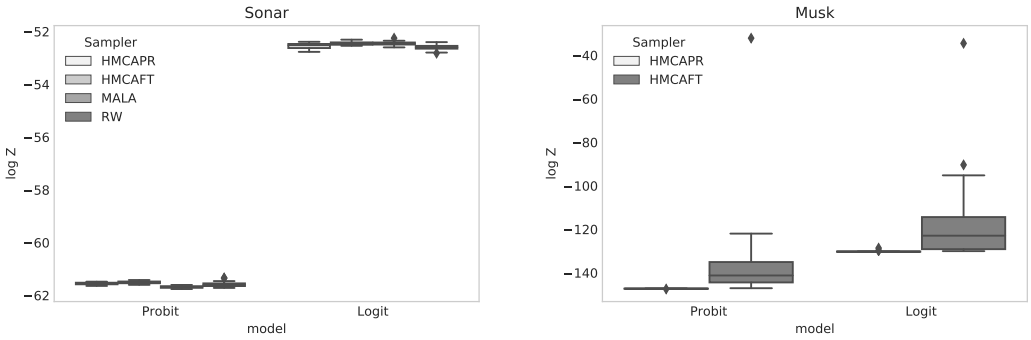
When using as link function the cumulative distribution function of a standard normal distribution, denoted Φ, one obtains the Bayesian Probit regression. In this case the unnormalized log posterior is given as

$$\gamma(\beta) = \sum_{j=1}^{J} \left[y_j \log \Phi(z_j^T \beta) + (1 - y_j) \log(1 - \Phi(z_j^T \beta)) \right] - 1/2 \, \|\beta\|_2^2 .$$

We set π_0 to a Gaussian approximation of the posterior obtained by Expectation Propagation (Minka, 2001) as in Chopin and Ridgway (2017).

We consider two datasets (both available in the UCI repository): sonar ($d = 61$ covariates, $J = 208$ observations) and Musk ($d = 95$, $J = 476$, after removing a few covariates that are highly correlated with the rest). In both cases, an intercept is added, and the predictors are normalised (to have mean 0 and variance 1).

We compare the following SMC samplers: RW, MALA, HMCAFT, HMCAPR; see Figure 5.7a for the estimation of the marginal likelihoods (which may be used to perform model choice) and Figure 5.7b for the estimation of the posterior expectation of the first component. For all samplers we adapt the number of move steps and the temperature steps.



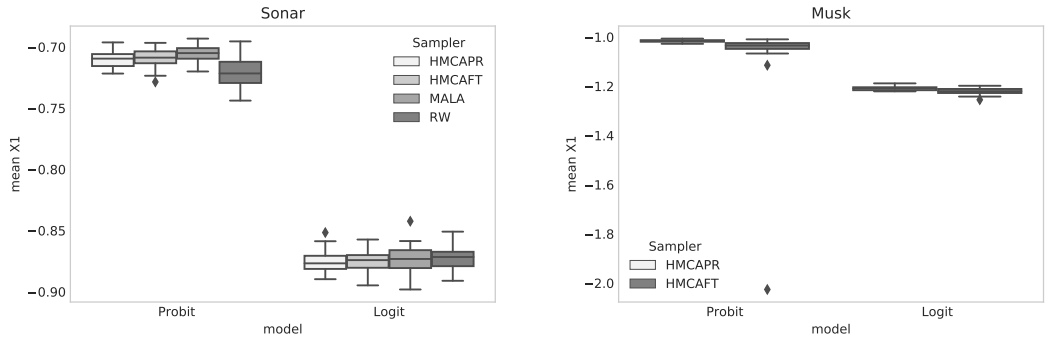
(A) (B)

FIGURE 5.7: Normalization constants obtained for the probit and logit regression based on 40 runs of the samplers. Figure 5.7a corresponds to the normalization constants obtained for the sonar dataset. The posterior has 61 dimensions. Figure 5.7b corresponds to the Musk dataset. The posterior has 95 dimensions.

As our results illustrate, all the samplers are able to recover the posterior distribution correctly. However, estimating the normalizing constant is more difficult. The sampler based on FT struggles for the Musk dataset. This may be due to the high correlation of the regressors. As FT adapts less easily to changing distributions, this tuning procedure fails here. MALA and RW-based kernels exceeded the fixed computational budget on the Musk dataset, therefore we omit the comparison. Moreover, the computation time necessary varies widely across the different approaches; see Table 5.2. The PR tuning for HMC based samplers works well in both settings and there

is no clear winner both in terms of estimating the normalizing constant and the first moment.

The MALA-based sampler performs overall better than RW, and the HMC-based samplers clearly perform better in terms of adjusted variance (variance times computational load). In the presence of strongly correlated features we recommend the adaptation procedure based on pre-tuning (PT), as this approach outperforms the adaptation of FT. This holds in particular when interest lies in calculating the evidence of the model. If the kernel mixes badly at some time during the tempering, this translates directly into a poor estimation of an intermediate normalizing constant. Contrary to the estimation of a moment of the final distribution, an error based on intermediate weights has a major impact on this quantity of interest, as a better mixing at subsequent temperatures cannot remove this previous error.



(A) (B)

FIGURE 5.8: Estimated mean of the first component of the posterior obtained for the probit and logit regression. Figure 5.8a corresponds to the sonar dataset. Figure 5.8b corresponds to the Musk dataset.

		Normalizing constant							
		Logit				Probit			
Dataset	Dim	HMCAPR	HMCAFT	MALA	RW	HMCAPR	HMCAFT	MALA	RW
Sonar	61	1.0	0.405	2.461	9.521	1.0	0.872	7.397	20.304
Musk	95	1.0	272.135	-	-	1.0	9.877	-	-
		Mean first component							
		Logit				Probit			
Dataset	Dim	HMCAPR	HMCAFT	MALA	RW	HMCAPR	HMCAFT	MALA	RW
Sonar	61	1.0	0.791	7.806	16.774	1.0	0.894	5.963	31.037
Musk	95	1.0	1.414	-	-	1.0	2.31	-	-

TABLE 5.2: Ratio of the variance of the normalizing constant multiplied by the mean number of gradient evaluations based on 40 runs of the different samplers. For the RWMH sampler we adjust by the number of likelihood evaluations.

