Acceleration Strategies of Markov Chain Monte Carlo for

Chang-Ye Wu

▶ **To cite this version:**

Chang-Ye Wu. Acceleration Strategies of Markov Chain Monte Carlo for Bayesian Computation. Probability [math.PR]. Université Paris sciences et lettres, 2018. English. NNT : 2018PSLED019 . tel-01943788

HAL Id: tel-01943788

https://theses.hal.science/tel-01943788

Submitted on 4 Dec 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

THÈSE DE DOCTORAT

de l'Université de recherche Paris Sciences et Lettres
PSL Research University

Préparée à l'Université Paris-Dauphine

Acceleration Strategies of Markov Chain Monte Carlo for Bayesian Computation

École doctorale n°543

ÉCOLE DOCTORALE DE DAUPHINE

Spécialité SCIENCES

COMPOSITION DU JURY :

M. Christian P. ROBERT
CEREMADE
Directeur de thèse

Mme. Kerrie MENGERSEN
Queensland University of Technology
Présidente du jury

M. Nicolas CHOPIN
ENSAE-CREST
Rapporteur

M. Arnaud DOUCET
University of Oxford
Rapporteur

M. Julien STOEHR
CEREMADE
Membre du jury

Soutenue par **Chang-Ye Wu**
le 4 octobre 2018

Dirigée par **Christian P. Robert**

DAUPHINE UNIVERSITÉ PARIS | PSL UNIVERSITÉ PARIS

Acceleration Strategies of Markov Chain Monte Carlo for Bayesian Computation

Chang-Ye Wu

Supervisor: Prof. Christian P. Robert

CEREMADE, CNRS, UMR 7534,

Université Paris Dauphine, PSL Research University, Paris, France

This dissertation is submitted for the degree of

Doctor of Philosophy

October 2018

Contents

Résumé

Dans l'analyse statistique, le statisticien interprète les observations, $(X_{1:n}) \subset \mathcal{X}$, avec un modèle qui appartient à une classe de distributions de probabilité $\mathcal{P} = \{P_\theta, \theta \in \Theta\}$ sur l'espace d'échantillonnage $(\mathcal{X}, \mathcal{A})$, où θ est appelé le paramètre du modèle et Θ est un ensemble arbitraire. Dans cette thèse, nous nous intéressons aux cas paramétriques - c'est-à-dire $\Theta \subset \mathbb{R}^d$, et supposons les distributions sont dominées par une mesure $\mu(dx)$. Dans les statistiques fréquentistes et les statistiques Bayésiennes, la probabilité joue un rôle crucial dans les inférences, ce qui décrit la plausibilité des valeurs des paramétres, compte tenu des données observées.

Dans les statistiques fréquentistes, il existe un vrai paramètre fixe θ_0 tel que P_{θ_0} peut correspondre aux observations et l'objectif principal de l'analyse statistique est de construire un estimateur, $\hat{\theta}_n$, du θ_0 tel que $\hat{\theta}_n$ converge vers θ_0 (en probabilité ou en distribution), comme $n \to \infty$. Cependant, les statistiques Bayésiennes considèrent le paramètre θ comme une variable aléatoire et le modèle θ avec une distribution à priori qui décrit nos croyances sur le paramètre et est indépendante des données. Par conséquent, l'espace des paramètres est équipé d'une structure de probabilité $(\Theta, \mathcal{B}, \pi_0)$ et l'analyse Bayésienne extrait les informations sur θ en combinant les croyances à priori, π_0, et l'information contenue dans l'ensemble de données observé. Dans le paradigme Bayésien, une fois donné l'a priori et la vraisemblance, l'information sur le paramètre est modélisée par la distribution a posteriori, définie comme suit.

Definition 0.0.1 *La distribution a posteriori est la distribution de probabilité du paramètre θ, étant donné les observations $(X_{1:n})$, sur l'espace des paramètres $(\Theta, \mathcal{B})$. Selon le théorème de Bayes, il a la forme suivante,*

$$\pi(d\theta|X_{1:n}) = \frac{\mathcal{L}(X_{1:n}|\theta)\pi_0(d\theta)}{\int_\Theta \mathcal{L}(X_{1:n}|\theta')\pi_0(d\theta')}$$

L'intégrale, $\int_\Theta \mathcal{L}(X_{1:n}|\theta')\pi_0(d\theta')$, dans le dénominateur est appelée l'evidence, ou le marginal probabilité et est noté $m_{\pi_0}(X_{1:n})$.

Comparées aux statistiques fréquentistes, les méthodes Bayésiennes utilisent explicitement les distributions de probabilité pour quantifier les incertitudes sur les quantités inconnues. Toutes les inférences dans les méthodes Bayésiennes sont exactes et non approximatives et les résultats sont interprétables afin que nous puissions facilement répondre directement à toute question scientifique. Cependant, en pratique, des problèmes surviennent si la distribution a posteriori n'a pas de forme fermée. Par exemple, l'evidence peut ne pas être explicitement disponible. Au début des statistiques Bayésiennes, l'analyse Bayésienne se limite aux problèmes où les distributions a posteriori sont explicitement disponibles, tels que les a priori conjugués.

Definition 0.0.2 *Si la distribution a posteriori $\pi(\theta|X_{1:n})$ est dans la même famille de distribution de probabilité de π_0, les lois a priori et a posteriori sont appelées distributions conjuguées, et l'a priori est appelé conjugué avant pour la fonction de vraisemblance $\mathcal{L}(X_{1:n}|\theta)$.*

Même si la conjugaison est certainement utile dans de nombreuses applications, elle est trop restrictive et n'est pas une solution universelle pour les problèmes généraux.

Les approches pour surmonter cette restriction peuvent être classées en deux groupes principaux: les méthodes d'approximation et les méthodes de Monte Carlo. Les méthodes d'approximation, telles que la propagation de l'espérance et Bayes variationnelles, projettent la distribution a posteriori d'intérêt dans une famille de probabilité traitable et l'approchent avec une famille proche. À moins que la distribution a posteriori ne soit dans la famille choisie, il existe un écart intrinsèque entre l'approximation et le postérieur d'intérêt. Les méthodes de Monte Carlo, qui sont également au centre de cette thése, ont un comportement différent et peuvent construire des estimateurs pour approcher les distributions ou les quantités d'intérêt dans toute précision désirée lorsque l'effort de calcul croît èa l'infini.

Monte Carlo est basé sur la loi des grands nombres (LLN) pour approximer les intégrales d'intérêt. Supposons que nous sommes intéressés par le calcul des intégrales de la forme

$$I_h := \mathbb{E}_P(h(X))$$

Le LLN dit que si $X_1, X_2, \cdots$, est une suite infinie de variables aléatoires indépendantes et identiquement distribuées (i.i.d.) selon la distribution de probabilité P, et si I_h existe, alors

$$\frac{1}{N}\sum_{i=1}^{N} h(X_i) \xrightarrow[N\to\infty]{P} I_h$$

Basé sur une hypothèse supplémentaire que $\sigma^2 := \mathbb{E}_P(h^2(X)) - I_h^2 < \infty$, le théorème central limite (CLT) donne un résultat plus fort,

$$\sqrt{N}\left(\sum_{i=1}^{N} h(X_i) - I_h\right) \xrightarrow[N\to\infty]{\mathcal{L}} \mathcal{N}(0, \sigma^2)$$

Par le CLT, l'estimateur de Monte Carlo converge vers I_h au taux $\mathcal{O}(N^{-1/2})$, indépendamment de la dimensionnalité de X_i. Nous présentons brièvement trois méthodes pour produire i.i.d. échantillons: la transformation inverse, le accepter et rejeter l'échantillonnage et l'échantillonnage d'importance.

Afin d'échantillonner à partir de distributions cibles plus complexes, nous décrivons une classe d'algorithmes d'échantillonnage - MCMC, basée sur des chaînes de Markov, qui produisent des échantillons corrélés pour approcher la distribution cible ou les intégrales d'intérêt. Les algorithmes Monte Carlo à chaîne de Markov (MCMC) sont utilisés depuis près de 60 ans, devenir une méthode de référence pour l'analyse des modèles complexes bayésiens au début 1990 (Gelfand and Smith, 1990). La force de cette méthode est qu'elle garantit la convergence vers la quantité (ou les quantités) d'intérêt avec un minimum exigences sur la distribution ciblée (également appelée *target*) derriére de telles quantités. En ce sens, MCMC les algorithmes sont robustes ou universels, contrairement aux méthodes de Monte Carlo les plus standards (see, e.g., Robert and Casella, 2004; Rubinstein, 1981) qui nécessitent simulations de la distribution cible.

Les méthodes MCMC ont un historique (see, e.g. Cappé and Robert, 2000) commence à peu près en même temps que les méthodes de Monte Carlo, en en conjonction avec la conception des premiers ordinateurs. Ils ont été conçus pour gérer la simulation de distributions cibles complexes, lorsque la complexité découle à partir

de la forme de la densité cible, de la taille des données associées, dimension de l'objet à simuler, ou à partir des exigences de temps. Par exemple, la densité cible $p(x)$ peut être exprimée en termes de des intégrales multiples qui ne peuvent être résolues analytiquement,

$$p(x) = \int \omega(x, \xi) \mathrm{d}\xi \,,$$

ce qui nécessite la simulation du vecteur entier (x, ξ). Dans les cas quand ξ est de la même dimension que les données, comme par exemple dans latent modèles variables, cette augmentation significative de la dimension de l'objet à simulé crée des difficultés de calcul pour les méthodes Monte Carlo standard, de gérer la nouvelle cible $\omega(x, \xi)$, à concevoir un nouveau et algorithme de simulation efficace. Un algorithme de chaîne de Markov Monte Carlo (MCMC) permet une résolution alternative de ce défi de calcul par simuler une chaîne de Markov qui explore l'espace d'intérét (et éventuellement espaces supplémentaires de variables auxiliaires) sans nécessiter de profondeur connaissances préliminaires sur la densité p, en plus de la capacité de calculer $p(x_0)$ pour une valeur de paramètre donnée x_0 (si jusqu'à une normalisation constante) et éventuellement le gradient $\nabla \log p(x_0)$.

La validation de la méthode (e.g., Robert and Casella, 2004) est que la chaîne de Markov est *ergodic* (e.g., Meyn and Tweedie, 1993), à savoir qu'elle converge vers distribution à la distribution avec densité π, peu importe où le Markov la chaîne est démarrée à l'instant $t = 0$.

L'algorithme de Metropolis - Hastings est une illustration générique du principe de MCMC. le algorithme de base est construit en choisissant une *proposition*, c'est-à-dire un densité $q(x'|x)$ (aussi connu sous le nom de *Markov kernel*), la chaîne de Markov $\{X_n\}_{n=1}^{\infty}$ étant ensuite dérivée par des simulations successives du transition

$$X_{n+1} = \begin{cases} X' \sim q(X'|X_n) & \text{w.p.} \left\{ \dfrac{p(X')}{p(X_n)} \times \dfrac{q(X_n|X')}{q(X'|X_n)} \right\} \wedge 1, \\ X_n & \text{sinon.} \end{cases}$$

Cette caractéristique d'acceptation-rejet de l'algorithme le rend approprié pour ciblant p comme sa distribution stationnaire si la chaîne de Markov qui en résulte $(X_n)_n$ est irréductible, c'est-à-dire qu'il a une probabilité positive de visiter région du support de p dans un nombre fini d'itérations. (Stationnarité peut facilement être montré, par exemple, en utilisant le soidisant *balance détaillée property* qui rend la chaîne réversible dans le temps, see, e.g., Robert and Casella, 2004.)

Hamiltonien (ou hybride) Monte Carlo (HMC) est une technique de variable auxiliaire qui tire parti d'une processus de Markov dans le temps pour échantillonner à partir de la cible p, qui a démontré son efficacité élevée dans l'échantillonnage à partir de distributions complexes de grande dimension. Cette approche vient de physique (Duane et al., 1987a) et a été popularisé dans les statistiques par Neal (1999, 2011) et MacKay (2002). Grâce à la dynamique Hamiltonienne, les propositions en HMC peuvent être loin de l'état actuel et être accepté avec une forte probabilité, ce qui peut réduire la marche aléatoire dans MCMC pour améliorer l'efficacité.

L'explosion de la collecte et de l'analyse des grands ensembles de données au cours des dernières années a apporté de nouveaux défis aux algorithmes MCMC qui sont utilisés pour l'inférence Bayésienne. En examinant si un nouvel échantillon proposé

est accepté ou non à l'étape d'acceptation-rejet, un algorithme MCMC tel que la version de Metropolis-Hastings doit balayer l'ensemble des données, à chaque et chaque itération, pour l'évaluation de la fonction de vraisemblance. Les algorithmes MCMC sont alors difficiles à intensifier, ce qui gêne fortement leur application dans les grands paramètres de données. Dans certains cas, les jeux de données peuvent être trop volumineux pour s'adapter à une seule machine. Il se peut aussi que des mesures de confidentialité imposent différentes bases de données pour se tenir sur des réseaux distincts, avec la possible ajouté fardeau des données cryptées (Aslett et al., 2015). la communication entre les machines séparées peut s'avérer impossible sur une échelle MCMC implique des milliers ou des centaines de milliers d'itérations. Ces dernières années, des efforts ont été faits pour concevoir des algorithmes *scalable*, à savoir des solutions capables de gérer des cibles à grande échelle en décomposant le problème en éléments maniables ou évolutifs. En général, ces méthodes peuvent être classées en deux catégories (Bardenet et al., 2015): approches de Diviser pour régner et approches de sous-échantillonnage. Nous donnons un bref aperçu des algorithmes MCMC évolutifs dans la section 1.3.3..

Tous les échantillonneurs MCMC ci-dessus sont basés sur des chaînes de Markov réversibles à temps discret, cependant, les échantillonneurs MCMC non réversibles à temps continu ont attiré l'attention des statisticiens computationnels, qui sont basés sur des processus de Markov déterministes par morceaux (PDMP). Même si PDMP a été proposé dès 1984 par Davis (1984), sa prédominance dans les statistiques pour les problèmes d'échantillonnage a commencé les applications remarquables par (Bierkens et al., 2016; Bouchard-Côté et al., 2017; Peters et al., 2012). Nous terminons le Chapitre 1 avec une introduction aux échantillonneurs basés sur PDMP.

Chapitre 2

Une technique d'échantillonnage puissante, la méthode Monte Carlo de chaîne de Markov (MCMC), (voir, par exemple, Christian and Casella (2004)) a été largement exploitée dans les statistiques informatiques pour devenir un outil standard dans l'inférence Bayésienne, où les distributions a posteriori sont souvent intraitables analytiquement et au mieux connu jusqu'à une constante de normalisation. Cependant, presque tous les algorithmes MCMC existants, tels que l'algorithme de Metropolis-Hastings (MH), Hamiltonien Monte Carlo (HMC) (Neal et al. (2011)) et l'algorithme de Langevin ajusté par Metropolis (MALA), satisfont des conditions détaillées d'équilibre, remontant à Metropolis et al. (1953) et Hastings (1970). Récemment, un type différent de méthode MCMC - le processus de Markov déterministe par morceaux (PDMP) - a été introduit dans les statistiques computationnelles, comme une approche qui est généralement irréversible et ne satisfait donc pas l'équilibre détaillé. La théorie de base de PDMP a été développée par Davis (1984) et Davis (1993), tandis qu'une application à la statistique computationnelle a d'abord été implémentée par Peters et al. (2012), Bierkens et al. (2016), et Bouchard-Côté et al. (2017).

Dans ce chapitre, nous proposons l'échantillonneur de coordonnées (CS), un nouvel échantillonneur MCMC basé sur PDMP qui est une variante de l'échantillonneur Zigzag (ZS) de Bierkens et al. (2016). Cependant, il diffère dans trois aspects importants. Premièrement, l'ensemble de vélocités utilisé dans l'échantillonneur de coordonnées est constitué d'une base orthonormale de l'espace euclidien $\mathbb{R}^d$, tandis

que celui de l'échantillonneur de Zigzag est limité à $\{-1, 1\}^d$, où d indique la dimension de la distribution cible. Deuxièmement, la fonction de taux d'événements dans l'échantillonneur Zigzag est beaucoup plus grande que celle de l'échantillonneur à coordonnées, en particulier pour les distributions de grande dimension. Cela signifie que les événements se produisent plus fréquemment dans l'échantillonneur Zigzag, ce qui réduit son efficacité par rapport à notre approche. Troisièmement, l'échantillonneur de coordonnées ne change qu'un seul composant à la fois lors de l'exploration de l'espace cible, et il conserve les autres composants inchangés, tandis que l'échantillonneur Zigzag modifie tous les composants en même temps.

Travail connexe: Comme les processus de Markov déterministes par échantillonnage pour les distributions ont été introduits par Peters et al. (2012), les algorithmes MCMC basés sur PDMP, à temps continu, non réversibles sont devenus des outils pertinents, de la probabilité appliquée (Bierkens et al., 2017b; Fontbona et al., 2016) à la physique (Harland et al., 2017; Michel et al., 2014; Peters et al., 2012), aux statistiques (Bierkens et al., 2017a, 2016; Bouchard-Côté et al., 2017; Fearnhead et al., 2016; Michel and Sénécal, 2017; Pakman et al., 2016; Vanetti et al., 2017). Cependant, presque tous les échantillonneurs MCMC basés sur PDMP sont basés sur deux versions originales: le BPS (Bouncy Particle Sampler) de Bouchard-Côté et al. (2017) et le Zigzag Sampler de Bierkens et al. (2016). Bouchard-Côté et al. (2017) montre que BPS peut fournir des performances de pointe comparées à la console HMC de référence pour les distributions de grande dimension, alors que Bierkens et al. (2016) montre que l'échantillonneur basé sur PDMP est plus facile à mettre à l'échelle bias, tandis que Bierkens et al. (2017a) considère l'application de PDMP pour les distributions sur des domaines restreints. Fearnhead et al. (2016) unifie l'échantillonneur BPS et Zigzag dans le cadre de PDMP et choisit la vitesse du processus, aux temps de l'événement, sur la sphère unité, basée sur le produit interne entre cette vitesse et le gradient de la fonction potentielle. (Cette perspective concerne la dynamique de transition utilisée dans notre article.) Pour surmonter la principale difficulté des échantillonneurs basés sur PDMP, qui est la simulation du processus de Poisson inhomogène dans le temps, Sherlock and Thiery (2017) et Vanetti et al. (2017) ont recours à un discrétisation de ces échantillonneurs à temps continu. En outre, le préconditionnement de l'ensemble de vitesses est montré pour accélérer les algorithmes, comme indiqué par Pakman et al. (2016).

Nous supposons que π est la distribution cible continue sur $\mathbb{R}^d$ et par commodité, nous utilisons aussi $\pi(\mathbf{x})$ pour la fonction de densité de probabilité de π , quand $\mathbf{x} \in \mathbb{R}^d$. Nous définissons $U(\mathbf{x})$ comme la fonction potentielle de $\pi(\mathbf{x})$, c'est-à-dire $\pi(\mathbf{x}) \propto \exp\{-U(\mathbf{x})\}$, avec U positif. Dans le framework PDMP, une variable auxiliaire, $\mathbf{V} \in \mathcal{V}$, est introduite et un sampler basé sur PDMP explore l'espace d'état augmenté $\mathbb{R}^d \times \mathcal{V}$, ciblant une variable $\mathbf{Z} = (\mathbf{X}, \mathbf{V})$ avec la distribution $\rho(d\mathbf{x}, d\mathbf{v})$ sur $\mathbb{R}^d \times \mathcal{V}$ comme sa distribution invariante. Par construction, la distribution ρ apprécie π comme sa distribution marginale dans $\mathbf{x}$. En pratique, les échantillonneurs basés sur PDMP choisissent $\mathcal{V}$ pour être l'espace euclidien $\mathbb{R}^d$, la sphère $\mathbb{S}^{d-1}$, ou la ensemble discret $\{\mathbf{v} = (v_1, \cdots, v_d) | v_i \in \{-1, 1\}, i = 1, \cdots, d\}$. Un processus de Markov déterministe par morceaux $\mathbf{Z}_t = (\mathbf{X}_t, \mathbf{V}_t)$ se compose de trois composants distincts: sa dynamique déterministe entre les événements, un taux d'occurrence d'événement et une dynamique de transition à l'heure de l'événement. Plus précisément,

1. **Dynamique déterministe**: entre deux événements, le processus de Markov

évolue de façon déterministe, selon une équation différentielle ordinaire:

$$\frac{d\mathbf{Z}_t}{dt} = \Psi(\mathbf{Z}_t).$$

2. **Taux d'occurrence d'événement**: un événement se produit à l'instant t avec le taux $\lambda(\mathbf{Z}_t)$.

3. **Dynamique de transition**: Lors d'un événement, τ, l'état précédant τ est noté $\mathbf{Z}_{\tau-}$, le nouvel état étant généré par $\mathbf{Z}_\tau \sim Q(\cdot|\mathbf{Z}_{\tau-})$

Ici, un "événement" se réfère à une occurrence d'un processus de Poisson inhomogène dans le temps avec le taux $\lambda(\cdot)$ (Kingman, 1992). Suivant (Davis, 1993, Theorem 26.14), ce processus de Markov avait un générateur étendu égal à

$$\mathcal{L}f(\mathbf{z}) = \nabla f(\mathbf{z}) \cdot \Psi(\mathbf{z}) + \lambda(\mathbf{z}) \int_{\mathbf{z}'} [f(\mathbf{z}') - f(\mathbf{z})]\, Q(d\mathbf{z}'|\mathbf{z}) \tag{1}$$

Afin de garantir l'invariance par rapport à $\rho(d\mathbf{z})$, le générateur étendu doit satisfaire $\int \mathcal{L}f(\mathbf{z})\rho(d\mathbf{z}) = 0$ pour tout f dans une classe de fonction appropriée sur $\mathbb{R}^d \times \mathcal{V}$ (Davis, 1993, Theorem 34.7).

En pratique, le choix d'une dynamique déterministe appropriée, d'un taux d'événement et d'une dynamique de transition produit une chaîne de Markov avec une distribution invariante unique $\rho(d\mathbf{z})$. Comme pour le MCMC régulier, générer une telle chaîne de Markov suffisamment longue pour T conduit à un estimateur, $\frac{1}{T}\int_{t=0}^{T} h(\mathbf{X}_t)dt$, convergeant vers l'intégrale d'intérêt, $I = \int h(\mathbf{x})\pi(d\mathbf{x})$, par la loi des grands nombres pour Markov processus (Glynn and Haas, 2006), sous des hypothèses appropriées. Plus précisement,

$$\frac{1}{T}\int_{t=0}^{T} g(\mathbf{Z}_t)dt \longrightarrow \int g(\mathbf{z})\rho(d\mathbf{z}), \quad \text{as } T \to \infty$$

et en définissant $g(\mathbf{z}) = g(\mathbf{x}, \mathbf{v}) := h(\mathbf{x})$ induit, comme $T \to \infty$,

$$\frac{1}{T}\int_{t=0}^{T} h(\mathbf{X}_t)dt = \frac{1}{T}\int_{t=0}^{T} g(\mathbf{Z}_t)dt \to \int g(\mathbf{z})\rho(d\mathbf{z}) = \int\int h(\mathbf{x})\pi(d\mathbf{x})p(d\mathbf{v}|\mathbf{x}) = \int h(\mathbf{x})\pi(d\mathbf{x}),$$

où $p(d\mathbf{v}|\mathbf{x})$ est la distribution conditionnelle de la variable $\mathbf{V}$, given $\mathbf{X} = \mathbf{x}$.

Cependant, dans de nombreux cas, l'évaluation de l'intégrale de chemin $\int_{t=0}^{T} h(\mathbf{Z}_t)dt$ peut être coûteuse, voire impossible, et une discrétisation de la trajectoire simulée est une alternative réalisable. Cela signifie estimer la quantité d'intérêt, I, par l'estimateur suivant

$$\hat{I} = \frac{1}{N}\sum_{n=1}^{N} h(\mathbf{X}_{\frac{nT}{N}}).$$

Presque tous les échantillonneurs PDMD existants sont basés sur deux échantillonneurs spécifiques, tous deux reposant sur une dynamique linéairement déterministe, une caractéristique qui facilite la détermination de l'état de la chaîne de Markov entre les événements de Poisson. Vanetti et al. (2017) utilise la dynamique hamiltonienne

sur une approximation de la distribution cible pour accélérer l'échantillonneur de particules bondissantes, mais l'efficacité de cette modification dépend de la qualité de l'approximation et transfère seulement la difficulté de la détermination de la dynamique déterministe au calcul de l'événement fonction de taux.

Pour l'échantillonneur Bouncy Particle, l'ensemble de vitesses $\mathcal{V}$ est soit l'espace euclidien $\mathbb{R}^d$, soit la sphère unité $\mathbb{S}^{d-1}$. La distribution cible augmentée associée est $\rho(d\mathbf{x}, d\mathbf{v}) = \pi(d\mathbf{x})\mathcal{N}(d\mathbf{v}|0, I_d)$, ou $\rho(d\mathbf{x}, d\mathbf{v}) = \pi(d\mathbf{x})\mathcal{U}_{\mathbb{S}^{d-1}}(d\mathbf{v})$, où $\mathcal{N}(\cdot|0, I_d)$ représente la distribution gaussienne d-dimensionnelle standard et $\mathcal{U}_{\mathbb{S}^{d-1}}(d\mathbf{v})$ indique la distribution uniforme sur $\mathbb{S}^{d-1}$, respectivement. La dynamique déterministe correspondante est

$$\frac{d\mathbf{X}_t}{dt} = \mathbf{V}_t, \quad \frac{d\mathbf{V}_t}{dt} = \mathbf{0},$$

$\lambda(\mathbf{z}) = \lambda(\mathbf{z}) = \langle \mathbf{v}, \nabla U(\mathbf{x})\rangle_+ + \lambda^{\text{ref}}$, où λ^{ref} est une constante non-négative choisie par l'utilisateur et la dynamique de transition est comme

$$Q((d\mathbf{x}', d\mathbf{v}')|(\mathbf{x}, \mathbf{v})) = \frac{\langle \mathbf{v}, \nabla U(\mathbf{x})\rangle_+}{\lambda(\mathbf{x}, \mathbf{v})}\delta_{\mathbf{x}}(d\mathbf{x}')\delta_{R_{\nabla U(\mathbf{x})}\mathbf{v}}(d\mathbf{v}') + \frac{\lambda^{\text{ref}}}{\lambda(\mathbf{x}, \mathbf{v})}\delta_{\mathbf{x}}(d\mathbf{x}')\varphi(d\mathbf{v}')$$

où $\varphi(d\mathbf{v}) = \mathcal{U}_{\mathbb{S}^{d-1}}(d\mathbf{v})$ ou $\varphi(d\mathbf{v}) = \mathcal{N}(d\mathbf{v}|0, I_d)$, selon le choix de l'ensemble de vélocités, et l'opérateur $R_{\mathbf{w}}$, for any non-zero vector $\mathbf{w} \in \mathbb{R}^d - \{\mathbf{0}\}$, is $R_{\mathbf{w}}\mathbf{v} = \mathbf{v} - 2\frac{\langle \mathbf{w}, \mathbf{v}\rangle}{\langle \mathbf{w}, \mathbf{w}\rangle}\mathbf{w}$.

Pour l'échantillonneur Zigzag, l'ensemble de vitesses, $\mathcal{V}$, est l'ensemble discret $\{\mathbf{v} = (v_1, \cdots, v_d)|v_i \in \{-1, 1\}, i = 1, \cdots, d\}$ et $\rho(d\mathbf{x}, d\mathbf{v}) = \pi(d\mathbf{x})\varphi(d\mathbf{v})$, où φ est la distribution uniforme sur $\mathcal{V}$. ZS utilise la même dynamique déterministe linéaire que BPS. Son taux d'événements est $\lambda(\mathbf{z}) = \sum_{i=1}^d \lambda_i(\mathbf{x}, \mathbf{v}) = \sum_{i=1}^d \left[\{v_i\nabla_i U(\mathbf{x})\}_+ + \lambda_i^{\text{ref}}\right]$, où le λ_i^{ref} sont des constantes non négatives choisies par l'utilisateur. La dynamique de transition est $Q((d\mathbf{x}', d\mathbf{v}')|(\mathbf{x}, \mathbf{v})) = \sum_{i=1}^d \frac{\lambda_i(\mathbf{x},\mathbf{v})}{\lambda(\mathbf{x},\mathbf{v})}\delta_{\mathbf{x}}(d\mathbf{x}')\delta_{F_i\mathbf{v}}(d\mathbf{v}')$, où F_i dénote un opérateur qui retourne le i-th composant de $\mathbf{v}$ et les autres inchangés. En pratique, ZS s'appuie sur le théorème de superposition. À chaque événement, ZS simule les processus d Poisson, avec les taux $\lambda_i(\mathbf{x} + t\mathbf{v}, \mathbf{v})$, calcule leur première occurrence et prend leur minimum, Par exemple, i-th, pour la durée entre l'événement courant et l'événement suivant, et retourne la composante i-th de la vélocité $\mathbf{v}$.

Nous décrivons maintenant l'échantillonneur de coordonnées, dans lequel une seule composante de $\mathbf{x}$ évolue et le reste reste inactif entre les événements. Pour CS, l'ensemble de vitesses $\mathcal{V}$ est choisi pour être $\{\pm e_i, i = 1, \cdots, d\}$, où e_i est le vecteur avec le composant i-th égal àÀ un et les autres mis à zéro. La distribution cible augmentée est $\rho(d\mathbf{x}, d\mathbf{v}) = \pi(d\mathbf{x})\varphi(d\mathbf{v})$, avec $\varphi(d\mathbf{v})$ la distribution uniforme sur $\mathcal{V}$. Les caractéristiques PDMP de CS sont donc

1. **Dynamique déterministe**:

$$\frac{d\mathbf{X}_t}{dt} = \mathbf{V}_t, \quad \frac{d\mathbf{V}_t}{dt} = \mathbf{0}\,.$$

2. **Taux d'occurrence d'événement**: $\lambda(\mathbf{z}) = \langle \mathbf{v}, \nabla U(\mathbf{x})\rangle_+ + \lambda^{\text{ref}}$, where λ^{ref} is a user-chosen non-negative constant.

3. **Dynamique de transition**:

$$Q((d\mathbf{x}', d\mathbf{v}')|(\mathbf{x}, \mathbf{v})) = \sum_{\mathbf{v}^* \in \mathcal{V}} \frac{\lambda(\mathbf{x}, -\mathbf{v}^*)}{\lambda(\mathbf{x})}\delta_{\mathbf{x}}(d\mathbf{x}')\delta_{\mathbf{v}^*}(d\mathbf{v}')$$

where $\lambda(\mathbf{x}) = \sum_{\mathbf{v}\in\mathcal{V}} \lambda(\mathbf{x},\mathbf{v}) = 2d\lambda^{\text{ref}} + \sum_{i=1}^{d} \left|\frac{\partial U(\mathbf{x})}{\partial x_i}\right|$,

qui se traduit par le pseudo-code Nous établissons maintenant que CS bénéficie de

Algorithm 0.1 Coordinate Sampler

Input: Start with position $\mathbf{X}_0$, velocity $\mathbf{V}_0$ and set simulation time threshold T^{total}.

Generate a set of event times $\{\tau_0, \tau_1, \cdots, \tau_M\}$ and their associated state $\{\mathbf{Z}_{\tau_0}, \mathbf{Z}_{\tau_1}, \cdots, \mathbf{Z}_{\tau_M}\}$, where $\tau_0 = 0$, $\tau_M \geq T^{total}$ and $\tau_{M-1} < T^{total}$, $\mathbf{Z}_0 = (\mathbf{X}_0, \mathbf{V}_0)$

Set $t \leftarrow 0$, $T \leftarrow 0$, $m \leftarrow 0$, $\tau_m \leftarrow 0$

while $T < T^{total}$ **do**

$m \leftarrow m + 1$

$u \leftarrow \text{Uniform}[0, 1]$

Solve the equation

$$\exp\left\{-\int_0^{\eta_m} \lambda_m(t)dt\right\} = u$$

with respect to η_m, where $\lambda_m(t) = \lambda\left(\mathbf{X}_{\tau_{m-1}} + t\mathbf{V}_{\tau_{m-1}}, \mathbf{v}_{\tau_{m-1}}\right)$.

$\tau_m \leftarrow \tau_{m-1} + \eta_m$, $T \leftarrow \tau_m$, $\mathbf{Z}_{\tau_m} \sim Q((\mathbf{X}_{\tau_{m-1}} + \eta_m\mathbf{V}_{\tau_{m-1}}, \mathbf{v}_{\tau_{m-1}}), \cdot)$.

end while

Output: A trajectory of the Markov chain over $[0, \tau_M]$, $\{\mathbf{Z}_t\}_{t=0}^{\tau_M}$, where $\mathbf{Z}_t = (\mathbf{X}_{\tau_m} + (t - \tau_m)\mathbf{V}_{\tau_m}, \mathbf{V}_{\tau_m})$ for $\tau_m \leq t < \tau_{m+1}$.

la distribution cible augmentée, $\rho(d\mathbf{x}, d\mathbf{v})$, comme sa distribution invariante sous la condition que $U : \mathbb{R}^d \to \mathbb{R}^+$ est C^1. En outre, dans les hypothèses suivantes, CS est V-uniformement ergodique pour la fonction de Lyapunov

$$V(\mathbf{x}, \mathbf{v}) = \frac{e^{U(\mathbf{x})/2}}{\sqrt{\lambda^{\text{ref}} + \langle \nabla U(\mathbf{x}), -\mathbf{v}\rangle_+}},$$

qui a été utilisé dans Deligiannidis et al. (2017).

Theorem 0.0.1 *Pour tout $\lambda^{ref} > 0$ positif, le PDMP produit par CS bénéficie de $\rho(d\mathbf{x}, d\mathbf{v})$ comme unique distribution invariante, à condition le potentiel U est C^1.*

Il est facile de vérifier que le générateur d'échantillonneur de coordonnées, $\mathcal{L}$, satisfait

$$\int \mathcal{L}f(\mathbf{z})\rho(d\mathbf{z}) = 0,$$

pour toutes les fonctions, f, dans son générateur étendu, ce qui signifie que ρ est une distribution invariante de CS (Davis, 1993, Theorem 34.7). En outre, l'unicité découle de la positivité de λ^{ref}, qui permet à la chaîne de Markov d'atteindre n'importe quel état $(\mathbf{x}^*, \mathbf{v}^*)$ de tout état initial $(\mathbf{x}_0, \mathbf{v}_0)$, en temps fini.

Hypothèses: Supposons que $U : \mathbb{R}^d \to \mathbb{R}^+$ vérifie les conditions suivantes, définies par Deligiannidis et al. (2017),

A.1 $\frac{\partial^2 U(\mathbf{x})}{\partial x_i x_j}$ est localement Lipschitz continue pour tous i, j,

A.2 $\int \left|\nabla U(\mathbf{x})\right| \pi(d\mathbf{x}) < \infty$,

A.3 $\underline{\lim}_{|\mathbf{x}|\to\infty} e^{U(\mathbf{x})/2} / \sqrt{\left|\nabla U(\mathbf{x})\right|} > 0$

A.4 $V \geq c_0$ pour une constante positive c_0.

Conditions: Nous fixons des conditions

C.1 $\underline{\lim}_{|x|\to\infty} \big|\nabla U(x)\big| = \infty$, $\overline{\lim}_{|x|\to\infty} \|\Delta U(x)\| \leq \alpha_1 < \infty$ and $\lambda^{\text{ref}} > \sqrt{8\alpha_1}$.

C.2 $\underline{\lim}_{|\mathbf{x}|\to\infty} \big|\nabla U(\mathbf{x})\big| = 2\alpha_2 > 0$, $\overline{\lim}_{|x|\to\infty} \|\Delta U(x)\| = 0$ and $\lambda^{\text{ref}} < \frac{\alpha_2}{14d}$.

où **C.1** correspond à des distributions dont les queues se désintègrent $\mathcal{O}(|\mathbf{x}|^{\beta})$, où $1 < \beta \leq 2$, et **C.2** à la distribution avec des queues d'ordre $\mathcal{O}(|\mathbf{x}|^{1})$.

Theorem 0.0.2 *Supposons que les hypothèses $A.1 - A.4$ tiennent et que l'une des conditions $C.1, C.2$ est vraie, alors CS est V - uniformément ergodique: Il existe des constantes $\Gamma < \infty$ et $0 < \gamma < 1$, tel que*

$$\|P^t(\mathbf{z}, \cdot) - \rho\|_V \leq V(\mathbf{z})\Gamma\gamma^t \,,$$

où $P^t(\mathbf{z}, \cdot)$ est la distribution de la chaîne de Markov avec l'état initial $\mathbf{z}$ à l'instant t, et la norme $\|\cdot\|_V$ est défini par

$$\|\mu\|_V = \sup_{|f|<V} \int f(\mathbf{z})\mu(d\mathbf{z}) \,.$$

Pour CS, chaque temps d'événement voit un changement d'une seule composante de $\mathbf{X}$, contrairement à l'échantillonneur Zigzag, qui modifie tous les composants en même temps. Une première impression est que CS est donc moins efficace dans son exploration de l'espace cible, en comparaison avec ZS, à cause de cette restriction. Cependant, cette intuition est trompeuse. Supposons que λ_i's, $i = 1, \cdots, d$ dans ZS et λ dans CS sont d'échelles similaires, par exemple prendre la durée attendue entre deux événements de Poisson pour avoir la même valeur ℓ.

Supposons en outre que le calcul d'un temps d'occurrence ait le même coût de calcul, c, pour tous les processus de Poisson. Dans ZS, le taux d'événements est la somme des taux $\lambda_1, \cdots, \lambda_d$. Par conséquent, la durée entre deux événements est $\frac{\ell}{d}$ et le coût de calcul induit est dc. Ainsi, que chaque composante de $\mathbf{X}$ évolue pendant une durée ℓ coûte d^2c pour ZS. En revanche, dans CS, un coût de calcul dc résultera du déplacement de la chaîne de Markov pendant une durée $d\ell$. Par conséquent, le coût de calcul pour surveiller chaque composante pendant une durée ℓ est également dc. Par conséquent, CS est $\mathcal{O}(d)$ fois plus efficace que ZS en termes d'évolution d'une composante donnée de $\mathbf{X}$.

Dans la section 2.4, nous comparons l'efficacité de l'échantillonneur à coordonnées et de l'échantillonneur en zigzag sur plusieurs exemples et démontrons l'amélioration de l'échantillonneur à coordonnées par rapport à l'échantillonneur en zigzag.

Chapitre 3

Comme un algorithme d'échantillonnage puissant et efficace, Hamiltonien Monte Carlo (HMC) (Duane et al., 1987b; Neal et al., 2011) ont démontré sa surperformance dans les problèmes de haute dimension sur de nombreux autres algorithmes MCMC (Neal et al., 2011). En recourant à la dynamique Hamiltonienne, HMC réduit considérablement le comportement de marche aléatoire en acceptant les propositions

éloignées avec une forte probabilité. Cependant, lors de la mise en œuvre de la console HMC, il faut actualiser l'élan à chaque itération, ce qui introduit un caractère aléatoire et rejeter toutes les propositions intermédiaires le long de chaque chemin de sautemouton, ce qui gaspille l'effort de calcul. Motivés par ces deux défis, nous proposons dans ce chapitre une nouvelle variante de HMC, qui coïncide avec la discrétisation des échantillonneurs PDMP (Processus de Markov déterministes), récemment contrôlés par morceaux, pour contrôler la fréquence de rafraîchissement de la quantité de mouvement et recycler propositions intermédiaires, tout en gardant l'exactitude en même temps.

Pour supprimer le comportement de marche aléatoire résultant de l'actualisation des moments entre les trajectoires, Horowitz (1991) a proposé de ne les rafraîchir que partiellement à chaque itération. Pour ne pas pouvoir recycler les propositions intermédiaires le long des trajectoires, seule une légère amélioration est obtenue par ce rafraîchissement partiel (Neal et al., 2011). Nishimura and Dunson (2015) a recyclé les propositions intermédiaires en augmentant les temps cibles L, où L est le nombre d'étapes leapfrog à chaque itération, alors que Bernton et al. (2015) les pondérait localement pour améliorer la console HMC. Par rapport à ces deux méthodes, notre méthode non seulement réutilise les propositions intermédiaires, mais contrôle la fréquence de rafraîchissement.

Comme il est introduit en physique par Peters et al. (2012) et popularisé dans les statistiques par (Bierkens et al., 2016; Bouchard-Côté et al., 2017), l'échantillonneur basé sur PDMP, en tant que méthode MCMC à temps continu, non réversible et sans rejets, a montré des performances exceptionnelles dans cas de grande dimension (Bouchard-Côté et al., 2017; Galbraith, 2016), même comparé à l'algorithme HMC. Par chance, notre variante de HMC peut être considérée comme une discrétisation d'un échantillonneur spécifique basé sur PDMP, d'après (Sherlock and Thiery, 2017; Vanetti et al., 2017). Par conséquent, nous nommons cette variante par un processus de Markov déterministe par morceaux discret avec une dynamique hamiltonienne (dPDMP-HMC). Jusqu'à présent, pour générer les trajectoires déterministes par morceaux, presque tous les échantillonneurs PDMP existants tirent parti de la dynamique linéaire, à l'exception du BPS hamiltonien de Vanetti et al. (2017), qui transfère l'obstacle de la dynamique non linéaire dans la fonction de taux d'événements et est basé sur une bonne approximation gaussienne. Les échantillonneurs discrets à base de PDMP peuvent contourner cette difficulté, sans perte d'efficacité.

Nous supposons que la distribution cible augmentée a une forme distincte, $\pi(\mathbf{x}, \mathbf{v}) = \pi(\mathbf{x})\varphi(\mathbf{v})$, où φ est une distribution sur $\mathbb{R}^d$. Pour simplifier, nous supposons que φ est la distribution gaussienne standard. Les processus PDMP-HMC utilisent la dynamique hamiltonienne, génèrent un temps d'événement avec un taux d'événements constant et actualisent la vitesse en fonction de sa distribution marginale au moment de l'événement. Plus précisément,

1. **dynamique déterministe**:

$$\frac{d\mathbf{x}_t}{dt} = -\nabla \log \varphi(\mathbf{v}_t) = \mathbf{v}_t$$
$$\frac{d\mathbf{v}_t}{dt} = \nabla \log \pi(\mathbf{x}_t)$$

2. **taux d'occurrence d'événement**: $\lambda(\mathbf{z}_t) = \lambda(\mathbf{x})$.

3. **dynamique de transition**: $Q((\mathbf{x},\mathbf{v}),(d\mathbf{x}',d\mathbf{v}')) = \delta_{\mathbf{x}}(d\mathbf{x}')\varphi(d\mathbf{v}')$

Theorem 0.0.3 *La chaîne de Markov déterministe par morceaux ci-dessus admet* $\rho(d\mathbf{x},\mathbf{v}) = \pi(d\mathbf{x}) \otimes \varphi(d\mathbf{v})$ *sur* $\mathbb{R}^{2d}$ *comme sa distribution invariante, où* $\varphi(\mathbf{v})$ *est le d-dimensionnelle distribution standard normale.*

En pratique, attendez-vous à quelques exemples spécifiques, nous ne pouvons pas implémenter PDMP-HMC pour échantillonner à partir de la distribution cible puisqu'il est difficile de calculer exactement les trajectoires selon la dynamique Hamiltonienne. Merci à la discrétisation de PDMP dans (Sherlock and Thiery, 2017; Vanetti et al., 2017), nous pouvons surmonter cette difficulté avec la même complexité de calcul de la console HMC.