5.4.5 Log Gaussian Cox model

In order to illustrate the advantage of HMC-based SMC samplers in high dimensions we consider the log Gaussian Cox point process model in Girolami and Calderhead (2011), applied to the Finnish pine saplings dataset. This dataset consists of the geographic position of 126 trees. The aim is to recover the latent process $X \in \mathbb{R}^{d\times d}$ from the realization of a Poisson process $Y = (y_{j,k})_{j,k}$ with intensity $m\Lambda(j,k) = m\exp(x_{j,k})$, where $m = 1/d^2$ and $X = (x_{j,k})_{j,k}$ is a Gaussian process with mean $\mathbb{E}[X] = \mu \times \mathbf{1}_{d\times d}$ and covariance function $\Sigma_{(j,k)(j',k')} = \sigma^2 \exp(-\delta(j,j',k,k')/(d\beta))$, where $\delta(j,j',k,k') = \sqrt{(j-j')^2 + (k-k')^2}$.

The posterior density of the model is given as

$$p(x|y,\mu,\sigma^2,\beta) \propto \left\{\prod_{j,k}^{d} \exp(y_{j,k}x_{j,k} - m\exp(x_{j,k}))\right\} \times \exp\left\{-1/2(x-\mu)^T\Sigma^{-1}(x-\mu)\right\},$$

where the second factor is the Gaussian prior density.

Following the results of Christensen et al. (2005) we fix $\beta = 1/33$, $\sigma^2 = 1.91$ and $\mu = \log(126) - \sigma^2/2$. We vary the dimension of the problem from $d = 100$ to $d = 4,096$ by considering different discretizations. We consider three SMC samplers: MALA, HMCAFT and HMCAPR. The starting distribution is the prior.

Figure 5.9b shows that all the samplers estimate well the normalizing constant, even for a large dimension d. Moreover, we estimate the cumulative mean throughout different dimensions with relatively low variance, see Figure 5.9b. We omitted the simulation for the MALA-based samplers, as the simulation took excessively long in dimension $4,096$ due to slow mixing of the kernel. The estimated posterior mean of the latent field for dimension 900 is plotted in Figure 5.9c.

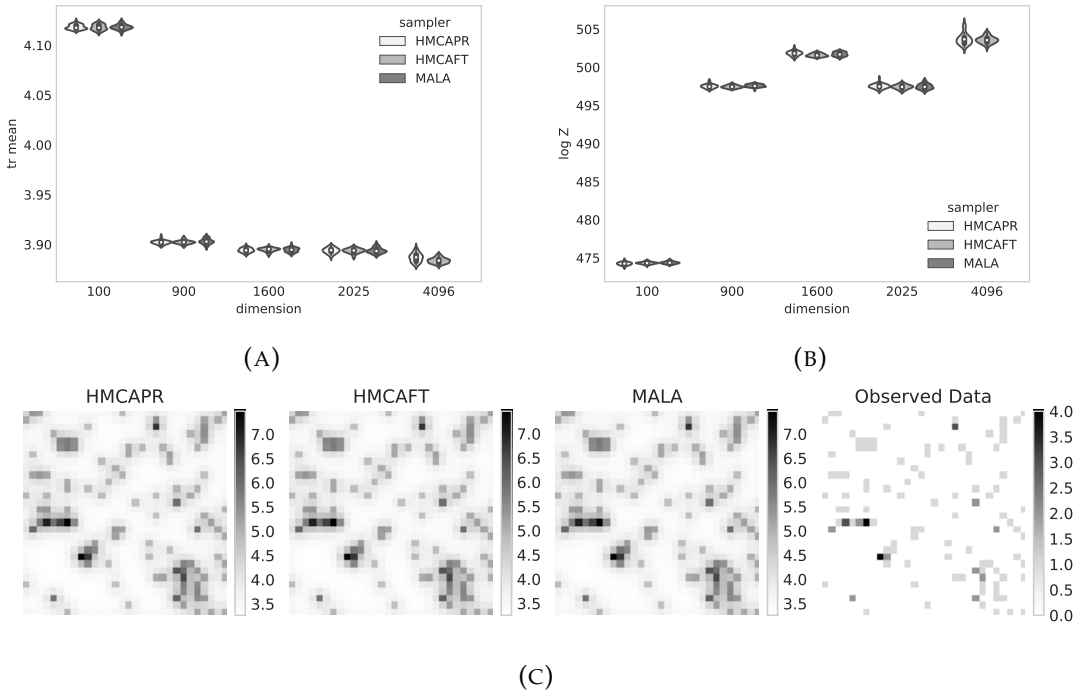


FIGURE 5.9: Tempering from a normal prior to the posterior of log Gaussian Cox process over various dimensions. Figure 5.9a illustrates the estimations of the normalizing constants. Figure 5.9b illustrates the estimated cumulative posterior mean. Figure 5.9c illustrates the recovered componentwise posterior mean of the process in dimension 900.

Table 5.3 reports adjusted variances (variances times computational load) for the considered SMC samplers. We see that the MALA-based sampler is not competitive when the dimension increases. Regarding the adaptation scheme, FT outperforms PR, especially in high dimension.

	Normalizing constant			Mean first component		
Dim	HMCAPR	HMCAFT	MALA	HMCAPR	HMCAFT	MALA
100	1.0	1.182	0.813	1.0	2.739	1.381
400	1.0	0.399	0.867	1.0	0.872	1.249
900	1.0	0.413	0.744	1.0	0.691	2.624
1,600	1.0	0.251	1.068	1.0	0.464	1.63
2,500	1.0	0.519	2.002	1.0	0.457	1.567
4,096	1.0	0.199	-	1.0	0.203	-

TABLE 5.3: Ratio of the variance of the normalizing constant and the mean of the particles multiplied by the mean number of likelihood evaluations based on 40 runs of the samplers.