En tant qu'algorithme HMC classique, nous discrétisons la dynamique hamiltonienne avec l'intégrateur leapfrog dans PDMP-HMC et étalons le biais avec l'étape d'acceptation de Metropolis-Hastings. Dans PDMP-HMC discret (dPDMP-HMC), nous introduisons trois noyaux de transition de Markov.

1. Noyau Leapfrog $P_1^{\epsilon,L}$: $(\mathbf{x}^+,\mathbf{v}^+) = \text{Leapfrog}(\mathbf{x},\mathbf{v},\epsilon,L_r)$ où $L_r \sim \text{Uniform}\{1,2,\cdots,L\}$.

   (a) $(\mathbf{x},\mathbf{v}) \to (\mathbf{x}^+,-\mathbf{v}^+)$ w.p.
   $$1 \wedge \frac{\pi(\mathbf{x}^+)\varphi(-\mathbf{v}^+)}{\pi(\mathbf{x})\varphi(\mathbf{v})}$$

   (b) $(\mathbf{x},\mathbf{v}) \to (\mathbf{x},\mathbf{v})$ w.p.
   $$1 - 1 \wedge \frac{\pi(\mathbf{x}^+)\varphi(-\mathbf{v}^+)}{\pi(\mathbf{x})\varphi(\mathbf{v})}$$

2. Noyau de flipP_2: retourner la vitesse, $(\mathbf{x},\mathbf{v}) \to (\mathbf{x},-\mathbf{v})$.

3. Noyau de rafraîchissement P_3: rafraîchir la vitesse, $(\mathbf{x},\mathbf{v}) \to (\mathbf{x},\mathbf{v}^*)$, où $\mathbf{v}^* \sim \varphi$.

Le noyau de transition de dPDMP-HMC est

$$P^{\epsilon,L,\eta} = (1-\eta)P_2 \circ P_1^{\epsilon,L} + \eta P_3$$

où $\epsilon > 0$ est la taille de pas de l'intégrateur de sautemouton et $\eta \in (0,1)$ est la probabilité d'actualisation de la vitesse. η est un paramètre pour contrôler la fréquence de rafraîchissement de la quantité de mouvement, $\mathbf{v}$, alors que ϵ et L ont les mêmes fonctions qu'eux dans la console HMC. Dans dPDMD-HMC, nous utilisons des sauts progressifs aléatoires à chaque itération, afin d'éliminer la répétition exacte des propositions visitées. Puisque P_1 et P_2 sont réversibles par rapport à $\pi(\mathbf{x},\mathbf{v})$, leur composition est également invariante par rapport à $\pi(\mathbf{x},\mathbf{v})$. Comme P_3 actualise l'impulsion en fonction de sa distribution conditionnelle, P est invariant par rapport à $\pi(\mathbf{x},\mathbf{v})$ aussi.

Un pseudo-code de dPDMP-HMC est décrit en détail dans Algorithm 3.2. Dans la description de l'algorithme 3.2, $(\mathbf{x}_k^{\mathcal{B}},\mathbf{v}_k^{\mathcal{B}})$ indique le k-th élément de $\mathcal{B}$.
Dans cette section 3.4, nous comparons l'efficacité de dPDMP-HMC et HMC classique sur trois exemples. D'après les expériences numériques, comparé à la HMC, le dPDMP-HMC présente une efficacité plus élevée - en termes de taille d'échantillon

Algorithm 0.2 Leapfrog Integrator

Input: start position $\mathbf{x}_0$, start velocity $\mathbf{v}_0$, stepsize ϵ, and steps L.
for $\ell = 1, 2, \cdots, L$ **do**

$$\begin{aligned}
\mathbf{v}_{\ell-\frac{1}{2}} &\leftarrow \mathbf{v}_{\ell-1} + \frac{1}{2}\epsilon\nabla \log(\pi(\mathbf{x}_{\ell-1})) \\
\mathbf{x}_{\ell} &\leftarrow \mathbf{x}_{\ell-1} + \epsilon\mathbf{v}_{\ell-\frac{1}{2}} \\
\mathbf{v}_{\ell} &\leftarrow \mathbf{v}_{\ell-\frac{1}{2}} + \frac{1}{2}\epsilon\nabla \log(\pi(\mathbf{x}_{\ell}))
\end{aligned}$$

end for
Output: Return $(\mathbf{x}_L, \mathbf{v}_L)$

efficace et de plus grande précision - en termes de test de Kolmogorov-Smirnov et de distance de Wasserstein.

Chapitre 4

En tant que méthode d'échantillonnage puissante, la méthode de Monte Carlo à chaîne de Markov (MCMC) a été largement utilisée dans les statistiques informatiques et est maintenant un outil standard dans l'inférence bayésienne, où la distribution postérieure est souvent analytiquement intraitable, connue jusqu'à une constante. Cependant, presque tous les algorithmes MCMC existants, tels que l'algorithme de Metropolis-Hastings (MH), Hamiltonian Monte Carlo (HMC) et l'algorithme de Langevin ajusté par Metropolis (MALA), sont basés sur une condition d'équilibre détaillée datant de (Metropolis et al. (1953), Hastings (1970)). Récemment, un nouveau type de méthode MCMC — le processus de Markov déterministe par morceaux (PDMP) — est apparu dans les statistiques computationnelles, méthode qui est généralement irréversible, ce qui signifie une violation de la condition d'équilibre détaillée. La théorie de PDMP est développée par (Davis (1984),Davis (1993)), alors que ses applications remarquables sur les statistiques de calcul sont implémentées par (Bouchard-Côté et al. (2017), Bierkens et al. (2017b), Bierkens et al. (2016)).

Comparé aux algorithmes MCMC traditionnels, PDMP est sans rejet, ce qui signifie qu'il n'y a pas de gaspillage d'échantillons de propositions. Sur la base des conditions d'équilibre détaillées, les algorithmes MCMC traditionnels sont réversibles. Cependant, certains travaux théoriques et expériences numériques (Hwang et al. (1993), Sun et al. (2010), Chen and Hwang (2013), Bierkens (2016)) ont montré que la chaîne de Markov irréversible peut surpasser la MCMC réversible par rapport au taux de mélange et asymptotique variance. Le PDMP est une chaîne de Markov typiquement irréversible, qui mérite d'être étudiée. L'échantillonneur de particules rebondissantes (BPS) génère une chaîne de Markov déterministe par morceaux, originaire de Peters et al. (2012) et explorée par Bouchard-Côté et al. (2017). L'échantillonneur de processus zig-zag Bierkens et al. (2016) est un autre exemple de PDMP et Fearnhead et al. (2016) unifie l'échantillonneur de processus BPS et zig-zag dans le cadre de PDMP. En outre, les algorithmes MCMC sont difficiles à mettre à l'échelle, car le calcul de chaque taux d'acceptation MH doit balayer l'ensemble des données. Cependant, selon leurs structures spéciales, les échantillonneurs de processus BPS

Algorithm 0.3 Discrete PDMP-HMC Sampler

Input: starting position $\mathbf{x}_0$, stepsize ϵ, refreshment probability η, iteration number N and upper bound of steps L.

Generate starting velocity $\mathbf{v}_0 \sim \mathcal{N}(\mathbf{0}, I_d)$. $\mathcal{B} \leftarrow \{(\mathbf{x}_0, \mathbf{v}_0)\}$, $v \leftarrow 1$ and $\gamma \leftarrow 1$.

for $i = 1, 2, \cdots, N$ **do**

  $u \sim \mathcal{U}[0,1]$

  if $u < \eta$ **then**

    $\mathbf{x}_i \leftarrow \mathbf{x}_{i-1}$, $\mathbf{v}_i \sim \mathcal{N}(\mathbf{0}, I_d)$, $\mathcal{B} \leftarrow \{(\mathbf{x}_i, \mathbf{v}_i)\}$, $v \leftarrow 1$ and $\gamma \leftarrow 1$.

  else

    $L_r \sim \text{Uniform}\{1, 2, \cdots, L\}$

    if $\gamma + L_r > \#\mathcal{B}$ and $v = 1$ **then**

      $L_\delta \leftarrow \gamma + L_r - \#\mathcal{B}$

      $\{(\mathbf{x}_k^*, \mathbf{v}_k^*)\}_{k=1,\cdots,L_\delta} \leftarrow$ all proposals of Leapfrog$(\mathbf{x}^{\mathcal{B}}_{\#\mathcal{B}}, \mathbf{v}^{\mathcal{B}}_{\#\mathcal{B}}, \epsilon, L_\delta)$, $\mathcal{B} \leftarrow \{\mathcal{B}, (\mathbf{x}_1^*, \mathbf{v}_1^*), \cdots, (\mathbf{x}_{L_\delta}^*, \mathbf{v}_{L_\delta}^*)\}$

      $u \sim \mathcal{U}[0,1]$, compute $p \leftarrow \frac{\pi(\mathbf{x}^*_{L_\delta}, \mathbf{v}^*_{L_\delta})}{\pi(\mathbf{x}_{i-1}, \mathbf{v}_{i-1})}$.

      if $u < 1 \wedge p$ **then**

        $(\mathbf{x}_i, \mathbf{v}_i) \leftarrow (\mathbf{x}^*_{L_\delta}, \mathbf{v}^*_{L_\delta})$, $\gamma \leftarrow \gamma + L_r$ and $v \leftarrow 1$.

      else

        $(\mathbf{x}_i, \mathbf{v}_i) \leftarrow (\mathbf{x}_{i-1}, -\mathbf{v}_{i-1})$, $\gamma \leftarrow \gamma$ and $v \leftarrow -1$

      end if

    else

      if $\gamma - L_r < 1$ and $v = -1$ **then**

        $L_\delta \leftarrow L_r + 1 - \gamma$

        $\{(\mathbf{x}_k^*, \mathbf{v}_k^*)\}_{k=1,\cdots,L_\delta} \leftarrow$ all proposals of Leapfrog$(\mathbf{x}_1^{\mathcal{B}}, \mathbf{v}_1^{\mathcal{B}}, -\epsilon, L_\delta)$

        $\mathcal{B} \leftarrow \{(\mathbf{x}^*_{L_\delta}, \mathbf{v}^*_{L_\delta}), \cdots, (\mathbf{x}_1^*, \mathbf{v}_1^*), \mathcal{B}\}$

        $u \sim \mathcal{U}[0,1]$, compute $p \leftarrow \frac{\pi(\mathbf{x}^*_{L_\delta}, \mathbf{v}^*_{L_\delta})}{\pi(\mathbf{x}_{i-1}, \mathbf{v}_{i-1})}$.

        if $u < 1 \wedge p$ **then**

          $(\mathbf{x}_i, \mathbf{v}_i) \leftarrow (\mathbf{x}^*_{L_\delta}, -\mathbf{v}^*_{L_\delta})$, $\gamma \leftarrow 1$ and $v \leftarrow -1$.

        else

          $(\mathbf{x}_i, \mathbf{v}_i) \leftarrow (\mathbf{x}_{i-1}, -\mathbf{v}_{i-1})$, $\gamma \leftarrow \gamma + L_\delta$ and $v \leftarrow 1$.

        end if

      else

        if $v = 1$ **then**

          $u \sim \mathcal{U}[0,1]$, compute $p \leftarrow \frac{\pi(\mathbf{x}^{\mathcal{B}}_{\gamma+L_r}, \mathbf{v}^{\mathcal{B}}_{\gamma+L_r})}{\pi(\mathbf{x}_{i-1}, \mathbf{v}_{i-1})}$

          if $u < 1 \wedge p$ **then**

            $(\mathbf{x}_i, \mathbf{v}_i) \leftarrow (\mathbf{x}^{\mathcal{B}}_{\gamma+L_r}, \mathbf{v}^{\mathcal{B}}_{\gamma+L_r})$, $\gamma \leftarrow \gamma + L_r$, and $v \leftarrow 1$.

          else

            $(\mathbf{x}_i, \mathbf{v}_i) \leftarrow (\mathbf{x}_i, -\mathbf{v}_i)$, $\gamma \leftarrow \gamma$, and $v \leftarrow -1$

          end if

        else

          $u \sim \mathcal{U}[0,1]$, compute $p \leftarrow \frac{\pi(\mathbf{x}^{\mathcal{B}}_{\gamma-L_r}, \mathbf{v}^{\mathcal{B}}_{\gamma-L_r})}{\pi(\mathbf{x}_{i-1}, \mathbf{v}_{i-1})}$

          if $u < 1 \wedge p$ **then**

            $(\mathbf{x}_i, \mathbf{v}_i) \leftarrow (\mathbf{x}^{\mathcal{B}}_{\gamma-L_r}, -\mathbf{v}^{\mathcal{B}}_{\gamma-L_r})$, $\gamma \leftarrow \gamma - L_r$, and $v \leftarrow -1$.

          else

            $(\mathbf{x}_i, \mathbf{v}_i) \leftarrow (\mathbf{x}_{i-1}, -\mathbf{v}_{i-1})$, $\gamma \leftarrow \gamma$, and $v \leftarrow 1$

          end if

        end if

      end if

    end if

  end if

end for

Output: $\{(\mathbf{x}_i, \mathbf{v}_i)\}_{i=1}^N$.

et zig-zag sont faciles à mettre à l'échelle pour les gros volumes de données. À l'exception des algorithmes MCMC de style PDMP, presque tous les autres algorithmes MCMC évolutifs existants (tels queScott et al. (2016a), Neiswanger et al. (2013a), Wang and Dunson (2013a), Minsker et al. (2014c), Quiroz et al. (2015), Bardenet et al. (2015)), sont approximatives, non exactes, qui n'admettent pas la distribution cible comme leur distribution invariante.

Nous généralisons l'échantillonneur de particules rebondissant - l'échantillonneur généralisé de particules rebondissantes (GBPS) - qui peut être traité comme une extension de l'échantillonneur de processus BPS et zig-zag. En BPS, la dynamique de transition à l'événement est déterministe. En revanche, la dynamique de transition dans GBPS est aléatoire par rapport à une certaine distribution. Dans l'échantillonneur de processus en zig-zag, nous décomposons la vitesse par rapport à un système de coordonnées fixe, tandis qu'en GBPS, nous utilisons un système de coordonnées mobiles pour décomposer la vitesse. Par ailleurs, le gain principal de GBPS par rapport à BPS est qu'il n'y a pas de paramètre à régler. En fait, pour BPS, afin de surmonter le problème de réductibilité, nous devons ajouter un processus de Poisson de rafraîchissement pour rafraîchir la vitesse de temps en temps, qui doit être réglé pour équilibrer l'efficacité et la précision. Mais pour GBPS, le caractère aléatoire du rafraîchissement est incorporé dans la dynamique de transition et il n'y a pas de paramètre à régler.

En BPS, à l'heure de l'événement, la vélocité change de manière déterministe. Cependant, nous trouvons que la vitesse peut être changée dans d'autres directions, selon une certaine distribution, à l'instant de l'événement, qui incorpore le caractère aléatoire du processus de Poisson de référence en BPS pour surmonter la réductibilité. Dans cette section, nous généralisons le BPS. Plus précisément, avaànt l'heure de l'événement, nous décomposons la vitesse en fonction du gradient de $\log \pi(\mathbf{x})$, retournons le sous-secteur parallèle et rééchantillonnons le sous-vecteur orthogonal par rapport à une distribution. Les détails sont les suivants:

1. **dynamique déterministe**:

$$\frac{dx_t^{(i)}}{dt} = v_t^{(i)}, \quad \frac{dv_t^{(i)}}{dt} = 0, \quad i = 1, \cdots, d$$

2. **taux d'occurrence d'événement**: $\lambda(\mathbf{z}_t) = \max\{0, -\langle \mathbf{v}_t, \nabla \log \pi(\mathbf{x}_t)\rangle\}$.

3. **dynamique de transition**: $Q(d\mathbf{x}', d\mathbf{v}'|\mathbf{x}, \mathbf{v}) = \delta_{\{\mathbf{x}\}}(d\mathbf{x}')\delta_{\{-\mathbf{v}_1\}}(d\mathbf{v}_1')\mathcal{N}_{\mathbf{v}_1^\perp}(d\mathbf{v}_2')$, où

$$\mathbf{v}_1 = \frac{\langle \mathbf{v}, \nabla \log \pi(\mathbf{x})\rangle}{\langle \nabla \log \pi(\mathbf{x}), \nabla \log \pi(\mathbf{x})\rangle} \nabla \log \pi(\mathbf{x}), \quad \mathbf{v}_2 = \mathbf{v} - \mathbf{v}_1$$

$$\mathbf{v}_1' = \frac{\langle \mathbf{v}', \nabla \log \pi(\mathbf{x})\rangle}{\langle \nabla \log \pi(\mathbf{x}), \nabla \log \pi(\mathbf{x})\rangle} \nabla \log \pi(\mathbf{x}), \quad \mathbf{v}_2' = \mathbf{v}' - \mathbf{v}_1'$$

$$\mathbf{v}_1^\perp = \left\{\mathbf{u} \in \mathbb{R}^d : \langle \mathbf{u}, \mathbf{v}_1\rangle = 0\right\}$$

   $\mathcal{N}_{\mathbf{v}_1^\perp}$ est la distribution normale standard en $(d-1)$-dimensions sur l'espace $\mathbf{v}_1^\perp$.

We summarize the GBPS in Algorithm 4.2.

Algorithm 0.4 Generalized Bouncy Particle Sampler

Initialize: $\mathbf{x}_0, \mathbf{v}_0, T_0 = 0$.
for $i = 1, 2, 3, \cdots$ **do**
  Generate $\tau \sim PP(\lambda(\mathbf{x}_t, \mathbf{v}_t))$
  $T_i \leftarrow T_{i-1} + \tau$
  $\mathbf{x}_i \leftarrow \mathbf{x}_{i-1} + \tau \mathbf{v}_{i-1}$
  $\mathbf{v}_i \leftarrow Q(d\mathbf{v}|\mathbf{x}_i, \mathbf{v}_{i-1})$
end for

Theorem 0.0.4 *La chaîne de Markov déterministe par morceaux ci-dessus admet* $\pi(\mathbf{x})d\mathbf{x} \otimes \psi_d(\mathbf{v})d\mathbf{v}$ *sur* $\mathbb{R}^{2d}$ *comme sa distribution invariante, où* $\psi_d(\mathbf{v})$ *est la fonction de densité de la distribution normale standard de d- dimension.*

Hypothèse 1: Pour deux points $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^d$ et toute vitesse $\mathbf{v} \in \mathbb{R}^d, \|\mathbf{v}\|_2 = 1$, il existe $t > 0$, tel que

$$\mathbf{x}_2 \in S^{\perp}(\mathbf{x}_1 + t\mathbf{v}, \mathbf{v})$$

Theorem 0.0.5 *Sous l'hypothèse 1, la chaîne de Markov* $\mathbf{z}'_t = (\mathbf{x}_t, \frac{\mathbf{v}_t}{\|\mathbf{v}_t\|})$ *induite par GBPS admet* $\pi(\mathbf{x}) \times \mathcal{U}(S_{d-1})$ *comme unique distribution invariante.*

Dans l'analyse bayésienne, nous supposons que les observations $\{y_1, y_2, \cdots, y_N\}$ sont i.i.d. des exemples d'une distribution de la famille $\{\mathbb{P}_{\mathbf{x}}, \mathbf{x} \in \mathbb{R}^d\}$ and let $\mathbb{P}_{\mathbf{x}}$ et laisser $\mathbb{P}_{\mathbf{x}}$ admet la densité $p_{\mathbf{x}}$ par rapport à la mesure de Lebesgue sur $\mathbb{R}^d$. tant donné un précédent $\pi_0(\mathbf{x})$ sur le paramètre $\mathbf{x}$, le postérieur est

$$\pi(\mathbf{x}) \overset{\text{def}}{=} \pi(\mathbf{x}|y_1, \cdots, y_N) \propto \pi_0(\mathbf{x}) \prod_{n=1}^{N} p_{\mathbf{x}}(y_n)$$

Les algorithmes MCMC traditionnels (avec étape MH) sont difficiles à mettre à l'échelle pour un grand ensemble de données, car chaque étape MH doit balayer l'ensemble des données. Cependant, comme indiqué dans Bierkens et al. (2016), le PDMP peut être très efficace en utilisant un sous-échantillonnage pour simuler des échantillons de la distribution cible si nous pouvons donner une limite supérieure de la fonction de taux. En GBPS, nous utilisons seulement le gradient du logarithme de la distribution cible, ce qui signifie que nous pouvons simuler le postérieur en le connaissant jusqu'à une constante. En outre, nous pouvons donner un estimateur non biaisé du gradient du logarithme du postérieur en utilisant sa structure de somme pour simuler le postérieur exactement:

$$\widehat{\nabla \log \pi(\mathbf{x})} = N \nabla \log \pi_I(\mathbf{x}) = \nabla \log \pi_0(\mathbf{x}) + N \nabla_{\mathbf{x}} \log p_{\mathbf{x}}(y_I), \quad I \sim \mathcal{U}\{1, 2, \cdots, N\}$$

Dans Algorithme 4.3, nous montrons le workflow de l'implémentation du sous-échanti-llonnage en GBPS. Notez que $\lambda(\Delta, \mathbf{v}_{i-1})$ est égal à $\lambda(\mathbf{x}, \mathbf{v}_{i-1})$ dans lequel $\nabla \log \pi(\mathbf{x})$ est remplacé par Δ. $\Lambda(t)$ est une limite supérieure de $\lambda(\mathbf{x}, \mathbf{v})$. Dans la section 4.4, nous appliquons l'algorithme GBPS sur trois expériences numériques. L'exemple 1 montre qu'un problème de réductibilité apparaît dans la distribution gaussienne isotrope pour BPS sans rafraîchissement mais n'est pas rencontré par GBPS. Dans l'exemple 2, nous pouvons trouver que GBPS fonctionne bien sur les distributions multimodes et avec des performances similaires avec BPS. Enfin, nous présentons le GBPS avec sous-échantillonnage sur le modèle logistique Bayésien.

Algorithm 0.5 Subsampling version

Initialize: $\mathbf{x}_0, \mathbf{v}_0, T_0 = 0$.
for $i = 1, 2, 3, \cdots$ **do**
  Generate $\tau \sim PP(\Lambda(t))$
  $T_i \leftarrow T_{i-1} + \tau$
  $\mathbf{x}_i \leftarrow \mathbf{x}_{i-1} + \tau \mathbf{v}_{i-1}$
  $I \sim \mathcal{U}(\{1, \cdots, N\})$
  $\Delta \leftarrow N \nabla \log \pi_I(\mathbf{x}_i)$
  $q \leftarrow \lambda(\Delta, \mathbf{v}_{i-1})/\Lambda(\tau)$
  $u \sim \mathcal{U}(0, 1)$
  if $u \leq q$ **then**
    $\mathbf{v}_i \leftarrow Q(d\mathbf{v}|\Delta, \mathbf{v}_{i-1})$
  else
    $\mathbf{v}_i \leftarrow \mathbf{v}_{i-1}$
  end if
end for

Chapitre 5

L'algorithme de chaîne de Markov Monte Carlo (MCMC), une méthode d'échantillonnage générique, est omniprésente dans les statistiques modernes, en particulier dans les champs Bayésiens. Algorithmes MCMC seulement besoin d'évaluer la cible avec précision jusqu'à une constante multiple dans afin d'en échantillonner asymptotiquement. Par exemple, dans l'analyse Bayésienne, le principal objet d'intérêt est la distribution a posteriori, qui n'a le plus souvent pas de forme fermée, et MCMC a devenir un cheval de trait standard dans ce domaine. Cependant, MCMC est assez délicat à l'échelle et ses applications sont limitées lorsque la taille de l'observation devient très grande, car il faut balayer l'ensemble de l'observation (ou échantillon) dans afin d'évaluer la fonction de vraisemblance à chaque itération. Récemment, plusieurs méthodes ont été proposées pour mettre à l'échelle l'algorithme MCMC contre soidisant "Big Data" et ils peuvent être grossièrement classé en deux groupes (Bardenet et al., 2015): méthodes de diviser pour régner et méthodes basées sur le sous-échantillonnage.

Les méthodes de diviser pour régner divisent l'ensemble de données en sous-ensembles ou en lots, exécute MCMC sur chaque sous-ensemble pour générer des échantillons de paramètres et les combine pour produire une approximation de la distribution postérieure. Selon comment un l'algorithme MCMC est appliqué à ces sous-ensembles, les méthodes peuvent être classées en deux sous-catégories. Tout d'abord, soit $\mathcal{X}$ soit l'ensemble des données et $\mathcal{X}_1, \cdots, \mathcal{X}_K$ sont les sous-ensembles. Notons π_0 comme distribution prioritaire sur le paramètre *theta*. Ensuite, une catégorie de proposition (Neiswanger et al., 2013a; Nemeth et al., 2017; Scott et al., 2016a; Wang and Dunson, 2013a; Wang et al., 2015) est de lancer MCMC sur

$$\pi_k(\theta|\mathcal{X}_k) \propto (\pi_0(\theta))^{1/K} \prod_{x \in \mathcal{X}_k} p(x|\theta).$$

La deuxième catégorie (Minsker et al., 2014a; Srivastava et al., 2015a) cible plutôt

$$\pi_k(\theta|\mathcal{X}_k) \propto \pi_0(\theta) \left(\prod_{x \in \mathcal{X}_k} p(x|\theta) \right)^K.$$

Les méthodes basées sur un sous-échantillonnage utilisent une partition de l'ensemble de données estimer le taux d'acceptation MH à chaque itération, en vue d'accélérer le les algorithmes MCMC résultants. Ces techniques peuvent également être subdivisées en deux plus fines classes: méthodes de sous-échantillonnage exactes et approximatives, en fonction de leurs sorties. Les approches de sous-échantillonnage exactes nécessitent généralement explorer un espace augmenté, en traitant la distribution cible comme son invariant distribution marginale. Une direction (Quiroz et al., 2016) est de prendre avantage de MCMC de pseudo-marginal (Andrieu and Roberts, 2009) via la construction de estimateurs non biaisés de la densité cible sous-ensembles de données. Un autre est de recourir à déterministe par morceaux processus de Markov (Bierkens et al., 2016; Bouchard-Côté et al., 2017; Davis, 1984, 1993; Fearnhead et al., 2016; Sherlock and Thiery, 2017; Vanetti et al., 2017), qui prennent les cibles comme marginales de leurs distributions invariantes correspondantes. Sous-échantillonnage approximatif les approches visent à construire une approximation des distributions cibles. Une approche (Bardenet et al., 2015) est de déterminer l'acceptation des propositions avec une probabilité élevée via des sous-ensembles des données. Une autre approche Chen et al. (2014a); Ding et al. (2014); Welling and Teh (2011a) est basé sur direct modifications de méthodes exactes. Le travail séminal dans cette direction est gradient stochastique Dynamique de Langevin (SGLD) (Welling and Teh, 2011a).

Nous proposons deux nouvelles méthodes pour mettre à l'échelle les algorithmes MCMC, basés sur la méthode de diviser pour régner. Contrairement aux anciennes solutions diviser pour régner, nous utilisons MCMC sur une version pondérée de postérieurs partiels

$$\pi_k(\theta|\mathcal{X}_k) \propto \left\{ (\pi_0(\theta))^{1/K} \prod_{x\in\mathcal{X}_k} p(x|\theta) \right\}^{\lambda},$$

où λ n'est pas nécessaire restreint aux valeurs 1 ou K. En outre, nous utilisons forêts aléatoires (Breiman, 2001) pour apprendre rapidement les approximations de ces subposteriors et nous profitons des solutions pour déduire une approximation de la vrai postérieur si une étape MCMC supplémentaire ou par l'échantillonnage d'importance.

Noté par $\mathcal{X} = \{x_1, x_2, \cdots, x_N\}$ le ensemble complet d'observations, où $x_i \sim p_\theta(\cdot)$ i.i.d. et $\theta \in \Theta \subset \mathbb{R}^d$. Soit $\pi_0(\theta)$ le précédent distribution sur Θ. Séparer l'ensemble de données $\mathcal{X}$ en sous-ensembles $\mathcal{X}_1, \cdots, \mathcal{X}_K$, chacun avec la même taille $m = \frac{N}{K}$. Dans l'analyse bayésienne, la cible d'intérêt est la distribution postérieure:

$$\pi(\theta|\mathcal{X}) \propto \pi_0(\theta) \prod_{i=1}^{N} p(x_i|\theta)$$

Pour chaque sous-ensemble $\mathcal{X}_k$, $k = 1, \cdots, K$, nous définissons le λ_k-subposterior associé comme

$$\pi_k^{\lambda_k}(\theta|\mathcal{X}_k) = \frac{(\gamma_k(\theta|\mathcal{X}_k))^{\lambda_k}}{Z_{k,\lambda_k}}, \quad \gamma_k(\theta|\mathcal{X}_k) = \pi_0(\theta)^{\frac{1}{K}} \prod_{x\in\mathcal{X}_k} p(x|\theta),$$

où Z_{k,λ_k} est la constante de normalisation de $(\gamma_k(\theta|\mathcal{X}_k))^{\lambda_k}$. Dans les stratégies de division et de conquête, on applique des algorithmes MCMC à chaque subposterior, générant ainsi des échantillons à partir de ces distributions, et combine ces

échantillons à approcher la vraie distribution a posteriori. Diviser et conquérir MCMC existant méthodes traite les puissances λ_k de deux façons possibles: l'une est de mettre $\lambda_k = 1$ pour tous $k = 1, \cdots, K$, comme dans les échantillonneurs de consensus Monte Carlo et Weierstrass; la l'autre est de mettre $\lambda_k = K$, comme dans WASP (Srivastava et al., 2015a) et M-posterior (Minsker et al., 2014a). Mais il n'y a aucune raison d'adopter cette restriction. En outre, les méthodes mentionnées ci-dessus, à l'exception de Nemeth et al. (2017), ignore un sous-produit précieux — à savoir, la valeur de la distribution cible calculée aux points d'échantillonnage proposés — de l'exécution de ces MCMC algorithmes. En comparaison avec Nemeth et al. (2017), nos méthodes diffèrent de deux façons: une mineure est que nous utilisons le subposterior à l'échelle et l'autre est que l'algorithme d'apprentissage que nous implémentons est bon marché aux étapes de formation et de prédiction.

Comme mentionné ci-dessus, l'idée centrale de notre approche est de intégrer une procédure de régression d'apprentissage automatique dans la méthodologie de division et de conquête vers la construction d'approximations de toutes les fonctions de densité subposterior et dérivent de ces approximations du vrai postérieur. Plus précisément, nous exécutons certains MCMC algorithmes, tels que Metropolis - Hastings, HMC, MALA, ou d'autres versions, au hasard $\pi_k^{\lambda_k}(\theta|\mathcal{X}_k)$ pour obtenir des exemples MCMC $\{\theta_1^k, \theta_2^k, \cdots, \theta_T^k\}$ qui sont (approximativement) de ces cibles. Dans la métropole - Hastings taux d'acceptation, nous évaluons donc $\log \gamma_k(\theta|\mathcal{X}_k)$ pour chaque proposition, plutôt que le plus difficile $\pi_k^{\lambda_k}(\theta|\mathcal{X}_k)$, en contournant la normalisation dérivation constante et problèmes numériques ultérieurs. En tant que sous-produit, nous obtenons ainsi évaluations de $\log \gamma_k(\theta|\mathcal{X}_k)$ pour certaines valeurs de paramètres $\{\vartheta_1^k, \vartheta_1^k, \cdots, \vartheta_{T_k}^k\}$, qui sont les quantités proposées par les algorithmes MCMC. Ainsi, appeler dans une forêt aléatoire ou une autre régression algorithme d'apprentissage automatique sur l'ensemble d'apprentissage

$$\left\{ \left(\vartheta_1^k, \log \gamma_k(\vartheta_1^k|\mathcal{X}_k) \right), \left(\vartheta_2^k, \log \gamma_k(\vartheta_2^k|\mathcal{X}_k) \right), \cdots, \left(\vartheta_{T_k}^k, \log \gamma_k(\vartheta_{T_k}^k|\mathcal{X}_k) \right) \right\}$$

fournit un estimateur, f_k, de la valeur des log-vraisemblances partielles non-normalisées, $\log\{\gamma_k(\theta|\mathcal{X}_k)\}$. Ces estimateurs f_k, $k = 1, \cdots, K$ peuvent ensuite être exploités de deux façons. Premièrement, ils fournissent une approximation de $\pi(\theta|\mathcal{X})$ par

$$f(\theta) = \exp\left\{ \sum_{k=1}^{K} f_k(\theta) \right\}.$$

Cette approximation peut alors être utilisée comme un substitut pour exécuter un nouvel algorithme MCMC ciblant f, afin de dériver un exemple de paramètre approximativement généré à partir de la distribution postérieure. Deuxiùmement, l'approximation de $\pi(\theta|\mathcal{X})$ by $f(\theta)$ peut aussi être recyclée en une étape d'échantillonnage d'importance en utilisant des simulations de $\gamma_k^{\lambda_k}(\theta|\mathcal{X}_k)$ et cible f.

Remarque 1: (Facteurs d'échelle) Le facteur d'échelle λ_k est utilisé pour contrôler l'incertitude supplémentaire résultant de l'utilisation de sous-balises. Informellement, ce terme contrôle la plage de la région sur laquelle l'algorithme de régression intégré va apprendre chaque sous-ordre. Lorsque le facteur d'échelle devient trop grand, par exemple, quand $\lambda_k = 100K$, $k = 1, \cdots, K$, chaque subposterior mis à l'échelle est doté de peu d'incertitude, ce qui peut conduire à ce que leurs échantillons MCMC respectifs ne se chevauchent pas. Lorsque cette situation se produit, le

approximations f_k ne peut pas fournir d'informations sur les régions où $\gamma_j(\theta|\mathcal{X}_j)$ est élevé pour $j \neq k$ et par conséquent ni le méthode MCMC supplémentaire ni l'importance de la version d'échantillonnage. Dans la situation inverse, à savoir, lorsque le facteur d'échelle est trop petit, par exemple $\lambda_k = 0.001$, $k = 1, \cdots, K$, l'approche nécessite plus de paires de paramètres et correspondant subposteriors pour être en mesure de former une bonne approximation, même si une partie ou la totalité les sous-colonnes se chevauchent. Dans les cas où nous pouvons obtenir des approximations adéquates et bon marché des moyens et des covariances à la fois le vrai postérieur et les subposteriors (où $\lambda_k = 1, k = 1, \cdots, K$), on peut choisir λ_k tel que suffisamment région de probabilité postérieure élevée est chargée par le subposterior pi_k. Plus précisément, nous procédons comme suit dans le cas unidimensionnel ($d = 1$): prenons $\hat{\theta}$, $\hat{\theta}_k$, $\hat{\sigma}$ et $\hat{\sigma}_k$ comme estimations du moyennes et écarts-types du vrai postérieur et du sub-postérieur π_k, respectivement. Nous dénotons

$$\delta_k = \max\{|\hat{\theta}_k - \hat{\theta} - 2\hat{\sigma}|, |\hat{\theta}_k - \hat{\theta} + 2\hat{\sigma}|\}$$

et nous choisissons $\lambda_k = (\delta_k/\hat{\sigma}_k)^{-2}$. Par l'inégalité de Markov, $\pi_k^{\lambda_k}$ couvre la région qui a une grande probabilité sous le vrai postérieur. Dans les cas multidimensionnels, c'est-à-dire lorsque $d > 1$, nous sélectionnons les facteurs d'échelle comme minimaux à travers les composants marginaux. Si de telles estimations ne sont pas bon marché disponibles, nous choisissons plutôt les facteurs d'échelle en exécutant notre méthode plusieurs fois sur des facteurs d'échelle prédéfinis, comme $\lambda_k = 0.1, 0.5, 1, 2, 5, 10$ et choisissez ceux qui amènent les sous-colonnes à se chevaucher.

Remarque 2: (Algorithmes de régression) Considérant que les forêts aléatoires bénéficient des caractéristiques attrayantes d'un mise en œuvre, une forte capacité d'apprentissage des relations non-linéaires, et robustesse, nous faisons le choix d'appliquer cette méthode pour construire notre aléatoire forêt, nous l'appliquons pour apprendre les approximations de la densité subposterior les fonctions. Il va sans dire que d'autres algorithmes de régression sont concurrents disponibles et potentiels aux forêts aléatoires. Cependant, par rapport à la forêt aléatoire, ces autres apprentissage de la machine techniques, comme les machines à vecteurs de support et les réseaux de neurones, nécessitent hyper-paramètres supplémentaires à régler. En outre, les capacités de prédiction des forêts aléatoires sont évolutives, c'est-à-dire, étant donné une formation ensemble de taille T, le coût de la prédiction de la sortie d'une nouvelle entrée est de l'ordre $\mathcal{O}(\log(T))$. Quand nous appelons ces approximations des subposteriors f_k pour exécuter un MCMC supplémentaire, l'évaluation d'une nouvelle proposition a un coût de $\mathcal{O}(K \log(T))$, ce qui correspond à un coût de $\mathcal{O}(KT)$ pour les algorithmes basés sur la mémoire, comme la régression linéaire ou polynomiale locale, la densité de noyau non paramétrique estimation, et les processus gaussiens.

Remarque 3: (Combinaison d'étapes d'échantillonnage d'importance) Pour notre deuxième méthode, nous utilisons l'échantillonnage d'importance sur chaque subposterior et combiner les échantillons résultants à approcher le vrai posterior, sans besoin d'un cycle MCMC supplémentaire. Cependant, considérant que les subposteriors sont répartis plus loin que le vrai postérieur, une grande partie de ces échantillons ont des poids extrêmement faibles et besoin d'être écarté pour conduire à une meilleure approximation du vrai postérieur. Plus précisément, supposons

$\left\{\theta_1^k, \cdots, \theta_T^k\right\}$ sont l'échantillon de $\pi_k^{\lambda_k}$, avec des poids $(t = 1, \cdots, T)$

$$w_t^k \propto \exp\left\{\sum_{j=1}^{K} f_j(\theta_t^k) - \lambda_k f_k(\theta_t^k)\right\}$$

Soit σ une permutation de $\{1, \cdots, T\}$ telle que $w_{\sigma(1)}^k \geq w_{\sigma(2)}^k \geq \cdots \geq w_{\sigma(T)}^k$. Après avoir sélectionné une probabilité de troncation p, nous choisissons alors $i_k = \min\{i : \sum_{\ell=1}^{i} w_{\sigma(\ell)}^k \geq p\}$ et considérons plutôt $\tilde{\pi}_k = \sum_{\ell=1}^{i_k} \tilde{w}_\ell^k \delta_{\theta_{\sigma\ell}^k}$, où $\tilde{w}_\ell^k = w_\ell^k / \sum_{r=1}^{i_k} w_r^k$. Par défaut, nous choisissez p dans $0.99, 0.999, 0.9999$. Considérant que les i_k s peuvent différer beaucoup entre les échantillons en vertu de Par simple stochasticité, certains $\tilde{\pi}_k$ vont profiter de plus ou de beaucoup plus d'atomes que d'autres. Nous pondérons donc les approximations $\tilde{\pi}_k$ proportionnellement à la taille effective de l'échantillon (ESS) (Liu, 2008), c'est-à-dire que nous approchons le vrai postérieur par

$$\hat{\pi} \propto \sum_{k=1}^{K} ESS_k \tilde{\pi}_k, \qquad ESS_k = i_k / \{1 + \mathbb{V}_k\}$$

où $\mathbb{V}_k$ est la variance de $\{i_k w_{\sigma(1)}^k, \cdots, i_k w_{\sigma(i_k)}^k\}$.

Remarque 4: (Calcul complexité) Le budget informatique de notre approche est composé de trois composants

- Au stade de la division et de la conquête, le coût de calcul est $\mathcal{O}(T_k N/K)$ sur chaque sous-échantillon et nous générons un total de points d'échantillons T, où T peut différer de T_k selon les techniques de rodage et d'amincissement du MCMC.
- À l'étape de formation à la régression, le coût de chaque forêt aléatoire est $\mathcal{O}(T_k \log T_k)$
- Au stade de la combinaison
  1. Le coût d'un MCMC supplémentaire est $\mathcal{O}(K \log T_k)$ par point proposé
  2. le coût de l'échantillonnage d'importance est $\mathcal{O}(KT \log T_k)$ pour pondérer tous les échantillons sur tous les sous-balises.

Chapitre 6

En raison de l'afflux massif de données, la puissance des algorithmes MCMC traditionnels est inhibée pour l'inférence bayésienne, car les algorithmes MCMC sont difficiles à mettre à l'échelle. En effet, les algorithmes MCMC, tels que les algorithmes Metropolis-Hastings (MH) (Robert (2004)), requièrent à chaque itération de balayer l'ensemble des données, ce qui est très coûteux sur les grands ensembles de données. Afin de surmonter cette lacune et de sauver les algorithmes MCMC pour le big data, beaucoup d'efforts ont été consacrés au cours des dernières années pour développer des algorithmes MCMC évolutifs. Ces approches peuvent être classées en deux classes (Angelino et al. (2016), Bardenet et al. (2017)): approches diviser-et-conquérir (Gelman et al. (2014), Minsker et al. (2014b), Neiswanger et al. (2013b), Srivastava et al. (2015b), Scott et al. (2016b), Wang and Dunson (2013b)) et des approches de sous-échantillonnage (Bardenet et al. (2014), Chen et al. (2014b), Korattikara et al. (2014), Welling and Teh (2011b)). Dans cet article, nous proposons une

nouvelle méthode appartenant à la catégorie divide-and-conquer. Plus précisément, nous divisons l'ensemble de données en lots et répétons chaque lot un certain nombre de fois, exécutons MCMC sur des lots répétés, recentrons tous les subposteriors ainsi obtenus et prenons leur moyenne comme une approximation du vrai postérieur.

nous étendons les algorithmes MCMC parallèles traditionnels dans trois directions. Tout d'abord, nous mettons à l'échelle chaque probabilité du subposterior avec un facteur tel qu'il pourrait être considéré comme une approximation de la vraisemblance vraie, par laquelle nous voulons dire chaque matrice de covariance subposterior dans la même échelle avec celle du vrai postérieur. Deuxièmement, notre méthode de combinaison est assez simple, a de solides justifications mathématiques et est efficace. Troisièmement, même si notre méthode est justifiée dans un cadre paramétrique, elle peut s'étendre à Bayesian non-paramétrique sans modification.

Soit $\mathcal{X} = \{X_1, \cdots, X_N\}$ l'ensemble de données et supposons que X_i soit i.i.d. observations d'une distribution commune P_θ possédant une densité $f(x|\theta)$ où $\theta \in \Theta$, un ensemble ouvert de $\mathbb{R}^d$. Nous fixons $\theta_0 \in \Theta$, ce qui peut être considéré comme la "vraie valeur" du paramètre. Supposons que l'ensemble de données $\mathcal{X}$ soit divisé en K sous-ensembles $\mathcal{X}_1, \cdots, \mathcal{X}_K$ avec la même taille $M = N/K$. Dénoter

$$\ell(\theta, x) = \log f(x|\theta)$$

$$\ell_i(\theta) = \sum_{j=1}^{M} \log f(x_{ij}|\theta)$$

$$L_N(\theta) = \sum_{i=1}^{K}\sum_{j=1}^{M} \log f(x_{ij}|\theta) = \sum_{i=1}^{K} \ell_i(\theta)$$

$$\hat{\theta}_i = \arg\max_{\theta\in\Theta} \ell_i(\theta), \quad \hat{\theta} = \arg\max_{\theta\in\Theta} L_N(\theta), \quad \bar{\theta} = \frac{1}{K}\sum_{i=1}^{K} \hat{\theta}_i$$

pour chaque $i \in \{1, \cdots, K\}$, $\mathcal{X}_i = \{x_{ij}\}_{j=1}^{M}$. Dans les approches parallèles classiques, on décompose la postérieure globale en un produit de subposteriors:

$$\pi(\theta|\mathcal{X}) \propto \prod_{i=1}^{K} \left(\pi(\theta)^{1/K} \exp(\ell_i(\theta))\right)$$

Même si cette décomposition est correcte mathématiquement, elle n'est pas raisonnable dans les statistiques. Dans l'analyse bayésienne, le type de prieur ne devrait pas changer avec la taille de l'ensemble de données. Par conséquent, l'utilisation d'un préalable qui dépend de la taille de l'observation n'est pas appropriée. Afin de surmonter cette lacune, nous pouvons créer un ensemble de données artificiel poéur chaque sous-ensemble, qui ne fait que répéter chaque point de données K fois pour chaque sous-ensemble. Par conséquent, nous pouvons appliquer l'avant global sur ces ensembles de données artificiels. C'est-à-dire que nous considérons les subposteriors rééchelonnés suivants comme des approximations de la face postérieure globale:

$$\pi_i(\theta|\mathcal{X}_i) \propto \exp\{K\ell_i(\theta)\}\pi(\theta)$$

Cette idée est également apparue dans Minsker et al. (2014b), Srivastava et al. (2015b). Dénoter

$$\theta_i^* = \mathbb{E}_{\pi_i}(\theta), \quad \bar{\theta}^* = \frac{1}{K}\sum_{i=1}^{K} \theta_i^*$$

Dans ces approximations, le facteur K redimensionne la variance de chaque sous-ensemble postérieur $\pi_i(\theta|\mathcal{X}_i)$ pour être approximativement du même ordre que celui de l'ensemble postérieur $\pi(\theta|\mathcal{X}) \propto \exp(L_N(\theta))\pi(\theta)$. Inspirés par ce phénomène, nous recentrons chaque sous-ensemble postérieur à leur moyenne commune, puis les calculons en fonction de la véritable postérieure. C'est-à-dire que le postérieur global $\pi(\theta|\mathcal{X})$ est approché par

$$\frac{1}{K}\sum_{i=1}^{K}\pi_i(\theta - \bar{\theta}^* + \theta_i^*|\mathcal{X}_i)$$

Afin de procéder à l'analyse théorique, nous faisons quelques suppositions légères

Algorithm 0.6 Average of Recentered Subposterior

Input: K subsets of data $\mathcal{X}_1, \cdots, \mathcal{X}_K$, each with size M.
Output: Samples to approximate the true posterior.
for $i = 1, \cdots, K$ (in parallel) **do**
  for $t = 1, \cdots, T$ **do**
    Draw θ_i^t from $\pi_i(\theta|\mathcal{X}_i)$ via MCMC.
  end for
  Calculate $\theta_i^* = \frac{1}{T}\sum_{t=1}^{T}\theta_i^t$.
end for
Calculate $\bar{\theta}^* = \frac{1}{K}\sum_{i=1}^{K}\theta_i^*$
for $i = 1, \cdots, K$ (in parallel) **do**
  for $t = 1, \cdots, T$ **do**
    $\theta_i^t \leftarrow \theta_i^t - \theta_i^* + \bar{\theta}^*$.
  end for
end for
Result: $\{\theta_i^t | i = 1, \cdots; K, t = 1, \cdots, T\}$ approximates the overall posterior.

sur la vraisemblance et le $\pi(\theta)$ antérieur. Ces hypothèses sont standard pour le théorème de Bernstein-von Mises (Ghosh et al. (2007)).

Hypothèse 1: L'ensemble de support $\{x : f(x|\theta) > 0\}$ est le même pour tout $\theta \in \Theta$.

Hypothèse 2: $\ell(\theta, x)$ est trois fois différentiable par rapport à θ dans un voisinage $\{\theta : ||\theta - \theta_0|| \leq \delta_0\}$ of θ_0. L'espérance de $\mathbb{E}_{\theta_0}\nabla\ell(\theta_0, X_1)$ et $\mathbb{E}_{\theta_0}\nabla^2\ell(\theta_0, X_1)$ sont à la fois finis et pour tout x and $p, q, r \in \{1, \cdots, d\}$,

$$\sup_{\theta:||\theta-\theta_0||\leq\delta_0}\frac{\partial^3}{\partial\theta_p\partial\theta_q\partial\theta_r}\ell(\theta, x) \leq M(x) \quad \text{and} \quad \mathbb{E}_{\theta_0}M(X_1) < \infty$$

Hypothèse 3: L'échange de l'ordre d'intégration par rapport à P_{θ_0}et la différenciation à θ_0 est justifié, de sorte que

$$\mathbb{E}_{\theta_0}\nabla\ell(\theta_0, X_1) = 0 \quad \text{and} \quad \mathbb{E}_{\theta_0}\nabla^2\ell(\theta_0, X_1) = -\mathbb{E}_{\theta_0}\nabla\ell(\theta_0, X_1)[\nabla\ell(\theta_0, X_1)]^T$$

Aussi l'information de Fisher $I(\theta_0) = \mathbb{E}_{\theta_0}\nabla\ell(\theta_0, X_1)[\nabla\ell(\theta_0, X_1)]^T$ est positive définitivement.

Hypothèse 4: Pour tout $\delta > 0$, il existe un $\epsilon > 0$, avec P_{θ_0}-probabilité un, tel que

$$\sup_{\theta:||\theta-\theta_0||>\delta}\frac{1}{N}(L_N(\theta) - L_N(\theta_0)) < -\epsilon$$

for all sufficiently large N.

Theorem 1: Si les hypothèses 1 -4 sont vérifiées, alors comme $N \to \infty$ et $M \to \infty$,

$$\left| \frac{1}{K} \sum_{i=1}^{K} \pi_i(\theta - \bar{\theta} + \hat{\theta}_i | \mathcal{X}_i) - \pi(\theta|\mathcal{X}) \right|_{TV} \to 0$$

$$\left| \frac{1}{K} \sum_{i=1}^{K} \pi_i(\theta - \bar{\theta}^* + \theta_i^* | \mathcal{X}_i) - \pi(\theta|\mathcal{X}) \right|_{TV} \to 0$$

Remarque 1: For the center $\bar{\theta}^*$, we have $\bar{\theta}^* - \hat{\theta} = \mathcal{O}_P(\frac{1}{\sqrt{N}})$. In order to improve its performance, we can resort to the Newton-Raphson method Kaw et al. (2008). Under mild conditions, Newton-Raphson method can converge quadratically. Given its impact, the Newton-Raphon method only needs to be called for a few steps.

Remarque 2: Dans le lemme 1, nous pouvons remplacer $I(\theta_0)$ par $I_i(\hat{\theta}_i)$. Alors $\sqrt{KM}(\theta - \hat{\theta}_i)$ a la même distribution de limite avec $\mathcal{N}(0, I^{-1}(\hat{\theta}_i))$ comme $M \to \infty$, où, $\theta \sim \pi_i(\theta|\mathcal{X}_i)$. En tant que tel, si $\theta \sim \frac{1}{K} \sum_{i=1}^{K} \pi_i(\theta - \bar{\theta} + \hat{\theta}_i | \mathcal{X}_i)$,

$$\mathrm{Var}(\sqrt{N}(\theta - \bar{\theta})) \asymp \frac{1}{K} \sum_{i=1}^{K} I_i^{-1}(\hat{\theta}_i)$$

Parce que, pour chaque $i = 1, \cdots, K$, $I_i(\hat{\theta}_i) - I(\theta_0) \to \mathcal{N}(0, \frac{1}{M}\Sigma_1)$, en conséquence, avec la méthode Delta, nous obtenons

$$\sqrt{N} \left(\frac{1}{K} \sum_{i=1}^{K} I_i^{-1}(\hat{\theta}_i) - I^{-1}(\theta_0) \right) \to \mathcal{N}(0, \Sigma_1)$$

Cela signifie que la matrice de covariance de notre échantillon converge vers la vraie avec $\mathcal{O}_P(\frac{1}{\sqrt{N}})$.

Remarque 3: M est un paramètre spécifié par l'utilisateur, qui détermine l'écart entre $\bar{\theta}$ et $\hat{\theta}$. En fait, M n'est pas nécessaire $\mathcal{O}(N^\gamma)$, à condition que chaque $\bar{\theta}_i$ soit une approximation raisonnable de θ_0, ce qui signifie que CLT fonctionne bien pour M.

Bibliography

Andrieu, C. and Roberts, G. (2009). The pseudo-marginal approach for efficient Monte Carlo computations. *The Annals of Statistics*, pages 697–725.

Angelino, E., Johnson, M. J., Adams, R. P., et al. (2016). Patterns of scalable bayesian inference. *Foundations and Trends® in Machine Learning*, 9(2-3):119–247.

Aslett, L. J., Esperança, P. M., and Holmes, C. C. (2015). A review of homomorphic encryption and software tools for encrypted statistical machine learning. *arXiv preprint arXiv:1508.06574*.

Bardenet, R., Doucet, A., and Holmes, C. (2014). Towards scaling up markov chain monte carlo: an adaptive subsampling approach. In *International Conference on Machine Learning*, pages 405–413.

Bardenet, R., Doucet, A., and Holmes, C. (2015). On Markov chain Monte Carlo methods for tall data. *arXiv preprint arXiv:1505.02827*.

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.

Bernton, E., Yang, S., Chen, Y., Shephard, N., and Liu, J. S. (2015). Locally weighted markov chain monte carlo. *arXiv preprint arXiv:1506.08852*.

Bierkens, J. (2016). Non-reversible Metropolis–Hastings. *Statistics and Computing*, 26(6):1213–1228.

Bierkens, J., Bouchard-Côté, A., Doucet, A., Duncan, A. B., Fearnhead, P., Roberts, G., and Vollmer, S. J. (2017a). Piecewise deterministic Markov processes for scalable Monte Carlo on restricted domains. *arXiv preprint arXiv:1701.04244*.

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*.

Bierkens, J., Roberts, G., et al. (2017b). A piecewise deterministic scaling limit of lifted metropolis–hastings in the curie–weiss model. *The Annals of Applied Probability*, 27(2):846–882.

Bouchard-Côté, A., Vollmer, S. J., and Doucet, A. (2017). The bouncy particle sampler: A non-reversible rejection-free Markov chain Monte Carlo method. *Journal of the American Statistical Association*, (to appear).

Breiman, L. (2001). Random forests. *Machine learning*, 45(1):5–32.

Cappé, O. and Robert, C. (2000). Ten years and still running! *J. American Statist. Assoc.*, 95(4):1282–1286.

Chen, T., Fox, E., and Guestrin, C. (2014a). Stochastic gradient hamiltonian monte carlo. pages 1683–1691.

Chen, T., Fox, E., and Guestrin, C. (2014b). Stochastic gradient hamiltonian monte carlo. In *International Conference on Machine Learning*, pages 1683–1691.

Chen, T. and Hwang, C. (2013). Accelerating reversible Markov chains. *Statistics & Probability Letters*, 83(9):1956–1962.

Christian, P. R. and Casella, G. (2004). Monte carlo statistical methods. *Springer New York*.

Davis, M. H. (1984). Piecewise-deterministic Markov processes: A general class of non-diffusion stochastic models. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 353–388.

Davis, M. H. (1993). *Markov Models & Optimization*, volume 49. CRC Press.

Deligiannidis, G., Bouchard-Côté, A., and Doucet, A. (2017). Exponential ergodicity of the bouncy particle sampler. *arXiv preprint arXiv:1705.04579*.

Ding, N., Fang, Y., Babbush, R., Chen, C., Skeel, R. D., and Neven, H. (2014). Bayesian sampling using stochastic gradient thermostats. pages 3203–3211.

Duane, S., Kennedy, A. D., Pendleton, B. J., and Roweth, D. (1987a). Hybrid Monte Carlo. *Phys. Lett. B*, 195:216–222.

Duane, S., Kennedy, A. D., Pendleton, B. J., and Roweth, D. (1987b). Hybrid monte carlo. *Physics letters B*, 195(2):216–222.

Fearnhead, P., Bierkens, J., Pollock, M., and Roberts, G. (2016). Piecewise deterministic Markov processes for continuous-time Monte Carlo. *arXiv preprint arXiv:1611.07873*.

Fontbona, J., Guérin, H., and Malrieu, F. (2016). Long time behavior of telegraph processes under convex potentials. *Stochastic Processes and their Applications*, 126(10):3077–3101.

Galbraith, N. (2016). On event-chain monte carlo methods.

Gelfand, A. and Smith, A. (1990). Sampling based approaches to calculating marginal densities. *J. American Statist. Assoc.*, 85:398–409.

Gelman, A., Vehtari, A., Jylänki, P., Robert, C., Chopin, N., and Cunningham, J. P. (2014). Expectation propagation as a way of life. *arXiv preprint arXiv:1412.4869*.

Ghosh, J. K., Delampady, M., and Samanta, T. (2007). *An introduction to Bayesian analysis: theory and methods*. Springer Science & Business Media.

Glynn, P. W. and Haas, P. J. (2006). Laws of large numbers and functional central limit theorems for generalized semi-markov processes. *Stochastic Models*, 22(2):201–231.

Harland, J., Michel, M., Kampmann, T. A., and Kierfeld, J. (2017). Event-chain monte carlo algorithms for three-and many-particle interactions. *EPL (Europhysics Letters)*, 117(3):30001.

Hastings, W. K. (1970). Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1):97–109.

Horowitz, A. M. (1991). A generalized guided monte carlo algorithm. *Physics Letters B*, 268(2):247–252.

Hwang, C.-R., Hwang-Ma, S.-Y., and Sheu, S.-J. (1993). Accelerating gaussian diffusions. *The Annals of Applied Probability*, pages 897–913.

Kaw, A., Kalu, E., and Ngyen, D. (2008). Numerical methods with applications.

Kingman, J. F. C. (1992). *Poisson processes*, volume 3. Clarendon Press.

Korattikara, A., Chen, Y., and Welling, M. (2014). Austerity in mcmc land: Cutting the metropolis-hastings budget. In *International Conference on Machine Learning*, pages 181–189.

Liu, J. S. (2008). *Monte Carlo strategies in scientific computing*. Springer Science & Business Media.

MacKay, D. J. C. (2002). *Information Theory, Inference & Learning Algorithms*. Cambridge University Press, Cambridge, UK.

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.

Meyn, S. and Tweedie, R. (1993). *Markov Chains and Stochastic Stability*. Springer-Verlag, New York.

Michel, M., Kapfer, S. C., and Krauth, W. (2014). Generalized event-chain monte carlo: Constructing rejection-free global-balance algorithms from infinitesimal steps. *The Journal of chemical physics*, 140(5):054116.

Michel, M. and Sénécal, S. (2017). Forward event-chain monte carlo: a general rejection-free and irreversible markov chain simulation method. *arXiv preprint arXiv:1702.08397*.

Minsker, S., Srivastava, S., Lin, L., and Dunson, D. (2014a). Scalable and robust Bayesian inference via the median posterior. pages 1656–1664.

Minsker, S., Srivastava, S., Lin, L., and Dunson, D. (2014b). Scalable and robust bayesian inference via the median posterior. In *International Conference on Machine Learning*, pages 1656–1664.

Minsker, S., Srivastava, S., Lin, L., and Dunson, D. B. (2014c). Robust and scalable bayes via a median of subset posterior measures. *arXiv preprint arXiv:1403.2660*.

Neal, R. (1999). *Bayesian Learning for Neural Networks*, volume 118. Springer–Verlag, New York. Lecture Notes.

Neal, R. (2011). MCMC using Hamiltonian dynamics. In Brooks, S., Gelman, A., Jones, G. L., and Meng, X.-L., editors, *In Handbook of Markov Chain Monte Carlo*. CRC Press, New York.

Neal, R. M. et al. (2011). Mcmc using hamiltonian dynamics. *Handbook of Markov Chain Monte Carlo*, 2(11).

Neiswanger, W., Wang, C., and Xing, E. (2013a). Asymptotically exact, embarrassingly parallel MCMC. *arXiv preprint arXiv:1311.4780*.

Neiswanger, W., Wang, C., and Xing, E. (2013b). Asymptotically exact, embarrassingly parallel mcmc. *arXiv preprint arXiv:1311.4780*.

Nemeth, C., Sherlock, C., et al. (2017). Merging mcmc subposteriors through gaussian-process approximations. *Bayesian Analysis*.

Nishimura, A. and Dunson, D. (2015). Recycling intermediate steps to improve hamiltonian monte carlo. *arXiv preprint arXiv:1511.06925*.

Pakman, A., Gilboa, D., Carlson, D., and Paninski, L. (2016). Stochastic bouncy particle sampler. *arXiv preprint arXiv:1609.00770*.

Peters, E. A. et al. (2012). Rejection-free Monte Carlo sampling for general potentials. *Physical Review E*, 85(2):026703.

Quiroz, M., Villani, M., and Kohn, R. (2015). Speeding up mcmc by efficient data subsampling.

Quiroz, M., Villani, M., and Kohn, R. (2016). Exact subsampling MCMC. *arXiv preprint arXiv:1603.08232*.

Robert, C. and Casella, G. (2004). *Monte Carlo Statistical Methods*. Springer-Verlag, New York, second edition.

Robert, C. P. (2004). *Monte carlo methods*. Wiley Online Library.

Rubinstein, R. Y. (1981). *Simulation and the Monte Carlo Method*. J. Wiley, New York.

Scott, S. L., Blocker, A. W., Bonassi, F. V., Chipman, H. A., George, E. I., and McCulloch, R. E. (2016a). Bayes and big data: The consensus Monte Carlo algorithm. *International Journal of Management Science and Engineering Management*, 11(2):78–88.

Scott, S. L., Blocker, A. W., Bonassi, F. V., Chipman, H. A., George, E. I., and McCulloch, R. E. (2016b). Bayes and big data: The consensus monte carlo algorithm. *International Journal of Management Science and Engineering Management*, 11(2):78–88.

Sherlock, C. and Thiery, A. H. (2017). A discrete bouncy particle sampler. *arXiv preprint arXiv:1707.05200*.

Srivastava, S., Cevher, V., Dinh, Q., and Dunson, D. (2015a). Wasp: Scalable Bayes via barycenters of subset posteriors. pages 912–920.

Srivastava, S., Cevher, V., Dinh, Q., and Dunson, D. (2015b). Wasp: Scalable bayes via barycenters of subset posteriors. In *Artificial Intelligence and Statistics*, pages 912–920.

Sun, Y., Schmidhuber, J., and Gomez, F. J. (2010). Improving the asymptotic performance of Markov chain Monte-carlo by inserting vortices. pages 2235–2243.

Vanetti, P., Bouchard-Côté, A., Deligiannidis, G., and Doucet, A. (2017). Piecewise deterministic Markov chain Monte Carlo. *arXiv preprint arXiv:1707.05296*.

Wang, X. and Dunson, D. (2013a). Parallelizing MCMC via weierstrass sampler. *arXiv preprint arXiv:1312.4605*.

Wang, X. and Dunson, D. B. (2013b). Parallelizing mcmc via weierstrass sampler. *arXiv preprint arXiv:1312.4605*.

Wang, X., Guo, F., Heller, K., and Dunson, D. (2015). Parallelizing MCMC with random partition trees. *Advances in Neural Information Processing Systems*, pages 451–459.

Welling, M. and Teh, Y. (2011a). Bayesian learning via stochastic gradient Langevin dynamics. pages 681–688.

Welling, M. and Teh, Y. W. (2011b). Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 681–688.

Introduction

Partially based on "Accelerating MCMC Algorithms", joint work with Christian P. Robert, Víctor Elvira and Nick Tawn.

In this thesis we study some approaches to accelerate Markov chain Monte Carlo (MCMC) algorithms, especially for parametric Bayesian with massive observations. This chapter is devoted to providing the necessary background for the following chapters of this manuscript.

1.1 Bayesian statistics

In statistical analysis, the statistician interpret the observations, $(X_{1:n}) \subset \mathcal{X}$, with a model which belongs to a class of probability distributions $\mathcal{P} = \{P_\theta, \theta \in \Theta\}$ over the sample space $(\mathcal{X}, \mathcal{A})$, where θ is called the model parameter and Θ is an arbitrary set. In this thesis, we focus on parametric cases – that is, $\Theta \subset \mathbb{R}^d$, and suppose the distributions to be dominated by some measure $\mu(dx)$. In both frequentist statistics and Bayesian statistics, the likelihood plays a crucial role in inferences, which describes the plausibility of the parameter values, given the observed data.

Definition 1.1.1 *The likelihood is defined as the probability of the observations $(X_{1:n})$ conditioned on the parameter θ and is denoted by $\mathcal{L}(X_{1:n}|\theta)$.*

In frequentist statistics, there exists a true, fixed parameter θ_0 such that P_{θ_0} can fit the observations and the main goal of statistical analysis is to construct an estimator, $\hat{\theta}_n$, of the very θ_0 such that $\hat{\theta}_n$ converges to θ_0 (in probability or in distribution), as $n \to \infty$. However, Bayesian statistics regard the parameter θ as a random variable and model θ with a prior distribution that describes our beliefs about the parameter and is independent of the data. As a result, the parameter space is equipped with a probability structure $(\Theta, \mathcal{B}, \pi_0)$ and Bayesian analysis extracts the information about θ by combining the prior beliefs, π_0, and information contained in the observed set of data. In Bayesian paradigm, once given the prior and the likelihood, the information about the parameter is modelled by the posterior distribution, which is defined as follows.

Definition 1.1.2 *The posterior distribution is the probability distribution of the parameter θ, given the observations $(X_{1:n})$, over the parameter space $(\Theta, \mathcal{B})$. According to Bayes' Theorem, it has the following form,*

$$\pi(d\theta|X_{1:n}) = \frac{\mathcal{L}(X_{1:n}|\theta)\pi_0(d\theta)}{\int_\Theta \mathcal{L}(X_{1:n}|\theta')\pi_0(d\theta')}$$

The integral, $\int_\Theta \mathcal{L}(X_{1:n}|\theta')\pi_0(d\theta')$, in the denominator is called the evidence, or the marginal likelihood and is denoted by $m_{\pi_0}(X_{1:n})$.

Compared with frequentist statistics, Bayesian methods explicitly use probability distributions as a way to quantify uncertainties about the unknown quantities. All inferences in Bayesian methods are exact and not approximated and the results are interpretable so that we can easily answer any scientific questions directly. However, in practice, problems arise if the posterior has no closed form. For example, the evidence may be not explicitly available. At the early era of Bayesian statistics, Bayesian analysis is confined to the problems where the posteriors are explicitly available, such as conjugate priors.

Definition 1.1.3 *If the posterior distribution $\pi(\theta|X_{1:n})$ is in the same probability distribution family as the prior π_0, the prior and posterior are then called conjugate distributions, and the prior is called a conjugate prior for the likelihood function $\mathcal{L}(X_{1:n}|\theta)$.*

Even though conjugacy is definitely useful in many applications, it is too restrictive and not a universal solution for general problems.

The approaches to overcome this restriction can be classified into two main groups: the approximation methods and the Monte Carlo methods. The approximation methods, such as expectation propagation and variational Bayes, project the posterior of interest into a tractable family of probability and approximate it with a closest one. Unless the posterior is in the chosen family, there is intrinsic gap between the approximation and the posterior of interest. The Monte Carlo methods, which are also the focus of this thesis, have different behaviour and can construct estimators to approximate the distributions or quantities of interest in any desired precision as the computation effort growing infinity.

1.2 Monte Carlo

Monte Carlo is based on the law of large numbers (LLN) to approximate the integrals of interest. Suppose we are interested in computing the integrals of the form

$$I_h := \mathbb{E}_P(h(X))$$

The LLN says that if $X_1, X_2, \cdots$, is an infinite sequence of independent and identically distributed (i.i.d.) random variables according to the probability distribution P, and if I_h exists, then

$$\frac{1}{N}\sum_{i=1}^{N} h(X_i) \xrightarrow[N\to\infty]{P} I_h$$

Based on an additional assumption that $\sigma^2 := \mathbb{E}_P(h^2(X)) - I_h^2 < \infty$, the central limit theorem (CLT) gives a stronger result,

$$\sqrt{N}\left(\sum_{i=1}^{N} h(X_i) - I_h\right) \xrightarrow[N\to\infty]{\mathcal{L}} \mathcal{N}(0, \sigma^2)$$

By the CLT, the Monte Carlo estimator converges to I_h at the rate $\mathcal{O}(N^{-1/2})$, regardless of the dimensionality of X_i's.

1.2.1 The Inverse Transform

When the distribution to sample from is one-dimensional with c.d.f F, we can sample $U \sim \mathcal{U}[0,1]$ and compute $X = F^{-1}(U)$, where F^{-1} is the generalized inverse function of F. It is easy to verify that $X \sim F$.

Definition 1.2.1 *For a non-decreasing function F on $\mathbb{R}$, the generalized inverse of F, F^{-1}, is the function defined by*

$$F^{-1}(u) = \inf\{x|F(x) \geq u\}$$

This method is only applicable to one-dimensional distributions and requires deep knowledge about the generalized inverse function F^{-1}, which restricts its applicability in practice. General transformation method generalizes this inverse transform by taking advantage of the linkages of the target distribution with some tractable distributions. Unfortunately, only a few distributions can be expressed as a transformation of other easier distributions. Even worse, the posterior distributions in Bayesian statistics, the main target distributions of this thesis, cannot be sampled from by this method. See (Robert and Casella, 2004) for more details about general transformation method and its examples.

1.2.2 Accept-Reject Sampling

Accept-Reject sampling only requires that the target p of interest is known up to a multiplicative constant and is based on the *Fundamental Theorem of Simulation*, which says that sampling uniformly in the epigraph of a probability density function results in samples marginally distributed according to the distribution.

Theorem 1.2.1 *(Fundamental Theorem of Simulation) Sampling from*

$$X \sim p$$

is equivalent to sample from

$$(X, U) \sim \mathcal{U}(\{(x, u)|0 < u < p(x)\})$$

and marginalise with respect to U.

In light of the simple fact that

$$p(x) = \int_0^{p(x)} du,$$

it is easy to show that if we can sample (X, U) from $\mathcal{F} = \{(x, u)|0 < u < f(x)\}$ uniformly, then the marginal X is distributed to our desired target p. In practice, sampling (X, U) is not always feasible or too expensive, we can sample from an bigger set and discard the pairs which are outside of $\mathcal{F}$ to bypass this difficulty. Generally, such auxiliary set is constructed by the epigraph of Mq, where q is a probability density function, which it is easy to sample from, and M is a known constant such that $p \leq Mq$ on the support of p. Of course the auxiliary distribution q will impact the efficiency of the algorithm.

Algorithm 1.1 Accept-Reject Algorithm - Sample from p

Sample $X \sim q$ and $U \sim \mathcal{U}[0, Mq(X)]$
if $u \leq f(X)$ **then**
  Accept X
else
  Reject X
end if

1.2.3 Importance Sampling

As accept-reject sampling, importance sampling (IS) introduces an auxiliary distribution, Q, and is based on the identity

$$\mathbb{E}_P(h(X)) = \mathbb{E}_Q\left(h(X)w(X)\right)$$

where Q dominates P and is called the *importance distribution*, and $w(x) = \frac{dP}{dQ}(x)$ is called the *weight function*. By the LLN, the integral of interest, I_h, can be approximated by

$$\hat{I}_h^N = \frac{1}{N}\sum_{i=1}^{N} h(X_i)w(X_i), \quad X_i \overset{iid}{\sim} Q$$

The remarkable advantage of importance sampling is that the weight function w can be known up to a multiplicative constant, which is extremely valuable for sampling from posterior in Bayesian inference. In fact, the multiplicative constant can be estimated by $\frac{1}{N}\sum_{i=1}^{N} w(X_i)$ and one can show that the normalized estimator

$$\frac{\sum_{i=1}^{N} h(X_i)w(X_i)}{\sum_{i=1}^{N} w(X_i)}$$

converges to the integral of interest.

Like in accept-reject sampling, the importance distribution Q has large influence of the efficiency of IS. The optimal Q, which minimises the variance of the estimator $\hat{I}_h^N$, not only depends of the target distribution P, but also has something to do with the integrand h. Actually, we always have

$$\mathrm{Var}_Q[h(X)w(X)] \leq \mathrm{Var}_P[h(X)],$$

where Q is optimal. For more details of optimal Q see Robert and Casella (2004).

So far, all the above requires a sequence of i.i.d. proposals and the obtained samples are also independent to each other. In the next section, we describe a class of sampling algorithms, based on Markov chains, which produce correlated samples to approximate the target distribution or the integrals of interest.

1.3 Markov chain Monte Carlo

Markov chain Monte Carlo (MCMC) algorithms have been used for nearly 60 years, becoming a reference method for analysing Bayesian complex models in the early 1990's (Gelfand and Smith, 1990). The strength of this method is that it guarantees convergence to the quantity (or quantities) of interest with minimal requirements on

the targeted distribution (also called *target*) behind such quantities. In that sense, MCMC algorithms are robust or universal, as opposed to the most standard Monte Carlo methods (see, e.g., Robert and Casella, 2004; Rubinstein, 1981) that require direct simulations from the target distribution.

MCMC methods have a history (see, e.g. Cappé and Robert, 2000) that starts at approximately the same time as the Monte Carlo methods, in conjunction with the conception of the first computers. They have been devised to handle the simulation of complex target distributions, when complexity stems from the shape of the target density, the size of the associated data, the dimension of the object to be simulated, or from time requirements. For instance, the target density $p(x)$ may happen to be expressed in terms of multiple integrals that cannot be solved analytically,

$$p(x) = \int \omega(x, \xi) \mathrm{d}\xi \,,$$

which requires the simulation of the entire vector (x, ξ). In cases when ξ is of the same dimension as the data, as for instance in latent variable models, this significant increase in the dimension of the object to be simulated creates computational difficulties for standard Monte Carlo methods, from managing the new target $\omega(x, \xi)$, to devising a new and efficient simulation algorithm. A Markov chain Monte Carlo (MCMC) algorithm allows for an alternative resolution of this computational challenge by simulating a Markov chain that explores the space of interest (and possibly supplementary spaces of auxiliary variables) without requiring a deep preliminary knowledge on the density p, besides the ability to compute $p(x_0)$ for a given parameter value x_0 (if up to a normalising constant) and possibly the gradient $\nabla \log p(x_0)$.

The validation of the method (e.g., Robert and Casella, 2004) is that the Markov chain is *ergodic* (e.g., Meyn and Tweedie, 1993), namely that it converges in distribution to the distribution with density π, no matter where the Markov chain is started at time $t = 0$. As the basic Monte Carlo, MCMC has its corresponding LLN and CLT.

Theorem 1.3.1 *(Ergodic Theorem (Robert and Casella, 2004)) If $(X_n)_{n\geq 0}$ is a positive Harris recurrent Markov chain with invariant measure P, then for every $h \in L_1(P)$, we have*

$$\frac{1}{N} \sum_{i=1}^{N} h(X_i) \xrightarrow[N\to\infty]{} \int h(x) P(dx)$$

Theorem 1.3.2 *(Markov Functional Central Limit Theorem (Robert and Casella, 2004)) If $(X_n)_{n\geq 0}$ is a positive Harris recurrent and irreducible Markov chain, geometrically ergodic with invariant measure P, and if the function h satisfies $\mathbb{E}_P[h(X)] = 0$ and $\mathbb{E}_P[|h(X)|^{2+\epsilon}] < \infty$ for some $\epsilon > 0$, the we have*

$$\frac{1}{N} \sum_{i=1}^{N} h(X_i) \xrightarrow[N\to\infty]{\mathcal{L}} \mathcal{N}(0, \sigma_h^2)$$

for some finite $\sigma_h^2 = \mathbb{E}_P[h(X_0)^2] + 2 \sum_{k=1}^{\infty} \mathbb{E}_P[h(X_0)h(X_k)] < \infty$.

See (Robert and Casella, 2004) for a comprehensive review of the Markov chain theory.

1.3.1 Metropolis-Hastings algorithm

The Metropolis–Hastings algorithm is a generic illustration of the principle of MCMC and is named after N. Metropolis, who first proposed to the algorithm by using symmetrical proposal distribution in Metropolis et al. (1953), and K. Hastings, who extended it to the more general case in Hastings (1970). The basic algorithm is constructed by choosing a *proposal*, that is, a conditional density $q(x'|x)$ (also known as a *Markov kernel*), the Markov chain $\{X_n\}_{n=1}^{\infty}$ being then derived by successive simulations of the transition

$$X_{n+1} = \begin{cases} X' \sim q(X'|X_n) & \text{with probability } \left\{ \dfrac{p(X')}{p(X_n)} \times \dfrac{q(X_n|X')}{q(X'|X_n)} \right\} \wedge 1, \\ X_n & \text{otherwise.} \end{cases}$$

This acceptance-rejection feature of the algorithm makes it appropriate for targeting p as its stationary distribution if the resulting Markov chain $(X_n)_n$ is irreducible, i.e., has a positive probability of visiting any region of the support of p in a finite number of iterations. (Stationarity can easily be shown, e.g., by using the so-called *detailed balance property* that makes the chain time-reversible, see, e.g., Robert and Casella, 2004.) The most widely used Markov kernel in Metropolis-Hastings might

Algorithm 1.2 Metropolis-Hastings algorithm

Input: the starting point X_0, the proposal distribution q and the number of iterations N.
for $n = 1, 2, \cdots, N$ **do**
  Sample $X' \sim q(\cdot|X_{n-1})$
  Compute the acceptance probability $\alpha(X_{n-1}, X')$, where

$$\alpha(X_{n-1}, X') = \min \left\{ 1, \frac{p(X')q(X_{n-1}|X')}{p(X_{n-1})q(X'|X_{n-1})} \right\}$$

  Sample $U \sim \mathcal{U}[0, 1]$;
  if $U < \alpha(X_{n-1}, X')$ **then**
    $X_n \to X'$
  else
    $X_n \to X_{n-1}$
  end if
end for

be the random walk proposals, in which $q(x|y) = q(y|x)$ and the density of proposal cancels in the acceptance ratio. Considering the initial goal of simulating samples from the target distribution p, the performances of MCMC methods like the Metropolis–Hastings algorithm above often vary quite a lot, depending primarily on the adequacy between the proposal q and the target p. For instance, if $q(\cdot|X_n) = p(\cdot)$, the Metropolis–Hastings algorithm reduces to i.i.d. sampling from the target, which is of course a formal option when i.i.d. sampling from p proves impossible. Although there exist rare instances when the Markov chain (X_n) leads to negative correlations between the successive terms of the chain, making it *more efficient* than regular i.i.d. sampling (Liu et al., 1995), the most common occurrence is one of positive correlation between the simulated values (sometimes uniformly, see Liu et al., 1994). This feature implies a reduced efficiency of the algorithm and hence requires a larger number of simulations to achieve the same precision as

an approximation based on i.i.d. simulations (without accounting for differences in computing time). More generally, an MCMC algorithm may require a large number of iterations to escape the attraction of its starting point X_0 and to reach stationarity, to the extent that some versions of such algorithms fail to converge in the time available (i.e., in practice if not in theory).

1.3.2 Hamiltonian Monte Carlo

Hamiltonian (or hybrid) Monte Carlo (HMC) is an auxiliary variable technique that takes advantage of a continuous time Markov process to sample from the target p. This approach comes from physics (Duane et al., 1987) and was popularised in statistics by Neal (1999, 2011) and MacKay (2002). Given a target $p(x)$, where $x \in \mathbb{R}^d$, an artificial auxiliary variable $v \in \mathbb{R}^d$ is introduced along with a density $\varphi(v|x)$ so that the joint distribution of (x, v) enjoys $p(x)$ as its marginal. While there is complete freedom in this representation, the HMC literature often calls v the *momentum* of a particle located at x by analogy with physics. Based on the representation of the joint distribution

$$\rho(x, v) = p(x)\varphi(v|x) \propto \exp\{-H(x, v)\}\,,$$

where $H(\cdot)$ is called the *Hamiltonian*, Hamiltonian Monte Carlo (HMC) is associated with the continuous time process (x_t, v_t) generated by the so-called *Hamiltonian equations*

$$\frac{\mathrm{d}x_t}{\mathrm{d}t} = \frac{\partial H}{\partial v}(x_t, v_t) \qquad \frac{\mathrm{d}v_t}{\mathrm{d}t} = -\frac{\partial H}{\partial x}(x_t, v_t)\,,$$

which keep the Hamiltonian target stable over time, as

$$\frac{\mathrm{d}H(x_t, v_t)}{\mathrm{d}t} = \frac{\partial H}{\partial v}(x_t, v_t)\,\frac{\mathrm{d}v_t}{\mathrm{d}t} + \frac{\partial H}{\partial x}(x_t, v_t)\,\frac{\mathrm{d}x_t}{\mathrm{d}t} = 0\,.$$

Obviously, the above continuous time Markov process is deterministic and only explores a given level set,

$$\{(x, v) : H(x, v) = H(x_0, v_0)\}\,,$$

instead of the whole augmented state space $\mathbb{R}^{2d}$, which induces an issue with irreducibility. An acceptable solution to this problem is to refresh the momentum, $v_t \sim \varphi(v|x_{t-})$, at random times $\{\tau_n\}_{n=1}^{\infty}$, where x_{t-} denotes the location of x immediately prior to time t, and the random durations $\{\tau_n - \tau_{n-1}\}_{n=2}^{\infty}$ follow an exponential distribution. By construction, continuous-time Hamiltonian Markov chain can be regarded as a specific piecewise deterministic Markov process using Hamiltonian dynamics (Bou-Rabee et al., 2017; Davis, 1984, 1993) and our target, π, is the marginal of its associated invariant distribution.

Before moving to the practical implementation of the concept, let us point out that the free cog in the machinery is the conditional density $\varphi(v|x)$, which is usually chosen as a Gaussian density with either a constant covariance matrix M corresponding to the target covariance or as a local curvature depending on x in Riemannian Hamiltonian Monte Carlo (Girolami and Calderhead, 2011). Betancourt (2017) argues in favour of these two cases against non-Gaussian alternatives and (Livingstone et al., 2017) analyses how different choices of kinetic energy in Hamiltonian Monte

Carlo affect algorithm performance. For a fixed covariance matrix, the Hamilton equations become

$$\frac{\mathrm{d}x_t}{\mathrm{d}t} = M^{-1}v_t \qquad \frac{\mathrm{d}v_t}{\mathrm{d}t} = \nabla \log p(x_t)\,,$$

which is the score function. The velocity (or momentum) of the process is thus driven by this score function, gradient of the log-target.

The above description remains quite conceptual in that there is no generic methodology for producing this continuous time process, since Hamilton equations cannot be solved exactly in most cases. Furthermore, standard numerical solvers like Euler's method create an unstable approximation that induces a bias as the process drifts away from its true trajectory. There exists however a discretisation simulation technique that produces a Markov chain and is well-suited to the Hamiltonian equations in that it preserves the stationary distribution (Betancourt, 2017). It is called the *symplectic integrator*, and one version in the independent case with constant covariance consists in the following (so-called *leapfrog*) steps

$$\begin{aligned} v_{t+\epsilon/2} &= v_t + \epsilon\nabla \log p(x_t)/2, \\ x_{t+\epsilon} &= x_t + \epsilon M^{-1} v_{t+\epsilon/2}, \\ v_{t+\epsilon} &= v_{t+\epsilon/2} + \epsilon\nabla \log p(x_{t+\epsilon})/2, \end{aligned}$$

where ϵ is the time-discretisation step. Using a proposal on v_0 drawn from the Gaussian auxiliary target and deciding on the acceptance of the value of $(x_{T\epsilon}, v_{T\epsilon})$ by a Metropolis–Hastings step can limit the danger of missing the target. Note that the first two leapfrog steps induce a Langevin move on x_t:

$$x_{t+\epsilon} = x_t + \epsilon^2 M^{-1}\nabla \log p(x_t)/2 + \epsilon M^{-1} v_t\,,$$

thus connecting with the MALA algorithm. In practice, it is important to note

Algorithm 1.3 Leapfrog(x_0, v_0, ϵ, L)

Input: the starting position x_0, the starting momentum v_0, the step-size ϵ, the steps L
for $\ell = 0, 1, \cdots, L-1$ **do**
  $v_{\ell+1/2} = v_\ell + \epsilon\nabla \log p(x_\ell)$
  $x_{\ell+1} = x_\ell + \epsilon M^{-1} v_{\ell+1/2}$
  $v_{\ell+1} = v_{\ell+1/2} + \epsilon\nabla \log p(x_{\ell+1})$
end for
Output: (x_L, v_L)

that discretising Hamiltonian dynamics introduces two free parameters, the step size ϵ and the trajectory length T, both to be calibrated. As an empirically successful and popular variant of HMC, the "no-U-turn sampler" (NUTS) of Hoffman and Gelman (2014) adapts the value of ϵ based on primal-dual averaging. It also eliminates the need to choose the trajectory length T via a recursive algorithm that builds a set of candidate proposals for a number of forward and backward leapfrog steps and stops automatically when the simulated path retraces.

1.3.3 Scalable MCMC

The explosion in the collection and analysis of "big" datasets in recent years has brought new challenges to the MCMC algorithms that are used for Bayesian inference. When examining whether or not a new proposed sample is accepted at

Algorithm 1.4 Hamiltonian Monte Carlo algorithm

Input: the step-size ϵ, the steps of leapfrog integrator L, starting position x_0, the desired number of iterations N.
for $n = 1, \cdots, N$ **do**
  Sample $v_{n-1} \sim \varphi(v)$;
  Compute $(x^*, v^*) \leftarrow \text{Leapfrog}(x_{n-1}, v_{n-1}, \epsilon, L)$;
  Compute the acceptance ratio α, where

$$\alpha = \min\left\{1, \frac{\exp(-H(x^*, -v^*))}{\exp(-H(x_{n-1}, v_{n-1}))}\right\};$$

  Sample $u \sim \mathcal{U}[0, 1]$;
  if $u < \alpha$ **then**
    $x_n \leftarrow x^*$
  else
    $x_n \leftarrow x_{n-1}$
  end if
end for

the accept-reject step, an MCMC algorithm such as the Metropolis-Hastings version needs to sweep over the whole data set, at each and every iteration, for the evaluation of the likelihood function. MCMC algorithms are then difficult to scale up, which strongly hinders their application in big data settings. In some cases, the datasets may be too large to fit on a single machine. It may also be that confidentiality measures impose different databases to stand on separate networks, with the possible added burden of encrypted data (Aslett et al., 2015). Communication between the separate machines may prove impossible on an MCMC scale that involves thousands or hundreds of thousands iterations.