5.5 Discussion

Our experiments indicate that the relative performance of HMC kernels within SMC depends on the dimension of the problem. For low to medium dimensions, RW and MALA are much faster, and tend to perform reasonably well. On the other hand, for high dimensions, HMC kernels outperform, sometimes significantly, RW and MALA kernels.

The key to good performance of SMC samplers (based on HMC or other kernels) is to adaptively tune the Markov kernels used in the propagation step. We have considered two approaches in this paper. On posterior distributions with reasonable correlation our adaption of the approach by Fearnhead and Taylor (2013) works best. Our approach based on pre-tuning of the HMC kernels is more robust to changes in the subsequent target distributions. From a practical point of view and if the structure of the posterior is unknown the second approach may be the more prudent choice.

All in all, our methodology provides a principled approach for an automatic adaption of SMC samplers, applicable over a range of various models in different dimensions.

Acknowledgments

The first author gratefully acknowledges a GENES research scholarship and a DAAD grant for visiting the third author. The second author is partly supported by Labex Ecodec (anr-11-labx-0047). The third author gratefully acknowledges support by the National Science Foundation through grants DMS-1712872.

Bibliography

Agapiou, S., Papaspiliopoulos, O., Sanz-Alonso, D., and Stuart, A. M. (2015). Importance sampling: computational complexity and intrinsic dimension. *arXiv preprint 1511.06196*.

Alquier, P., Friel, N., Everitt, R., and Boland, A. (2016a). Noisy monte carlo: Convergence of markov chains with approximate transition kernels. *Statistics and Computing*, 26(1-2):29–47.

Alquier, P., Ridgway, J., and Chopin, N. (2016b). On the properties of variational approximations of gibbs posteriors. *The Journal of Machine Learning Research*, 17(1):8374–8414.

Andrieu, C., Doucet, A., and Holenstein, R. (2010). Particle markov chain monte carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(3):269–342.

Andrieu, C. and Roberts, G. O. (2009). The pseudo-marginal approach for efficient Monte Carlo computations. *Ann. Statist.*, 37(2):697–725.

Andrieu, C. and Thoms, J. (2008). A tutorial on adaptive MCMC. *Statistics and computing*, 18(4):343–373.

Antonov, I. A. and Saleev, V. (1979). An economic method of computing LPτ-sequences. *USSR Computational Mathematics and Mathematical Physics*, 19(1):252–256.

Atchadé, Y. F. and Rosenthal, J. S. (2005). On adaptive markov chain monte carlo algorithms. *Bernoulli*, 11(5):815–828.

Bardenet, R., Doucet, A., and Holmes, C. (2017). On markov chain monte carlo methods for tall data. *The Journal of Machine Learning Research*, 18(1):1515–1557.

Barthelmé, S. and Chopin, N. (2014). Expectation propagation for likelihood-free inference. *Journal of the American Statistical Association*, 109(505):315–333.

Beaumont, M. A. (2003). Estimation of population growth or decline in genetically monitored populations. *Genetics*, 164(3):1139–1160.

Beaumont, M. A., Cornuet, J.-M., Marin, J.-M., and Robert, C. P. (2009). Adaptive approximate Bayesian computation. *Biometrika*, 96(4):983–990.

Bernton, E., Jacob, P. E., Gerber, M., and Robert, C. P. (2017). Inference in generative models using the Wasserstein distance. *arXiv preprint arXiv:1701.05146*.

Beskos, A., Jasra, A., Kantas, N., and Thiery, A. (2016). On the convergence of adaptive sequential Monte Carlo methods. *The Annals of Applied Probability*, 26(2):1111–1146.

Beskos, A., Pillai, N., Roberts, G., Sanz-Serna, J.-M., Stuart, A., et al. (2013). Optimal tuning of the hybrid Monte Carlo algorithm. *Bernoulli*, 19(5A):1501–1534.

Betancourt, M. (2016). Identifying the optimal integration time in Hamiltonian Monte Carlo. *arXiv preprint arXiv:1601.00225*.

Betancourt, M., Byrne, S., and Girolami, M. (2014). Optimizing the integrator step size for Hamiltonian Monte Carlo. *arXiv preprint arXiv:1411.6669*.

Bierkens, J., Fearnhead, P., and Roberts, G. (2016). The zig-zag process and super-efficient sampling for bayesian analysis of big data. *arXiv preprint arXiv:1607.03188*.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg.

Blei, D. M., Kucukelbir, A., and McAuliffe, J. D. (2017). Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518):859–877.

Blum, M. G. (2010). Approximate Bayesian computation: A nonparametric perspective. *Journal of the American Statistical Association*, 105(491):1178–1187.

Bornn, L., Pillai, N. S., Smith, A., and Woodard, D. (2015). The use of a single pseudo-sample in approximate Bayesian computation. *Statistics and Computing*, 27(3):1–14.

Bottou, L., Curtis, F. E., and Nocedal, J. (2016). Optimization methods for large-scale machine learning. *arXiv preprint arXiv:1606.04838*.

Bou-Rabee, N. and Sanz-Serna, J. M. (2018). Geometric integrators and the Hamiltonian Monte Carlo method. *Acta Numerica*, 27:113–206.

Bouchard-Côté, A., Vollmer, S. J., and Doucet, A. (2018). The bouncy particle sampler: A nonreversible rejection-free markov chain monte carlo method. *Journal of the American Statistical Association*, pages 1–13.

Buchholz, A. and Chopin, N. (2017). Improving approximate Bayesian computation via quasi Monte Carlo. *arXiv preprint arXiv:1710.01057*.

Burda, Y., Grosse, R., and Salakhutdinov, R. (2016). Importance weighted autoencoders. *Proceedings of the International Conference on Learning Representations*.