In the recent years, efforts have been made to design *scalable* algorithms, namely, solutions that manage to handle large scale targets by breaking the problem into manageable or scalable pieces. Roughly speaking, these methods can be classified into two categories (Bardenet et al., 2017): divide-and-conquer approaches and subsampling approaches.

Divide-and-conquer approaches partition the whole data set, denoted $\mathcal{D}$, into batches, $\{\mathcal{D}_1, \cdots, \mathcal{D}_k\}$, and run separate MCMC algorithms on each data batch, independently, as if they were independent Bayesian inference problems.[1] These methods then combine the simulated parameter outcomes together to approximate the original posterior distribution. Depending on the treatments of the batches selected in the MCMC stages, these approaches can be further subdivided into two finer groups: sub-posterior methods and boosted sub-posterior methods. Sub-posterior methods are motivated by the independent product equation:

$$\pi(\theta|\mathcal{D}) \propto \prod_{i=1}^{k} \left(\pi_0(\theta)^{1/k} \prod_{\ell \in \mathcal{X}_i} p(x_\ell|\theta)\right) = \prod_{i=1}^{k} \pi_i(\theta)\,, \tag{1.1}$$

[1] In order to keep the notations consistent, we still denote the target density by π, with the prior density denoted as π_0 and the sampling distribution of one observation x as $p(x|\theta)$. The dependence on the sample $\mathcal{D}$ is not reported unless necessary.

and they target the densities $\pi_i(\theta)$ (up to a constant) in their respective MCMC steps. They thus bypass communication costs (Scott et al., 2016), by running MCMC samplers independently on each batch, and they most often increase MCMC mixing rates (in effective samples sizes produced by second), given that the sub-posterior distributions $\pi_i(\theta)$ are based on smaller datasets. For instance, Scott et al. (2016) combine the samples from the sub-posteriors, $\pi_i(\theta)$, by a Gaussian reweighting. Neiswanger et al. (2013) estimate the sub-posteriors $\pi_i(\theta)$ by non-parametric and semi-parametric methods, and they run additional MCMC samplers on the product of these estimators towards approximating the true posterior $\pi(\theta)$. Wang and Dunson (2013) refine this product estimator with an additional Weierstrass sampler, while Wang et al. (2015) estimate the posterior by partitioning the space of samples with step functions.

As an alternative to sampling from the sub-posteriors, boosted sub-posterior methods target instead the components

$$\tilde{\pi}_i(\theta) \propto \pi_0(\theta) \left(\prod_{\ell \in \mathcal{X}_i} p(x_\ell|\theta) \right)^k \tag{1.2}$$

in separate MCMC runs. Since they formally amount to repeating each batch k times towards producing pseudo data sets with the same size as the true one, the resulting boosted sub-posteriors, $\tilde{\pi}_1(\theta), \cdots, \tilde{\pi}_k(\theta)$, have the same scale in variance of each component of the parameters, θ, as the true posterior, and can thus be treated as a group of estimators of the true posterior. In the subsequent combining stage, these sub-posteriors are merged together to construct a better approximation of the target distribution. For instance, Minsker et al. (2014) approximate the posterior with the geometric median of the boosted sub-posteriors, embedding them into associated reproducing kernel Hilbert spaces (rkhs), while Srivastava et al. (2015) achieve this goal using the barycentres of $\tilde{\pi}_1, \cdots, \tilde{\pi}_k$, these barycentres being computed with respect to a Wasserstein distance.

In a perspective different from the above parallel scheme of divide-and-conquer approaches, sub-sampling approaches aim at reducing the number of individual datapoint likelihood evaluations operated at each iteration towards accelerating MCMC algorithms. From a general perspective, these approaches can be further classified into two finer classes: exact subsampling methods and approximate subsampling methods, depending on their resulting outputs. Exact subsampling approaches typically require subsets of data of random size at each iteration. One solution to this effect is taking advantage of pseudo-marginal MCMC via constructing unbiased estimators of the target density evaluated on subsets of the data (Andrieu and Roberts, 2009). Quiroz et al. (2016) follow this direction by combining the debiasing technique of Rhee and Glynn (2015) and the correlated pseudo-marginal MCMC approach of Deligiannidis et al. (2015). Another direction is to use piecewise deterministic Markov processes (PDMP) (Davis, 1984, 1993), which enjoy the target distribution as the marginal of their invariant distribution. This PDMP version requires unbiased estimators of the gradients of the logarithm of the likelihood function, instead of the likelihood itself. By using a tight enough bound on the event rate function of the associated Poisson processes PDMP can produce super-efficient scalable MCMC algorithms. The bouncy particle sampler (Bouchard-Côté et al., 2018) and the zig-zag sampler (Bierkens et al., 2016) are two competing PDMP algorithm, while Bierkens et al. (2018) unifies and extends these two methods. Besides, one

should note that PDMP produces a non-reversible Markov chain, which means that the algorithm should be more efficient in terms of mixing rate and asymptotic variance, when compared with reversible MCMC algorithms, such as MH, HMC and MALA, as observed in some theoretical and experimental works (Bierkens, 2016; Chen and Hwang, 2013; Hwang et al., 1993; Sun et al., 2010).

Approximate subsampling approaches aim at constructing an approximation of the target distribution. One direction is to approximate the acceptance probability with high accuracy by using subsets of the data (Bardenet et al., 2014, 2017). Another solution is based on a direct modification of exact methods. The seminal work (Welling and Teh, 2011) in this direction, SGLD, is to exploit the Langevin diffusion

$$\mathrm{d}\theta_t = \frac{1}{2}\mathbf{\Lambda}\nabla \log \pi(\theta_t)\mathrm{d}t + \mathbf{\Lambda}^{1/2}\mathrm{d}\mathbf{B}_t, \quad \theta_0 \in \mathbb{R}^d, t \in [0, \infty) \tag{1.3}$$

where $\mathbf{\Lambda}$ is a user-specified matrix, π is the target distribution and $\mathbf{B}_t$ is a d-dimensional Brownian process. By virtue of the Euler-Maruyama discretisation and using unbiased estimators of the gradient of the log-target density, SGLD and its variants (Chen et al., 2014; Ding et al., 2014) often produce fast and accurate results in practice when compared with MCMC algorithms using MH steps.

1.3.4 Continuous-Time MCMC Sampler

All the above MCMC samplers are based on discrete-time, reversible Markov chains, however, continuous-time, non-reversible MCMC samplers have been drawing the attention of computational statisticians, which are based on piecewise deterministic Markov processes (PDMP). Even though PDMP was proposed as early as 1984 by Davis (1984), its prevalence in statistics for sampling problems began the remarkable applications by (Bierkens et al., 2016; Bouchard-Côté et al., 2018; Peters et al., 2012).

Suppose p be the continuous target distribution over $\mathbb{R}^d$ and for convenience sake, we also use $p(x)$ for the probability density function of p. Like HMC, an auxiliary variable, $v \in \mathcal{V}$ is introduced in PDMP framework and PDMP-based sampler explores the augmented space $\mathbb{R}^d \times \mathcal{V}$, targeting a variable $z = (x, v)$ with distribution $\rho(dx, dv)$ over $\mathbb{R}^d \times \mathcal{V}$ as its invariant distribution. By construction, the distribution ρ enjoys p as its marginal distribution in x. In practice, the distribution of v is often chosen to be independent to x and we denote it $\varphi(v)$. A piecewise deterministic Markov process $z_t = (x_t, v_t)$ consists of three distinct components: its deterministic dynamic between events, an event occurrence rate and a transition dynamic at event time. Specifically,

1. **Deterministic dynamic**: between two events, the Markov process evolves deterministically, according to some ordinary differential equation:

$$\frac{dz_t}{dt} = \Phi(z_t).$$

2. **Event occurrence rate**: an event occurs at time t with rate $\lambda(z_t)$.

3. **Transition dynamic**: At an event time, τ, the state prior to τ is denoted by $z_{\tau-}$, with the new state being generated by $z_\tau \sim Q(\cdot|z_{\tau-})$.

Algorithm 1.5 Simulation of PDMP

Initialize the starting point z_0, $\tau_0 \leftarrow 0$.
for $k = 1, 2, 3, \cdots$ **do**
Sample inter-event time η_k from following distribution

$$\mathbb{P}(\eta_k > t) = \exp\left\{-\int_0^t \lambda(z_{\tau_{k-1}+s})ds\right\}.$$

$\tau_k \leftarrow \tau_{k-1} + \eta_k$, $z_{\tau_{k-1}+s} \leftarrow \Psi_s(z_{\tau_{k-1}})$, for $s \in (0, \eta_k)$, where Ψ is the ODE flow of Φ.
$z_{\tau_k-} \leftarrow \Psi_{\eta_k}(z_{\tau_{k-1}})$, $z_{\tau_k} \sim Q(\cdot|z_{\tau_k-})$.
end for

Here, an "event" refers to an occurrence of a time-inhomogeneous Poisson process with rate $\lambda(\cdot)$ (Kingman, 1992). The powerful tool to analyse PDMP is the extended generator, which is defined as follows.

Definition 1.3.1 *(Davis, 1993) Let $\mathcal{D}(\mathcal{L})$ denote the set of measurable functions $f : \mathcal{Z} \to \mathbb{R}$ with the following property: there exists a measurable function $h : \mathcal{Z} \to \mathbb{R}$ such that the function $t \to h(z_t)$ is integrable P_z-a.s. for each $z \in \mathcal{Z}$ and the process*

$$C_t^f = f(z_t) - f(z_0) - \int_0^t h(z_s)ds$$

is a local martingale. Then we write $h = \mathcal{L}f$ and call $(\mathcal{L}, \mathcal{D}(\mathcal{L}))$ the extended generator of the process $\{z_t\}_{t\geq 0}$.

Actually, we can compute the generator of above PDMP explicitly, by (Davis, 1993, Theorem 26.14).

Theorem 1.3.3 *(Davis, 1993) The generator, $\mathcal{L}$, of above PDMP is, for $f \in \mathcal{D}(\mathcal{L})$*

$$\mathcal{L}f(z) = \nabla f(z) \cdot \Phi(z) + \lambda(z) \int_{z'} [f(z') - f(z)]\, Q(dz'|z).$$

Furthermore, $\rho(dz)$ is an invariant distribution of above PDMP, if

$$\int \mathcal{L}f(z)\rho(dz) = 0, \quad \text{for all } f \in \mathcal{D}(\mathcal{L}).$$

By choosing appropriate deterministic dynamic, rate function and transition dynamic, it is easy to make sure the specific PDMP admit the desired distribution, ρ, as its invariant distribution. However, there are two main difficulties in implementing such PDMP-based MCMC sampler. The first one is the computation of the ODE flow, Ψ. Almost all existing PDMP-based samplers adopt the linear dynamic, which means that

$$\frac{dx_t}{dt} = v_t, \quad \frac{dv_t}{dt} = 0.$$

(Vanetti et al., 2017) uses an approximation, $\hat{p}$, of the target p and adopts the Hamiltonian dynamic of $\hat{p} \times \varphi$ in HMC-BPS. The second difficulty comes from the generation of inter-event time, which corresponds to the first occurrence time of inhomogeneous Poisson process. The comment techniques to overcome such difficulty are based on the following two theorems.

Theorem 1.3.4 *(Superposition Theorem (Kingman, 1992)) Let $\Pi_1, \Pi_2, \cdots$, be a countable collection of independent Poisson processes on state space $\mathbb{R}^+$ and let Π_n have rate $\lambda_n(\cdot)$ for each n. If $\sum_{n=1}^{\infty}\lambda_n(t) < \infty$ for all t, then the superposition*

$$\Pi = \bigcup_{n=1}^{\infty} \Pi_n$$

is a Poisson process with rate

$$\lambda(t) = \sum_{n=1}^{\infty} \lambda_n(t)$$

Theorem 1.3.5 *(Thinning Theorem (Lewis and Shedler, 1979)) Let $\lambda : \mathbb{R}^+ \to \mathbb{R}^+$ and $\Lambda : \mathbb{R}^+ \to \mathbb{R}^+$ be continuous functions such that $\lambda(t) \leq \Lambda(t)$ for all $t \geq 0$. Let $\tau_1, \tau_2, \cdots$, be the increasing finite or infinite sequence of a Poisson process with rate $\Lambda(\cdot)$. For all i, if the point τ_i is removed from the sequence with probability $1-\lambda(t)/\Lambda(t)$, then the remaining points $\tilde{\tau}_1, \tilde{\tau}_2, \cdots$ form a non-homogeneous Poisson process with rate $\lambda(\cdot)$.*

In order to estimate the integral of interest $I_h = \int h(x)P(dx)$, there are two approaches used to construct the estimator by PDMP-based samplers. Given a simulated path $(x_t, v_t)_{t=0}^T$, the first estimator is

$$\tilde{I}_h^T = \frac{1}{T}\int_0^T h(x_t)dt.$$

By discretising the path uniformly with respect to t, we can construct another estimator as

$$\hat{I}_h^T = \frac{1}{N}\sum_{n=1}^{N} h(x_{nT/N}).$$

1.4 Overview

This thesis consists of 6 chapters, we present abstracts of each of the following chapters.

1.4.1 Coordinate Sampler

In Chapter 2, we derive a novel non-reversible, continuous-time Markov chain Monte Carlo (MCMC) sampler, called Coordinate Sampler, based on a piecewise deterministic Markov process (PDMP), which can be seen as a variant of the Zigzag sampler (Bierkens et al. (2016)). In addition to proving a theoretical validation for this new sampling algorithm, we show that the Markov chain it induces exhibits geometrical ergodicity convergence, for distributions whose tails decay at least as fast as an exponential distribution and at most as fast as a Gaussian distribution. Several numerical examples highlight that our coordinate sampler is more efficient than the Zigzag sampler, in terms of effective sample size.

1.4.2 Discrete PDMP using Hamiltonian dynamics

In Hamiltonian Monte Carlo (HMC) algorithms, one refreshes the momentum at each iteration and discards all proposals, expect for the last one, proposed during leapfrog integrators. Such refreshment introduces much randomness in terms of exploring the target space, while discarding the intermediate proposals wastes computation effort. In Chapter 3, we propose a randomness-controlled HMC algorithm, which can overcome above two potential drawbacks and coincides with the recently prevailing piecewise deterministic Markov process sampler.

1.4.3 Generalized Bouncy Particle Sampler

As a special example of piecewise deterministic Markov process, bouncy particle sampler is a rejection-free, irreversible Markov chain Monte Carlo algorithm and can draw samples from target distribution efficiently. In Chapter 4, we generalize bouncy particle sampler in terms of its transition dynamics. In BPS, the transition dynamic at event time is deterministic, but in GBPS, it is random. With the help of this randomness, GBPS can overcome the reducibility problem in BPS without refreshment.

1.4.4 Parallelizing MCMC via Random Forest

For Bayesian analyses of so-called big data models, the divide-and-conquer MCMC approach splits the whole data set into smaller batches, runs MCMC algorithm over each batch to produce parameter samples, and combines these towards producing an approximation of the posterior distribution. In this spirit, we introduce random forests into this method and use each subposterior or partial posterior as a proposal distribution to implement importance sampling in Chapter 5. Unlike the existing divide-and-conquer MCMC solutions, our method is based on scaled subposteriors, whose scale factor is not necessarily restricted to 1 or to the number of batches. Through several experiments, we show that our methods performs satisfactorily against the existing solutions in low-dimensional setting, for both Gaussian and severely non-Gaussian cases, and under model misspecification.

1.4.5 Average of Recentered Parallel MCMC for Big Data

In big data context, traditional MCMC methods, such as Metropolis-Hastings algorithms and hybrid Monte Carlo, scale poorly because of their need to evaluate the likelihood over the whole data set at each iteration. In order to resurrect MCMC methods, numerous approaches belonging to two categories: divide-and-conquer and subsampling, are proposed. In Chapter 6, we study the parallel MCMC and propose a new combination method in the divide-and-conquer framework. Compared with some parallel MCMC methods, such as consensus Monte Carlo, Weierstrass Sampler, instead of sampling from subposteriors, our method runs MCMC on rescaled subposteriors, but share the same computation cost in the parallel stage. We also give the mathematical justification of our method and show its performance in several models. Besides, even though our new methods is proposed in parametric framework, it can been applied to non-parametric cases without difficulty.

Bibliography

Andrieu, C. and Roberts, G. (2009). The pseudo-marginal approach for efficient Monte Carlo computations. *The Annals of Statistics*, pages 697–725.

Aslett, L. J., Esperança, P. M., and Holmes, C. C. (2015). A review of homomorphic encryption and software tools for encrypted statistical machine learning. *arXiv preprint arXiv:1508.06574*.

Bardenet, R., Doucet, A., and Holmes, C. (2014). Towards scaling up Markov chain Monte Carlo: an adaptive subsampling approach. *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 405–413.

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.

Betancourt, M. (2017). A conceptual introduction to Hamiltonian Monte Carlo. *arXiv preprint arXiv:1701.02434*.

Bierkens, J. (2016). Non-reversible Metropolis–Hastings. *Statistics and Computing*, 26(6):1213–1228.

Bierkens, J., Bouchard-Côté, A., Doucet, A., Duncan, A. B., Fearnhead, P., Lienart, T., Roberts, G., and Vollmer, S. J. (2018). Piecewise deterministic Markov processes for scalable Monte Carlo on restricted domains. *Statistics & Probability Letters*, 136:148–154.

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*.

Bou-Rabee, N., Sanz-Serna, J. M., et al. (2017). Randomized Hamiltonian Monte Carlo. *The Annals of Applied Probability*, 27(4):2159–2194.

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.

Cappé, O. and Robert, C. (2000). Ten years and still running! *J. American Statist. Assoc.*, 95(4):1282–1286.

Chen, T., Fox, E., and Guestrin, C. (2014). Stochastic gradient Hamiltonian Monte Carlo. *International Conference on Machine Learning*, pages 1683–1691.

Chen, T. and Hwang, C. (2013). Accelerating reversible Markov chains. *Statistics & Probability Letters*, 83(9):1956–1962.

Davis, M. H. (1984). Piecewise-deterministic Markov processes: A general class of non-diffusion stochastic models. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 353–388.

Davis, M. H. (1993). *Markov Models & Optimization*, volume 49. CRC Press.

Deligiannidis, G., Doucet, A., and Pitt, M. K. (2015). The correlated pseudo-marginal method. *arXiv preprint arXiv:1511.04992*.

Ding, N., Fang, Y., Babbush, R., Chen, C., Skeel, R. D., and Neven, H. (2014). Bayesian sampling using stochastic gradient thermostats. *Advances in neural information processing systems*, pages 3203–3211.

Duane, S., Kennedy, A. D., Pendleton, B. J., and Roweth, D. (1987). Hybrid Monte Carlo. *Phys. Lett. B*, 195:216–222.

Gelfand, A. and Smith, A. (1990). Sampling based approaches to calculating marginal densities. *J. American Statist. Assoc.*, 85:398–409.

Girolami, M. and Calderhead, B. (2011). Riemann manifold Langevin and Hamiltonian Monte Carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 73:123–214.

Hastings, W. K. (1970). Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109.

Hoffman, M. D. and Gelman, A. (2014). The No-U-turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo. *The Annals of Applied Probability*, 27(4):2159–2194.

Hwang, C.-R., Hwang-Ma, S.-Y., and Sheu, S.-J. (1993). Accelerating Gaussian diffusions. *The Annals of Applied Probability*, pages 897–913.

Kingman, J. F. C. (1992). *Poisson processes*, volume 3. Clarendon Press.

Lewis, P. A. and Shedler, G. S. (1979). Simulation of nonhomogeneous poisson processes by thinning. *Naval Research Logistics (NRL)*, 26(3):403–413.

Liu, J., Wong, W., and Kong, A. (1994). Covariance structure of the Gibbs sampler with application to the comparison of estimators and augmentation schemes. *Biometrika*, 81:27–40.

Liu, J., Wong, W., and Kong, A. (1995). Covariance structure and convergence rates of the Gibbs sampler with various scans. *Journal of Royal Statistical Society*, B 57:157–169.

Livingstone, S., Faulkner, M. F., and Roberts, G. O. (2017). Kinetic energy choice in Hamiltonian/hybrid Monte Carlo. *arXiv preprint arXiv:1706.02649*.

MacKay, D. J. C. (2002). *Information Theory, Inference & Learning Algorithms*. Cambridge University Press, Cambridge, UK.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. (1953). Equations of state calculations by fast computing machines. *J. Chem. Phys.*, 21:1087–1092.

Meyn, S. and Tweedie, R. (1993). *Markov Chains and Stochastic Stability*. Springer-Verlag, New York.

Minsker, S., Srivastava, S., Lin, L., and Dunson, D. (2014). Scalable and robust Bayesian inference via the median posterior. *International Conference on Machine Learning*, pages 1656–1664.

Neal, R. (1999). *Bayesian Learning for Neural Networks*, volume 118. Springer–Verlag, New York. Lecture Notes.

Neal, R. (2011). MCMC using Hamiltonian dynamics. In Brooks, S., Gelman, A., Jones, G. L., and Meng, X.-L., editors, *In Handbook of Markov Chain Monte Carlo*. CRC Press, New York.

Neiswanger, W., Wang, C., and Xing, E. (2013). Asymptotically exact, embarrassingly parallel MCMC. *arXiv preprint arXiv:1311.4780*.

Peters, E. A. et al. (2012). Rejection-free Monte Carlo sampling for general potentials. *Physical Review E*, 85(2):026703.

Quiroz, M., Villani, M., and Kohn, R. (2016). Exact subsampling MCMC. *arXiv preprint arXiv:1603.08232*.

Rhee, C.-h. and Glynn, P. W. (2015). Unbiased estimation with square root convergence for sde models. *Operations Research*, 63(5):1026–1043.

Robert, C. and Casella, G. (2004). *Monte Carlo Statistical Methods*. Springer-Verlag, New York, second edition.

Rubinstein, R. Y. (1981). *Simulation and the Monte Carlo Method*. J. Wiley, New York.

Scott, S. L., Blocker, A. W., Bonassi, F. V., Chipman, H. A., George, E. I., and McCulloch, R. E. (2016). Bayes and big data: The consensus Monte Carlo algorithm. *International Journal of Management Science and Engineering Management*, 11(2):78–88.

Srivastava, S., Cevher, V., Dinh, Q., and Dunson, D. (2015). WASP: Scalable Bayes via barycenters of subset posteriors. *Artificial Intelligence and Statistics*, pages 912–920.

Sun, Y., Schmidhuber, J., and Gomez, F. J. (2010). Improving the asymptotic performance of Markov chain Monte-carlo by inserting vortices. *Advances in Neural Information Processing Systems*, pages 2235–2243.

Vanetti, P., Bouchard-Côté, A., Deligiannidis, G., and Doucet, A. (2017). Piecewise deterministic Markov chain Monte Carlo. *arXiv preprint arXiv:1707.05296*.

Wang, X. and Dunson, D. (2013). Parallelizing MCMC via Weierstrass sampler. *arXiv preprint arXiv:1312.4605*.

Wang, X., Guo, F., Heller, K., and Dunson, D. (2015). Parallelizing MCMC with random partition trees. *Advances in Neural Information Processing Systems*, pages 451–459.

Welling, M. and Teh, Y. (2011). Bayesian learning via stochastic gradient Langevin dynamics. *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 681–688.

Coordinate Sampler

Joint work with Christian P. ROBERT

2.1 Introduction

A powerful sampling technique, the Markov chain Monte Carlo (MCMC) method, (see, e.g., Robert and Casella (2004)) has been widely exploited in computational statistics to become a standard tool in Bayesian inference, where posterior distributions are often intractable analytically and at best known up to a normalizing constant. However, almost all existing MCMC algorithms, such as Metropolis-Hastings algorithm (MH), Hamiltonian Monte Carlo (HMC) (Neal et al. (2011)) and Metropolis adjusted Langevin algorithm (MALA), satisfy detailed balance conditions, dating back to Metropolis et al. (1953) and Hastings (1970). Recently, a different type of MCMC method – piecewise deterministic Markov process (PDMP) – was introduced in computational statistics, as an approach that is generally irreversible and hence does not satisfy detailed balance. The basic theory of PDMP was developed by Davis (1984) and Davis (1993), while an application to computational statistics was first implemented by Peters et al. (2012), Bierkens et al. (2016), and Bouchard-Côté et al. (2018).

In this chapter, we propose the Coordinate Sampler (CS), a novel PDMP-based MCMC sampler that is a variant of the Zigzag sampler (ZS) of Bierkens et al. (2016). However, it differs in three important aspects. First, the velocity set used in the coordinate sampler consists of an orthonormal basis of the Euclidean space $\mathbb{R}^d$, while the one of the Zigzag sampler is restricted to $\{-1, 1\}^d$, where d denotes the dimension of the target distribution. Second, the event rate function in the Zigzag sampler is much larger the one for the coordinate sampler, especially for high dimensional distributions. This means that events occur more frequently in the Zigzag sampler and hence this lowers its efficiency compared with our approach. Thirdly, the coordinate sampler targets only one component at a time when exploring the target space, and it keeps the other components unchanged, while Zigzag sampler modifies all components at the same time.

Related work Since piecewise deterministic Markov processes for sampling from distributions was introduced by Peters et al. (2012), PDMP-based, continuous-time, non-reversible, MCMC algorithms have become relevant tools, from applied probability (Bierkens et al., 2017; Fontbona et al., 2016) to physics (Harland et al., 2017; Michel et al., 2014; Peters et al., 2012), to statistics (Bierkens et al., 2018, 2016; Bouchard-Côté et al., 2018; Fearnhead et al., 2016; Michel and Sénécal, 2017; Pakman et al., 2016; Vanetti et al., 2017). However, almost all existing PDMP-based

MCMC samplers are based on two original versions: the Bouncy Particle Sampler (BPS) of Bouchard-Côté et al. (2018) and and the Zigzag Sampler of Bierkens et al. (2016). Bouchard-Côté et al. (2018) exhibits that BPS can provide state-of-the-art performance compared with the reference HMC for high dimensional distributions, while Bierkens et al. (2016) shows that PDMP-based sampler is easier to scale in big data settings without introducing bias, while Bierkens et al. (2018) considers the application of PDMP for distributions on restricted domains. Fearnhead et al. (2016) unifies BPS and Zigzag sampler in the framework of PDMP and they choose the process velocity, at event times, over the unit sphere, based on the inner product between this velocity and the gradient of the potential function. (This perspective relates to the transition dynamics used in our paper.) To overcome the main difficulty in PDMP-based samplers, which is the simulation of time-inhomogeneous Poisson process, Sherlock and Thiery (2017) and Vanetti et al. (2017) resort to a discretisation of such continuous-time samplers. Furthermore, pre-conditioning the velocity set is shown to accelerate the algorithms, as shown by Pakman et al. (2016).

The outline of this chapter is as follows. Section 2 introduces the necessary background of PDMP-based MCMC samplers, the techniques used in its implementation, and two specified samplers, BPS and the Zigzag sampler. In Section 3, we describe the methodology behind the coordinate sampler, provide some theoretical validation along with a proof of geometrical ergodicity, obtained under quite mild conditions, and we compare this proposal with the Zigzag sampler in an informal analysis. Section 4 further compares the efficiency of both approaches on banana-shaped distributions, multivariate Gaussian distributions and a Bayesian logistic model, when effective sample size is measuring efficiency. Section 5 concludes by pointing out further research directions about this special MCMC sampler.

2.2 Piecewise deterministic Markov process

In this section, we briefly introduce piecewise deterministic Markov processes (PDMP) and describe how to apply this methodology into statistical computing problems. We describe two specified PDMP-based MCMC samplers: the bouncy particle sampler (BPS) and the Zigzag sampler (ZS).

2.2.1 PDMP-based Sampler

In this chapter, we assume π be the continuous target distribution over $\mathbb{R}^d$ and for convenience sake, we also use $\pi(\mathbf{x})$ for the probability density function of π, when $\mathbf{x} \in \mathbb{R}^d$. We define $U(\mathbf{x})$ as the potential function of $\pi(\mathbf{x})$, that is, $\pi(\mathbf{x}) \propto \exp\{-U(\mathbf{x})\}$, with U positive. In the PDMP framework, an auxiliary variable, $\mathbf{V} \in \mathcal{V}$, is introduced and a PDMP-based sampler explores the augmented state space $\mathbb{R}^d \times \mathcal{V}$, targeting a variable $\mathbf{Z} = (\mathbf{X}, \mathbf{V})$ with distribution $\rho(d\mathbf{x}, d\mathbf{v})$ over $\mathbb{R}^d \times \mathcal{V}$ as its invariant distribution. By construction, the distribution ρ enjoys π as its marginal distribution in $\mathbf{x}$. In practice, the existing PDMP-based samplers choose $\mathcal{V}$ to be the Euclidean space $\mathbb{R}^d$, the sphere $\mathbb{S}^{d-1}$, or the discrete set $\{\mathbf{v} = (v_1, \cdots, v_d) | v_i \in \{-1, 1\}, i = 1, \cdots, d\}$. A piecewise deterministic Markov process $\mathbf{Z}_t = (\mathbf{X}_t, \mathbf{V}_t)$ consists of three distinct components: its deterministic dynamic between events, an event occurrence rate, and a transition dynamic at event time. Specifically,

1. **Deterministic dynamic**: between two events, the Markov process evolves

deterministically, according to some ordinary differential equation:

$$\frac{d\mathbf{Z}_t}{dt} = \Psi(\mathbf{Z}_t).$$

2. **Event occurrence rate**: an event occurs at time t with rate $\lambda(\mathbf{Z}_t)$.

3. **Transition dynamic**: At an event time, τ, the state prior to τ is denoted by $\mathbf{Z}_{\tau-}$, with the new state being generated by $\mathbf{Z}_\tau \sim Q(\cdot|\mathbf{Z}_{\tau-})$.

Here, an "event" refers to an occurrence of a time-inhomogeneous Poisson process with rate $\lambda(\cdot)$ (Kingman, 1992). Following (Davis, 1993, Theorem 26.14), this Markov process had an extended generator equal to

$$\mathcal{L}f(\mathbf{z}) = \nabla f(\mathbf{z}) \cdot \Psi(\mathbf{z}) + \lambda(\mathbf{z}) \int_{\mathbf{z}'} [f(\mathbf{z}') - f(\mathbf{z})]\, Q(d\mathbf{z}'|\mathbf{z}) \tag{2.1}$$

In order to guarantee invariance with respect to $\rho(d\mathbf{z})$, the extended generator need satisfy $\int \mathcal{L}f(\mathbf{z})\rho(d\mathbf{z}) = 0$ for all f in an appropriate function class on $\mathbb{R}^d \times \mathcal{V}$ (Davis, 1993, Theorem 34.7).

2.2.2 Implementation of a PDMP-based Sampler

In practice, choosing an appropriate deterministic dynamic, an event rate and a transition dynamic, produces a Markov chain with unique invariant distribution $\rho(d\mathbf{z})$. As for regular MCMC, generating such a Markov chain for T long enough, leads to an estimator, $\frac{1}{T}\int_{t=0}^{T} h(\mathbf{X}_t)dt$, converging to the integral of interest, $I = \int h(\mathbf{x})\pi(d\mathbf{x})$, by the Law of Large Numbers for Markov processes (Glynn and Haas, 2006), under appropriate assumptions. More specifically,

$$\frac{1}{T}\int_{t=0}^{T} g(\mathbf{Z}_t)dt \longrightarrow \int g(\mathbf{z})\rho(d\mathbf{z}), \quad \text{as } T \to \infty$$

and defining $g(\mathbf{z}) = g(\mathbf{x}, \mathbf{v}) := h(\mathbf{x})$ induces, as $T \to \infty$,

$$\frac{1}{T}\int_{t=0}^{T} h(\mathbf{X}_t)dt = \frac{1}{T}\int_{t=0}^{T} g(\mathbf{Z}_t)dt \to \int g(\mathbf{z})\rho(d\mathbf{z}) = \int\int h(\mathbf{x})\pi(d\mathbf{x})p(d\mathbf{v}|\mathbf{x}) = \int h(\mathbf{x})\pi(d\mathbf{x}),$$

where $p(d\mathbf{v}|\mathbf{x})$ is the conditional distribution of the variable $\mathbf{V}$, given $\mathbf{X} = \mathbf{x}$. Algorithm 2.1 contains a pseudo-code to simulate the PDMP in practice:

Algorithm 2.1 General PDMP-based sampler

1: **Input:** start at position $\mathbf{x}_0$, velocity $\mathbf{v}_0$ and simulation time threshold T^{total}.
2: Generate a set of event time of PDMP $\{\tau_0, \tau_1, \cdots, \tau_M\}$ and their associated state $\{\mathbf{Z}_{\tau_0}, \mathbf{Z}_{\tau_1}, \cdots, \mathbf{z}_{\tau_M}\}$, where $\tau_0 = 0$, $\tau_{M-1} < T^{total}$, $\tau_M \geq T^{total}$. Set $\mathbf{Z}_0 = (\mathbf{X}_0, \mathbf{V}_0)$.
3: Set $t \leftarrow 0$, $T \leftarrow 0$, $m \leftarrow 0$, $\tau_m \leftarrow 0$
4: **while** $T < T^{total}$ **do**
5: $m \leftarrow m + 1$
6: $u \leftarrow \text{Uniform}[0, 1]$
7: Solve the equation

$$\exp\left\{-\int_0^{\eta_m} \lambda_m(t)dt\right\} = u, \tag{2.2}$$

to obtain η_m, where $\lambda_m(t) = \lambda\left(\Phi_t(\mathbf{X}_{\tau_{m-1}}, \mathbf{V}_{\tau_{m-1}})\right)$, and $\Phi_t(\cdot,\cdot)$ is the flow of the deterministic dynamic.
8: $\tau_m \leftarrow \tau_{m-1} + \eta_m$, $T \leftarrow \tau_m$, $\mathbf{Z}_{\tau_m} \sim Q(\Phi_{\eta_m}(\mathbf{X}_{\tau_{m-1}}, \mathbf{V}_{\tau_{m-1}}), \cdot)$
9: **end while**
10: **Output:** A trajectory of the Markov chain up to time τ_M, $\{\mathbf{Z}_t\}_{t=0}^{\tau_M}$, where $\mathbf{Z}_t = \Phi_{t-\tau_m}(\mathbf{Z}_{\tau_m})$ for $\tau_m \leq t < \tau_{m+1}$.

However, in many cases, evaluating the path integral $\int_{t=0}^{T} h(\mathbf{Z}_t)dt$ may be expensive, or even impossible, and a discretisation of the simulated trajectory is a feasible alternative. This means estimating the quantity of interest, I, by the following estimator

$$\hat{I} = \frac{1}{N}\sum_{n=1}^{N} h(\mathbf{X}_{\frac{nT}{N}}).$$

In practice, the main difficult in implementing a PDMP-based sampler is the generation of the occurrence times of the associated time-inhomogeneous Poisson process with event rate $\lambda(\cdot)$. Fortunately, the following two theorems alleviate this difficulty.

Theorem 2.2.1 (Superposition Theorem) *(Kingman, 1992) Let $\Pi_1, \Pi_2, \cdots$, be a countable collection of independent Poisson processes on state space $\mathbb{R}^+$ and let Π_n have rate $\lambda_n(\cdot)$ for each n. If $\sum_{n=1}^{\infty}\lambda_n(t) < \infty$ for all t, then the superposition*

$$\Pi = \bigcup_{n=1}^{\infty} \Pi_n$$

is a Poisson process with rate

$$\lambda(t) = \sum_{n=1}^{\infty} \lambda_n(t)$$

Theorem 2.2.2 (Thinning Theorem) *(Lewis and Shedler, 1979) Let $\lambda : \mathbb{R}^+ \to \mathbb{R}^+$ and $\Lambda : \mathbb{R}^+ \to \mathbb{R}^+$ be continuous functions such that $\lambda(t) \leq \Lambda(t)$ for all $t \geq 0$. Let $\tau^1, \tau^2, \cdots$, be the increasing finite or infinite sequence of a Poisson process with rate $\Lambda(\cdot)$. For all i, if the point τ^i is removed from the sequence with probability $1-\lambda(t)/\Lambda(t)$, then the remaining points $\tilde{\tau}^1, \tilde{\tau}^2, \cdots$ form a non-homogeneous Poisson process with rate $\lambda(\cdot)$.*

2.2.3 Two reference PDMD-based samplers

Almost all existing PDMD-based samplers are based on two specific ones, both of which rely on linearly deterministic dynamics, a feature that facilitates the determination of the state of the Markov chain between Poisson events. Vanetti et al. (2017) uses Hamiltonian dynamics over an approximation of the target distribution to accelerate the bouncy particle sampler, but the efficiency of that modification depends on the quality of the approximation and it only transfer the difficulty from setting the deterministic dynamics to computing the event rate function.

Bouncy Particle sampler

For the Bouncy Particle sampler, the velocity set $\mathcal{V}$ is either the Euclidean space $\mathbb{R}^d$, or the unit sphere $\mathbb{S}^{d-1}$. The associated augmented target distribution is $\rho(d\mathbf{x}, d\mathbf{v}) = \pi(d\mathbf{x})\mathcal{N}(d\mathbf{v}|0, I_d)$, or $\rho(d\mathbf{x}, d\mathbf{v}) = \pi(d\mathbf{x})\mathcal{U}_{\mathbb{S}^{d-1}}(d\mathbf{v})$, where $\mathcal{N}(\cdot|0, I_d)$ represents the standard d-dimensional Gaussian distribution and $\mathcal{U}_{\mathbb{S}^{d-1}}(d\mathbf{v})$ denotes the uniform distribution over $\mathbb{S}^{d-1}$, respectively. The corresponding deterministic dynamic is

$$\frac{d\mathbf{X}_t}{dt} = \mathbf{V}_t, \quad \frac{d\mathbf{V}_t}{dt} = \mathbf{0},$$

$\lambda(\mathbf{z}) = \lambda(\mathbf{x}, \mathbf{v}) = \langle \mathbf{v}, \nabla U(\mathbf{x}) \rangle_+ + \lambda^{\text{ref}}$, where λ^{ref} is a user-chosen non-negative constant and the transition dynamic is as

$$Q((d\mathbf{x}', d\mathbf{v}')|(\mathbf{x}, \mathbf{v})) = \frac{\langle \mathbf{v}, \nabla U(\mathbf{x}) \rangle_+}{\lambda(\mathbf{x}, \mathbf{v})} \delta_{\mathbf{x}}(d\mathbf{x}') \delta_{R_{\nabla U(\mathbf{x})}\mathbf{v}}(d\mathbf{v}') + \frac{\lambda^{\text{ref}}}{\lambda(\mathbf{x}, \mathbf{v})} \delta_{\mathbf{x}}(d\mathbf{x}') \varphi(d\mathbf{v}')$$

where $\varphi(d\mathbf{v}) = \mathcal{U}_{\mathbb{S}^{d-1}}(d\mathbf{v})$ or $\varphi(d\mathbf{v}) = \mathcal{N}(d\mathbf{v}|0, I_d)$, depending on the choice of the velocity set, and the operator $R_{\mathbf{w}}$, for any non-zero vector $\mathbf{w} \in \mathbb{R}^d - \{\mathbf{0}\}$, is $R_{\mathbf{w}}\mathbf{v} = \mathbf{v} - 2\frac{\langle \mathbf{w}, \mathbf{v} \rangle}{\langle \mathbf{w}, \mathbf{w} \rangle}\mathbf{w}$.

Zigzag sampler

For the Zigzag sampler, the velocity set, $\mathcal{V}$, is the discrete set $\{\mathbf{v} = (v_1, \cdots, v_d) | v_i \in \{-1, 1\}, i = 1, \cdots, d\}$ and $\rho(d\mathbf{x}, d\mathbf{v}) = \pi(d\mathbf{x})\varphi(d\mathbf{v})$, where φ is the uniform distribution over $\mathcal{V}$. ZS uses the same linear deterministic dynamics as BPS. Its event rate is $\lambda(\mathbf{z}) = \sum_{i=1}^d \lambda_i(\mathbf{x}, \mathbf{v}) = \sum_{i=1}^d \left[\{v_i \nabla_i U(\mathbf{x})\}_+ + \lambda_i^{\text{ref}} \right]$, where the λ_i^{ref}'s are user-chosen non-negative constants. The transition dynamics is $Q((d\mathbf{x}', d\mathbf{v}')|(\mathbf{x}, \mathbf{v})) = \sum_{i=1}^d \frac{\lambda_i(\mathbf{x}, \mathbf{v})}{\lambda(\mathbf{x}, \mathbf{v})} \delta_{\mathbf{x}}(d\mathbf{x}') \delta_{F_i\mathbf{v}}(d\mathbf{v}')$, where F_i denotes an operator that flips the i-th component of $\mathbf{v}$ and keep the others unchanged. In practice, ZS relies on the Superposition Theorem. At each event time, ZS simulates d Poisson processes, with rates $\lambda_i(\mathbf{x}+t\mathbf{v}, \mathbf{v})$, computes their first occurrence time and takes their minimum, e.g., the i-th, for the duration between current and next event, and flips the i-th component of the velocity $\mathbf{v}$.

2.3 Coordinate sampler

We now describe the coordinate sampler, in which only one component of $\mathbf{x}$ evolves and the rest remains inactive between event times. For CS, the velocity set $\mathcal{V}$ is chosen to be $\{\pm e_i, i = 1, \cdots, d\}$, where e_i is the vector with i-th component equal to one and the others set to zero. The augmented target distribution is $\rho(d\mathbf{x}, d\mathbf{v}) = \pi(d\mathbf{x})\varphi(d\mathbf{v})$, with $\varphi(d\mathbf{v})$ the uniform distribution over $\mathcal{V}$. The PDMP characteristics of CS are thus

1. **Deterministic dynamic**:

$$\frac{d\mathbf{X}_t}{dt} = \mathbf{V}_t, \quad \frac{d\mathbf{V}_t}{dt} = \mathbf{0}\,.$$

2. **Event occurrence rate**: $\lambda(\mathbf{z}) = \langle \mathbf{v}, \nabla U(\mathbf{x})\rangle_+ + \lambda^{\text{ref}}$, where λ^{ref} is a user-chosen non-negative constant.

3. **Transition dynamic**:

$$Q((d\mathbf{x}', d\mathbf{v}')|(\mathbf{x}, \mathbf{v})) = \sum_{\mathbf{v}^* \in \mathcal{V}} \frac{\lambda(\mathbf{x}, -\mathbf{v}^*)}{\lambda(\mathbf{x})} \delta_{\mathbf{x}}(d\mathbf{x}')\delta_{\mathbf{v}^*}(d\mathbf{v}')$$

where $\lambda(\mathbf{x}) = \sum_{\mathbf{v}\in\mathcal{V}} \lambda(\mathbf{x}, \mathbf{v}) = 2d\lambda^{\text{ref}} + \sum_{i=1}^{d} \left|\frac{\partial U(\mathbf{x})}{\partial x_i}\right|$,

which translates into the pseudo-code

Algorithm 2.2 Coordinate Sampler

Input: Start with position $\mathbf{X}_0$, velocity $\mathbf{V}_0$ and set simulation time threshold T^{total}.

Generate a set of event times $\{\tau_0, \tau_1, \cdots, \tau_M\}$ and their associated state $\{\mathbf{Z}_{\tau_0}, \mathbf{Z}_{\tau_1}, \cdots, \mathbf{Z}_{\tau_M}\}$, where $\tau_0 = 0$, $\tau_M \geq T^{total}$ and $\tau_{M-1} < T^{total}$, $\mathbf{Z}_0 = (\mathbf{X}_0, \mathbf{V}_0)$

Set $t \leftarrow 0$, $T \leftarrow 0$, $m \leftarrow 0$, $\tau_m \leftarrow 0$

while $T < T^{total}$ **do**

$m \leftarrow m + 1$

$u \leftarrow \text{Uniform}[0, 1]$

Solve the equation

$$\exp\left\{-\int_0^{\eta_m} \lambda_m(t)dt\right\} = u$$

with respect to η_m, where $\lambda_m(t) = \lambda\left(\mathbf{X}_{\tau_{m-1}} + t\mathbf{V}_{\tau_{m-1}}, \mathbf{v}_{\tau_{m-1}}\right)$.

$\tau_m \leftarrow \tau_{m-1} + \eta_m$, $T \leftarrow \tau_m$, $\mathbf{Z}_{\tau_m} \sim Q((\mathbf{X}_{\tau_{m-1}} + \eta_m\mathbf{V}_{\tau_{m-1}}, \mathbf{v}_{\tau_{m-1}}), \cdot)$.

end while

Output: A trajectory of the Markov chain over $[0, \tau_M]$, $\{\mathbf{Z}_t\}_{t=0}^{\tau_M}$, where $\mathbf{Z}_t = (\mathbf{X}_{\tau_m} + (t - \tau_m)\mathbf{V}_{\tau_m}, \mathbf{V}_{\tau_m})$ for $\tau_m \leq t < \tau_{m+1}$.

2.3.1 Theoretical properties of the coordinate sampler

We now establish that CS enjoys the augmented target distribution, $\rho(d\mathbf{x}, d\mathbf{v})$, as its invariant distribution under the condition that $U : \mathbb{R}^d \to \mathbb{R}^+$ is C^1. Besides, under the following assumptions, CS is V-uniformly ergodic for the Lyapunov function

$$V(\mathbf{x}, \mathbf{v}) = \frac{e^{U(\mathbf{x})/2}}{\sqrt{\lambda^{\text{ref}} + \langle \nabla U(\mathbf{x}), -\mathbf{v}\rangle_+}},$$

which was used in Deligiannidis et al. (2017).

Theorem 2.3.1 *For any positive $\lambda^{ref} > 0$, the PDMP produced by CS enjoys $\rho(d\mathbf{x}, d\mathbf{v})$ as its unique invariant distribution, provided the potential U is C^1.*

It is easy to check that the generator of coordinate sampler, $\mathcal{L}$, satisfies

$$\int \mathcal{L}f(\mathbf{z})\rho(d\mathbf{z}) = 0,$$

for all functions, f, in its extended generator, which means ρ is an invariant distribution of CS (Davis, 1993, Theorem 34.7) . Besides, uniqueness follows from the positivity of λ^{ref}, which enables the Markov chain to reach any state $(\mathbf{x}^*, \mathbf{v}^*)$ from any starting state $(\mathbf{x}_0, \mathbf{v}_0)$, in finite time. The details of the proof is postponed till the supplementary document. In practice, it appears that the constraint $\lambda^{\text{ref}} > 0$ is not necessary for convergence in many examples.

Assumptions: Assume $U : \mathbb{R}^d \to \mathbb{R}^+$ satisfy the following conditions, set by Deligiannidis et al. (2017),

A.1 $\frac{\partial^2 U(\mathbf{x})}{\partial x_i x_j}$ is locally Lipschitz continuous for all i, j,

A.2 $\int \big|\nabla U(\mathbf{x})\big|\pi(d\mathbf{x}) < \infty$,

A.3 $\underline{\lim}_{|\mathbf{x}|\to\infty} e^{U(\mathbf{x})/2}/\sqrt{\big|\nabla U(\mathbf{x})\big|} > 0$

A.4 $V \geq c_0$ for some positive constant c_0.

Conditions: We set conditions

C.1 $\underline{\lim}_{|x|\to\infty}\big|\nabla U(x)\big| = \infty$, $\overline{\lim}_{|x|\to\infty}\|\Delta U(x)\| \leq \alpha_1 < \infty$ and $\lambda^{\text{ref}} > \sqrt{8\alpha_1}$.

C.2 $\underline{\lim}_{|\mathbf{x}|\to\infty}\big|\nabla U(\mathbf{x})\big| = 2\alpha_2 > 0$, $\overline{\lim}_{|x|\to\infty}\|\Delta U(x)\| = 0$ and $\lambda^{\text{ref}} < \frac{\alpha_2}{14d}$.

where **C.1** corresponds to distributions whose tails decay at rate $\mathcal{O}(|\mathbf{x}|^\beta)$, where $1 < \beta \leq 2$, and **C.2** to distribution with tails of order $\mathcal{O}(|\mathbf{x}|^1)$.

Theorem 2.3.2 *Suppose the assumptions $A.1 - A.4$ hold and one of the conditions $C.1, C.2$ holds, then CS is V-uniformly ergodic: There exist constants $\Gamma < \infty$ and $0 < \gamma < 1$, such that*

$$\|P^t(\mathbf{z}, \cdot) - \rho\|_V \leq V(\mathbf{z})\Gamma\gamma^t\,,$$

where $P^t(\mathbf{z}, \cdot)$ is the distribution of the Markov chain with starting state $\mathbf{z}$ at time t, and the norm $\|\cdot\|_V$ is defined by

$$\|\mu\|_V = \sup_{|f|<V} \int f(\mathbf{z})\mu(d\mathbf{z})\,.$$

The proof appears in the supplementary materials, based on techniques similar to those in Deligiannidis et al. (2017).

2.3.2 An informal comparison between Zigzag and coordinate sampler

For CS, each event time sees a change of a single component of $\mathbf{X}$, in contrast with the Zigzag sampler, which modifies all components at the same time. A first impression is that CS is thus less efficient in its exploration of the target space, when compared with ZS, because of this restriction. However, this intuition is misleading. Suppose

that the λ_i's, $i = 1, \cdots, d$ in ZS and λ in CS are of similar scales, for instance taking the expected duration between two Poisson events to be the same value ℓ. Assume further that computing an occurrence time have the same computation cost, c, for all Poisson processes. In ZS, the event rate is the summation of the rates $\lambda_1, \cdots, \lambda_d$. Therefore, the time duration between two events is $\frac{\ell}{d}$ and the induced computation cost is dc. Thus, that each component of $\mathbf{X}$ evolves for a time duration ℓ costs d^2c for ZS. By contrast, in CS, a dc computation cost will result from the Markov chain moving for a duration time $d\ell$. Hence, the computation cost for monitoring each component for a time duration ℓ is also dc. As a result, CS is $\mathcal{O}(d)$ times more efficient than ZS in terms of the evolution of a given component of $\mathbf{X}$.

2.4 Numerical experiments

In this section, we compare the efficiency of both samplers over benchmarks (a banana-shaped distribution, two multivariate Gaussian distributions, and a Bayesian logistic model). In each model, we run both samplers for the same computer time and we compare their efficiency in terms of effective sample size (ESS) (Liu, 2008) per second. The models are reproduced ten times to produce an averaged efficiency ratio, namely the ratio of ESS per second for CS over the one for ZS. We use the function *ess* of package *mcmcse* in R to compute ESS of samples. In the first three experiments, we use the canonical ZS and CS, meaning that $\lambda_i^{\text{ref}} = 0, i = 1, \cdots, d$ in ZS and $\lambda^{\text{ref}} = 0$ in CS, since such setting guarantees ergodicity. In Bayesian logistic model, we set $\lambda_i^{\text{ref}} = 1, i = 1, \cdots, d$ and $\lambda^{\text{ref}} = 1$. In log-Gaussian Cox point process, we set $\lambda_i^{\text{ref}} = 0.1, i = 1, \cdots, d$ and $\lambda^{\text{ref}} = 0.1$ to achieve 10 percent complete refreshment of velocity.

Banana-Shaped Distribution: The target distribution is a 2-dimensional banana-shaped distribution with density

$$\pi(\mathbf{x}) \propto \exp\left\{-(x_1 - 1)^2 - \kappa(x_2 - x_1^2)^2\right\}$$

where κ controls the similarity between x_2 and x_1^2. A high κ enforces the approximate constraint $x_2 \simeq x_1^2$. The comparison between Zigzag and coordinate samplers runs over the configurations $2^{-2} \leq \kappa \leq 2^5$. With an increase in κ, the distribution becomes more difficult to simulate and the event rate functions in CS and ZS make the generations of time durations more costly. Figure 2.1 shows that CS is more efficient than ZS across a large range of κ in this model.

Strongly Correlated Multivariate Gaussian Distribution (MVN1): Here, the target is a multivariate Gaussian distribution with zero mean and covariance matrix equal to $A \in \mathbb{R}^{d\times d}$, where $A_{ii} = 1$ and $A_{ij} = 0.9, i \neq j$. The density function is

$$\pi(\mathbf{x}) \propto \exp\{-\frac{1}{2}\mathbf{x}^T A^{-1}\mathbf{x}\}$$

We consider the values $d = 10, 20, \ldots, 100$ in the comparison of the sampling methods.

Correlated Multivariate Gaussian Distribution (MVN2): In this scenario, the target distribution is again a multivariate Gaussian distribution with zero mean and covariance matrix such that $A_{ii} = 1$ and $A_{ij} = 0.9^{|i-j|}$. Once again, the comparison runs for $d = 10, \ldots, 100$.

Figure 2.2 presents the comparison between CS and ZS for these models **MVN1** and **MVN2** in terms of the minimal ESS, mean ESS and maximal ESS taken across all d components. In **MVN1**, the three curves are similar by virtue of the symmetry across the components. In both models, the efficiency ratio and thus the improvement brought by CS over ZS increases with the dimension d.

In Table 2.1, we compare CS with several popular algorithms over a 20-dimensional multivariate normal distribution with zero mean and $A \in \mathbb{R}^{20\times 20}$ covariance matrix, where $A_{ii=1}$ and $A_{ij} = 0.9^{|i-j|}, i \neq j$, in terms of Kolmogorov-Smirnov statistic and Wasserstein distances. Since it is infeasible to compute such quantities for multivariate dimensional distributions, we compute the marginal distances between the samples of each algorithm and the target across the coordinates and take the median of them as the summary of efficiency. We compare the samplers under same computation time (60 seconds).

Table 2.1: Comparison for a 20-dimensional MVN over 40 repetitions in terms of median of the marginal distances across the components in each criterion. The smaller the value is, the better the algorithm performs.

Sampler	Kolmogorov statistic	Wasserstein L_1	Wasserstein L_2
Bouncy Particle Sampler	4.34×10^{-2}	7.66×10^{-2}	9.65×10^{-2}
Coordinate Sampler	$\mathbf{3.73 \times 10^{-2}}$	$\mathbf{6.29 \times 10^{-2}}$	$\mathbf{7.76 \times 10^{-2}}$
Zigzag Sampler	6.62×10^{-2}	1.348×10^{-1}	1.561×10^{-1}
Gibbs Sampler	4.90×10^{-2}	9.26×10^{-2}	1.145×10^{-1}
Metropolis-Hastings	1.513×10^{-1}	3.125×10^{-1}	3.576×10^{-1}

Bayesian Logistic Model: In this example, the target is the posterior of a Bayesian logistic model under a flat prior, with no intercept. The simulated dataset contains N observations $\{(\mathbf{r}_n, t_n)\}_{n=1}^N$, where each $r_{n,i}$, $n = 1, \cdots, N, i = 1, \cdots, d$, is drawn from standard normal distribution and t_n is drawn from $\{-1, 1\}$ uniformly. The targeted density function is thus

$$\pi(\mathbf{x}) \propto \prod_{n=1}^{N} \frac{\exp(t_n \mathbf{x}^T \mathbf{r}_n)}{1 + \exp(\mathbf{x}^T \mathbf{r}_n)}$$

In the simulations, we set $N = 40, d = 10$, and $\lambda^{\text{ref}} = 1$ for CS, and $\lambda_i^{\text{ref}} = 1, i = 1, \cdots, d$ for ZS. Figure 2.3 presents the comparison between the two samplers, with a massive improvement brought by our proposal.

Log-Gaussian Cox Point Process In this example, as described by **?**, the observations $\mathbf{Y} = \{y_{ij}\}$ are Poisson distributed and conditionally independent given a latent intensity process $\mathbf{\Lambda} = \{\lambda_{ij}\}$ with means $s\lambda_{ij} = s\exp(x_{ij})$, where $s = 1/d^2$. The underlying process $\mathbf{X} = \{x_{ij}\}$ is a Gaussian process with mean function $m(x_{ij}) = \mu\mathbf{1}$ and covariance function $\Sigma(x_{i,j}, x_{i',j'}) = \sigma^2 \exp(-\delta(i, i', j, j')/(\beta d))$, where $\delta(i, i', j, j') = \sqrt{(i - i')^2 + (j - j')^2}$. In our experiment, we set $d = 20$ and choose $\sigma^2 = 1.91, \mu = \log(126) - \sigma^2/2$ and $\beta = 1/6$. The target is, based on the observations $\mathbf{Y}$,

$$\pi(\mathbf{X}|\mathbf{Y}, \mu, \sigma, \beta) \propto \exp\left\{\sum_{i,j=1}^{d} (y_{ij}x_{ij} - s\exp(x_{ij})) - \frac{1}{2}(\mathbf{X} - \mu\mathbf{1})^T \Sigma^{-1} (\mathbf{X} - \mu\mathbf{1})\right\}$$

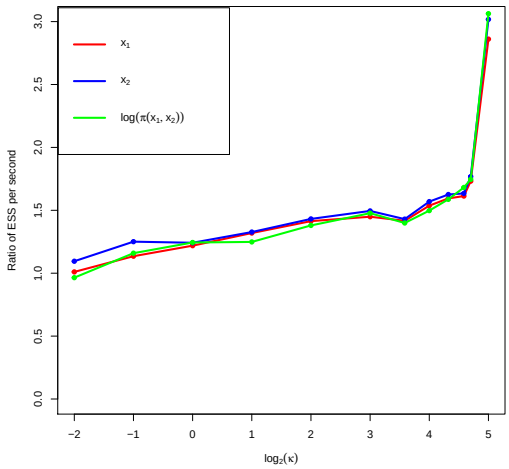

Figure 2.1: Banana-Shaped Distribution: the x-axis is indexed by $\log(\kappa)$, the y-axis corresponds to the ratio of ESS per second of coordinate versus Zigzag samplers. The red line shows the efficiency ratio for the first component, the blue line for the second component, and the green line for the log probability.

We run CS and ZZ samplers for 160 seconds respectively and obtain about $1,000$ draws from each samplers. Figure 2.4 shows the first two components of the samples generated by CS and ZZ. In Figure 2.5, the lhs is the plot of values of log-density of the samples generated by CS (red) and ZZ (blue) and the rhs compares the last component of the generated samples.

2.5 Conclusion

We have introduced and studied the coordinate sampler as an alternative to the Zigzag sampler of Bierkens et al. (2016) and compared the efficiencies of the two samplers in terms of effective sample size over several models. In these examples, coordinate sampler exhibits a higher efficiency, which gain increases with the dimensionality of the target distribution, while enjoying the same ergodicity guarantees. While intuition about a component-wise implementation is at the basis of our proposal, the deeper reason for the improvement brought by CS needs further investigation.

We also note that, among PDMP-based MCMC samplers, this coordinate sampler is easy to scale for big data problem, similarly to the Zigzag sampler. Besides, taking advantage of the techniques set in Bierkens et al. (2018), it can also be applied to distributions defined on restricted domains. In this setting, since only one component of the target distribution is active between Poisson events, this may lower its efficiency relatively to ZS, especially in cases where the variances across the components are of different magnitudes. A appropriate reparametrization of the target distribution should alleviate this problem, and accelerate CS, being equivalent to a pre-conditioning of the velocity set. Another promising solution is to take advantage of the curvature of the target by Riemann manifold techniques (Girolami and Calderhead, 2011).

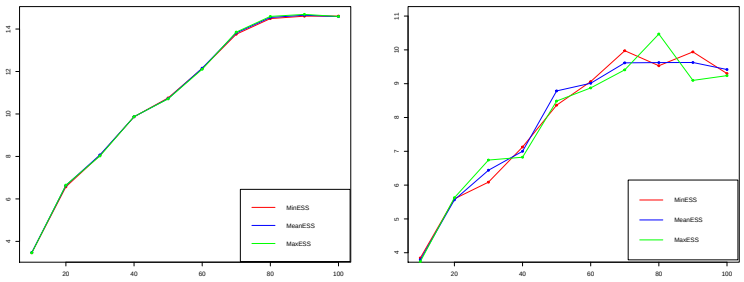

Figure 2.2: The lhs plot shows the results for MVN1 and the rhs for MVN2. The x-axis indexes the dimension d of the distribution, and the y-axis the efficiency ratios of CS over ZS in terms of minimal (red), mean (red), and maximal ESS(red) across the components.

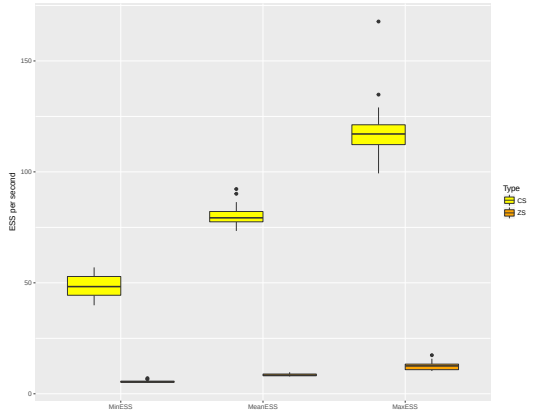

Figure 2.3: Comparison of CS versus ZS for the Bayesian logistic model: the y-axis represents the ESS per second.

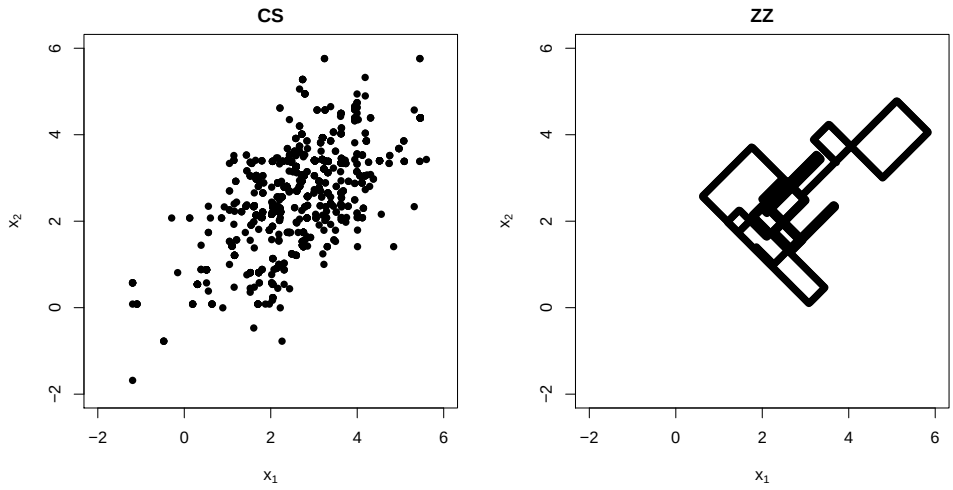

Figure 2.4: Samples generated by CS and ZZ samplers under same computation time for log-Gaussian Cox point process. Only the first two components are showed.

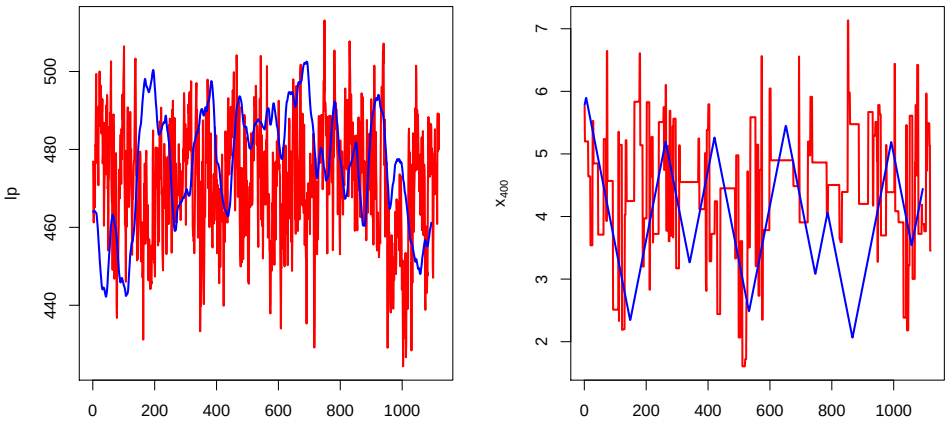

Figure 2.5: The plots of log-density and the final component of the samples generate by CS (red) and ZZ (blue) samplers under same computation time for log-Gaussian Cox point process.

2.6 Appendix

In this section, we give the details of the proofs of the theorems in this chapter.

2.6.1 Proof of Theorem 3

Lemma 1 *$\rho(d\mathbf{x}, \mathbf{v}) = \pi(d\mathbf{x})\varphi(d\mathbf{v})$ is the invariant distribution of Markov process induced by Coordinate Sampler.*

Proof: The generator of the Markov process induced by Coordinate Sampler is, by (Davis (1993), Theorem 26.14),

$$\begin{aligned}\mathcal{L}f(\mathbf{x},\mathbf{v}) &= \left\langle \frac{\partial f(\mathbf{x},\mathbf{v})}{\partial \mathbf{x}}, \mathbf{v} \right\rangle + \lambda(\mathbf{x},\mathbf{v}) \int_{\mathbf{x}'} \int_{v'} \left(f(\mathbf{x}',\mathbf{v}') - f(\mathbf{x},\mathbf{v})\right) Q((\mathbf{x},\mathbf{v}),(d\mathbf{x}',d\mathbf{v}')) \\ &= \left\langle \frac{\partial f(\mathbf{x},\mathbf{v})}{\partial \mathbf{x}}, \mathbf{v} \right\rangle + \lambda(\mathbf{x},\mathbf{v}) \sum_{i=1}^{2d} \frac{\lambda(\mathbf{x},-\mathbf{v}_i)}{\lambda(\mathbf{x})} f(\mathbf{x},\mathbf{v}_i) - \lambda(\mathbf{x},\mathbf{v}) f(\mathbf{x},\mathbf{v})\end{aligned}$$

Since we have

$$\begin{aligned}&\lambda(\mathbf{x},-\mathbf{v}) - \lambda(\mathbf{x},\mathbf{v}) \\ &= \lambda^{\text{ref}} + \langle \nabla U(\mathbf{x}), -\mathbf{v} \rangle_+ - \lambda_0 - \langle \nabla U(\mathbf{x}), \mathbf{v} \rangle_+ \\ &= \langle \nabla U(\mathbf{x}), \mathbf{v} \rangle\end{aligned}$$

as a result, $\int_{\mathbf{x}}\int_{\mathbf{v}} \mathcal{L}f(\mathbf{x},\mathbf{v})\pi(\mathbf{x})\varphi(\mathbf{v})d\mathbf{x}d\mathbf{v} = 0$, for all $f \in \mathcal{D}(\mathcal{L})$. That is,

$$
\begin{aligned}
&\int_{\mathbf{x}}\int_{v} \mathcal{L}f(\mathbf{x},\mathbf{v})\pi(\mathbf{x})\varphi(\mathbf{v})d\mathbf{x}d\mathbf{v} \\
&= \frac{1}{2d}\sum_{i=1}^{2d}\int_{\mathbf{x}} \mathcal{A}f(\mathbf{x},\mathbf{v}_i)\pi(\mathbf{x})d\mathbf{x} \\
&= \frac{1}{2d}\sum_{i=1}^{2d}\int_{\mathbf{x}} \left\langle \frac{\partial f(\mathbf{x},\mathbf{v})}{\partial \mathbf{x}}, \mathbf{v} \right\rangle \pi(\mathbf{x})d\mathbf{x} + \frac{1}{2d}\sum_{i=1}^{2d}\int_{\mathbf{x}} \lambda(\mathbf{x},\mathbf{v}_i)\sum_{j=1}^{2d}\frac{\lambda(\mathbf{x},-\mathbf{v}_j)}{\lambda(\mathbf{x})} f(\mathbf{x},\mathbf{v}_j)\pi(\mathbf{x})d\mathbf{x} \\
&\quad - \frac{1}{2d}\sum_{i=1}^{2d}\int_{\mathbf{x}} \lambda(\mathbf{x},\mathbf{v}_i) f(\mathbf{x},\mathbf{v}_i)\pi(\mathbf{x})d\mathbf{x} \\
&= \frac{1}{2d}\sum_{i=1}^{2d}\int_{\mathbf{x}} \langle -\nabla U(\mathbf{x}),\mathbf{v}\rangle f(\mathbf{x},\mathbf{v})\pi(\mathbf{x})d\mathbf{x} \\
&\quad + \frac{1}{2d}\sum_{j=1}^{2d}\int_{\mathbf{x}} \left(\sum_{i=1}^{2d}\lambda(\mathbf{x},\mathbf{v}_i)\right)\frac{\lambda(\mathbf{x},-\mathbf{v}_j)}{\lambda(\mathbf{x})} f(\mathbf{x},\mathbf{v}_j)\pi(\mathbf{x})d\mathbf{x} \\
&\quad - \frac{1}{2d}\sum_{i=1}^{2d}\int_{\mathbf{x}} \lambda(\mathbf{x},\mathbf{v}_i) f(\mathbf{x},\mathbf{v}_i)\pi(\mathbf{x})d\mathbf{x} \\
&= \frac{1}{2d}\sum_{i=1}^{2d}\int_{\mathbf{x}} \langle -\nabla U(\mathbf{x}),v\rangle f(\mathbf{x},\mathbf{v})\pi(\mathbf{x})d\mathbf{x} + \frac{1}{2d}\sum_{j=1}^{2d}\int_{\mathbf{x}} \lambda(\mathbf{x},-\mathbf{v}_j) f(\mathbf{x},\mathbf{v}_j)\pi(\mathbf{x})d\mathbf{x} \\
&\quad - \frac{1}{2d}\sum_{i=1}^{2d}\int_{\mathbf{x}} \lambda(\mathbf{x},\mathbf{v}_i) f(\mathbf{x},\mathbf{v}_i)\pi(\mathbf{x})d\mathbf{x} \\
&= \frac{1}{2}\sum_{i=1}^{2d}\int_{\mathbf{x}} \left\{\langle -\nabla U(\mathbf{x}),v\rangle + \lambda(\mathbf{x},-\mathbf{v}_i) - \lambda(\mathbf{x},\mathbf{v}_i)\right\} f(\mathbf{x},\mathbf{v}_i)\pi(\mathbf{x})d\mathbf{x} \\
&= 0
\end{aligned}
$$

By (Davis (1993), Theorem 34.7), ρ is the invariant distribution of the Markov chain induced by Coordinate Sampler.

Lemma 2 *For all $T > 0$, $\mathbf{z}_0 = (\mathbf{x}_0,\mathbf{v}_0) \in B(0,\frac{T}{6})\times\mathcal{V}$, and Borel set $A \subset B(0,\frac{T}{6})\times\mathcal{V}$*

$$\mathbb{P}(\mathbf{z}_0, \mathbf{Z}_T \in A) \geq C(T,d,\lambda^{ref})\int\int_A \varphi(d\mathbf{v})d\mathbf{x}$$

for some constant $C > 0$ depending only on T, d, λ^{ref}. Hence, all compacts are small and the Markov process induced by Coordinate Sampler is irreducible.

This lemma is just the Lemma 2 of Deligiannidis et al. (2017), we reproduces its proof for the Markov chain induced by Coordinate Sampler with some modification.

Proof Let E the event that there are exactly $d+2$ events during the time interval

$[0,T]$. Suppose $f : B(0,\frac{T}{6}) \times \mathcal{V} \to [0,\infty)$ be a bounded, positive function.Then

$$
\begin{aligned}
&\mathbb{E}^{\mathbf{z}}[f(\mathbf{Z}_T)] \\
&\geq \mathbb{E}^{\mathbf{z}}[f(\mathbf{Z}_T \mathbb{1}_E)] \\
&= \int_{\mathbf{v}_1} \varphi(d\mathbf{v}_1) \int_{t_1=0}^{T} dt_1 \Big(\frac{\lambda(\mathbf{x}_0 + t_1\mathbf{v}_0, -\mathbf{v}_1)}{\lambda(\mathbf{x}_0 + t_1\mathbf{v}_0)} \\
&\quad \exp\Big\{ - \int_{u_1=0}^{t_1} \lambda(\mathbf{x}_0 + u_1\mathbf{v}_0, \mathbf{v}_0) du_1 \Big\} \lambda(\mathbf{x}_0 + t_1\mathbf{v}_0, \mathbf{v}_0) \Big) \\
&\times \int_{\mathbf{v}_2} \varphi(d\mathbf{v}_2) \int_{t_2=0}^{T-T_1} dt_2 \Big(\frac{\lambda(\mathbf{x}_{T_1} + t_2\mathbf{v}_1, -\mathbf{v}_2)}{\lambda(\mathbf{x}_{T_1} + t_2\mathbf{v}_1)} \\
&\quad \exp\Big\{ - \int_{u_2=0}^{t_2} \lambda(\mathbf{x}_{T_1} + u_2\mathbf{v}_1, \mathbf{v}_1) du_2 \Big\} \lambda(\mathbf{x}_{T_1} + t_2\mathbf{v}_1, \mathbf{v}_1) \Big) \\
&\times \cdots \\
&\times \int_{\mathbf{v}_{d+2}} \varphi(d\mathbf{v}_{d+2}) \int_{t_{d+2}=0}^{T-T_{d+1}} dt_{d+2} \Big\{ \frac{\lambda(\mathbf{x}_{T_{d+1}} + t_{d+2}\mathbf{v}_{d+1}, -\mathbf{v}_{d+2})}{\lambda(\mathbf{x}_{T_{d+1}} + t_{d+2}\mathbf{v}_{d+1})} \\
&\times \exp\left\{ - \int_{u_{d+2}=0}^{t_{d+2}} \lambda(\mathbf{x}_{T_{d+1}} + u_{d+2}\mathbf{v}_{d+1}, \mathbf{v}_{d+1}) du_{d+2} \right\} \\
&\quad \times \lambda(\mathbf{x}_{T_{d+1}} + t_{d+2}\mathbf{v}_{d+1}, \mathbf{v}_{d+1}) \Big\} \\
&\times \exp\left\{ - \int_{u_{d+3}=0}^{T-T_{d+2}} \lambda(\mathbf{x}_{T_{d+2}} + u_{d+3}\mathbf{v}_{d+2}, \mathbf{v}_{d+2}) du_{d+3} \right\} \\
&\quad \times f\left(\mathbf{x}_{T_{d+2}} + (T - T_{d+2})\mathbf{v}_{d+2}, \mathbf{v}_{d+2} \right)
\end{aligned}
$$

where $T_i = \sum_{k=1}^{i} t_k$ and $\mathbf{x}_{T_i} = \mathbf{x}_0 + \sum_{k=1}^{i} t_k \mathbf{v}_{k-1}$. Since $\mathbf{x}_0 \in B(0,T)$, then $\mathbf{x}_t \in B(0,2T)$ for all $t \in [0,T]$. As a result, there exists constant $K \leq \infty$, such that

$$
K \geq \sup_{\mathbf{x} \in B(0,2T)} \Big| \nabla U(\mathbf{x}) \Big|
$$

Since $\lambda(\mathbf{x},\mathbf{v}) \geq \lambda^{\text{ref}}$ for all $(\mathbf{x},\mathbf{v}) \in \mathbb{R}^d \times \mathcal{V}$, then

$$
\begin{aligned}
&\mathbb{E}^{\mathbf{z}}[f(\mathbf{Z}_T)] \\
&\geq \int_{\mathbf{v}_1} \varphi(d\mathbf{v}_1) \int_{t_1=0}^{T} dt_1 \left\{ \frac{\lambda^{\text{ref}}}{\lambda^{\text{ref}}+K} \exp\left\{ -\int_{u_1=0}^{t_1} \left(\lambda^{\text{ref}}+K\right) du_1 \right\} \lambda^{\text{ref}} \right\} \\
&\times \int_{\mathbf{v}_2} \varphi(d\mathbf{v}_2) \int_{t_2=0}^{T-T_1} dt_2 \left\{ \frac{\lambda^{\text{ref}}}{\lambda^{\text{ref}}+K} \exp\left\{ -\int_{u_2=0}^{t_2} \left(\lambda^{\text{ref}}+K\right) du_2 \right\} \lambda^{\text{ref}} \right\} \\
&\times \cdots \\
&\times \int_{t_{d+2}=0}^{T-T_{d+1}} dt_{d+2} \left\{ \frac{\lambda^{\text{ref}}}{\lambda^{\text{ref}}+K} \exp\left\{ -\int_{u_{d+2}=0}^{t_{d+2}} \left(\lambda^{\text{ref}}+K\right) du_{d+2} \right\} \lambda^{\text{ref}} \right\} \\
&\times \int_{\mathbf{v}_{d+2}} \varphi(d\mathbf{v}_{d+2}) \exp\left\{ -\int_{u_{d+3}=0}^{T-T_{d+2}} \left(\lambda^{\text{ref}}+K\right) du_{d+3} \right\} f\left(\mathbf{x}_{T_{d+2}} + (T-T_{d+2})\mathbf{v}_{d+2}, \mathbf{v}_{d+2}\right) \\
&= \left(\frac{(\lambda^{\text{ref}})^2}{\lambda^{\text{ref}}+K} \right)^{d+2} \exp\left\{ -T\left(\lambda^{\text{ref}}+K\right) \right\} \int_{\mathbf{v}_1} \varphi(d\mathbf{v}_1) \int_{\mathbf{v}_2} \varphi(d\mathbf{v}_2) \cdots \int_{\mathbf{v}_{d+2}} \varphi(d\mathbf{v}_{d+2}) \\
&\times \int_{t_1=0}^{T} dt_1 \int_{t_2=0}^{T-T_1} dt_2 \cdots \int_{t_{d+2}=0}^{T-T_{d+1}} dt_{d+2} f\left(\mathbf{x}_{T_{d+2}} + (T-T_{d+2})\mathbf{v}_{d+2}, \mathbf{v}_{d+2}\right) \\
&= \left(\frac{(\lambda^{\text{ref}})^2}{\lambda^{\text{ref}}+K} \right)^{d+2} \exp\left\{ -T\left(\lambda^{\text{ref}}+K\right) \right\} \int_{\mathbf{v}_1} \varphi(d\mathbf{v}_1) \int_{\mathbf{v}_2} \varphi(d\mathbf{v}_2) \cdots \int_{\mathbf{v}_{d+2}} \varphi(d\mathbf{v}_{d+2}) \\
&\times \int_{t=\frac{5T}{6}}^{T} dt \times \int_{r_1=0}^{1} dr_1 \int_{r_2=0}^{1-r_1} dr_2 \cdots \int_{r_{d+1}=0}^{1-r_1-\cdots-r_d} dr_{d+1} \\
&\quad f(\mathbf{x}_0 + t\sum_{k=1}^{d+1} r_k \mathbf{v}_{k-1} + t(1-\sum_{k=1}^{d+1} r_k)\mathbf{v}_{d+1} + (T-t)\mathbf{v}_{d+2})
\end{aligned}
$$

Fix $t > \frac{5T}{6}$ and $\mathbf{v}_{d+2}$, then $\mathbf{x}' = \mathbf{x}_0 + (T-t)\mathbf{v}_{d+2}$ is fixed now. Since $\mathbf{x}_0 \in B(0, \frac{T}{6})$, then $\mathbf{x}' \in B(0, \frac{T}{3})$. For any $\mathbf{x}'' \in B(0, \frac{T}{6})$, then $\|\mathbf{x}' - \mathbf{x}''\| < \frac{T}{2}$. There exist $r_1^*, \cdots, r_{d+1}^*$ and $\mathbf{v}_1^* \cdots, \mathbf{v}_{d+1}^*$ such that

$$
t\left(r_1^* \mathbf{v}_0 + r_2^* \mathbf{v}_1^* + \cdots + r_{d+1}^* \mathbf{v}_d^* + (1 - \sum_{k=1}^{d+1} r_k^*)\mathbf{v}_{d+1}^* \right) = \mathbf{x}'' - \mathbf{x}',
$$

$$
r_k^* \in [0,1], \quad \mathbf{v}_k^* \in \mathcal{V}, \text{ for } k = 1, \cdots, d+1, \text{ and } \sum_{k=1}^{d+1} r_k^* \leq 1
$$

Let $\mathbf{R} = (R_1, \cdots, R_{d+1}, R_{d+2}) \sim \text{Dirichlet}(1, 1, \cdots, 1)$ and $\mathbf{V} = (\mathbf{v}_1, \cdots, \mathbf{v}_{d+1}) \sim \varphi^{d+1}$ be independent, then

$$
\begin{aligned}
&\int_{\mathcal{V}} \varphi(d\mathbf{v}_1) \cdots \int_{\mathcal{V}} \varphi(d\mathbf{v}_{d+1}) \int_{r_1=0}^{1} dr_1 \int_{r_2=0}^{1-r_1} dr_2 \cdots \int_{r_{d+1}=0}^{1-\sum_{k=1}^{d} r_k} dr_{d+1} \\
&\times \mathbb{1}_{B(\mathbf{x}'', \delta)} \left(\mathbf{x}' + t \left(r_1 \mathbf{v}_0 + r_2 \mathbf{v}_1 + \cdots + r_{d+1} \mathbf{v}_d + (1 - \sum_{k=1}^{d+1} r_k) \mathbf{v}_{d+1} \right) \right) \\
&= \Gamma(d+2) \mathbb{P} \left\{ \left| \mathbf{x}' + t \left(R_1 \mathbf{v}_0 + R_2 \mathbf{V}_1 + \cdots + R_{d+1} \mathbf{V}_d + (1 - \sum_{k=1}^{d+1} R_k) \mathbf{V}_{d+1} \right) - \mathbf{x}'' \right| \leq \delta \right\} \\
&= \Gamma(d+2) \mathbb{P} \Big\{ \Big| (R_1 \mathbf{v}_0 + R_2 \mathbf{V}_1 + \cdots + R_{d+1} \mathbf{V}_d + (1 - \sum_{k=1}^{d+1} R_k) \mathbf{V}_{d+1}) \\
&\qquad - (r_1^* \mathbf{v}_0 + r_2^* \mathbf{v}_1^* + \cdots + r_{d+1}^* \mathbf{v}_d^* + (1 - \sum_{k=1}^{d+1} r_k^*) \mathbf{v}_{d+1}^*) \Big| \leq \frac{\delta}{t} \Big\} \\
&\geq \Gamma(d+2) \mathbb{P} \Big\{ \{ \Big| (R_1 \mathbf{v}_0 + R_2 \mathbf{V}_1 + \cdots + R_{d+1} \mathbf{V}_d + (1 - \sum_{k=1}^{d+1} R_k) \mathbf{V}_{d+1}) - \Big(r_1^* \mathbf{v}_0 + r_2^* \mathbf{v}_1^* \\
&\qquad + \cdots + r_{d+1}^* \mathbf{v}_d^* + (1 - \sum_{k=1}^{d+1} r_k^*) \mathbf{v}_{d+1}^* \Big) \Big| \leq \frac{\delta}{t} \} \cap \{ \mathbf{V_k} = \mathbf{v}_k^*, k = 1, \cdots, d+1 \} \Big\} \\
&\geq \left(\frac{\lambda^{\text{ref}}}{2d\lambda^{\text{ref}} + K} \right)^{d+1} \Gamma(d+2) \mathbb{P} \Big\{ \Big| (R_1 - r_1^*) \mathbf{v}_0 + \sum_{k=2}^{d+1} (R_k - r_k^*) \mathbf{v}_{k-1}^* \\
&\qquad - \sum_{k=1}^{d+1} (R_k - r_k^*) \mathbf{v}_{d+1}^* \Big| \leq \frac{\delta}{t} \Big\} \\
&\geq \left(\frac{\lambda^{\text{ref}}}{2d\lambda^{\text{ref}} + K} \right)^{d+1} \Gamma(d+2) \mathbb{P} \Big\{ \Big| (R_1 - r_1^*) \mathbf{v}_0 \Big| + \sum_{k=2}^{d+1} \Big| (R_k - r_k^*) \mathbf{v}_{k-1}^* \Big| \\
&\qquad + \sum_{k=1}^{d+1} \Big| (R_k - r_k^*) \mathbf{v}_{d+1}^* \Big| \leq \frac{\delta}{t} \Big\} \\
&\geq \left(\frac{\lambda^{\text{ref}}}{2d\lambda^{\text{ref}} + K} \right)^{d+1} \Gamma(d+2) \mathbb{P} \Big\{ \Big| (R_1 - r_1^*) \mathbf{v}_0 \Big| \leq \frac{\delta}{2(d+1)t}, \\
&\qquad \Big| (R_k - r_k^*) \mathbf{v}_{k-1}^* \Big| \leq \frac{\delta}{2(d+1)t}, k = 2, \cdots, d+1, \Big| (R_k - r_k^*) \mathbf{v}_{d+1}^* \\
&\qquad \Big| \leq \frac{\delta}{2(d+1)t}, k = 1, \cdots, d+1 \Big\} \\
&= \left(\frac{\lambda^{\text{ref}}}{2d\lambda^{\text{ref}} + K} \right)^{d+1} \Gamma(d+2) \mathbb{P} \Big\{ \Big| R_k - r_k^* \Big| \leq \frac{\delta}{2(d+1)t}, k = 1, \cdots, d+1 \Big\} \\
&\geq \left(\frac{\lambda^{\text{ref}}}{2d\lambda^{\text{ref}} + K} \right)^{d+1} \Gamma(d+2) \mathbb{P} \Big\{ \Big| R_k - r_k^* \Big| \leq \frac{\delta}{2(d+1)T}, k = 1, \cdots, d+1 \Big\} \\
&\geq \left(\frac{\lambda^{\text{ref}}}{2d\lambda^{\text{ref}} + K} \right)^{d+1} \Gamma(d+2) C_1(d, T, \delta)
\end{aligned}
$$

As a result, we have, for any $t > \frac{5T}{6}$,

$$\begin{aligned}
&\int_{\mathcal{V}} \varphi(d\mathbf{v}_1) \cdots \int_{\mathcal{V}} \varphi(d\mathbf{v}_{d+1}) \int_{r_1=0}^{1} dr_1 \int_{r_2=0}^{1-r_1} dr_2 \cdots \\
&\times \int_{r_{d+1}=0}^{1-\sum_{k=1}^{d} r_k} dr_{d+1} f(\mathbf{x}_0 + t\sum_{k=1}^{d+1} r_k \mathbf{v}_{k-1} + t(1 - \sum_{k=1}^{d+1} r_k)\mathbf{v}_{d+1} + (T-t)\mathbf{v}_{d+2}) \\
&\geq C_2(T,d) \int_{B(0,\frac{T}{6})} f(\mathbf{x}'', \mathbf{v}_{d+2}) d\mathbf{x}''
\end{aligned}$$

$$\begin{aligned}
&\mathbb{E}^{\mathbf{z}}[f(\mathbf{Z}_T)] \\
&\geq \left(\frac{(\lambda^{\text{ref}})^2}{\lambda^{\text{ref}} + K}\right)^{d+2} \exp\left\{-T\left(\lambda^{\text{ref}} + K\right)\right\} \int_{\mathcal{V}} \varphi(d\mathbf{v}_{d+2}) \times \int_{t=\frac{5T}{6}}^{T} dt \int_{B(0,\frac{T}{6})} f(\mathbf{x}'', \mathbf{v}_{d+2}) d\mathbf{x}'' \\
&\geq C_3(T,d,\lambda^{\text{ref}}) \int_{\mathcal{V}} \varphi(d\mathbf{v}_{d+2}) \int_{t=\frac{5T}{6}}^{T} dt \int_{B(0,\frac{T}{6})} f(\mathbf{x}'', \mathbf{v}_{d+2}) d\mathbf{x}'' \\
&\geq C_4(T,d,\lambda^{\text{ref}}) \int_{\mathcal{V}} \varphi(d\mathbf{v}_{d+2}) \int_{B(0,\frac{T}{6})} f(\mathbf{x}'', \mathbf{v}_{d+2}) d\mathbf{x}''
\end{aligned}$$

As a result, for any Borel set $A \subset \mathbb{R}^d \times \mathcal{V}$ and $\mathbf{z}_0 = (\mathbf{x}_0, \mathbf{v}_0) \in \mathbb{R}^d \times \mathcal{V}$, set $f = \mathbb{1}_A$ and use above arguments, we have

$$\mathbb{P}(\mathbf{z}_0, \mathbf{Z}_T \in A) \geq C_4(T,d,\lambda^{\text{ref}}) \int \int_A \varphi(d\mathbf{v}) d\mathbf{x}$$

As a result, for any $R > 0$, the set $B(0,R) \times \mathcal{V}$ is petite. Hence, any compact set is small and irreducibility follows.