Cappé, O., Guillin, A., Marin, J.-M., and Robert, C. P. (2004). Population monte carlo. *Journal of Computational and Graphical Statistics*, 13(4):907–929.

Cappé, O., Rydén, O., Moulines, E., Ryden, T., and Rydén, T. (2005). *Inference in Hidden Markov Models*. Springer Series in Statistics. Springer.

Carpenter, B., Gelman, A., Hoffman, M., Lee, D., Goodrich, B., Betancourt, M., Brubaker, M., Guo, J., Li, P., and Riddell, A. (2017). Stan: A Probabilistic Programming Language. *Journal of Statistical Software, Articles*, 76(1):1–32.

Chatterjee, S., Diaconis, P., et al. (2018). The sample size required in importance sampling. *The Annals of Applied Probability*, 28(2):1099–1135.

Chen, S., Dick, J., Owen, A. B., et al. (2011). Consistency of markov chain quasi-monte carlo on continuous state spaces. *The Annals of Statistics*, 39(2):673–701.

Chen, T., Fox, E., and Guestrin, C. (2014). Stochastic gradient hamiltonian monte carlo. In *International Conference on Machine Learning*, pages 1683–1691.

Chérief-Abdellatif, B.-E. and Alquier, P. (2018). Consistency of variational bayes inference for estimation and model selection in mixtures. *arXiv preprint arXiv:1805.05054*.

Chopin, N. (2002). A sequential particle filter method for static models. *Biometrika*, 89(3):539–552.

Chopin, N. and Ridgway, J. (2017). Leave Pima Indians alone: binary regression as a benchmark for Bayesian computation. *Statistical Science*, 32(1):64–87.

Chopin, N., Rousseau, J., and Liseo, B. (2013). Computational aspects of Bayesian spectral density estimation. *Journal of Computational and Graphical Statistics*, 22(3):533–557.

Christensen, O. F., Roberts, G. O., and Rosenthal, J. S. (2005). Scaling limits for the transient phase of local Metropolis–Hastings algorithms. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):253–268.

Christophe, D. and Petr, S. (2015). *randtoolbox: Generating and Testing Random Numbers*. R package version 1.17.

Cornuet, J., Marin, J.-M., Mira, A., and Robert, C. P. (2012). Adaptive multiple importance sampling. *Scandinavian Journal of Statistics*, 39(4):798–812.

Cranley, R. and Patterson, T. N. (1976). Randomization of number theoretic methods for multiple integration. *SIAM Journal on Numerical Analysis*, 13(6):904–914.

Defazio, A., Bach, F., and Lacoste-Julien, S. (2014). Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in Neural Information Processing Systems*, pages 1646–1654.

Del Moral, P. (2004). Feynman-kac formulae: genealogical and interacting particle systems with applications. In *Feynman-Kac Formulae*, pages 47–93. Springer.

Del Moral, P., Doucet, A., and Jasra, A. (2006). Sequential Monte Carlo samplers. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(3):411–436.

Del Moral, P., Doucet, A., and Jasra, A. (2007). Sequential Monte Carlo for Bayesian Computation. *Bayesian Statistics*, (8):1–34.

Del Moral, P., Doucet, A., and Jasra, A. (2012). An adaptive sequential Monte Carlo method for approximate Bayesian computation. *Statistics and Computing*, 22(5):1009–1020.

Devroye, L. (1986). *Non-Uniform Random Variate Generation*. Springer-Verlag.

Dick, J., Kuo, F. Y., and Sloan, I. H. (2013). High-dimensional integration: the quasi-Monte Carlo way. *Acta Numerica*, 22:133–288.

Dick, J. and Pillichshammer, F. (2010). *Digital nets and sequences: discrepancy theory and quasi–Monte Carlo integration*. Cambridge University Press.

Drew, S. S. and Homem-de Mello, T. (2006). Quasi-monte carlo strategies for stochastic optimization. In *Proceedings of the 38th conference on Winter simulation*, pages 774–782. Winter Simulation Conference.

Drovandi, C. C. and Tran, M.-N. (2018). Improving the Efficiency of Fully Bayesian Optimal Design of Experiments Using Randomised Quasi-Monte Carlo. *Bayesian Anal.*, 13(1):139–162.

Duane, S., Kennedy, A. D., Pendleton, B. J., and Roweth, D. (1987). Hybrid Monte Carlo. *Physics letters B*, 195(2):216–222.

Duchi, J., Hazan, E., and Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159.

Fearnhead, P. and Prangle, D. (2012). Constructing summary statistics for approximate Bayesian computation: Semi-automatic approximate Bayesian computation. *Journal of the Royal Statistical Society. Series B: Statistical Methodology*, 74(3):419–474.

Fearnhead, P. and Taylor, B. M. (2013). An adaptive sequential Monte Carlo sampler. *Bayesian Analysis*, 8(2):411–438.

Frazier, D. T., Martin, G. M., Robert, C. P., and Rousseau, J. (2018). Asymptotic properties of approximate bayesian computation. *Biometrika*, page asy027.

Gelman, A. and Hill, J. (2006). *Data analysis using regression and multilevel/hierarchical models*. Cambridge university press.

Gelman, A., Kiss, A., and Fagan, J. (2006). An analysis of the nypd's stop-and-frisk policy in the context of claims of racial bias. *Columbia Public Law & Legal Theory Working Papers*, page 0595.

Geman, S. and Geman, D. (1984). Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on pattern analysis and machine intelligence*, (6):721–741.

Gerber, M. (2015). On integration methods based on scrambled nets of arbitrary size. *Journal of Complexity*, 31(6):798–816.

Gerber, M. and Bornn, L. (2017). Improving simulated annealing through derandomization. *Journal of Global Optimization*, 68(1):189–217.

Gerber, M. and Chopin, N. (2015). Sequential quasi Monte Carlo. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 77(3):509–579.