Proof of Theorem 3: Using the same arguments as that of Lemma 3 of Deligiannidis et al. (2017) and by above **Lemma 2**, the Markov process induced by our Coordinate Sampler is ergodic, hence its invariant distribution is unique. By **Lemma 1**, $\rho(d\mathbf{z})$ is the unique invariant distribution of the coordinate sampler.

2.6.2 Proof of Theorem 4

In this section, we use the same techniques developed by Deligiannidis et al. (2017). We'll reproduces the proof that V is the desired Lyapunov function, since there is some difference between our proof and the original one in Deligiannidis et al. (2017).

Lemma 3 (Down et al. (1995), Theorem 5.2) *Let $\{\mathbf{Z}_t : t \geq 0\}$ be a Borel right Markov process taking values in a locally compact, separable metric space $\mathcal{Z}$ and assume it is non-explosive, irreducible and aperiodic. Let $(\mathcal{L}, \mathcal{D}(\mathcal{L}))$ be its extended generator. Suppose that there exists a measurable function $V : \mathcal{Z} \to [1, \infty)$ such that $V \in \mathcal{D}(\mathcal{L})$, and that for a petite set $C \in \mathcal{B}(\mathcal{Z})$ and constants $b, c > 0$, we have*

$$\mathcal{L}V \leq -cV + b\mathbb{I}_C,$$

Then $\{\mathbf{Z}_t : t \geq 0\}$ is V-uniformly ergodic.

Proof of Theorem 4: By the Section 5.1 in Deligiannidis et al. (2017), V defined in the paper belongs to the extended generator $\mathcal{D}(\mathcal{L})$, given Assumptions $A.1 - A.4$. We next show that V is a Lyapunov function.

Case 1: $\langle \nabla U(\mathbf{x}), \mathbf{v} \rangle > 0$. $V(\mathbf{x}, \mathbf{v}) = \frac{e^{U(\mathbf{x})/2}}{\sqrt{\lambda^{\text{ref}}}}$ and $\nabla_{\mathbf{x}} V(\mathbf{x}, \mathbf{v}) = \frac{1}{2} V(\mathbf{x}, \mathbf{v}) \nabla U(\mathbf{x})$.

$$
\begin{aligned}
&\mathcal{L}V(\mathbf{x}, \mathbf{v}) \\
&= \frac{1}{2} V(\mathbf{x}, \mathbf{v}) \langle \nabla U(\mathbf{x}), \mathbf{v} \rangle + \left(\lambda^{\text{ref}} + \langle \nabla U(\mathbf{x}), \mathbf{v} \rangle_+ \right) \\
&\times \sum_{i=1}^{d} \frac{\lambda^{\text{ref}} + \langle \nabla U(\mathbf{x}), -\mathbf{v}_i \rangle_+}{2d\lambda^{\text{ref}} + \left| \nabla U(\mathbf{x}) \right|_1} V(\mathbf{x}, \mathbf{v}) \sqrt{\frac{\lambda^{\text{ref}}}{\lambda^{\text{ref}} + \langle \nabla U(\mathbf{x}), -\mathbf{v}_i \rangle_+}} \\
&- \left(\lambda^{\text{ref}} + \langle \nabla U(\mathbf{x}), \mathbf{v} \rangle_+ \right) V(\mathbf{x}, \mathbf{v}) \\
&= V(\mathbf{x}, \mathbf{v}) \Big\{ -\frac{1}{2} \langle \nabla U(\mathbf{x}), \mathbf{v} \rangle - \lambda^{\text{ref}} + \left(\lambda^{\text{ref}} + \langle \nabla U(\mathbf{x}), \mathbf{v} \rangle \right) \\
&\qquad \times \sum_{i=1}^{d} \frac{\lambda^{\text{ref}} + \sqrt{\lambda^{\text{ref}} \left(\lambda^{\text{ref}} + \left| \nabla_i U(\mathbf{x}) \right| \right)}}{2d\lambda^{\text{ref}} + \left| \nabla U(\mathbf{x}) \right|_1} \Big\} \\
&= V(\mathbf{x}, \mathbf{v}) \Big\{ \lambda^{\text{ref}} \left[\sum_{i=1}^{d} \frac{\lambda^{\text{ref}} + \sqrt{\lambda^{\text{ref}} \left(\lambda^{\text{ref}} + \left| \nabla_i U(\mathbf{x}) \right| \right)}}{2d\lambda^{\text{ref}} + \left| \nabla U(\mathbf{x}) \right|_1} - 1 \right] \\
&+ \langle \nabla U(\mathbf{x}), \mathbf{v} \rangle \left[\sum_{i=1}^{d} \frac{\lambda^{\text{ref}} + \sqrt{\lambda^{\text{ref}} \left(\lambda^{\text{ref}} + \left| \nabla_i U(\mathbf{x}) \right| \right)}}{2d\lambda^{\text{ref}} + \left| \nabla U(\mathbf{x}) \right|_1} - \frac{1}{2} \right] \Big\}
\end{aligned}
\tag{2.3}
$$

Case 2: $\langle \nabla U(\mathbf{x}), \mathbf{v} \rangle < 0$.

$$
V(\mathbf{x}, \mathbf{v}) = \frac{e^{U(\mathbf{x})/2}}{\sqrt{\lambda^{\text{ref}} + \langle \nabla U(\mathbf{x}), -\mathbf{v} \rangle}}
$$

$$
\frac{\partial V(\mathbf{x}, \mathbf{v})}{\partial \mathbf{x}} = \frac{1}{2} V(\mathbf{x}, \mathbf{v}) \nabla U(\mathbf{x}) + \frac{1}{2} V(\mathbf{x}, \mathbf{v}) \frac{\Delta U(\mathbf{x}) \mathbf{v}}{\lambda^{\text{ref}} + \langle \nabla U(\mathbf{x}), -\mathbf{v} \rangle}
$$

$$
\begin{aligned}
&\mathcal{L}V(\mathbf{x}, \mathbf{v}) \\
&= \frac{1}{2} V(\mathbf{x}, \mathbf{v}) \langle \nabla U(\mathbf{x}), \mathbf{v} \rangle + \frac{1}{2} V(\mathbf{x}, \mathbf{v}) \frac{\langle \mathbf{v}, \Delta U(\mathbf{x}) \mathbf{v} \rangle}{\lambda^{\text{ref}} + \langle \nabla U(\mathbf{x}), -\mathbf{v} \rangle} - \lambda^{\text{ref}} V(\mathbf{x}, \mathbf{v}) \\
&+ \lambda^{\text{ref}} \sum_{i=1}^{2d} \frac{\lambda^{\text{ref}} + \langle \nabla U(\mathbf{x}), -\mathbf{v}_i \rangle_+}{2d\lambda^{\text{ref}} + \left| \nabla U(\mathbf{x}) \right|_1} V(\mathbf{x}, \mathbf{v}) \sqrt{\frac{\lambda^{\text{ref}} + \langle \nabla U(\mathbf{x}), -\mathbf{v} \rangle}{\lambda^{\text{ref}} + \langle \nabla U(\mathbf{x}), -\mathbf{v}_i \rangle_+}} \\
&= V(\mathbf{x}, \mathbf{v}) \left\{ \frac{1}{2} \langle \nabla U(\mathbf{x}), \mathbf{v} \rangle - \lambda^{\text{ref}} + \frac{1}{2} \frac{\langle \mathbf{v}, \Delta U(\mathbf{x}) \mathbf{v} \rangle}{\lambda^{\text{ref}} + \langle \nabla U(\mathbf{x}), -\mathbf{v} \rangle} \right\} \\
&+ V(\mathbf{x}, \mathbf{v}) \left\{ \lambda^{\text{ref}} \sqrt{\lambda^{\text{ref}} + \langle \nabla U(\mathbf{x}), -\mathbf{v} \rangle} \sum_{i=1}^{d} \frac{\sqrt{\lambda^{\text{ref}}} + \sqrt{\lambda^{\text{ref}} + \left| \nabla_i U(\mathbf{x}) \right|}}{2d\lambda^{\text{ref}} + \left| \nabla U(\mathbf{x}) \right|_1} \right\}
\end{aligned}
$$

Case 3: $\langle U(\mathbf{x}), \mathbf{v} \rangle = 0$. The generator is defined as

$$
\mathcal{L}V(\mathbf{x}, \mathbf{v}) = \frac{dV(\mathbf{x} + t\mathbf{v}, \mathbf{v})}{dt} \Big|_{t=0+} + \lambda(\mathbf{x}, \mathbf{v}) \sum_{\mathbf{v}' \in \mathcal{V}} \frac{\lambda(\mathbf{x}, -\mathbf{v}')}{\lambda(\mathbf{x})} \left(f(\mathbf{x}, \mathbf{v}') - f(\mathbf{x}, \mathbf{v}) \right)
$$

(i): If $\langle \mathbf{v}, \Delta U(\mathbf{x})\mathbf{v}\rangle > 0$, then

$$\begin{aligned}
&\frac{dV(\mathbf{x}+t\mathbf{v},\mathbf{v})}{dt}\Big|_{t=0+} \\
&= \lim_{t\to 0+}\frac{1}{t}\left\{\frac{e^{U(\mathbf{x}+t\mathbf{v})/2}}{\sqrt{\lambda^{\text{ref}}+\langle\nabla U(\mathbf{x}+t\mathbf{v}),-\mathbf{v}\rangle_+}}-\frac{e^{U(\mathbf{x})/2}}{\sqrt{\lambda^{\text{ref}}}}\right\} \\
&= \frac{1}{2}\frac{e^{U(\mathbf{x}+t\mathbf{v})/2}}{\sqrt{\lambda^{\text{ref}}+\langle\nabla U(\mathbf{x}+t\mathbf{v}),-\mathbf{v}\rangle_+}}\langle\nabla U(\mathbf{x}+t\mathbf{v}),\mathbf{v}\rangle\Big|_{t=0+}=0
\end{aligned}$$

(ii): $\langle \mathbf{v}, \Delta U(\mathbf{x})\mathbf{v}\rangle < 0$, then

$$\begin{aligned}
&\frac{dV(\mathbf{x}+t\mathbf{v},\mathbf{v})}{dt}\Big|_{t=0+} \\
&= \lim_{t\to 0+}\frac{1}{t}\left\{\frac{e^{U(\mathbf{x}+t\mathbf{v})/2}}{\sqrt{\lambda^{\text{ref}}+\langle\nabla U(\mathbf{x}+t\mathbf{v}),-\mathbf{v}\rangle_+}}-\frac{e^{U(\mathbf{x})/2}}{\sqrt{\lambda^{\text{ref}}}}\right\} \\
&= \frac{1}{2}\frac{e^{U(\mathbf{x}+t\mathbf{v})/2}}{\sqrt{\lambda^{\text{ref}}+\langle\nabla U(\mathbf{x}+t\mathbf{v}),-\mathbf{v}\rangle_+}}\langle\nabla U(\mathbf{x}+t\mathbf{v}),\mathbf{v}\rangle\Big|_{t=0+} \\
&\quad -\frac{1}{2}\frac{e^{U(\mathbf{x}+t\mathbf{v})/2}\langle\mathbf{v},-\Delta U(\mathbf{x})\mathbf{v}\rangle}{\left(\lambda^{\text{ref}}+\langle\nabla U(\mathbf{x}+t\mathbf{v}),-\mathbf{v}\rangle_+\right)^{3/2}}\Big|_{t=0+} \\
&= 0+\frac{1}{2}\frac{e^{U(\mathbf{x})/2}\langle\mathbf{v},\Delta U(\mathbf{x})\mathbf{v}\rangle}{(\sqrt{\lambda^{\text{ref}}})^3}
\end{aligned}$$

As a result,

$$\frac{dV(\mathbf{x}+t\mathbf{v},\mathbf{v})}{dt}\Big|_{t=0+}=-\frac{1}{2}\frac{e^{U(\mathbf{x})/2}\langle\mathbf{v},-\Delta U(\mathbf{x})\mathbf{v}\rangle_+}{(\sqrt{\lambda^{\text{ref}}})^3}$$

$$\begin{aligned}
&\mathcal{L}V(\mathbf{x},\mathbf{v}) \\
&= V(\mathbf{x},\mathbf{v})\left\{-\frac{1}{2}\frac{\langle\mathbf{v},-\Delta U(\mathbf{x})\mathbf{v}\rangle_+}{\lambda^{\text{ref}}}+\lambda^{\text{ref}}\left[\sum_{i=1}^d\frac{\lambda^{\text{ref}}+\sqrt{\lambda^{\text{ref}}\left(\lambda^{\text{ref}}+\left|\nabla_i U(\mathbf{x})\right|\right)}}{2d\lambda^{\text{ref}}+\left|\nabla U(\mathbf{x})\right|_1}-1\right]\right\}
\end{aligned}$$

Condition 1: $\overline{\lim}_{|x|\to\infty}\|\Delta U(x)\| \le \alpha_1 < \infty$, $\underline{\lim}_{|x|\to\infty}\|\nabla U(x)\| = \infty$ and $\lambda^{\text{ref}} > \sqrt{8\alpha_1}$.

$$\begin{aligned}
&\sum_{i=1}^d\frac{\lambda^{\text{ref}}+\sqrt{\lambda^{\text{ref}}\left(\lambda^{\text{ref}}+\left|\nabla U(\mathbf{x})\right|_1/d\right)}}{2d\lambda^{\text{ref}}+\left|\nabla U(\mathbf{x})\right|_1}<\frac{1}{2} \\
\iff\ &\frac{d\lambda^{\text{ref}}+d\sqrt{\lambda^{\text{ref}}\left(\lambda^{\text{ref}}+\left|\nabla U(\mathbf{x})\right|_1/d\right)}}{2d\lambda^{\text{ref}}+\left|\nabla U(\mathbf{x})\right|_1}<\frac{1}{2} \\
\iff\ &2\sqrt{d}\sqrt{d(\lambda^{\text{ref}})^2+\lambda^{\text{ref}}\left|\nabla U(\mathbf{x})\right|_1}<\left|\nabla U(\mathbf{x})\right|_1 \\
\iff\ &4d^2(\lambda^{\text{ref}})^2+4d\lambda^{\text{ref}}\left|\nabla U(\mathbf{x})\right|_1<\left|\nabla U(\mathbf{x})\right|_1^2 \\
\iff\ &\left|\nabla U(\mathbf{x})\right|_1>2(\sqrt{2}+1)d\lambda^{\text{ref}}
\end{aligned}$$

Denote $K = K_1 \cup K_2 \cup K_3$, where $K_1 = \{\mathbf{x} : \left|\nabla U(\mathbf{x})\right|_1 \leq 2(\sqrt{2}+1)d\lambda^{\text{ref}}\}$, $K_1 = \{\mathbf{x} : \left|\nabla U(\mathbf{x})\right|_1 < 16d\lambda^{\text{ref}}\}$ and $K_3 = \{\mathbf{x} : \|\Delta U(\mathbf{x})\| \leq 2\alpha_1\}$. On K^c, we have

$$\begin{aligned}
&\sum_{i=1}^{d} \frac{\lambda^{\text{ref}} + \sqrt{\lambda^{\text{ref}}\left(\lambda^{\text{ref}} + \left|\nabla_i U(\mathbf{x})\right|\right)}}{2d\lambda^{\text{ref}} + \left|\nabla U(\mathbf{x})\right|_1} \\
\leq &\sum_{i=1}^{d} \frac{\lambda^{\text{ref}} + \sqrt{\lambda^{\text{ref}}\left(\lambda^{\text{ref}} + \left|\nabla U(\mathbf{x})\right|_1/d\right)}}{2d\lambda^{\text{ref}} + \left|\nabla U(\mathbf{x})\right|_1} \text{ (Jensen's inequality)} \\
< &\frac{1}{2}
\end{aligned}$$

As a result, for case 1 and case 3, we have

$$\frac{\mathcal{L}V(\mathbf{x},\mathbf{v})}{V(\mathbf{x},\mathbf{v})} \leq -\frac{1}{2}\lambda^{\text{ref}}, \quad \text{On } K^c \times \mathcal{V}$$

For case 2, we have

$$\begin{aligned}
&\frac{\mathcal{L}V(\mathbf{x},\mathbf{v})}{V(\mathbf{x},\mathbf{v})} \\
&= \left\{\frac{1}{2}\langle \nabla U(\mathbf{x}), \mathbf{v}\rangle - \lambda^{\text{ref}} + \frac{1}{2}\frac{\langle \mathbf{v}, \Delta U(\mathbf{x})\mathbf{v}\rangle}{\lambda^{\text{ref}} + \langle \nabla U(\mathbf{x}), -\mathbf{v}\rangle}\right\} \\
&+ \left\{\lambda^{\text{ref}}\sqrt{\lambda^{\text{ref}} + \langle \nabla U(\mathbf{x}), -\mathbf{v}\rangle}\sum_{i=1}^{d}\frac{\sqrt{\lambda^{\text{ref}}} + \sqrt{\lambda^{\text{ref}} + \left|\nabla_i U(\mathbf{x})\right|}}{2d\lambda^{\text{ref}} + \left|\nabla U(\mathbf{x})\right|_1}\right\} \\
&\leq \left\{\frac{1}{2}\langle \nabla U(\mathbf{x}), \mathbf{v}\rangle - \lambda^{\text{ref}} + \frac{\alpha_1}{\lambda^{\text{ref}} + \langle \nabla U(\mathbf{x}), -\mathbf{v}\rangle}\right\} \\
&+ \left\{\lambda^{\text{ref}}\sqrt{\lambda^{\text{ref}} + \langle \nabla U(\mathbf{x}), -\mathbf{v}\rangle}\frac{d\sqrt{\lambda^{\text{ref}}} + d\sqrt{\lambda^{\text{ref}} + \left|\nabla U(\mathbf{x})\right|_1/d}}{2d\lambda^{\text{ref}} + \left|\nabla U(\mathbf{x})\right|_1}\right\} \text{ (Jensen's Inequality)} \\
&\leq \left\{\frac{1}{2}\langle \nabla U(\mathbf{x}), \mathbf{v}\rangle - \lambda^{\text{ref}} + \frac{\alpha_1}{\lambda^{\text{ref}} + \langle \nabla U(\mathbf{x}), -\mathbf{v}\rangle}\right\} \\
&+ \sqrt{\lambda^{\text{ref}} + \langle \nabla U(\mathbf{x}), -\mathbf{v}\rangle} \times \left\{\frac{\sqrt{\lambda^{\text{ref}}}}{2} + \frac{\sqrt{d\lambda^{\text{ref}}}}{\sqrt{2d + \left|\nabla U(\mathbf{x})\right|_1/\lambda^{\text{ref}}}}\right\} \\
&\leq \left\{\frac{1}{2}\langle \nabla U(\mathbf{x}), \mathbf{v}\rangle - \lambda^{\text{ref}} + \frac{\alpha_1}{\lambda^{\text{ref}} + \langle \nabla U(\mathbf{x}), -\mathbf{v}\rangle}\right\} \\
&+ \sqrt{\lambda^{\text{ref}} + \langle \nabla U(\mathbf{x}), -\mathbf{v}\rangle} \times \left\{\frac{\sqrt{\lambda^{\text{ref}}}}{2} + \frac{\sqrt{\lambda^{\text{ref}}}}{4}\right\} (\text{ since } \left|\nabla U(\mathbf{x})\right|_1 > 16d\lambda^{\text{ref}}) \\
&= \frac{1}{2}\langle \nabla U(\mathbf{x}), \mathbf{v}\rangle - \lambda^{\text{ref}} + \frac{\alpha_1}{\lambda^{\text{ref}} + \langle \nabla U(\mathbf{x}), -\mathbf{v}\rangle} + \frac{3}{4}\sqrt{\lambda^{\text{ref}}}\sqrt{\lambda^{\text{ref}} + \langle \nabla U(\mathbf{x}), -\mathbf{v}\rangle}
\end{aligned}$$

Denote $w_0 = \langle \nabla U(\mathbf{x}), -\mathbf{v}\rangle$ and define $g : \mathbb{R}^+ \to \mathbb{R}$ as

$$g(w) = -\frac{1}{2}w - \lambda^{\text{ref}} + \frac{\alpha_1}{\lambda^{\text{ref}} + w} + \frac{3}{4}\sqrt{\lambda^{\text{ref}}}\sqrt{\lambda^{\text{ref}} + w}$$

$$g'(w) = -\frac{1}{2} - \frac{\alpha_1}{\lambda^{(\text{ref}} + w)^2} + \frac{3}{8}\frac{\sqrt{\lambda^{\text{ref}}}}{\sqrt{\lambda^{\text{ref}} + w}} < -\frac{1}{8} < 0$$

As a result, g is a decreasing function on $[0, \infty)$, and

$$g(w_0) \leq g(0) = -\lambda^{\text{ref}} + \frac{\alpha_1}{\lambda^{\text{ref}}} + \frac{3}{4}\lambda^{\text{ref}} \leq -\frac{1}{8}\lambda^{\text{ref}}, \quad (\text{since } \lambda^{\text{ref}} > \sqrt{8\alpha_1})$$

As a result, on $K^c \times \mathcal{V}$, we have

$$\mathcal{L}V(\mathbf{x}, \mathbf{v}) \leq -\frac{1}{8}\lambda^{\text{ref}} V(\mathbf{x}, \mathbf{v})$$

Since K is compact and $\mathcal{V}$ is finite, hence, $K \times \mathcal{V}$ is compact. As a result, there exists $b > 0$ such that

$$\mathcal{L}V(\mathbf{x}, \mathbf{v}) \leq b, \quad \text{for all } (\mathbf{x}, \mathbf{v}) \in K \times \mathcal{V}$$

Hence, under condition 1, there exist constants $b > 0$, $c > 0$ and function V such that

$$\mathcal{L}V(\mathbf{x}, \mathbf{v}) \leq -cV(\mathbf{x}, \mathbf{v}) + b\mathbb{I}_{K \times \mathcal{V}}$$

Condtion 2: $\underline{\lim}_{|\mathbf{x}| \to \infty} \left|\nabla U(\mathbf{x})\right|_1 = 2\alpha_2 > 0$, $\overline{\lim}_{|x| \to \infty} \|\Delta U(x)\| = 0$ and $\lambda^{\text{ref}} < \frac{\alpha_2}{14d}$. Denote $K_1 = \{\mathbf{x} : \left|\nabla U(\mathbf{x})\right|_1 \leq \alpha_2\}$, by the same arguments as above, we have, on $(\mathbf{x}, \mathbf{v}) \in K_1^c \times \mathcal{V}$,

$$\begin{aligned}
&\sum_{i=1}^{d} \frac{\lambda^{\text{ref}} + \sqrt{\lambda^{\text{ref}}\left(\lambda^{\text{ref}} + \left|\nabla_i U(\mathbf{x})\right|\right)}}{2d\lambda^{\text{ref}} + \left|\nabla U(\mathbf{x})\right|_1} \\
\leq &\sum_{i=1}^{d} \frac{\lambda^{\text{ref}} + \sqrt{\lambda^{\text{ref}}\left(\lambda^{\text{ref}} + \left|\nabla U(\mathbf{x})\right|_1/d\right)}}{2d\lambda^{\text{ref}} + \left|\nabla U(\mathbf{x})\right|_1} \text{ (Jensen's inequality)} \\
< &\frac{1}{2} \quad (\text{since } \left|\nabla U(\mathbf{x})\right|_1 > \alpha_2 > 14d\lambda^{\text{ref}} > 2(\sqrt{2}+1)d\lambda^{\text{ref}})
\end{aligned}$$

In cases 1 and 3, we have, for all $(\mathbf{x}, \mathbf{v}) \in K_1^c \times \mathcal{V}$

$$\mathcal{L}V(\mathbf{x}, \mathbf{v}) \leq -\frac{1}{2}V(\mathbf{x}, \mathbf{v})$$

Since $\overline{\lim}_{|x| \to \infty} \|\Delta U(x)\| = 0$, there exist $M > 0$ and *epsilon* $< (\lambda^{\text{ref}})^2/8$, such that $\|\Delta U(x)\| < \epsilon$, for all $x \in K_2^c = \{\mathbf{x} : \left|\mathbf{x}\right| > M\}$. Define $K = K_1 \cup K_2$, then, in case

2, for all $(\mathbf{x}, \mathbf{v}) \in K^c \times \mathcal{V}$

$$
\begin{aligned}
&\frac{\mathcal{L}V(\mathbf{x},\mathbf{v})}{V(\mathbf{x},\mathbf{v})} \\
&= \left\{ \frac{1}{2}\langle \nabla U(\mathbf{x}), \mathbf{v}\rangle - \lambda^{\text{ref}} + \frac{1}{2}\frac{\langle \mathbf{v}, \Delta U(\mathbf{x})\mathbf{v}\rangle}{\lambda^{\text{ref}} + \langle \nabla U(\mathbf{x}), -\mathbf{v}\rangle} \right\} \\
&+ \left\{ \lambda^{\text{ref}} \sqrt{\lambda^{\text{ref}} + \langle \nabla U(\mathbf{x}), -\mathbf{v}\rangle} \sum_{i=1}^{d} \frac{\sqrt{\lambda^{\text{ref}}} + \sqrt{\lambda^{\text{ref}} + \left|\nabla_i U(\mathbf{x})\right|}}{2d\lambda^{\text{ref}} + \left|\nabla U(\mathbf{x})\right|_1} \right\} \\
&\leq \left\{ \frac{1}{2}\langle \nabla U(\mathbf{x}), \mathbf{v}\rangle - \lambda^{\text{ref}} + \frac{\epsilon}{\lambda^{\text{ref}} + \langle \nabla U(\mathbf{x}), -\mathbf{v}\rangle} \right\} \\
&+ \left\{ \lambda^{\text{ref}} \sqrt{\lambda^{\text{ref}} + \langle \nabla U(\mathbf{x}), -\mathbf{v}\rangle} \frac{d\sqrt{\lambda^{\text{ref}}} + d\sqrt{\lambda^{\text{ref}} + \left|\nabla U(\mathbf{x})\right|_1/d}}{2d\lambda^{\text{ref}} + \left|\nabla U(\mathbf{x})\right|_1} \right\} \text{ (Jensen's Inequality)} \\
&\leq \left\{ \frac{1}{2}\langle \nabla U(\mathbf{x}), \mathbf{v}\rangle - \lambda^{\text{ref}} + \frac{\epsilon}{\lambda^{\text{ref}} + \langle \nabla U(\mathbf{x}), -\mathbf{v}\rangle} \right\} \\
&+ \sqrt{\lambda^{\text{ref}} + \langle \nabla U(\mathbf{x}), -\mathbf{v}\rangle} \times \left\{ \frac{\sqrt{\lambda^{\text{ref}}}}{2} + \frac{\sqrt{\lambda^{\text{ref}}}}{\sqrt{2 + \frac{\left|\nabla U(\mathbf{x})\right|_1}{d\lambda^{\text{ref}}}}} \right\} \\
&\leq \left\{ \frac{1}{2}\langle \nabla U(\mathbf{x}), \mathbf{v}\rangle - \lambda^{\text{ref}} + \frac{\epsilon}{\lambda^{\text{ref}} + \langle \nabla U(\mathbf{x}), -\mathbf{v}\rangle} \right\} \\
&+ \sqrt{\lambda^{\text{ref}} + \langle \nabla U(\mathbf{x}), -\mathbf{v}\rangle} \times \left\{ \frac{\sqrt{\lambda^{\text{ref}}}}{2} + \frac{\sqrt{\lambda^{\text{ref}}}}{4} \right\} (\text{ since } \left|\nabla U(\mathbf{x})\right|_1 > \alpha_2 \text{ and } \frac{\alpha_2}{d\lambda^{\text{ref}}} > 14) \\
&= \frac{1}{2}\langle \nabla U(\mathbf{x}), \mathbf{v}\rangle - \lambda^{\text{ref}} + \frac{\epsilon}{\lambda^{\text{ref}} + \langle \nabla U(\mathbf{x}), -\mathbf{v}\rangle} + \frac{3}{4}\sqrt{\lambda^{\text{ref}}}\sqrt{\lambda^{\text{ref}} + \langle \nabla U(\mathbf{x}), -\mathbf{v}\rangle}
\end{aligned}
$$

Denote $w_0 = \langle \nabla U(\mathbf{x}), -\mathbf{v}\rangle$, and define $g : \mathbb{R}^+ \to \mathbb{R}$ as

$$
\begin{aligned}
g(w) &= -\frac{1}{2}w - \lambda^{\text{ref}} + \frac{\epsilon}{\lambda^{\text{ref}} + w} + \frac{3}{4}\sqrt{\lambda^{\text{ref}}}\sqrt{\lambda^{\text{ref}} + w} \\
g'(w) &= -\frac{1}{2} - \frac{\epsilon}{\lambda^{(\text{ref}} + w)^2} + \frac{3}{8}\frac{\sqrt{\lambda^{\text{ref}}}}{\sqrt{\lambda^{\text{ref}} + w}} < -\frac{1}{8} < 0
\end{aligned}
$$

As a result, g is a decreasing function on $[0, \infty)$, and

$$
g(w_0) \leq g(0) = -\lambda^{\text{ref}} + \frac{\epsilon}{\lambda^{\text{ref}}} + \frac{3}{4}\lambda^{\text{ref}} \leq -\frac{1}{8}\lambda^{\text{ref}}, \quad (\text{since } \lambda^{\text{ref}} > \sqrt{8\epsilon})
$$

As a result, on $K^c \times \mathcal{V}$, for all three cases, we have

$$
\mathcal{L}V(\mathbf{x},\mathbf{v}) \leq -\frac{1}{8}\lambda^{\text{ref}} V(\mathbf{x},\mathbf{v})
$$

Since $K_1 \subset K$ and K is compact, $\mathcal{V}$ is finite, therefore, $K \times \mathcal{V}$ is compact. As a result, there exists $b > 0$ such that

$$
\mathcal{L}V(\mathbf{x},\mathbf{v}) \leq b, \quad \text{for all } (\mathbf{x},\mathbf{v}) \in K \times \mathcal{V}
$$

Hence, under condition 2, there exist constants $b > 0$, $c > 0$ and function V such that

$$
\mathcal{L}V(\mathbf{x},\mathbf{v}) \leq -cV(\mathbf{x},\mathbf{v}) + b\mathbb{I}_{K\times\mathcal{V}}
$$

Since each compact set is small, V is hence a Lyapunov function. As a result, by **Lemma 3**, Theorem 4 in the paper is proved.

2.7 The event rate of each experiment

In this section, we produce the form of the event rate $\lambda(\mathbf{x}, \mathbf{v})$ of each model for both Zigzag and Coordinate samplers.

2.7.1 Banana-shaped Distribution

In this example, the potential function is

$$U(\mathbf{x}) = U(x_1, x_2) = (x_1 - 1)^2 + \kappa(x_2 - x_1^2)^2,$$

and its gradient is as follows,

$$\frac{\partial U(\mathbf{x})}{\partial x_1} = 2(x_1 - 1) + 4\kappa(x_1^2 - x_2)x_1, \quad \frac{\partial U(\mathbf{x})}{\partial x_2} = 2\kappa(x_2 - x_1^2)$$

For the Zigzag Sampler, recall that we set $\lambda_i^{\text{ref}} = 0$, $i = 1, 2$,

$$\begin{aligned}
\lambda_1(x + tv, v) &= \left\{ v_1 \frac{\partial U(x + tv)}{\partial x_1} \right\}_+ \\
&= \left\{ v_1 \left\{ 2(x_1 + tv_1 - 1) + 4\kappa(x_1 + tv_1)((x_1 + tv_1)^2 - (x_2 + tv_2)) \right\} \right\}_+ \\
&= \left\{ 2x_1v_1 + 2tv_1^2 - 2v_1 + 4\kappa v_1(x_1 + tv_1)^3 - 4\kappa v_1(x_1 + tv_1)(x_2 + tv_2) \right\}_+ \\
&= \Big\{ 2x_1v_1 + 2tv_1^2 - 2v_1 + 4\kappa v_1(x_1^3 + 3x_1^2v_1t + 3x_1v_1^2t^2 + v_1^3t^3) \\
&\quad - 4\kappa v_1(x_1x_2 + (x_1v_2 + x_2v_1)t + v_1v_2t^2) \Big\}_+ \\
&= \Big\{ (4\kappa v_1^4)t^3 + (12\kappa x_1v_1^3 - 4\kappa v_1^2v_2)t^2 + (2v_1^2 + 12\kappa x_1^2v_1^2 - 4\kappa x_1v_1v_2 - 4\kappa x_2v_1^2)t \\
&\quad + (2x_1v_1 - 2v_1 + 4\kappa v_1x_1^3 - 4\kappa v_1x_1x_2) \Big\}_+ \\
&= \left\{ a_{1,3}t^3 + a_{1,2}t^2 + a_{1,1}t + a_{1,0} \right\}_+
\end{aligned}$$

where $a_{1,3} = 4\kappa v_1^4$, $a_{1,2} = 12\kappa x_1v_1^3 - 4\kappa v_1^2v_2$, $a_{1,1} = 2v_1^2 + 12\kappa x_1^2v_1^2 - 4\kappa x_1v_1v_2 - 4\kappa x_2v_1^2$, and $a_{1,0} = 2x_1v_1 - 2v_1 + 4\kappa v_1x_1^3 - 4\kappa v_1x_1x_2$.

$$\begin{aligned}
\lambda_2(x + tv, v) &= \left\{ v_2 \frac{\partial U(x + tv)}{\partial x_2} \right\}_+ \\
&= \left\{ v_2 2\kappa \left(x_2 + tv_2 - (x_1 + tv_1)^2 \right) \right\}_+ \\
&= \left\{ (-2\kappa v_1^2v_2)t^2 + (2\kappa v_2(v_2 - 2x_1v_1))\, t + \left(2\kappa v_2(x_2 - x_1^2) \right) \right\}_+ \\
&= \left\{ a_{2,2}t^2 + a_{2,1}t + a_{2,0} \right\}_+
\end{aligned}$$

where $a_{2,2} = -2\kappa v_1^2v_2$, $a_{2,1} = 2\kappa v_2(v_2 - 2x_1v_1)$ and $a_{2,0} = 2\kappa v_2(x_2 - x_1^2)$. As a result, we have the following upper bounds for λ_1 and λ_2.

$$\lambda_1(\mathbf{x} + t\mathbf{v}, \mathbf{v}) \le \bar{\lambda}_1(t) := \left(\sum_{i=0}^{3} \max\{0, a_{1,i}\} \right) \max\{1, t^3\}$$

$$\lambda_2(\mathbf{x} + t\mathbf{v}, \mathbf{v}) \le \bar{\lambda}_2(t) := \left(\sum_{i=0}^{2} \max\{0, a_{2,i}\} \right) \max\{1, t^2\}$$

First, we use the Superposition Theorem: we set $T_1 = 0$ and generate a time duration, τ, from the Poisson process with rate $\bar{\lambda}_1(t)$, then compute $p = \lambda_1(\mathbf{x} + t\mathbf{v}, \mathbf{v})/\bar{\lambda}_1(t)$ and accept τ with probability p. If it is rejected, we update $\mathbf{x} \leftarrow \mathbf{x} + \tau\mathbf{v}, \mathbf{v} \leftarrow \mathbf{v}, T_1 \leftarrow T_1 + \tau$ and repeat the above process, until we obtain one τ and set $T_1 = T_1 + \tau$. Apply this procedure on λ_2 and get T_2. By the Thinning Theorem, $\min\{T_1, T_2\}$ follows the Poisson process with rate $\lambda_1 + \lambda_2$.

For the Coordinate Sampler, if $\mathbf{v} = (v_1, 0)$, where $v_1 \in \{-1, 1\}$, then

$$\begin{aligned}
\lambda(x + tv, v) &= \left\{ v_1 \frac{\partial U(x + tv)}{\partial x_1} \right\}_+ \\
&= \left\{ v_1 \left\{ 2(x_1 + tv_1 - 1) + 4\kappa(x_1 + tv_1)((x_1 + tv_1)^2 - x_2)) \right\} \right\}_+ \\
&= \left\{ 2x_1v_1 + 2tv_1^2 - 2v_1 + 4\kappa v_1(x_1 + tv_1)^3 - 4\kappa v_1(x_1 + tv_1)x_2 \right\}_+ \\
&= \Big\{ (4\kappa v_1^4)t^3 + (12\kappa x_1 v_1^3)t^2 + (2v_1^2 + 12\kappa x_1^2 v_1^2 - 4\kappa x_2 v_1^2)t \\
&\quad + (2x_1v_1 - 2v_1 + 4\kappa x_1^3 v_1 - 4\kappa x_1 x_2 v_1) \Big\}_+ \\
&= \left\{ b_{1,3}t^3 + b_{1,2}t^2 + b_{1,1}t + b_{1,0} \right\}_+
\end{aligned}$$

if $\mathbf{v} = (0, v_2)$, where $v_2 \in \{-1, 1\}$, then

$$\begin{aligned}
\lambda_1(x + tv, v) &= \left\{ v_1 \frac{\partial U(x + tv)}{\partial x_1} \right\}_+ \\
&= \left\{ v_2 \left\{ 2\kappa(x_2 + tv_2 - x_1^2) \right\} \right\}_+ \\
&= \left\{ (2\kappa v_2^2)t + \left(2\kappa v_2(x_2 - x_1^2) \right) \right\}_+ \\
&= \left\{ b_{2,1}t + b_{2,0} \right\}_+
\end{aligned}$$

At current event time, if $\mathbf{v} = (v_1, 0)$, we generate the event duration as above via the Superposition Theorem. If $\mathbf{v} = (0, v_2)$, we generate the event duration directly. That is, $U \sim \text{Uniform}[0, 1]$, if $b_{2,0} > 0$, then the time duration is

$$\left(\sqrt{-2\log(U)b_{2,1} + b_{2,0}^2} - b_{2,0} \right) \Big/ b_{2,1}$$

. Otherwise, the time duration is

$$\left(\sqrt{-2\log(U)b_{2,1}} - b_{2,0} \right) \Big/ b_{2,1}$$

.

2.7.2 Multivariate Gaussian Distribution

In this model,

$$U(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T A^{-1}\mathbf{x}, \text{ and } \nabla U(\mathbf{x}) = A^{-1}\mathbf{x}$$

For simplicity, we denote $B = A^{-1}$ and $B = (b_{ij})_{i,j=1,\cdots,d}$.

In Zigzag Sampler,

$$\lambda_i(\mathbf{x} + t\mathbf{v}, \mathbf{v}) = \left\{ v_i \sum_{j=1}^{d} b_{ij}(x_j + tv_j) \right\}_+ = \left\{ \left(v_i \sum_{j=1}^{d} b_{ij}v_j \right) t + \left(v_i \sum_{j=1}^{d} b_{ij}x_j \right) \right\}_+$$

In Coordinate Sampler, if only $v_i \neq 0$ for $\mathbf{v}$, then

$$\lambda(\mathbf{x}+t\mathbf{v},\mathbf{v}) = \left\{ v_i \sum_{j=1}^{d} b_{ij}x_j + t b_{ii} v_i^2 \right\}_+$$

2.7.3 Bayesian Logistic Model

In this example, we sample from the posterior of a Bayesian logistic model with flat prior and without intercept. Let $\mathbf{r} \in \mathbb{R}^d$ be independent variable, $s \in \{0,1\}$ be the response variable. The model is

$$\mathbb{P}(s=1|\mathbf{r}) = \frac{\exp(\mathbf{r}^T\mathbf{x})}{1+\exp(\mathbf{r}^T\mathbf{x})}$$

where $\mathbf{x}$ denotes the set of parameters. For a sample of observations $\{(\mathbf{r}_n, s_n)\}_{n=1}^N$, the likelihood function is

$$L(\mathbf{x}) = \prod_{n=1}^{N} \left(\frac{\exp(\mathbf{r}_n^T\mathbf{x})}{1+\exp(\mathbf{r}_n^T\mathbf{x})} \right)^{s_n} \left(\frac{1}{1+\exp(\mathbf{r}_n^T\mathbf{x})} \right)^{1-s_n} = \prod_{n=1}^{N} \frac{\exp(s_n r_n^T\mathbf{x})}{1+\exp(\mathbf{r}_n^T\mathbf{x})}$$

The potential function is

$$U(\mathbf{x}) = \sum_{n=1}^{N} \left\{ \log\left(1+\exp(\mathbf{r}_n^T\mathbf{x})\right) - s_n \mathbf{r}_n^T\mathbf{x} \right\}$$

$$\nabla U(\mathbf{x}) = \sum_{n=1}^{N} \left\{ \frac{\exp(\mathbf{r}_n^T\mathbf{x})}{1+\exp(\mathbf{r}_n^T\mathbf{x})} - s_n \right\} \mathbf{r}_n$$

$$\nabla_i U(\mathbf{x}) = \sum_{n=1}^{N} \left\{ \frac{\exp(\mathbf{r}_n^T\mathbf{x})}{1+\exp(\mathbf{r}_n^T\mathbf{x})} - s_n \right\} r_{n,i}$$

$$\left|\nabla_i U(\mathbf{x})\right| = \left| \sum_{n=1}^{N} \left\{ \frac{\exp(\mathbf{r}_n^T\mathbf{x})}{1+\exp(\mathbf{r}_n^T\mathbf{x})} - s_n \right\} r_{n,i} \right| \leq \sum_{n=1}^{d} \left| r_{n,i} \right|$$

For the Zigzag Sampler, the event rate is

$$\lambda_i(\mathbf{x}+t\mathbf{v},\mathbf{v}) \leq \sum_{n=1}^{N} \left| r_{n,i} \right| + \lambda_i^{\text{ref}}$$

For the Coordinate Sampler, if $v_i \neq 0$ of $\mathbf{v}$, then

$$\lambda(\mathbf{x}+t\mathbf{v},\mathbf{v}) \leq \sum_{n=1}^{N} \left| r_{n,i} \right| + \lambda^{\text{ref}}$$

2.7.4 Log-Cox Gaussian Model

The energy function is

$$U(\mathbf{x}) = \sum_{i,j} \left(-y_{i,j}x_{i,j} + s\exp\{x_{ij}\} \right) + \frac{1}{2}(\mathbf{x}-\mu\mathbf{1})^T\Sigma^{-1}(\mathbf{x}-\mu\mathbf{1})$$

$$\nabla_{ij}U(\mathbf{x}) = -y_{ij} + s\exp\{x_{ij}\} + (\Sigma^{-1})_{ij\cdot}(\mathbf{x} - \mu\mathbf{1})$$

Denote $x_k = x_{30*(i-1)+j}$, $B = \Sigma^{-1}$,

$$\nabla_k U(\mathbf{x}) = -y_k + s\exp\{x_k\} + (\Sigma^{-1})_{k\cdot}(\mathbf{x} - \mu\mathbf{1})$$

For Coordinate sampler:

$$\begin{aligned}
&\lambda_k(\mathbf{x}+t\mathbf{v},\mathbf{v}) = \{v_k\nabla_k U(\mathbf{x}+t\mathbf{v})\}_+ + \lambda_0 \\
&= \left\{-y_k v_k + sv_k\exp\{x_k+tv_k\} + v_k\sum_{\ell\neq k} b_{k\ell}(x_\ell-\mu) + b_{kk}(x_k+tv_k-\mu)v_k\right\}_+ + \lambda_0 \\
&= \left\{-y_k v_k + sv_k\exp\{x_k+tv_k\} + v_k\sum_{\ell} b_{k\ell}(x_\ell-\mu) + tb_{kk}v_k^2\right\}_+ + \lambda_0 \\
&= \left\{v_k\left(\sum_\ell b_{k\ell}(x_\ell-\mu) - y_k\right) + tb_{kk}v_k^2 + sv_k\exp\{x_k\}\exp\{tv_k\}\right\}_+ + \lambda_0 \\
&\leq \left\{v_k\left(\sum_\ell b_{k\ell}(x_\ell-\mu) - y_k\right) + tb_{kk}v_k^2\right\}_+ + se^{x_k}e^{tv_k}\{v_k\}_+ + \lambda_0
\end{aligned}$$

For Zig-Zag sampler:

$$\begin{aligned}
&\lambda_k(\mathbf{x}+t\mathbf{v},\mathbf{v}) = \{v_k\nabla_k U(\mathbf{x}+t\mathbf{v})\}_+ + \lambda_0 \\
&= \left\{-y_k v_k + sv_k\exp\{x_k+tv_k\} + v_k(\Sigma^{-1})_{k\cdot}(\mathbf{x}+t\mathbf{v}-\mu\mathbf{1})\right\}_+ + \lambda_0 \\
&= \left\{-y_k v_k + sv_k\exp\{x_k+tv_k\} + v_k\sum_{\ell=1}^{d} b_{k\ell}(x_\ell+tv_\ell-\mu)\right\}_+ + \lambda_0 \\
&= \left\{v_k\left(\sum_{\ell=1}^{d} b_{k\ell}(x_\ell-\mu) - y_k\right) + \left(v_k\sum_{\ell=1}^{d} b_{k\ell}v_\ell\right)t + sv_k\exp\{x_k+tv_k\}\right\}_+ + \lambda_0 \\
&\leq \left\{v_k\left(\sum_\ell b_{k\ell}(x_\ell-\mu) - y_k\right) + \left(v_k\sum_{\ell} b_{k\ell}v_\ell\right)t\right\}_+ + se^{x_k}e^{tv_k}\{v_k\}_+ + \lambda_0
\end{aligned}$$

Bibliography

Bierkens, J., Bouchard-Côté, A., Doucet, A., Duncan, A. B., Fearnhead, P., Lienart, T., Roberts, G., and Vollmer, S. J. (2018). Piecewise deterministic Markov processes for scalable Monte Carlo on restricted domains. *Statistics & Probability Letters*, 136:148–154.

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*.

Bierkens, J., Roberts, G., et al. (2017). A piecewise deterministic scaling limit of lifted Metropolis–Hastings in the Curie–Weiss model. *The Annals of Applied Probability*, 27(2):846–882.

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.

Davis, M. H. (1984). Piecewise-deterministic Markov processes: A general class of non-diffusion stochastic models. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 353–388.

Davis, M. H. (1993). *Markov Models & Optimization*, volume 49. CRC Press.

Deligiannidis, G., Bouchard-Côté, A., and Doucet, A. (2017). Exponential Ergodicity of the Bouncy Particle Sampler. *arXiv preprint arXiv:1705.04579*.

Down, D., Meyn, S. P., and Tweedie, R. L. (1995). Exponential and uniform ergodicity of Markov processes. *The Annals of Probability*, 23(4):1671–1691.

Fearnhead, P., Bierkens, J., Pollock, M., and Roberts, G. O. (2016). Piecewise Deterministic Markov Processes for Continuous-Time Monte Carlo. *arXiv preprint arXiv:1611.07873*.

Fontbona, J., Guérin, H., and Malrieu, F. (2016). Long time behavior of telegraph processes under convex potentials. *Stochastic Processes and their Applications*, 126(10):3077–3101.

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.

Glynn, P. W. and Haas, P. J. (2006). Laws of large numbers and functional central limit theorems for generalized semi-Markov processes. *Stochastic Models*, 22(2):201–231.

Harland, J., Michel, M., Kampmann, T. A., and Kierfeld, J. (2017). Event-chain Monte Carlo algorithms for three-and many-particle interactions. *EPL (Europhysics Letters)*, 117(3):30001.

Hastings, W. K. (1970). Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109.

Kingman, J. F. C. (1992). *Poisson processes*, volume 3. Clarendon Press.

Lewis, P. A. and Shedler, G. S. (1979). Simulation of nonhomogeneous Poisson processes by thinning. *Naval Research Logistics (NRL)*, 26(3):403–413.

Liu, J. S. (2008). *Monte Carlo strategies in scientific computing*. Springer Science & Business Media.

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.

Michel, M., Kapfer, S. C., and Krauth, W. (2014). Generalized event-chain Monte Carlo: Constructing rejection-free global-balance algorithms from infinitesimal steps. *The Journal of chemical physics*, 140(5):054116.

Michel, M. and Sénécal, S. (2017). Forward Event-Chain Monte Carlo: a general rejection-free and irreversible Markov chain simulation method. *arXiv preprint arXiv:1702.08397*.

Neal, R. M. et al. (2011). MCMC using Hamiltonian dynamics. *Handbook of Markov Chain Monte Carlo*, 2(11).

Pakman, A., Gilboa, D., Carlson, D., and Paninski, L. (2016). Stochastic bouncy particle sampler. *arXiv preprint arXiv:1609.00770*.

Peters, E. A. et al. (2012). Rejection-free Monte Carlo sampling for general potentials. *Physical Review E*, 85(2):026703.

Robert, C. P. and Casella, G. (2004). Monte Carlo statistical methods. *Springer New York*.

Sherlock, C. and Thiery, A. H. (2017). A Discrete Bouncy Particle Sampler. *arXiv preprint arXiv:1707.05200*.

Vanetti, P., Bouchard-Côté, A., Deligiannidis, G., and Doucet, A. (2017). Piecewise Deterministic Markov Chain Monte Carlo. *arXiv preprint arXiv:1707.05296*.

Discrete PDMP using Hamiltonian dynamics

Joint work with Christian P. ROBERT and Julien STOEHR

3.1 Introduction

As a powerful and efficient sampling algorithm, Hamiltonian Monte Carlo (HMC) (Duane et al., 1987; Neal, 2010) have demonstrated its outperformance in high dimensional problems over many other MCMC algorithms (Neal, 2010). Resorting to Hamiltonian dynamics, HMC greatly reduces random walk behaviour by accepting remote proposals with high probability. However, in implementation of HMC, one needs to refresh the momentum at each iteration, which introduces randomness, and to discard all intermediate proposals along each leapfrog path, which wastes computation effort. Motivated by these two challenges, in this chapter, we propose a novel variant of HMC, , which coincides with the discretisation of recently prevailing piecewise deterministic Markov process (PDMP) samplers, to control the frequency of refreshment of the momentum and to recycle previously visited intermediate proposals, while keeping the correctness at the same time.

To suppress the random walk behaviour resulting from refreshing the momentums between trajectories, Horowitz (1991) proposed to only partially refresh them at each iteration. For not being able to recycle the intermediate proposals along the trajectories, only a small improvement is obtained by this partial refreshment (Neal, 2010). Nishimura and Dunson (2015) recycled the intermediate proposals by augmenting the target L times, where L is the number of leapfrog steps at each iteration, while Bernton et al. (2015) locally weighted them to improve HMC. Compared with these two methods, our method not only reuses the intermediate proposals, but controls the refreshment frequency.

Since it is introduced in physics by Peters et al. (2012) and popularised in statistics by (Bierkens et al., 2016; Bouchard-Côté et al., 2018), PDMP-based sampler, as continuous-time, non-reversible, rejection-free MCMC method, has exhibited outstanding performance in high dimensional cases (Bouchard-Côté et al., 2018; Galbraith, 2016), even compared with HMC algorithm. By chance, our variant of HMC can be regarded as a discretisation of a specific PDMP-based sampler, according to (Sherlock and Thiery, 2017; Vanetti et al., 2017). As a result, we name this variant by discrete piecewise deterministic Markov process with Hamiltonian dynamics (dPDMP-HMC). So far, in order to generate the piecewise deterministic trajectories, almost all existing PDMP-based samplers take advantage of linear dynamics,

except for Hamiltonian BPS of Vanetti et al. (2017), which transfers the obstacle of non-linear dynamics into the event rate function and is based on good Gaussian approximation. Discretizing PDMP-based samplers can bypass such difficulty, without loss of efficiency.

In this chapter, we briefly introduce Hamiltonian Monte Carlo and piecewise deterministic Markov processes in Section 2. Section 3 describes dPDMP-HMC sampler and we compare it with HMC over several numerical experiments in Section 4.

3.2 Hamiltonian Monte Carlo and Piecewise Deterministic Markov Process

In this section, we assume $\pi(\mathbf{x})$ is the probability density function of the target distribution, where $\mathbf{x} \in \mathbb{R}^d$ and denote $U(\mathbf{x}) = -\log \pi(\mathbf{x})$ the potential of the target distribution.

3.2.1 Hamiltonian Monte Carlo

For HMC, an artificial auxiliary variable $\mathbf{v}$, called *momentum*, is introduced along with a density $\varphi(\mathbf{v}|\mathbf{x})$ such that the joint distribution of $(\mathbf{x}, \mathbf{v})$ takes $\pi(\mathbf{x})$ as its marginal. For simplicity, we suppose $\varphi(\mathbf{v}|\mathbf{x})$ is the d-dimensional standard Gaussian distribution $\mathcal{N}(0, I_d)$. Based on the so-called Hamiltonian dynamic,

$$\frac{d\mathbf{x}_t}{dt} = \mathbf{v}_t, \quad \frac{d\mathbf{v}_t}{dt} = \nabla \log \pi(\mathbf{x}_t)$$

HMC can propose proposals far away from current position and accept it with high probability, by approximating above dynamics with leapfrog integrator. In practice, HMC requires two parameters for leapfrog integrator: the step size ϵ and the leapfrog steps L. The guidance of choosing these parameters is to ensure that the acceptance rate is around 65%. In our experiments, we run leapfrog integrator r steps, where $r \sim \text{Uniform}\{1, \cdots, L\}$, with fixed step size ϵ.

3.2.2 Piecewise Deterministic Markov Process

As in HMC, we introduce an auxiliary variable, $\mathbf{v} \in \mathbb{R}^d$, called velocity. Denote $\mathbf{z} = (\mathbf{x}, \mathbf{v}) \in \mathbb{R}^d \times \mathbb{R}^d$. The only constraint on $\mathbf{z}$ is that $\pi(\mathbf{x})$ is the marginal density of $\pi(\mathbf{x}, \mathbf{v})$ with respect to $\mathbf{x}$. Let $\{\mathbf{z}_t\}$ denote a piecewise deterministic Markov chain of $\mathbf{z}$ on the augmented space $(\mathbf{x}, \mathbf{v}) \in \mathbb{R}^d \times \mathbb{R}^d$, to be specified below. In fact, the dynamics of PDMP consist of three types of dynamics, namely, deterministic dynamics, event occurrence and transition dynamics. Specifically,

1. **The deterministic dynamics**: between two event times, the Markov process evolves deterministically, according to some ordinary differential equation:

$$\frac{dz_t^{(i)}}{dt} = \Psi^{(i)}(\mathbf{z}_t), \quad i = 1, \cdots, 2d$$

2. **The event occurrence**: the event occurs at rate $\lambda(\mathbf{z}_t)$.

3. **The transition dynamics**: At the event time, τ, we denote $\mathbf{z}_{\tau-}$ the state prior to τ, then $\mathbf{z}_\tau \sim Q(\cdot|\mathbf{z}_{\tau-})$

Following from (Davis, 1993, Theorem 26.14) , this Markov process's extension generator is

$$\mathcal{A}f(\mathbf{z}) = \nabla f(\mathbf{z}) \cdot \Psi(\mathbf{z}) + \lambda(\mathbf{z}) \int_{\mathbb{R}^{2d}} (f(\mathbf{z}') - f(\mathbf{z}))\, Q(d\mathbf{z}'; \mathbf{z})$$

In order to establish the invariance with respect to $\pi(\mathbf{z})$, we just need $\int \mathcal{A}f(\mathbf{z})\pi(d\mathbf{z}) = 0$ for all f in an appropriate class of functions on $\mathbb{R}^{2d}$ (Davis, 1993). For the existing PDMP algorithms, in order to compute the trajectories of the deterministic dynamics exactly, BPS Bouchard-Côté et al. (2018) and Zigzag Sampler Bierkens et al. (2016) resort to the linear dynamics,

$$\frac{d\mathbf{x}_t}{dt} = \mathbf{v}_t, \quad \frac{d\mathbf{v}_t}{dt} = \mathbf{0},$$

while Hamiltonian BPS Vanetti et al. (2017) takes advantage of the Hamiltonian dynamic of a Gaussian approximation to the augmented target distribution $\pi(\mathbf{x}, \mathbf{v})$. However, linear dynamics ignores the geometry of the target, while it might be difficult to find good Gaussian approximations in Hamiltonian BPS.

3.3 PDMP with Hamiltonian Dynamics

In this section, we assume that the augmented target distribution has a separate form, $\pi(\mathbf{x}, \mathbf{v}) = \pi(\mathbf{x})\varphi(\mathbf{v})$, where φ is a distribution over $\mathbb{R}^d$. For simplicity, we assume φ is the standard Gaussian distribution.

3.3.1 PDMP-HMC

PDMP-HMC processes use Hamiltonian dynamics, generate event time with constant event rate and refresh the velocity according to its marginal distribution at event time. Specifically,

1. **The deterministic dynamics**:
$$\frac{d\mathbf{x}_t}{dt} = -\nabla \log \varphi(\mathbf{v}_t) = \mathbf{v}_t$$
$$\frac{d\mathbf{v}_t}{dt} = \nabla \log \pi(\mathbf{x}_t)$$
2. **The event occurrence**: $\lambda(\mathbf{z}_t) = \lambda(\mathbf{x})$.
3. **The transition dynamics**: $Q((\mathbf{x}, \mathbf{v}), (d\mathbf{x}', d\mathbf{v}')) = \delta_{\mathbf{x}}(d\mathbf{x}')\varphi(d\mathbf{v}')$

Theorem 3.3.1 *The above piecewise deterministic Markov chain admits $\rho(d\mathbf{x}, \mathbf{v}) = \pi(d\mathbf{x}) \otimes \varphi(d\mathbf{v})$ over $\mathbb{R}^{2d}$ as its invariant distribution, where $\varphi(\mathbf{v})$ is the d-dimensional standard normal distribution.*

Proof: The generator of above PDMP is, by (Davis (1993), Theorem 26.14),

$$\begin{aligned}
\mathcal{L}f(\mathbf{x}, \mathbf{v}) &= \left\langle \frac{\partial f(\mathbf{x}, \mathbf{v})}{\partial \mathbf{x}}, \mathbf{v} \right\rangle + \left\langle \frac{\partial f(\mathbf{x}, \mathbf{v})}{\partial \mathbf{v}}, \nabla \log \pi(\mathbf{x}) \right\rangle \\
&\quad + \lambda(\mathbf{x}, \mathbf{v}) \int_{\mathbf{x}'} \int_{v'} (f(\mathbf{x}', \mathbf{v}') - f(\mathbf{x}, \mathbf{v}))\, Q((\mathbf{x}, \mathbf{v}), (d\mathbf{x}', d\mathbf{v}')) \\
&= \left\langle \frac{\partial f(\mathbf{x}, \mathbf{v})}{\partial \mathbf{x}}, \frac{\partial H(\mathbf{x}, \mathbf{v})}{\partial \mathbf{v}} \right\rangle + \left\langle \frac{\partial f(\mathbf{x}, \mathbf{v})}{\partial \mathbf{v}}, -\frac{\partial H(\mathbf{x}, \mathbf{v})}{\partial \mathbf{x}} \right\rangle \\
&\quad + \lambda(\mathbf{x}) \left(\int_{\mathbf{v}'} f(\mathbf{x}, \mathbf{v}')\varphi(\mathbf{v}')d\mathbf{v}' - f(\mathbf{x}, \mathbf{v}) \right)
\end{aligned}$$

For all $f \in \mathcal{D}(\mathcal{L})$, we have,

$$
\begin{aligned}
&\int_{\mathbf{x}}\int_{\mathbf{v}} \mathcal{L}f(\mathbf{x},\mathbf{v})\pi(\mathbf{x})\varphi(\mathbf{v})d\mathbf{x}d\mathbf{v} \\
&= \int_{\mathbf{x}}\int_{\mathbf{v}} \left[\left\langle \frac{\partial f(\mathbf{x},\mathbf{v})}{\partial \mathbf{x}}, \frac{\partial H(\mathbf{x},\mathbf{v})}{\partial \mathbf{v}} \right\rangle + \left\langle \frac{\partial f(\mathbf{x},\mathbf{v})}{\partial \mathbf{v}}, -\frac{\partial H(\mathbf{x},\mathbf{v})}{\partial \mathbf{x}} \right\rangle\right] \pi(\mathbf{x})\varphi(\mathbf{v})d\mathbf{x}d\mathbf{v} \\
&+ \int_{\mathbf{x}}\int_{\mathbf{v}}\int_{\mathbf{v}'} \lambda(\mathbf{x}) f(\mathbf{x},\mathbf{v}')\varphi(\mathbf{v}')\pi(\mathbf{x})\varphi(\mathbf{v})d\mathbf{v}'d\mathbf{x}d\mathbf{v} - \int_{\mathbf{x}}\int_{\mathbf{v}} \lambda(\mathbf{x}) f(\mathbf{x},\mathbf{v})\pi(\mathbf{x})\varphi(\mathbf{v})d\mathbf{x}d\mathbf{v} \\
&= \int_{\mathbf{x}}\int_{\mathbf{v}} \left[-\left\langle \frac{\partial \log \pi(\mathbf{x})}{\partial \mathbf{x}}, \frac{\partial H(\mathbf{x},\mathbf{v})}{\partial \mathbf{v}} \right\rangle - \left\langle \frac{\partial \log \varphi(\mathbf{v})}{\partial \mathbf{v}}, -\frac{\partial H(\mathbf{x},\mathbf{v})}{\partial \mathbf{x}} \right\rangle\right] f(\mathbf{x},\mathbf{v})\pi(\mathbf{x})\varphi(\mathbf{v})d\mathbf{x}d\mathbf{v} \\
&+ \int_{\mathbf{x}}\int_{\mathbf{v}'} \lambda(\mathbf{x}) f(\mathbf{x},\mathbf{v}')\varphi(\mathbf{v}')\pi(\mathbf{x})d\mathbf{v}'d\mathbf{x} - \int_{\mathbf{x}}\int_{\mathbf{v}} \lambda(\mathbf{x}) f(\mathbf{x},\mathbf{v}) f(\mathbf{x},\mathbf{v}) f(\mathbf{x},\mathbf{v})\pi(\mathbf{x})\varphi(\mathbf{v})d\mathbf{x}d\mathbf{v} \\
&= \int_{\mathbf{x}}\int_{\mathbf{v}} \left[\left\langle \frac{\partial H(\mathbf{x},\mathbf{v})}{\partial \mathbf{x}}, \frac{\partial H(\mathbf{x},\mathbf{v})}{\partial \mathbf{v}} \right\rangle + \left\langle \frac{\partial H(\mathbf{x},\mathbf{v})}{\partial \mathbf{v}}, -\frac{\partial H(\mathbf{x},\mathbf{v})}{\partial \mathbf{x}} \right\rangle\right] f(\mathbf{x},\mathbf{v})\pi(\mathbf{x})\varphi(\mathbf{v})d\mathbf{x}d\mathbf{v} \\
&= 0
\end{aligned}
$$

By (Davis (1993), Theorem 34.7), ρ is the invariant distribution of above PDMP.

In practice, expect for some specific examples, we can not implement PDMP-HMC to sample from the target distribution since it is difficult to compute the trajectories according to the Hamiltonian dynamics exactly. Thanks to the discretisation of PDMP in (Sherlock and Thiery, 2017; Vanetti et al., 2017), we can overcome such difficulty with the same computational complexity of HMC.

3.3.2 Discrete PDMP-HMC

As the classical HMC algorithm, we discretize the Hamiltonian dynamics with leapfrog integrator in PDMP-HMC and calibrate the bias with Metropolis-Hastings accept step. In discrete PDMP-HMC (dPDMP-HMC), we introduce three Markov transition kernels.

1. Leapfrog kernel $P_1^{\epsilon,L}$: denote $(\mathbf{x}^+,\mathbf{v}^+) = \text{Leapfrog}(\mathbf{x},\mathbf{v},\epsilon,L_r)$ where $L_r \sim \text{Uniform}\{1,2,\cdots,L\}$.
   (a) $(\mathbf{x},\mathbf{v}) \to (\mathbf{x}^+,-\mathbf{v}^+)$ w.p.
   $$1 \wedge \frac{\pi(\mathbf{x}^+)\varphi(-\mathbf{v}^+)}{\pi(\mathbf{x})\varphi(\mathbf{v})}$$
   (b) $(\mathbf{x},\mathbf{v}) \to (\mathbf{x},\mathbf{v})$ w.p.
   $$1 - 1 \wedge \frac{\pi(\mathbf{x}^+)\varphi(-\mathbf{v}^+)}{\pi(\mathbf{x})\varphi(\mathbf{v})}$$
2. Flip kernel P_2: flipping the velocity, $(\mathbf{x},\mathbf{v}) \to (\mathbf{x},-\mathbf{v})$.
3. Refreshment kernel P_3: refreshing the velocity, $(\mathbf{x},\mathbf{v}) \to (\mathbf{x},\mathbf{v}^*)$, where $\mathbf{v}^* \sim \varphi$.

The transition kernel of dPDMP-HMC is

$$P^{\epsilon,L,\eta} = (1-\eta)P_2 \circ P_1^{\epsilon,L} + \eta P_3$$

where $\epsilon > 0$ is the step size of leapfrog integrator and $\eta \in (0,1)$ is the probability of refreshing the velocity. η is a parameter to control the refreshment frequency of

momentum, $\mathbf{v}$, while ϵ and L have same functions as them in HMC. In dPDMD-HMC, we use random leapfrog steps at each iteration, in order to remove the exact repetition of visited proposals. Since both P_1 and P_2 are reversible with respect to $\pi(\mathbf{x}, \mathbf{v})$, their composition is also invariant with respect to $\pi(\mathbf{x}, \mathbf{v})$. As P_3 refreshes the momentum according to its conditional distribution, P is invariant with respect to $\pi(\mathbf{x}, \mathbf{v})$ too.

A pseudo-code of dPDMP-HMC is described in details in Algorithm 3.2. In the description of Algorithm 3.2, $(\mathbf{x}_k^{\mathcal{B}}, \mathbf{v}_k^{\mathcal{B}})$ denotes the k-th element of $\mathcal{B}$.

Algorithm 3.1 Leapfrog Integrator

Input: start position $\mathbf{x}_0$, start velocity $\mathbf{v}_0$, stepsize ϵ, and steps L.
for $\ell = 1, 2, \cdots, L$ **do**

$$\begin{aligned}
\mathbf{v}_{\ell-\frac{1}{2}} &\leftarrow \mathbf{v}_{\ell-1} + \frac{1}{2}\epsilon \nabla \log(\pi(\mathbf{x}_{\ell-1})) \\
\mathbf{x}_{\ell} &\leftarrow \mathbf{x}_{\ell-1} + \epsilon \mathbf{v}_{\ell-\frac{1}{2}} \\
\mathbf{v}_{\ell} &\leftarrow \mathbf{v}_{\ell-\frac{1}{2}} + \frac{1}{2}\epsilon \nabla \log(\pi(\mathbf{x}_{\ell}))
\end{aligned}$$

end for
Output: Return $(\mathbf{x}_L, \mathbf{v}_L)$

3.4 Empirical Evaluation

In this section, we compare the efficiency of dPDMP-HMC and classical HMC over two examples. In all experiments, we tune the step size ϵ and steps L for HMC over some appropriate grid and choose the settings such that they maximize the minimum effective sample size across components per gradient of $\log \pi$ or likelihood evaluation.

3.4.1 Correlated Multivariate Normal Distribution

In this experiment, the target distribution is a 50-dimensional multivariate normal distribution with zero-mean and covariance matrix A, where $A_{i,j} = 0.9^{|i-j|}$, that is,

$$\pi(\mathbf{x}) \propto \exp\left\{-\frac{1}{2}\mathbf{x}^T A^{-1}\mathbf{x}\right\}$$

We set $\epsilon = 0.167$ and $L = 60$ for HMC algorithm. That is, at each iteration in HMC, the leapfrog integrator runs r steps, where $r \sim \text{Uniform}\{1, \cdots, 60\}$, with step size $\epsilon = 0.167$. For dPDMP-HMC, we choose $(\epsilon, L, \eta) = (0.167, 60, 0.04)$. Under same computation budget, HMC generates 5,000 samples, while dPDMP-HMC gives 9,000 samples. We repeat each algorithm 40 times and summary their comparison in Figure 3.1.

3.4.2 Independent Multivariate Normal Distribution

In this experiment, the target distribution is a 100-dimensional multivariate normal distribution with zero-mean and diagonal covariance matrix A, where $A_{i,i} = \frac{i^2}{10000}$.

Algorithm 3.2 Discrete PDMP-HMC Sampler

Input: starting position $\mathbf{x}_0$, stepsize ϵ, refreshment probability η, iteration number N and upper bound of steps L.
Generate starting velocity $\mathbf{v}_0 \sim \mathcal{N}(\mathbf{0}, I_d)$. $\mathcal{B} \leftarrow \{(\mathbf{x}_0, \mathbf{v}_0)\}$, $v \leftarrow 1$ and $\gamma \leftarrow 1$.
for $i = 1, 2, \cdots, N$ **do**
  $u \sim \mathcal{U}[0,1]$
  if $u < \eta$ **then**
    $\mathbf{x}_i \leftarrow \mathbf{x}_{i-1}$, $\mathbf{v}_i \sim \mathcal{N}(\mathbf{0}, I_d)$, $\mathcal{B} \leftarrow \{(\mathbf{x}_i, \mathbf{v}_i)\}$, $v \leftarrow 1$ and $\gamma \leftarrow 1$.
  else
    $L_r \sim \text{Uniform}\{1, 2, \cdots, L\}$
    if $\gamma + L_r > \#\mathcal{B}$ and $v = 1$ **then**
      $L_\delta \leftarrow \gamma + L_r - \#\mathcal{B}$
      $\{(\mathbf{x}_k^*, \mathbf{v}_k^*)\}_{k=1,\cdots,L_\delta} \leftarrow$ all proposals of Leapfrog$(\mathbf{x}_{\#\mathcal{B}}^{\mathcal{B}}, \mathbf{v}_{\#\mathcal{B}}^{\mathcal{B}}, \epsilon, L_\delta)$, $\mathcal{B} \leftarrow \{\mathcal{B}, (\mathbf{x}_1^*, \mathbf{v}_1^*), \cdots, (\mathbf{x}_{L_\delta}^*, \mathbf{v}_{L_\delta}^*)\}$
      $u \sim \mathcal{U}[0,1]$, compute $p \leftarrow \frac{\pi(\mathbf{x}_{L_\delta}^*, \mathbf{v}_{L_\delta}^*)}{\pi(\mathbf{x}_{i-1}, \mathbf{v}_{i-1})}$.
      if $u < 1 \wedge p$ **then**
        $(\mathbf{x}_i, \mathbf{v}_i) \leftarrow (\mathbf{x}_{L_\delta}^*, \mathbf{v}_{L_\delta}^*)$, $\gamma \leftarrow \gamma + L_r$ and $v \leftarrow 1$.
      else
        $(\mathbf{x}_i, \mathbf{v}_i) \leftarrow (\mathbf{x}_{i-1}, -\mathbf{v}_{i-1})$, $\gamma \leftarrow \gamma$ and $v \leftarrow -1$
      end if
    else
      if $\gamma - L_r < 1$ and $v = -1$ **then**
        $L_\delta \leftarrow L_r + 1 - \gamma$
        $\{(\mathbf{x}_k^*, \mathbf{v}_k^*)\}_{k=1,\cdots,L_\delta} \leftarrow$ all proposals of Leapfrog$(\mathbf{x}_1^{\mathcal{B}}, \mathbf{v}_1^{\mathcal{B}}, -\epsilon, L_\delta)$
        $\mathcal{B} \leftarrow \{(\mathbf{x}_{L_\delta}^*, \mathbf{v}_{L_\delta}^*), \cdots, (\mathbf{x}_1^*, \mathbf{v}_1^*), \mathcal{B}\}$
        $u \sim \mathcal{U}[0,1]$, compute $p \leftarrow \frac{\pi(\mathbf{x}_{L_\delta}^*, \mathbf{v}_{L_\delta}^*)}{\pi(\mathbf{x}_{i-1}, \mathbf{v}_{i-1})}$.
        if $u < 1 \wedge p$ **then**
          $(\mathbf{x}_i, \mathbf{v}_i) \leftarrow (\mathbf{x}_{L_\delta}^*, -\mathbf{v}_{L_\delta}^*)$, $\gamma \leftarrow 1$ and $v \leftarrow -1$.
        else
          $(\mathbf{x}_i, \mathbf{v}_i) \leftarrow (\mathbf{x}_{i-1}, -\mathbf{v}_{i-1})$, $\gamma \leftarrow \gamma + L_\delta$ and $v \leftarrow 1$.
        end if
      else
        if $v = 1$ **then**
          $u \sim \mathcal{U}[0,1]$, compute $p \leftarrow \frac{\pi(\mathbf{x}_{\gamma+L_r}^{\mathcal{B}}, \mathbf{v}_{\gamma+L_r}^{\mathcal{B}})}{\pi(\mathbf{x}_{i-1}, \mathbf{v}_{i-1})}$
          if $u < 1 \wedge p$ **then**
            $(\mathbf{x}_i, \mathbf{v}_i) \leftarrow (\mathbf{x}_{\gamma+L_r}^{\mathcal{B}}, \mathbf{v}_{\gamma+L_r}^{\mathcal{B}})$, $\gamma \leftarrow \gamma + L_r$, and $v \leftarrow 1$.
          else
            $(\mathbf{x}_i, \mathbf{v}_i) \leftarrow (\mathbf{x}_i, -\mathbf{v}_i)$, $\gamma \leftarrow \gamma$, and $v \leftarrow -1$
          end if
        else
          $u \sim \mathcal{U}[0,1]$, compute $p \leftarrow \frac{\pi(\mathbf{x}_{\gamma-L_r}^{\mathcal{B}}, \mathbf{v}_{\gamma-L_r}^{\mathcal{B}})}{\pi(\mathbf{x}_{i-1}, \mathbf{v}_{i-1})}$
          if $u < 1 \wedge p$ **then**
            $(\mathbf{x}_i, \mathbf{v}_i) \leftarrow (\mathbf{x}_{\gamma-L_r}^{\mathcal{B}}, -\mathbf{v}_{\gamma-L_r}^{\mathcal{B}})$, $\gamma \leftarrow \gamma - L_r$, and $v \leftarrow -1$.
          else
            $(\mathbf{x}_i, \mathbf{v}_i) \leftarrow (\mathbf{x}_{i-1}, -\mathbf{v}_{i-1})$, $\gamma \leftarrow \gamma$, and $v \leftarrow 1$
          end if
        end if
      end if
    end if
  end if
end for
Output: $\{(\mathbf{x}_i, \mathbf{v}_i)\}_{i=1}^N$.

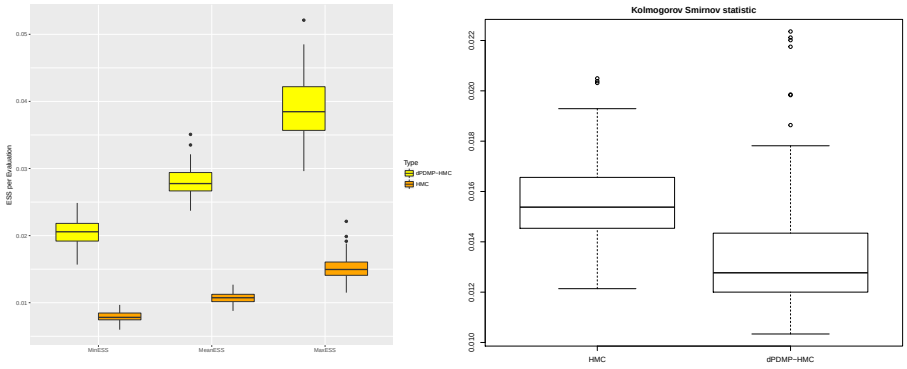

Figure 3.1: The lhs represents the comparison in terms of effective sample size, while rhs compares the two samplers in terms of Kolmogorov-Smirnov statistic.

For HMC, we set $\epsilon = 0.016$ and $L = 160$. For dPDMP-HMC, we choose $(\epsilon, L, \eta) = (0.016, 100, 0.1)$. Under same computation budget, HMC generates 5,000 samples, while dPDMP-HMC gives 11,500 samples. We repeat each algorithm 40 times and summary their comparison in Figure 3.2.

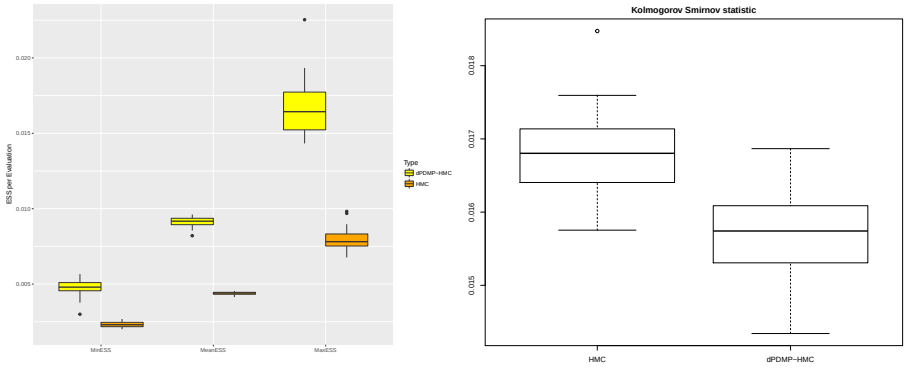

Figure 3.2: The lhs represents the comparison in terms of effective sample size, while rhs compares the two samplers in terms of Kolmogorov-Smirnov statistic.

3.4.3 Log-Gaussian Cox Point Process

In this example, the observations $\mathbf{Y} = \{y_{ij}\}$ are Poisson distributed and conditionally independent given a latent intensity process $\mathbf{\Lambda} = \{\lambda_{ij}\}$ with means $s\lambda_{ij} = s\exp(x_{ij})$, where $s = \frac{1}{d^2}$. The rates $\mathbf{X} = \{x_{ij}\}$ are obtained from a Gaussian process with mean function $m(x_{ij}) = \mu\mathbf{1}$ and covariance function $\Sigma(x_{ij}, x_{i'j'}) = \sigma^2 \exp(-\delta(i, i', j, j')/\beta d)$, where $\delta(i, i', j, j') = \sqrt{(i - i')^2 + (j - j')^2}$. In our experiment, we consider $d = 30$ and set $\sigma^2 = 1.91$, $\mu = \log(126) - \sigma^2/2$, and $\beta = \frac{1}{6}$. Then the target is, based on the synthetic data set $\mathbf{Y}$,

$$\pi(\mathbf{X}|\mathbf{Y}, \mu, \sigma, \beta) \propto \exp\left\{\sum_{i,j}(y_{ij}x_{ij} - s\exp(x_{ij}) - \frac{1}{2}(\mathbf{X} - \mu\mathbf{1})^T\Sigma^{-1}(\mathbf{X} - \mu\mathbf{1})\right\}$$

Following Galbraith (2016), we run HMC $L_r \sim \text{Uniform}\{1, \cdots, 100\}$ steps with $\epsilon = 0.15$ at each iteration. For dPDMP-HMC, we set $L = 100, \epsilon = 0.1$ and $\eta = 0.01$. We run each sampler for 100 repetitions under same computation budget, which refers to the number of gradient and likelihood evaluations, for a fixed synthetic data set. For HMC, we obtain 5,000 samples, while dPDMP-HMC sampler obtain

10,000 samples. We thin samples of dPDMP-HMC by 2 to get samples with same size as those of HMC. We run an extremely long Markov chain using the HMC sampler and obtain 500,000 samples, which are regarded as the ground truth. We compare the two samplers in terms of the mean of Wasserstein 1 and 2 distance over their variances and the mean of Kolmogorov-Smirnov statistic across the components.

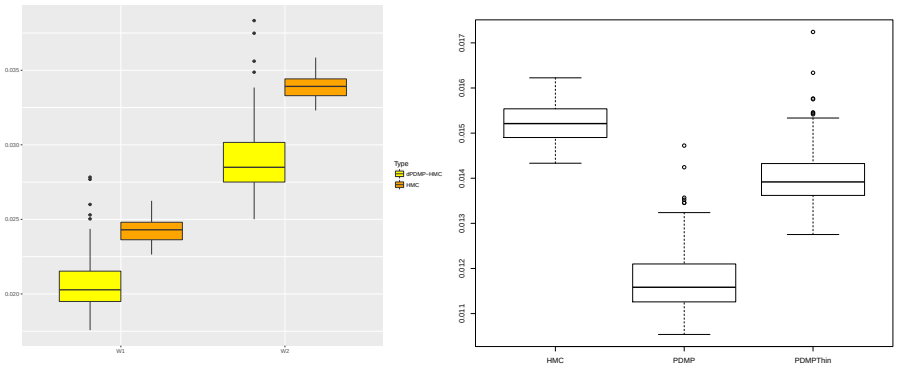

Figure 3.3: The lhs represents the comparison in terms of Wasserstein 1 and 2 distance, while rhs compares the two samplers in terms of Kolmogorov-Smirnov statistic.

3.5 Conclusion

In this article, we propose a randomness-controlled HMC algorithm, which coincides with the discrete piecewise deterministic Markov process sampler with Hamiltonian dynamics and compared its efficiency with basic HMC. As we expected, several examples show that dPDMP-HMC is more efficient than HMC in terms of minimal ESS per gradient or target evaluation, since we reuse the previously visited proposals in leapfrog integrator while the algorithm keeps its correctness at the same time. The future direction of this work is to investigate the theoretical properties of PDMP-HMC and dPDMP-HMC, such as if they are geometrically ergodic. How to tune the three parameters (ϵ, L, η) automatically also deserves further study. In addition, since HMM and dPDMP-HMC can both be viewed discretizations of PDMP with Hamiltonian dynamics, other ways of discretization also deserve to investigate.

Bibliography

Bernton, E., Yang, S., Chen, Y., Shephard, N., and Liu, J. S. (2015). Locally weighted Markov chain Monte Carlo. *arXiv preprint arXiv:1506.08852*.

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*.

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.

Davis, M. H. (1993). *Markov Models & Optimization*, volume 49. CRC Press.

Duane, S., Kennedy, A. D., Pendleton, B. J., and Roweth, D. (1987). Hybrid Monte Carlo. *Physics letters B*, 195(2):216–222.

Galbraith, N. (2016). On Event-Chain Monte Carlo Methods. *Master's thesis, Department of Statistics, Oxford University*.

Horowitz, A. M. (1991). A generalized guided Monte Carlo algorithm. *Physics Letters B*, 268(2):247–252.

Neal, R. M. (2010). MCMC using Hamiltonian dynamics. *Handbook of Markov Chain Monte Carlo*, 54(113-162).

Nishimura, A. and Dunson, D. (2015). Recycling intermediate steps to improve Hamiltonian Monte Carlo. *arXiv preprint arXiv:1511.06925*.

Peters, E. A. et al. (2012). Rejection-free Monte Carlo sampling for general potentials. *Physical Review E*, 85(2):026703.

Sherlock, C. and Thiery, A. H. (2017). A Discrete Bouncy Particle Sampler. *arXiv preprint arXiv:1707.05200*.

Vanetti, P., Bouchard-Côté, A., Deligiannidis, G., and Doucet, A. (2017). Piecewise Deterministic Markov Chain Monte Carlo. *arXiv preprint arXiv:1707.05296*.

Generalized Bouncy Particle Sampler

Joint work with Christian P. ROBERT

4.1 Introduction

As a powerful sampling technique, Markov chain Monte Carlo (MCMC) method has been widely used in computational statistics and is now a standard tool in Bayesian inference, where posterior distribution is often intractable analytically, known up to a constant. However, almost all existing MCMC algorithms, such as Metropolis-Hastings algorithm (MH), Hamiltonian Monte Carlo (HMC) and Metropolis adjusted Langevin algorithm (MALA), are based on detailed balance condition, dating back to (Metropolis et al. (1953), Hastings (1970)). Recently, a novel type of MCMC method — piecewise deterministic Markov process (PDMP) — appeared in computational statistics, method that is generally irreversible, meaning a violation of the detailed balance condition. The theory of PDMP is developed by (Davis (1984), Davis (1993)), while its quite remarkable applications on computational statistics are implemented by (Bouchard-Côté et al. (2018), Bierkens et al. (2017), Bierkens et al. (2016)).