Gerber, M., Chopin, N., and Whiteley, N. (2017). Negative association, ordering and convergence of resampling methods. *arXiv preprint arXiv:1707.01845*.

Germain, P., Bach, F., Lacoste, A., and Lacoste-Julien, S. (2016). Pac-bayesian theory meets bayesian inference. In Lee, D. D., Sugiyama, M., Luxburg, U. V., Guyon, I., and Garnett, R., editors, *Advances in Neural Information Processing Systems 29*, pages 1884–1892. Curran Associates, Inc.

Geweke, J. (1989). Bayesian inference in econometric models using Monte Carlo integration. *Econometrica: Journal of the Econometric Society*, pages 1317–1339.

Girolami, M. and Calderhead, B. (2011). Riemann manifold Langevin and Hamiltonian Monte Carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 73(2):123–214.

Glasserman, P. (2013). *Monte Carlo methods in financial engineering*, volume 53. Springer Science & Business Media.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680.

Gorham, J. and Mackey, L. (2015). Measuring sample quality with Stein's method. In *Advances in Neural Information Processing Systems*, pages 226–234.

Gunawan, D., Kohn, R., Quiroz, M., Dang, K.-D., and Tran, M.-N. (2018). Subsampling Sequential Monte Carlo for Static Bayesian Models. *arXiv preprint arXiv:1805.03317*.

Gutmann, M. and Corander, J. (2016). Bayesian optimization for likelihood-free inference of simulator-based statistical models. *Journal of Machine Learning Research*, 17(125):1–47.

Hairer, E., Lubich, C., and Wanner, G. (2003). Geometric numerical integration illustrated by the Störmer–Verlet method. *Acta numerica*, 12:399–450.

Hairer, E., Lubich, C., and Wanner, G. (2006). *Geometric numerical integration: structure-preserving algorithms for ordinary differential equations*, volume 31. Springer Science & Business Media.

Halton, J. H. (1964). Algorithm 247: Radical-inverse quasi-random point sequence. *Communications of the ACM*, 7(12):701–702.

Hardy, G. H. (1905). On double *Fourier* series, and especially those which represent the double zeta-function with real and incommensurable parameters. *Quart. J.*, 37:53–79.

Hastings, W. K. (1970). Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1):97–109.

Heng, J. and Jacob, P. E. (2017). Unbiased Hamiltonian Monte Carlo with couplings. *arXiv preprint arXiv:1709.00404*.

Hickernell, F. J. (2006). *Koksma–Hlawka Inequality*. American Cancer Society.

Hickernell, F. J., Lemieux, C., Owen, A. B., et al. (2005). Control variates for quasi-monte carlo. *Statistical Science*, 20(1):1–31.

Hoffman, M. D., Blei, D. M., Wang, C., and Paisley, J. (2013). Stochastic variational inference. *The Journal of Machine Learning Research*, 14(1):1303–1347.

Hoffman, M. D. and Gelman, A. (2014). The No-U-turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research*, 15(1):1593–1623.

Huggins, J. H. and Roy, D. M. (2015). Sequential Monte Carlo as Approximate Sampling: bounds, adaptive resampling via ∞-ESS, and an application to Particle Gibbs. *arXiv preprint arXiv:1503.00966*.

Jaakkola, T. S. and Jordan, M. I. (2000). Bayesian parameter estimation via variational methods. *Statistics and Computing*, 10(1):25–37.

Jacob, P. E., O'Leary, J., and Atchadé, Y. F. (2017). Unbiased Markov chain Monte Carlo with couplings. *arXiv preprint arXiv:1708.03625*.

Jasra, A., Paulin, D., and Thiery, A. H. (2015). Error Bounds for Sequential Monte Carlo Samplers for Multimodal Distributions. *arXiv preprint arXiv:1509.08775*.

Jasra, A., Stephens, D. A., Doucet, A., and Tsagaris, T. (2011). Inference for Lévy-Driven Stochastic Volatility Models via Adaptive Sequential Monte Carlo. *Scandinavian Journal of Statistics*, 38(1):1–22.

Johnson, N. L., Kemp, A. W., and Kotz, S. (2005). *Univariate discrete distributions*, volume 444. John Wiley & Sons.

Johnson, R. and Zhang, T. (2013). Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in neural information processing systems*, pages 315–323.

Jordan, M. I., Ghahramani, Z., Jaakkola, T. S., and Saul, L. K. (1999). An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233.

Joy, C., Boyle, P. P., and Tan, K. S. (1996). Quasi-monte carlo methods in numerical finance. *Management Science*, 42(6):926–938.

Kiefer, J. and Wolfowitz, J. (1952). Stochastic estimation of the maximum of a regression function. *Ann. Math. Statist.*, 23(3):462–466.

Kingma, D. and Ba, J. (2015). Adam: A method for stochastic optimization. *Proceedings of the International Conference on Learning Representations*.

Kingma, D. P. and Welling, M. (2014). Auto-encoding variational bayes. *Proceedings of the International Conference on Learning Representations*.

Kong, A., Liu, J. S., and Wong, W. H. (1994). Sequential imputation and Bayesian missing data problems. *Journal of the American statistical association*, 89:278–288.

Kuipers, L. and Niederreiter, H. (2012). *Uniform distribution of sequences*. Courier Corporation.

L'Ecuyer, P. (2016). Randomized Quasi-Monte Carlo: An Introduction for Practitioners. In *12th International Conference on Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing (MCQMC 2016)*.

L'Ecuyer, P., Lécot, C., and Tuffin, B. (2008). A randomized quasi-monte carlo simulation method for markov chains. *Operations Research*, 56(4):958–975.

L'Ecuyer, P. and Lemieux, C. (2005). Recent advances in randomized quasi-Monte Carlo methods. In *Modeling uncertainty*, pages 419–474. Springer.

Lee, A. (2012). On the the choice of MCMC kernels for approximate Bayesian computation with SMC samplers. In *Simulation Conference (WSC), Proceedings of the 2012 Winter*, pages 1–12. IEEE.

Lee, A. and Łatuszyński, K. (2014). Variance bounding and geometric ergodicity of Markov chain Monte Carlo kernels for approximate Bayesian computation. *Biometrika*, 101(3):655–671.

Lee, A. and Whiteley, N. (2018). Variance estimation in the particle filter. *Biometrika*, page asy028.

Leimkuhler, B. and Matthews, C. (2016). *Molecular Dynamics*. Springer.

Lemieux, C. and L'Ecuyer, P. (2001). On the use of quasi-monte carlo methods in computational finance. In *International Conference on Computational Science*, pages 607–616. Springer.

Leobacher, G. and Pillichshammer, F. (2014). *Introduction to quasi-Monte Carlo integration and applications*. Springer.

Levy, D., Hoffman, M. D., and Sohl-Dickstein, J. (2018). Generalizing hamiltonian monte carlo with neural networks. In *International Conference on Learning Representations*.

Lintusaari, J., Gutmann, M. U., Dutta, R., Kaski, S., and Corander, J. (2017). Fundamentals and recent developments in approximate Bayesian computation. *Systematic biology*, 66(1):e66–e82.

L'Ecuyer, P. (2009). Quasi-monte carlo methods with applications in finance. *Finance and Stochastics*, 13(3):307–349.

L'Ecuyer, P. and Sanvido, C. (2010). Coupling from the past with randomized quasi-monte carlo. *Mathematics and Computers in Simulation*, 81(3):476 – 489. The Sixth IMACS Seminar on Monte Carlo Methods Applied Scientific Computing VII. Forward Numerical Grid Generation, Approximation and Simulation.

Mandt, S., Hoffman, M. D., and Blei, D. M. (2017). Stochastic Gradient Descent as Approximate Bayesian Inference. *Journal of Machine Learning Research*, 18(134):1–35.

Mangoubi, O., Pillai, N. S., and Smith, A. (2018). Does Hamiltonian Monte Carlo mix faster than a random walk on multimodal densities? *arXiv preprint arXiv:1808.03230*.

Mangoubi, O. and Smith, A. (2017). Rapid mixing of Hamiltonian Monte Carlo on strongly log-concave distributions. *arXiv preprint arXiv:1708.07114*.

Marin, J.-M., Pudlo, P., Robert, C., and Ryder, R. (2012). Approximate Bayesian computational methods. *Statistics and Computing*, 22(6):1167–1180.

Marin, J.-M., Raynal, L., Pudlo, P., Ribatet, M., and Robert, C. P. (2016). ABC random forests for Bayesian parameter inference. *arXiv preprint arXiv:1605.05537*.

Marjoram, P., Molitor, J., Plagnol, V., and Tavare, S. (2003). Markov chain Monte Carlo without likelihoods. *Proceedings of the National Academy of Sciences of the United States of America*, 100(26):15324–8.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. (1953). Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6):1087–1092.

Metropolis, N. and Ulam, S. (1949). The monte carlo method. *Journal of the American Statistical Association*, 44(247):335–341.

Miller, A. C., Foti, N., D'Amour, A., and Adams, R. P. (2017). Reducing reparameterization gradient variance. In *Advances in Neural Information Processing Systems*.

Minka, T. P. (2001). Expectation propagation for approximate Bayesian inference. In *Proceedings of the Seventeenth conference on Uncertainty in artificial intelligence*, pages 362–369. Morgan Kaufmann Publishers Inc.

Mohamed, S., de Freitas, N., and Wang, Z. (2013). Adaptive Hamiltonian and Riemann manifold Monte Carlo samplers. *arXiv preprint arXiv:1302.6182*.

Moulines, E. and Bach, F. R. (2011). Non-asymptotic analysis of stochastic approximation algorithms for machine learning. In *Advances in Neural Information Processing Systems*.

Murray, L. M., Lee, A., and Jacob, P. E. (2016). Parallel resampling in the particle filter. *Journal of Computational and Graphical Statistics*, 25(3):789–805.

Neal, R. M. (1993). Bayesian learning via stochastic dynamics. In *Advances in neural information processing systems*, pages 475–482.

Neal, R. M. (2001). Annealed importance sampling. *Statistics and computing*, 11(2):125–139.

Neal, R. M. (2011). MCMC using Hamiltonian dynamics. *Handbook of Markov Chain Monte Carlo*, 2(11).

Nesterov, Y. (2013). *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media.

Nesterov, Y. E. (1983). A method for solving the convex programming problem with convergence rate $o(1/k^2)$. *Dokl. Akad. Nauk SSSR*, 269:543–547.

Niederreiter, H. (1978). Quasi-Monte Carlo methods and pseudo-random numbers. *Bulletin of the American Mathematical Society*, 84(6):957–1041.

Niederreiter, H. (1992). *Random number generation and quasi-Monte Carlo methods*. SIAM.

Oates, C. and Girolami, M. (2016). Control functionals for quasi-monte carlo integration. In *Artificial Intelligence and Statistics*, pages 56–65.

Oates, C. J., Girolami, M., and Chopin, N. (2017). Control functionals for Monte Carlo integration. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 79(3):695–718.

Oh, M.-S. and Berger, J. O. (1992). Adaptive importance sampling in monte carlo integration. *Journal of Statistical Computation and Simulation*, 41(3-4):143–168.

Ökten, G., Tuffin, B., and Burago, V. (2006). A central limit theorem and improved error bounds for a hybrid-Monte Carlo sequence with applications in computational finance. *Journal of Complexity*, 22(4):435–458.

Owen, A. (1998). Monte Carlo extension of quasi-Monte Carlo. *1998 Winter Simulation Conference. Proceedings (Cat. No.98CH36274)*, 1(1):571–577.

Owen, A. B. (1997). Scrambled net variance for integrals of smooth functions. *The Annals of Statistics*, 25(4):1541–1562.

Owen, A. B. (2008). Local antithetic sampling with scrambled nets. *The Annals of Statistics*, 36(5):2319–2343.

Owen, A. B. and Tribble, S. D. (2005). A quasi-monte carlo metropolis algorithm. *Proceedings of the National Academy of Sciences*, 102(25):8844–8849.

Paisley, J., Blei, D., and Jordan, M. (2012). Variational Bayesian inference with stochastic search. *International Conference on Machine Learning*.

Pakman, A., Gilboa, D., Carlson, D., and Paninski, L. (2017). Stochastic bouncy particle sampler. In *International Conference on Machine Learning*, pages 2741–2750.

Papamakarios, G. and Murray, I. (2016). Fast ε-free inference of simulation models with Bayesian conditional density estimation. In *Advances in Neural Information Processing Systems*, pages 1028–1036.

Pasarica, C. and Gelman, A. (2010). Adaptively scaling the Metropolis algorithm using expected squared jumped distance. *Statistica Sinica*, pages 343–364.

Pitt, M. K. and Shephard, N. (1999). Filtering via simulation: Auxiliary particle filters. *Journal of the American statistical association*, 94(446):590–599.

Price, L. F., Drovandi, C. C., Lee, A., and Nott, D. J. (2018). Bayesian synthetic likelihood. *Journal of Computational and Graphical Statistics*, 27(1):1–11.

Pudlo, P., Marin, J.-M., Estoup, A., Cornuet, J.-M., Gautier, M., and Robert, C. P. (2016). Reliable abc model choice via random forests. *Bioinformatics*, 32(6):859–866.

Ranganath, R., Gerrish, S., and Blei, D. M. (2014). Black box variational inference. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*.

Rezende, D. J., Mohamed, S., and Wierstra, D. (2014). Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of the International Conference on Machine Learning*.

Ridgway, J. (2016). Computation of Gaussian orthant probabilities in high dimension. *Statistics and computing*, 26(4):899–916.

Robbins, H. and Monro, S. (1951). A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407.

Robert, C. (2007). *The Bayesian choice: from decision-theoretic foundations to computational implementation*. Springer Science & Business Media.

Robert, C. and Casella, G. (2013). *Monte Carlo Statistical Methods*. Springer Science & Business Media.

Roberts, G. O., Gelman, A., and Gilks, W. R. (1997). Weak convergence and optimal scaling of random walk Metropolis algorithms. *The annals of applied probability*, 7(1):110–120.

Roberts, G. O. and Rosenthal, J. S. (1998). Optimal scaling of discrete approximations to Langevin diffusions. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 60(1):255–268.

Roberts, G. O. and Rosenthal, J. S. (2001). Optimal scaling for various metropolis-hastings algorithms. *Statist. Sci.*, 16(4):351–367.

Roberts, G. O. and Tweedie, R. L. (1996). Exponential convergence of langevin distributions and their discrete approximations. *Bernoulli*, 2(4):341–363.

Roeder, G., Wu, Y., and Duvenaud, D. K. (2017). Sticking the landing: Simple, lower-variance gradient estimators for variational inference. In *Advances in Neural Information Processing Systems*.

Rosenblatt, M. (1952). Remarks on a multivariate transformation. *Ann. Math. Statist.*, 23(3):470–472.

Ruiz, F. J. R., Titsias, M. K., and Blei, D. M. (2016a). Overdispersed black-box variational inference. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*.

Ruiz, F. R., Titsias, M., and Blei, D. (2016b). The generalized reparameterization gradient. In *Advances in Neural Information Processing Systems*.

Ryder, T., Golightly, A., McGough, A. S., and Prangle, D. (2018). Black-box variational inference for stochastic differential equations. *arXiv preprint arXiv:1802.03335*.

Salomone, R., South, L. F., Drovandi, C. C., and Kroese, D. P. (2018). Unbiased and consistent nested sampling via sequential Monte Carlo. *arXiv preprint arXiv:1805.03924*.

Schäfer, C. and Chopin, N. (2013). Sequential Monte Carlo on large binary sampling spaces. *Statistics and Computing*, pages 1–22.

Schwedes, T. and Calderhead, B. (2018). Quasi markov chain monte carlo methods. *arXiv preprint arXiv:1807.00070*.

Schweizer, N. (2012a). Non-asymptotic error bounds for sequential MCMC and stability of Feynman-Kac propagators. *arXiv preprint arXiv:1204.2382*.

Schweizer, N. (2012b). Non-asymptotic error bounds for sequential MCMC methods in multimodal settings. *arXiv preprint arXiv:1205.6733*.

Sedki, M., Pudlo, P., Marin, J.-M., Robert, C. P., and Cornuet, J.-M. (2012). Efficient learning in ABC algorithms. *arXiv preprint 1210.1388*.

Sherlock, C. and Thiery, A. H. (2017). A discrete bouncy particle sampler. *arXiv preprint arXiv:1707.05200*.

Sisson, S. A., Fan, Y., Tanaka, M. M., Rogers, A., Huang, Y., Njegic, B., Wayne, L., Gordon, M. S., Dabdub, D., Gerber, R. B., and Finlayson-pitts, B. J. (2009). Correction for Sisson et al., Sequential Monte Carlo without likelihoods. *Proceedings of the National Academy of Sciences*, 106(39):16889–16889.

Snoek, J., Larochelle, H., and Adams, R. P. (2012). Practical Bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems*, pages 2951–2959.

Sutton, R. S. and Barto, A. G. (1998). *Reinforcement learning: An introduction*. MIT press Cambridge.

Tanaka, M. M., Francis, A. R., Luciani, F., and Sisson, S. A. (2006). Using approximate Bayesian computation to estimate tuberculosis transmission parameters from genotype data. *Genetics*, 173(3):1511–1520.

Tavaré, S., Balding, D. J., Griffiths, R. C., and Donnelly, P. (1997). Inferring coalescence times from DNA sequence data. *Genetics*, 145(2):505–518.

Tierney, L. (1994). Markov chains for exploring posterior distributions. *Ann. Statist.*, 22(4):1701–1728.

Toni, T., Welch, D., Strelkowa, N., Ipsen, A., and Stumpf, M. P. (2009). Approximate Bayesian computation scheme for parameter inference and model selection in dynamical systems. *Journal of The Royal Society Interface*, 6(31):187–202.

Tran, D., Kucukelbir, A., Dieng, A. B., Rudolph, M., Liang, D., and Blei, D. M. (2016). Edward: A library for probabilistic modeling, inference, and criticism. *arXiv preprint arXiv:1610.09787*.

Tran, D., Ranganath, R., and Blei, D. (2017a). Hierarchical implicit models and likelihood-free variational inference. In *Advances in Neural Information Processing Systems*, pages 5523–5533.

Tran, M.-N., Nott, D. J., and Kohn, R. (2017b). Variational bayes with intractable likelihood. *Journal of Computational and Graphical Statistics*, 26(4):873–882.

Vanetti, P., Bouchard-Côté, A., Deligiannidis, G., and Doucet, A. (2017). Piecewise Deterministic Markov Chain Monte Carlo. *arXiv preprint arXiv:1707.05296*.

Vats, D., Flegal, J. M., and Jones, G. L. (2015). Multivariate output analysis for Markov chain Monte Carlo. *arXiv preprint arXiv:1512.07713*.

Wang, Y. and Blei, D. M. (2018). Frequentist consistency of variational bayes. *Journal of the American Statistical Association*, (just-accepted):1–85.

Wang, Z., Mohamed, S., and Freitas, N. (2013). Adaptive hamiltonian and riemann manifold monte carlo. In *International Conference on Machine Learning*, pages 1462–1470.

Welling, M. and Teh, Y. W. (2011). Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 681–688.

Wibisono, A., Wilson, A. C., and Jordan, M. I. (2016). A variational perspective on accelerated methods in optimization. *Proceedings of the National Academy of Sciences*, 113(47):E7351–E7358.

Wilkinson, R. D. (2014). Accelerating ABC methods using Gaussian processes. *Proceedings of the 17 th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 33:1015–1023.

Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256.

Yang, J., Sindhwani, V., Avron, H., and Mahoney, M. (2014). Quasi-monte carlo feature maps for shift-invariant kernels. In *International Conference on Machine Learning*, pages 485–493.

Zhang, C., Butepage, J., Kjellstrom, H., and Mandt, S. (2017a). Advances in variational inference. *arXiv preprint arXiv:1711.05597*.

Zhang, C., Kjellström, H., and Mandt, S. (2017b). Determinantal point processes for mini-batch diversification. In *Uncertainty in Artificial Intelligence, UAI 2017*.

Zhou, Y., Johansen, A. M., and Aston, J. A. (2016). Toward Automatic Model Comparison: An Adaptive Sequential Monte Carlo Approach. *Journal of Computational and Graphical Statistics*, 25(3):701–726.

université PARIS-SACLAY | École doctorale de mathématiques Hadamard (EDMH)

Titre : Computation bayésienne en grande dimension

Mots Clefs : Monte Carlo séquentiel, Statistique bayésienne, Quasi-Monte Carlo, Inférence variationelle, Calcul bayésien approximatif

Résumé :
La statistique bayésienne computationnelle construit des approximations de la distribution *a posteriori* soit par échantillonnage, soit en construisant des approximations tractables. La contribution de cette thèse au domaine des stastiques bayésiennes est le développement de nouvelles méthodologies combinant des méthodes existantes. Nos approches sont mieux adaptées à la dimension ou entraînent une réduction du coût de calcul par rapport aux méthodes existantes. Notre première contribution améliore le calcul bayésien approximatif (ABC) en utilisant le quasi-Monte Carlo (QMC). ABC permet l'inférence bayésienne dans les modèles avec une vraisemblance intractable. QMC est une technique de réduction de variance qui fournit des estimateurs plus précis d'intégrales. Notre deuxième contribution utilise le QMC pour l'inférence variationnelle (VI). VI est une méthode pour construire des approximations tractables de la distribution *a posteriori*. La troisième contribution développe une approche pour adapter les échantillonneurs Monte Carlo séquentiel (SMC) lorsqu'on utilise des noyaux de mutation Hamiltonian Monte Carlo (HMC). Les échantillonneurs SMC permettent une estimation non biaisée de l'evidence du modèle, mais ils ont tendance à perdre en performance lorsque la dimension croît. HMC est une technique de Monte Carlo par chaîne de Markov qui présente des propriétés intéressantes lorsque la dimension de l'espace cible augmente mais elle est difficile à adapter. En combinant les deux, nous construisons un échantillonneur qui tire avantage des deux.

Title : High dimensional Bayesian computation

Keys words : Sequential Monte Carlo, Bayesian statistics, Quasi-Monte Carlo, Variational inference, Approximiate Bayesian computation

Abstract :
Computational Bayesian statistics builds approximations of the posterior distribution either by sampling or by constructing tractable approximations. The contribution of this thesis to the field of Bayesian stastics is the development of new methodology by combining existing methods. Our approaches either scale better with the dimension or result in reduced computational cost compared to existing methods. Our first contribution improves approximate Bayesian computation (ABC) by using quasi-Monte Carlo (QMC). ABC allows Bayesian inference in models with intractable likelihoods. QMC is a variance reduction technique that yields precise estimations of integrals. Our second contribution takes advantage of QMC for Variational Inference (VI). VI is a method for constructing tractable approximations to the posterior distribution. The third contribution develops an approach for tuning Sequential Monte Carlo (SMC) samplers when using Hamiltonian Monte Carlo (HMC) mutation kernels. SMC samplers allow the unbiased estimation of the model evidence but tend to struggle with increasing dimension. HMC is a Markov chain Monte Carlo technique that has appealing properties when the dimension of the target space increases but is difficult to tune. By combining the two we construct a sampler that takes advantage of the two.