Compared with traditional MCMC algorithms, PDMP is rejection-free, which means that there is no waste of proposal samples. Based on detailed balance condition, traditional MCMC algorithms are reversible. However, some theoretic work and numerical experiments (Hwang et al. (1993), Sun et al. (2010), Chen and Hwang (2013), Bierkens (2016)) have shown that irreversible Markov chain can outperform reversible MCMC with respect to mixing rate and asymptotic variance. PDMP is a typically irreversible Markov chain, which is of interest to investigate. Bouncy particle sampler (BPS) generates a special piecewise deterministic Markov chain, originating in Peters et al. (2012) and being explored by Bouchard-Côté et al. (2018). Zig-zag process sampler Bierkens et al. (2016) is another PDMP example and Fearnhead et al. (2016) unifies the BPS and zig-zag process sampler in the framework of PDMP. Besides, MCMC algorithms are difficult to scale, since computing each MH acceptance ratio needs to sweep over the whole data set. However, according to their special structures, BPS and zig-zag process sampler are easy to scale for big data. Except PDMP style MCMC algorithms, almost all other existing scalable MCMC algorithms (such as, Scott et al. (2016), Neiswanger et al. (2013), Wang and Dunson (2013), Minsker et al. (2017), Quiroz et al. (2018), Bardenet et al. (2017)), are approximate, not exact, which do not admit the target distribution as their invariant distribution.

In this chapter, we generalize bouncy particle sampler – generalized bouncy particle sampler (GBPS) – which can be treated as an extension of BPS and zig-zag process sampler. In BPS, the transition dynamic at event time is deterministic. In opposition, the transition dynamic in GBPS is random with respect to some distribution. In zig-zag process sampler, we decompose the velocity with respect to some fixed coordinate system, while in GBPS, we use a moving coordinate system to decompose the velocity. Besides, the main gain of GBPS compared with BPS is that there is no parameter to tune. In fact, for BPS, in order to overcome the reducibility problem, we need to add a refreshment Poisson process to refresh the velocity occasionally, which needs to be tuned to balance the efficiency and accuracy. But for GBPS, the randomness of refreshment is incorporated into the transition dynamics and there is no parameter to tune.

This chapter is organized as follows: we introduce piecewise deterministic Markov process and bouncy particle sampler in section 2, followed by the introduction of generalized bouncy particle sampler (GBPS) and its implementation issues in section 3. In section 4, we present three numerical experiments of GBPS. At last, we discuss some questions and conclude in section 5.

4.2 Piecewise Deterministic Markov Process

In this section, suppose $\pi(\mathbf{x})$ be the target distribution, where $\mathbf{x} \in \mathbb{R}^d$. We introduce an auxiliary variable, $\mathbf{v} \in \mathbb{R}^d$, called velocity, which is restricted to be of unit length, $\|\mathbf{v}\|_2 = 1$. Denote $\mathbf{z} = (\mathbf{x}, \mathbf{v}) \in \mathbb{R}^d \times S_{d-1}$. In order to obtain the target, we just need to force $\pi(\mathbf{x})$ to be the marginal distribution of $\pi(\mathbf{x}, \mathbf{v})$ with respect to $\mathbf{x}$. Let $\{\mathbf{z}_t\}$ denote a piecewise deterministic Markov chain of $\mathbf{z}$ on the augmented space $(\mathbf{x}, \mathbf{v}) \in \mathbb{R}^d \times S_{d-1}$. The dynamics of PDMP consist of three types of dynamics, namely, deterministic dynamic, event occurrence and transition dynamic. Specifically,

1. **The deterministic dynamic**: between two event times, the Markov process evolves deterministically, according to some partial differential equation:

$$\frac{dz_t^{(i)}}{dt} = \Psi^{(i)}(\mathbf{z}_t), \quad i = 1, \cdots, 2d$$

2. **The event occurrence**: the event occurs at the rate: $\lambda(\mathbf{z}_t)$.

3. **The transition dynamic**: At the event time, τ, we denote $\mathbf{z}_{\tau-}$ the state prior to τ, then $\mathbf{z}_\tau \sim Q(\cdot|\mathbf{z}_{\tau-})$

Following from (Davis (1993), Theorem 26.14), this Markov process's extension generator is

$$\mathcal{A}f(\mathbf{z}) = \nabla f(\mathbf{z}) \cdot \Psi(\mathbf{z}) + \lambda(\mathbf{z}) \int_{\mathbb{R}^d \times S_{d-1}} (f(\mathbf{z}') - f(\mathbf{z}))\, Q(d\mathbf{z}'; \mathbf{z})$$

4.2.1 Bouncy Particle Sampler

Bouncy particle sampler (BPS) is a specific piecewise deterministic Markov process, which admits $\pi(\mathbf{x})d\mathbf{x} \otimes d\mathbf{v}$ over the state space $\mathbb{R}^d \times S_{d-1}$ as its invariant distribution, by specifying the event rate $\lambda(\mathbf{z})$ and the transition dynamic $Q(d\mathbf{z}'; \mathbf{z})$.

1. **The deterministic dynamic**:

$$\frac{dx_t^{(i)}}{dt} = v_t^{(i)}, \quad \frac{dv_t^{(i)}}{dt} = 0, \quad i = 1, \cdots, d$$

2. **The event occurrence**: $\lambda(\mathbf{z}_t) = \max\{0, -\mathbf{v}_t \cdot \nabla \log \pi(\mathbf{x}_t)\}$.

3. **The transition dynamic**: $Q(\cdot|\mathbf{x}, \mathbf{v}) = \delta_{(\mathbf{x}, P_{\mathbf{x}}\mathbf{v})}(\cdot)$, where

$$P_{\mathbf{x}}\mathbf{v} = \mathbf{v} - 2\frac{\langle \mathbf{v}, \nabla \log \pi(\mathbf{x})\rangle}{\langle \nabla \log \pi(\mathbf{x}), \nabla \log \pi(\mathbf{x})\rangle}\nabla \log \pi(\mathbf{x})$$

Bouchard-Côté et al. (2018) has shown that BPS admits $\pi(\mathbf{x})d\mathbf{x} \otimes d\mathbf{v}$ as its invariant distribution. However, the authors also find that pure BPS (specified above) meets with a reducibility problem and add a reference Poisson process into BPS to overcome it. The workflow of BPS with refreshment is shown in Algorithm 4.1.

Algorithm 4.1 Bouncy Particle Sampler

Initialize: $\mathbf{x}_0, \mathbf{v}_0, T_0 = 0$.
for $i = 1, 2, 3, \cdots$ **do**
  Generate $\tau \sim PP(\lambda(\mathbf{x}_t, \mathbf{v}_t))$
  Generate $\tau^{\text{ref}} \sim PP(\lambda^{\text{ref}})$
  if $\tau \leq \tau^{\text{ref}}$ **then**
    $T_i \leftarrow T_{i-1} + \tau$
    $\mathbf{x}_i \leftarrow \mathbf{x}_{i-1} + \tau \mathbf{v}_{i-1}$
    $\mathbf{v}_i \leftarrow \mathbf{v}_{i-1} - 2\frac{\langle \mathbf{v}_{i-1}, \nabla \log \pi(\mathbf{x}_i)\rangle}{\langle \nabla \log \pi(\mathbf{x}_i), \nabla \log \pi(\mathbf{x}_i)\rangle}\nabla \log \pi(\mathbf{x}_i)$
  else
    $T_i \leftarrow T_{i-1} + \tau^{\text{ref}}$
    $\mathbf{x}_i \leftarrow \mathbf{x}_{i-1} + \tau^{\text{ref}} \mathbf{v}_{i-1}$
    $\mathbf{v}_i \sim \mathcal{U}(S_{d-1})$
  end if
end for

4.3 Generalized Bouncy Particle Sampler

In BPS, at event time, the velocity changes deterministically. However, we find that the velocity can be changed into other directions, according to some distribution, at event time, which incorporates the randomness of the reference Poisson process in BPS to overcome the reducibility. In this section, we generalize the BPS. Specifically, prior to event time, we decompose the velocity according to the gradient of $\log \pi(\mathbf{x})$, flip the parallel subvector and resample the orthogonal subvector with respect to some distribution. The details are as follows:

1. **The deterministic dynamic**:

$$\frac{dx_t^{(i)}}{dt} = v_t^{(i)}, \quad \frac{dv_t^{(i)}}{dt} = 0, \quad i = 1, \cdots, d$$

2. **The event occurrence**: $\lambda(\mathbf{z}_t) = \max\{0, -\langle \mathbf{v}_t, \nabla \log \pi(\mathbf{x}_t)\rangle\}$.

3. **The transition dynamic**: $Q(d\mathbf{x}', d\mathbf{v}'|\mathbf{x}, \mathbf{v}) = \delta_{\{\mathbf{x}\}}(d\mathbf{x}')\delta_{\{-\mathbf{v}_1\}}(d\mathbf{v}_1')\mathcal{N}_{\mathbf{v}_1^\perp}(d\mathbf{v}_2')$, where

$$\mathbf{v}_1 = \frac{\langle \mathbf{v}, \nabla \log \pi(\mathbf{x})\rangle}{\langle \nabla \log \pi(\mathbf{x}), \nabla \log \pi(\mathbf{x})\rangle} \nabla \log \pi(\mathbf{x}), \quad \mathbf{v}_2 = \mathbf{v} - \mathbf{v}_1$$

$$\mathbf{v}_1' = \frac{\langle \mathbf{v}', \nabla \log \pi(\mathbf{x})\rangle}{\langle \nabla \log \pi(\mathbf{x}), \nabla \log \pi(\mathbf{x})\rangle} \nabla \log \pi(\mathbf{x}), \quad \mathbf{v}_2' = \mathbf{v}' - \mathbf{v}_1'$$

$$\mathbf{v}_1^\perp = \left\{\mathbf{u} \in \mathbb{R}^d : \langle \mathbf{u}, \mathbf{v}_1\rangle = 0\right\}$$

$\mathcal{N}_{\mathbf{v}_1^\perp}$ is the $(d-1)-$dimensional standard normal distribution over the space $\mathbf{v}_1^\perp$.

We summarize the GBPS in Algorithm 4.2.

Algorithm 4.2 Generalized Bouncy Particle Sampler

Initialize: $\mathbf{x}_0, \mathbf{v}_0, T_0 = 0$.
for $i = 1, 2, 3, \cdots$ **do**
  Generate $\tau \sim PP(\lambda(\mathbf{x}_t, \mathbf{v}_t))$
  $T_i \leftarrow T_{i-1} + \tau$
  $\mathbf{x}_i \leftarrow \mathbf{x}_{i-1} + \tau \mathbf{v}_{i-1}$
  $\mathbf{v}_i \leftarrow Q(d\mathbf{v}|\mathbf{x}_i, \mathbf{v}_{i-1})$
end for

Theorem 4.3.1 *The above piecewise deterministic Markov chain admits* $\pi(\mathbf{x})d\mathbf{x} \otimes \psi_d(\mathbf{v})d\mathbf{v}$ *over* $\mathbb{R}^{2d}$ *as its invariant distribution, where* $\psi_d(\mathbf{v})$ *is the density function of* $d-$*dimensional standard normal distribution.*

Proof: In order to prove $\pi(\mathbf{x})d\mathbf{x} \otimes \psi_d(\mathbf{v})d\mathbf{v}$ is the invariant distribution of generator $\mathcal{A}$ of the above Markov chain, we just need to prove the following equation is satisfied by appropriate functions f:

$$\int_{\mathbb{R}^d}\int_{\mathbb{R}^d} \mathcal{A}f(\mathbf{x}, \mathbf{v})\pi(\mathbf{x})\psi_d(\mathbf{v})d\mathbf{x}d\mathbf{v} = 0$$

where by Theorem 26.14, Davis (1993)

$$\mathcal{A}f(\mathbf{z}) = \nabla f(\mathbf{z}) \cdot \Psi(\mathbf{z}) + \lambda(\mathbf{x}, \mathbf{v}) \int_{\mathbf{v}' \in \mathbb{R}^d} f(\mathbf{x}, \mathbf{v}')Q(d\mathbf{v}'|\mathbf{x}, \mathbf{v}) - \lambda(\mathbf{x}, \mathbf{v})f(\mathbf{x}, \mathbf{v})$$

Since for bounded f,

$$\begin{aligned}
&\int_{\mathbb{R}^d}\int_{\mathbb{R}^d} \langle \nabla f(\mathbf{z}), \Psi(\mathbf{z})\rangle \pi(\mathbf{x})\psi_d(\mathbf{v})d\mathbf{x}d\mathbf{v} \\
&= \int_{\mathbb{R}^d}\int_{\mathbb{R}^d} \langle \nabla_{\mathbf{x}} f(\mathbf{x}, \mathbf{v}), \mathbf{v}\rangle \pi(\mathbf{x})\psi_d(\mathbf{v})d\mathbf{x}d\mathbf{v} \\
&= \int_{\mathbb{R}^d}\int_{\mathbb{R}^d} \langle -\nabla \log \pi(\mathbf{x}), \mathbf{v}\rangle f(\mathbf{x}, \mathbf{v})\pi(\mathbf{x})\psi_d(\mathbf{v})d\mathbf{x}d\mathbf{v}
\end{aligned}$$

For each $\mathbf{x}$, decomposing the velocity spaces, $\mathbb{R}^d$, into the direct sum, $\mathbf{v}_1^{\|} \oplus \mathbf{v}_1^\perp$, such that $\mathbf{v}_1 = (\|\mathbf{v}_1\|_2, 0, \cdots, 0)$, then

$$\lambda(\mathbf{x}, (v_1, \cdots, v_d)) = \max\{0, -\langle (v_1, 0, \cdots, 0), \nabla \log \pi(\mathbf{x})\rangle\}$$

$$
\begin{aligned}
&\int_{\mathbf{v}\in\mathbb{R}^d}\int_{\mathbf{v}'\in\mathbb{R}^d} f(\mathbf{x},\mathbf{v}')\lambda(\mathbf{x},\mathbf{v})\pi(\mathbf{x})\psi_d(\mathbf{v})Q(d\mathbf{v}'|\mathbf{x},\mathbf{v})d\mathbf{v}\\
&=\int_{v_1\in\mathbb{R}}\int_{(v_2,\cdots,v_d)\in\mathbb{R}^{d-1}}\int_{v_1'\in\mathbb{R}}\int_{(v_2',\cdots,v_d')\in\mathbb{R}^{d-1}} f(\mathbf{x},(v_1',\cdots,v_d'))\\
&\times\lambda(\mathbf{x},(v_1,\cdots,v_d))\pi(\mathbf{x})\psi_1(v_1)\psi_{d-1}(v_2,\cdots,v_d)\psi_{d-1}(v_2',\cdots,v_d')\\
&\times\delta_{\{-v_1\}}(v_1')dv_1dv_2\cdots dv_ddv_1'dv_2'\cdots dv_d'\\
&=\int_{v_1\in\mathbb{R}}\int_{(v_2,\cdots,v_d)\in\mathbb{R}^{d-1}}\int_{(v_2',\cdots,v_d')\in\mathbb{R}^{d-1}} f(\mathbf{x},(-v_1,v_2',\cdots,v_d'))\\
&\times\max\left\{0,-\langle(v_1,0,\cdots,0),\nabla\log\pi(\mathbf{x})\rangle\right\}\pi(\mathbf{x})\psi_1(v_1)\psi_{d-1}(v_2,\cdots,v_d)\\
&\times\psi_{d-1}(v_2',\cdots,v_d')dv_1dv_2\cdots dv_ddv_2'\cdots dv_d'\\
&\qquad\text{(by the change of variable: } v_1\to -v_1 \text{ and } \psi_1(-v_1)=\psi_1(v_1))\\
&=\int_{v_1\in\mathbb{R}}\int_{(v_2,\cdots,v_d)\in\mathbb{R}^{d-1}}\int_{(v_2',\cdots,v_d')\in\mathbb{R}^{d-1}} f(\mathbf{x},(v_1,v_2',\cdots,v_d'))\\
&\times\max\left\{0,-\langle(-v_1,0,\cdots,0),\nabla\log\pi(\mathbf{x})\rangle\right\}\pi(\mathbf{x})\psi_1(-v_1)\psi_{d-1}(v_2,\cdots,v_d)\\
&\times\psi_{d-1}(v_2',\cdots,v_d')dv_1dv_2\cdots dv_ddv_2'\cdots dv_d'\\
&=\int_{v_1\in\mathbb{R}}\int_{(v_2,\cdots,v_d)\in\mathbb{R}^{d-1}}\int_{(v_2',\cdots,v_d')\in\mathbb{R}^{d-1}} f(\mathbf{x},(v_1,v_2',\cdots,v_d'))\\
&\times\max\{0,\langle(v_1,0,\cdots,0),\nabla\log\pi(\mathbf{x})\rangle\}\pi(\mathbf{x})\psi_1(v_1)\psi_{d-1}(v_2,\cdots,v_d)\\
&\times\psi_{d-1}(v_2',\cdots,v_d')dv_1dv_2\cdots dv_ddv_2'\cdots dv_d'\\
&=\int_{v_1\in\mathbb{R}}\int_{(v_2',\cdots,v_d')\in\mathbb{R}^{d-1}} f(\mathbf{x},(v_1,v_2',\cdots,v_d'))\psi_1(v_1)\psi_{d-1}(v_2',\cdots,v_d')\\
&\times\max\{0,\langle(v_1,0,\cdots,0),\nabla\log\pi(\mathbf{x})\rangle\}\pi(\mathbf{x})dv_1dv_2'\cdots dv_d'\\
&=\int_{\mathbf{v}\in\mathbb{R}^d} f(\mathbf{x},\mathbf{v})\lambda(\mathbf{x},-\mathbf{v})\pi(\mathbf{x})\psi_d(\mathbf{v})d\mathbf{v}
\end{aligned}
$$

As a result,

$$
\begin{aligned}
&\int_{\mathbf{x}\in\mathbb{R}^d}\int_{\mathbf{v}\in\mathbb{R}^d}\mathcal{A}f(\mathbf{x},\mathbf{v})\pi(\mathbf{x})\psi_d(\mathbf{v})d\mathbf{x}d\mathbf{v}\\
&=\int_{\mathbf{x}\in\mathbb{R}^d}\int_{\mathbf{v}\in\mathbb{R}^d}\left[\langle-\nabla\log\pi(\mathbf{x}),\mathbf{v}\rangle\right]f(\mathbf{x},\mathbf{v})\pi(\mathbf{x})\psi_d(\mathbf{v})d\mathbf{x}d\mathbf{v}\\
&+\int_{\mathbf{x}\in\mathbb{R}^d}\int_{\mathbf{v}\in\mathbb{R}^d}\left[\lambda(\mathbf{x},-\mathbf{v})-\lambda(\mathbf{x},\mathbf{v})\right]f(\mathbf{x},\mathbf{v})\pi(\mathbf{x})\psi_d(\mathbf{v})d\mathbf{x}d\mathbf{v}\\
&\qquad\text{(since } \lambda(\mathbf{x},-\mathbf{v})-\lambda(\mathbf{x},\mathbf{v})=\langle\mathbf{v},\nabla\log\pi(\mathbf{x})\rangle)\\
&=\int_{\mathbb{R}^d}\int_{\mathbb{R}^d}\left[\langle-\nabla\log\pi(\mathbf{x}),\mathbf{v}\rangle+\langle\mathbf{v},\nabla\log\pi(\mathbf{x})\rangle\right]f(\mathbf{x},\mathbf{v})\pi(\mathbf{x})d\mathbf{x}d\mathbf{v}\\
&=0
\end{aligned}
$$

proof In order to establish the ergodicity theorem of GBPS, we propose an assumption on the target distribution $\pi(\mathbf{x})$.

Assumption 1: For any two points $\mathbf{x}_1,\mathbf{x}_2\in\mathbb{R}^d$ and any velocity $\mathbf{v}\in\mathbb{R}^d$, $\|\mathbf{v}\|_2=1$, there exists $t>0$, such that

$$\mathbf{x}_2\in S^{\perp}(\mathbf{x}_1+t\mathbf{v},\mathbf{v})$$

Theorem 4.3.2 *Under Assumption 1, the Markov chain* $\mathbf{z}_t'=(\mathbf{x}_t,\frac{\mathbf{v}_t}{\|\mathbf{v}_t\|})$ *induced by GBPS admits* $\pi(\mathbf{x})\times\mathcal{U}(S_{d-1})$ *as its unique invariant distribution.*

The proof of Theorem 4.3.2 and the definitions of notations in Assumption 1 can be found in Appendix. Whether Theorem 4.3.2 is still correct without Assumption 1 is an open question.

4.3.1 Construction of Estimator

While constructing an unbiased estimator of $I = \int h(\mathbf{x})\pi(d\mathbf{x})$, we cannot use the skeleton of the simulated GBPS path directly. In fact, such an estimator is biased. Suppose $\{\mathbf{x}_i, \mathbf{v}_i, T_i\}_{i=0}^M$ be the skeleton of an simulated trajectory, which means that at event time T_i, the state is $(\mathbf{x}_i, \mathbf{v}_i)$. Then, the whole trajectory $\mathbf{x}_{[0,T_M]}$ is filled up with

$$\mathbf{x}_t = \mathbf{x}_i + (t - T_i)\mathbf{v}_i, \quad T_i \leq t < T_{i+1}$$

Let n be the number of data points selected from this trajectory, then an estimator of I is constructed as

$$\hat{I} = \frac{1}{n}\sum_{i=1}^{n} h(\mathbf{x}_{\frac{iT_M}{n}})$$

4.3.2 Implementation

The main difficult to implement BPS and GBPS is to simulate event time, which follows a Poisson process. The common techniques are based on the thinning and superposition theorems of Poisson process.

Theorem 4.3.3 (Superposition Theorem Kingman (1993)) *Let $\Pi_1, \Pi_2, \cdots$, be a countable collection of independent Poisson processes on state space $\mathbb{R}^+$ and let Π_n have rate $\lambda_n(t)$ for each n. If $\sum_{n=1}^{\infty}\lambda_n(t) < \infty$ for all t, then the superposition*

$$\Pi = \bigcup_{n=1}^{\infty} \Pi_n$$

is a Poisson process with rate

$$\lambda(t) = \sum_{n=1}^{\infty} \lambda_n(t)$$

Theorem 4.3.4 (Thinning Theorem Lewis and Shedler (1979)) *Let $\lambda : \mathbb{R}^+ \to \mathbb{R}^+$ and $\Lambda : \mathbb{R}^+ \to \mathbb{R}^+$ be continuous such that $\lambda(t) \leq \Lambda(t)$ for all $t \geq 0$. Let $\tau^1, \tau^2, \cdots$, be the increasing finite or infinite sequence of points of a Poisson process with rate $\Lambda(t)$. For all i, delete the point τ^i with probability $1 - \lambda(t)/\Lambda(t)$. Then the remaining points $\tilde{\tau}^1, \tilde{\tau}^2, \cdots$ form a non-homogeneous Poisson process with rate $\lambda(t)$.*

In GBPS, from a given state $(\mathbf{x}, \mathbf{v})$, the associated Poisson process $\Pi_{\mathbf{x},\mathbf{v}}$ has a rate function $\lambda(t) = \lambda(\mathbf{x} + t\mathbf{v}, \mathbf{v})$. With the help of the above two theorems, we can simulate a sample from $\Pi_{\mathbf{x},\mathbf{v}}$ feasibly.

Let $\eta(t) = \int_0^t \lambda(s)ds$, then the first event time, τ, of Poisson process Π, whose rate function is $\lambda(t)$, satisfies

$$\mathbb{P}(\tau > u) = \mathbb{P}(\Pi \cap [0, u] = \emptyset) = \exp(-\eta(u))$$

By the inverse theorem, τ can be simulated with the help of a uniform variate $V \sim \mathcal{U}(0,1)$ via:

$$\tau = \eta^{-1}(-\log(V))$$

If we can compute η^{-1} analytically, it is easy to simulate the event times. Otherwise, the simulations commonly depend on the superposition and thinning theorems.

4.3.3 GBPS with Sub-sampling in Big Data

In Bayesian analysis, we suppose the observations $\{y_1, y_2, \cdots, y_N\}$ are i.i.d. samples from some distribution in the family $\{\mathbb{P}_{\mathbf{x}}, \mathbf{x} \in \mathbb{R}^d\}$ and let $\mathbb{P}_{\mathbf{x}}$ admit the density $p_{\mathbf{x}}$ with respect to the Lebesgue measure on $\mathbb{R}^d$. Given a prior $\pi_0(\mathbf{x})$ over the parameter $\mathbf{x}$, the posterior is

$$\pi(\mathbf{x}) \overset{\text{def}}{=} \pi(\mathbf{x}|y_1, \cdots, y_N) \propto \pi_0(\mathbf{x}) \prod_{n=1}^{N} p_{\mathbf{x}}(y_n)$$

Traditional MCMC algorithms (with MH step) are difficult to scale for large data set, since each MH step needs to sweep over the whole data set. However, as indicated in Bierkens et al. (2016), PDMP may be super-efficient by using sub-sampling to simulate samples from the target distribution if we can give a tight upper bound of the rate function. In GBPS, we only use the gradient of the logarithm of the target distribution, which means we can simulate the posterior by knowing it up to a constant. Besides, we can give an unbiased estimator of the gradient of the logarithm of the posterior by using its sum structure to simulate the posterior exactly:

$$\widehat{\nabla \log \pi(\mathbf{x})} = N \nabla \log \pi_I(\mathbf{x}) = \nabla \log \pi_0(\mathbf{x}) + N \nabla_{\mathbf{x}} \log p_{\mathbf{x}}(y_I), \quad I \sim \mathcal{U}\{1, 2, \cdots, N\}$$

In Algorithm 4.3, we show the workflow of the implementation of subsampling in GBPS. Notice that $\lambda(\Delta, \mathbf{v}_{i-1})$ equals to $\lambda(\mathbf{x}, \mathbf{v}_{i-1})$ in which $\nabla \log \pi(\mathbf{x})$ is replaced by Δ. $\Lambda(t)$ is an upper bound of $\lambda(\mathbf{x}, \mathbf{v})$.

Algorithm 4.3 Subsampling version

Initialize: $\mathbf{x}_0, \mathbf{v}_0, T_0 = 0$.
for $i = 1, 2, 3, \cdots$ **do**
  Generate $\tau \sim PP(\Lambda(t))$
  $T_i \leftarrow T_{i-1} + \tau$
  $\mathbf{x}_i \leftarrow \mathbf{x}_{i-1} + \tau \mathbf{v}_{i-1}$
  $I \sim \mathcal{U}(\{1, \cdots, N\})$
  $\Delta \leftarrow N \nabla \log \pi_I(\mathbf{x}_i)$
  $q \leftarrow \lambda(\Delta, \mathbf{v}_{i-1}) / \Lambda(\tau)$
  $u \sim \mathcal{U}(0,1)$
  if $u \leq q$ **then**
    $\mathbf{v}_i \leftarrow Q(d\mathbf{v}|\Delta, \mathbf{v}_{i-1})$
  else
    $\mathbf{v}_i \leftarrow \mathbf{v}_{i-1}$
  end if
end for

4.4 Numerical simulations

In this section, we apply GBPS algorithm on three numerical experiments. Example 1 shows that reducibility problem appears in isotropic Gaussian distribution for BPS without refreshment but is not encountered by GBPS. In Example 2, we can find that GBPS works well on multimode distributions and with similar performance with BPS. Finally, we present the GBPS with sub-sampling on Bayesian logistic model.

Example 1: (isotropic Gaussian distribution) In this example, we show the reducibility problem of BPS without refreshment. The target distribution is

$$\pi(\mathbf{x}) = \frac{1}{2\pi} \exp\left\{-\frac{x_1^2 + x_2^2}{2}\right\}$$

First we apply the BPS without reference Poisson process and show its reducibility in Figure 4.1. Compared with BPS without refreshment, GBPS is irreducible, shown

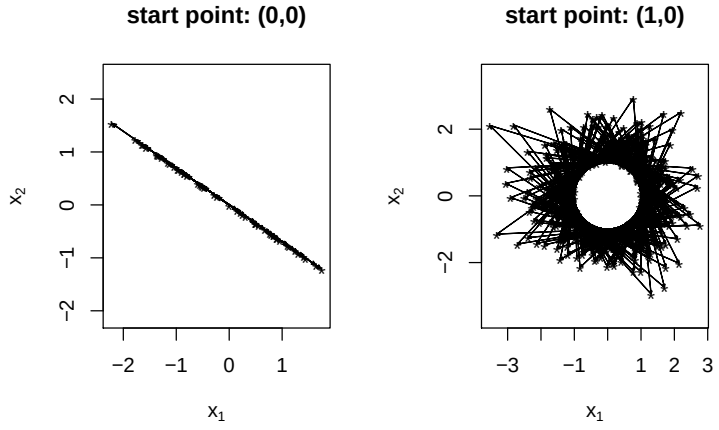

Figure 4.1: Reducibility problem in isotropic Gaussian distributions: (left) the first 50 segments of a BPS path without refreshment which starts from the center of the Gaussian distribution, the trajectory is on a line; (right) the first 500 segments of another BPS path with $\lambda^{\text{ref}} = 0$ starting from an point except the center, the trajectory cannot explore the center area.

in Figure 4.2.

Secondly, we compare the performance of GBPS and BPS with refreshment. For BPS, we set $\lambda^{\text{ref}} = \{0.01, 0.1, 0.2, 0.5, 1\}$. Each method is run 50 times and each sampled path has length 10^4. For each path, we sample 10^4 points with length gap 1. Figure 4.3 shows the errors of the first and second moments of each component and Figure 4.4 presents the errors in terms of Wasserstein-2 distance with respect to the target distribution and the effective sample size of each method.

For BPS, we need to tune the rate of reference Poisson process to balance the efficiency and accuracy. Event though BPS is ergodic for every positive refreshment rate λ^{ref} in theory, the value of λ^{ref} matters in implementation. The smaller the refreshment rate, the larger the effective sample size (high efficiency), the more slowly the chain mixes. The larger the refreshment rate, the smaller the effective sample

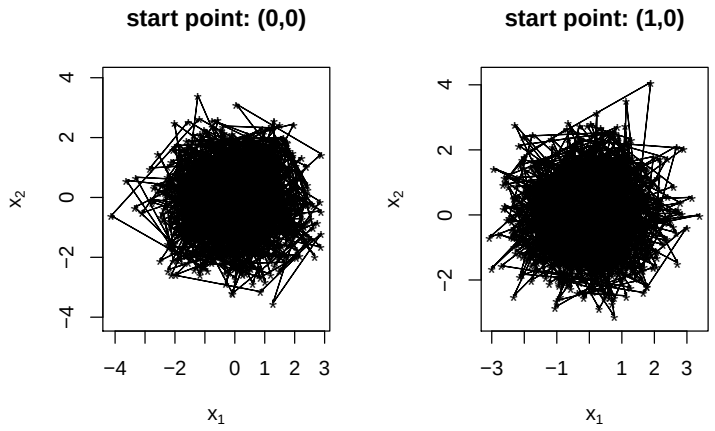

Figure 4.2: GBPS is irreducible in isotropic Gaussian distribution. (left) the first 1000 segments of a GBPS path which starts from the center of the Gaussian distribution; (right) the first 1000 segments of another GBPS path starting from an point except the center

size (low efficiency), the faster the chain mixes. However, when the refreshment rate is extremely large or small, BPS will produce chains approximating the target distribution poorly. On the other hand, there is no hyper-parameter to tune in GBPS, which incorporates the randomness of BPS in refreshment into transition dynamics. Compared to BPS with different refreshment rates, GBPS performs modest in terms of the first and second moments of each component. In terms of Wasserstein-2 distance, GBPS outperforms BPS.

Example 2: (mixture of Gaussian model) In this example, we show how to simulate the event time by using superposition and thinning theorems. The target is a mixture of Gaussian distributions:

$$\pi(x_1, x_2) = \frac{p}{2\pi\sigma_1\sigma_2} \exp\left\{-\frac{(x_1-3)^2}{2\sigma_1^2} - \frac{x_2^2}{2\sigma_2^2}\right\} + \frac{1-p}{2\pi\sigma_3\sigma_4} \exp\left\{-\frac{x_1^2}{2\sigma_3^2} - \frac{(x_2-3)^2}{2\sigma_4^2}\right\}$$

In our experiment, we set $p = 0.5, (\sigma_1, \sigma_2, \sigma_3, \sigma_4) = (1, 1.5, 2, 1)$. The gradient is

$$\begin{aligned}\frac{\partial\pi(x_1, x_2)}{\partial x_1} &= \frac{p}{2\pi\sigma_1\sigma_2} \exp\left\{-\frac{(x_1-3)^2}{2\sigma_1^2} - \frac{x_2^2}{2\sigma_2^2}\right\}\left(-\frac{(x_1-3)}{\sigma_1^2}\right) \\ &+ \frac{1-p}{2\pi\sigma_3\sigma_4} \exp\left\{-\frac{x_1^2}{2\sigma_3^2} - \frac{(x_2-3)^2}{2\sigma_4^2}\right\}\left(-\frac{x_1}{\sigma_3^2}\right)\end{aligned}$$

$$\begin{aligned}\frac{\partial\pi(x_1, x_2)}{\partial x_2} &= \frac{p}{2\pi\sigma_1\sigma_2} \exp\left\{-\frac{(x_1-3)^2}{2\sigma_1^2} - \frac{x_2^2}{2\sigma_2^2}\right\}\left(-\frac{x_2}{\sigma_2^2}\right) \\ &+ \frac{1-p}{2\pi\sigma_3\sigma_4} \exp\left\{-\frac{x_1^2}{2\sigma_3^2} - \frac{(x_2-3)^2}{2\sigma_4^2}\right\}\left(-\frac{(x_2-3)}{\sigma_4^2}\right)\end{aligned}$$

We can give an upper bound for the norm of the gradient of the logarithm of the target density function:

$$\|\nabla \log \pi(x_1, x_2)\|_2 \leq \frac{|x_1-3|}{\sigma_1^2} + \frac{|x_1|}{\sigma_3^2} + \frac{|x_2|}{\sigma_2^2} + \frac{|x_2-3|}{\sigma_4^2}$$

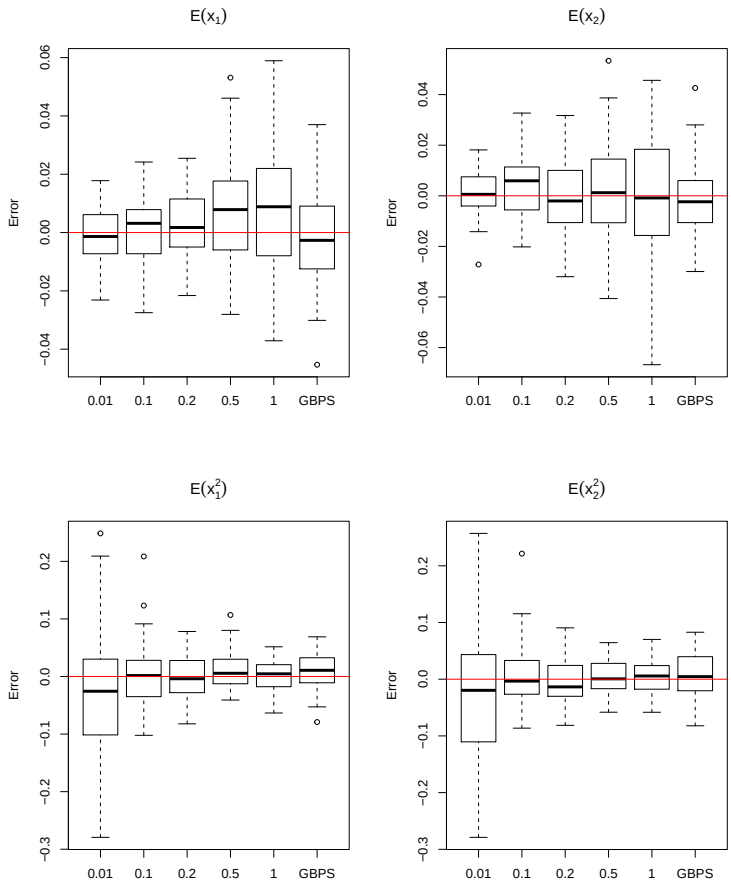

Figure 4.3: Comparison between BPS and GBPS in isotropic Gaussian distribution. For each graph, the first five boxes represent the BPS method with different refreshment rates $\lambda^{\text{ref}} = \{0.01, 0.1, 0.2, 0.5, 1\}$.

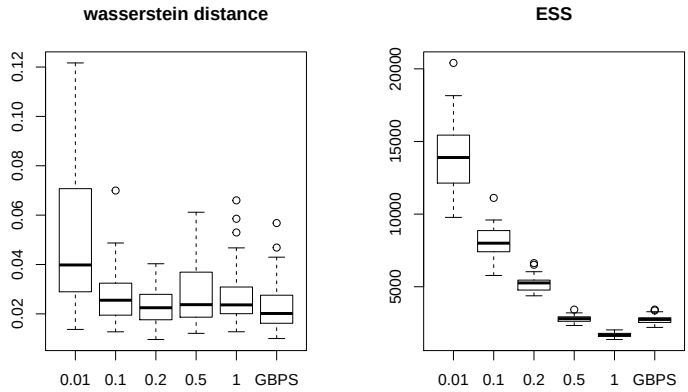

Figure 4.4: Comparison between BPS and GBPS in isotropic Gaussian distribution in terms of Wasserstein distance and effective sample size. For each graph, the first five boxes represent the BPS method with different refreshment rates $\lambda^{\text{ref}} = \{0.01, 0.1, 0.2, 0.5, 1\}$.

Then an upper bound for $\lambda(\mathbf{x}, \mathbf{v})$ is given as

$$\lambda(\mathbf{x}, \mathbf{v}) \leq \left(\frac{|x_1 - 3|}{\sigma_1^2} + \frac{|x_1|}{\sigma_3^2} + \frac{|x_2|}{\sigma_2^2} + \frac{|x_2 - 3|}{\sigma_4^2} \right) * \|\mathbf{v}\|_2$$

By superposition, we need only focus on Poisson process whose rate function has such form: $\lambda(x, v) = \frac{|x-\mu|}{\sigma^2}$. Let $\lambda_s(x, v) = \lambda(x + sv, v) = \frac{|x+sv-\mu|}{\sigma^2}$. Define

$$\eta(t) = \int_0^t \lambda_s(x, v) ds$$

i): If $x > \mu, v > 0$,

$$\eta(t) = \int_0^t \frac{(x-\mu) + sv}{\sigma^2} ds = \frac{\frac{1}{2}vt^2 + (x-\mu)t}{\sigma^2} = \frac{v}{2\sigma^2} \left(t^2 + \frac{2(x-\mu)}{v} t \right)$$
$$= \frac{v}{2\sigma^2} \left[\left(t + \frac{(x-\mu)}{v} \right)^2 - \frac{(x-\mu)^2}{v^2} \right]$$

$$\eta^{-1}(z) = \sqrt{\frac{2\sigma^2 z}{v} + \frac{(x-\mu)^2}{v^2}} - \frac{(x-\mu)}{v}$$

ii) : If $x < \mu, v < 0$, then

$$\eta(t) = \int_0^t \frac{-(x-\mu) - sv}{\sigma^2} ds = \frac{-\frac{1}{2}vt^2 - (x-\mu)t}{\sigma^2} = -\frac{v}{2\sigma^2} \left(t^2 + \frac{2(x-\mu)}{v} t \right)$$
$$= -\frac{v}{2\sigma^2} \left[\left(t + \frac{(x-\mu)}{v} \right)^2 - \frac{(x-\mu)^2}{v^2} \right]$$

$$\eta^{-1}(z) = \sqrt{-\frac{2\sigma^2 z}{v} + \frac{(x-\mu)^2}{v^2}} - \frac{(x-\mu)}{v}$$

iii) : If $x > \mu, v \leq 0$:

$$\eta\left(-\frac{x-\mu}{v} \right) = \int_0^{-\frac{x-\mu}{v}} \frac{sv + (x-\mu)}{\sigma^2} ds = -\frac{(x-\mu)^2}{2v\sigma^2}$$

1. If $z > -\frac{(x-\mu)^2}{2v\sigma^2}$: $t_0 = -\frac{x-\mu}{v}$

$$-\frac{(x-\mu)^2}{2v\sigma^2} + \int_0^t -\frac{sv}{\sigma^2} ds = z, \quad t = \sqrt{-\frac{2\sigma^2 z}{v} - \frac{(x-\mu)^2}{v^2}}$$

$$\eta^{-1}(z) = \sqrt{-\frac{2\sigma^2 z}{v} - \frac{(x-\mu)^2}{v^2}} + \left(-\frac{x-\mu}{v} \right)$$

2. If $z \leq -\frac{(x-\mu)^2}{2v\sigma^2}$:

$$\int_0^t \frac{sv + (x-\mu)}{\sigma^2} ds = \frac{v}{2\sigma^2} \left(t^2 + \frac{2(x-\mu)}{v} t \right) = z$$

$$\eta^{-1}(z) = -\sqrt{\frac{2\sigma^2 z}{v} + \frac{(x-\mu)^2}{v^2}} + \left(-\frac{x-\mu}{v} \right)$$

iv) : If $x \leq \mu, v > 0$:

$$\eta\left(-\frac{x-\mu}{v}\right) = \int_0^{-\frac{x-\mu}{v}} \frac{-sv-(x-\mu)}{\sigma^2} ds = \frac{(x-\mu)^2}{2v\sigma^2}$$

1. If $z > \frac{(x-\mu)^2}{2v\sigma^2}$: $t_0 = -\frac{x-\mu}{v}$

$$\frac{(x-\mu)^2}{2v\sigma^2} + \int_0^t \frac{sv}{\sigma^2} ds = z, \quad t = \sqrt{\frac{2\sigma^2 z}{v} - \frac{(x-\mu)^2}{v^2}}$$

$$\eta^{-1}(z) = \sqrt{\frac{2\sigma^2 z}{v} - \frac{(x-\mu)^2}{v^2}} + \left(-\frac{x-\mu}{v}\right)$$

2. If $z \leq \frac{(x-\mu)^2}{2v\sigma^2}$:

$$\int_0^t \frac{-sv-(x-\mu)}{\sigma^2} ds = -\frac{v}{2\sigma^2}\left(t^2 + \frac{2(x-\mu)}{v}t\right) = z$$

$$\eta^{-1}(z) = -\sqrt{-\frac{2\sigma^2 z}{v} + \frac{(x-\mu)^2}{v^2}} + \left(-\frac{x-\mu}{v}\right)$$

In Figure 4.5, we show the trajectory of the simulated GBPS path and associated samples. Figure 4.6 shows the marginal density functions of the target distribution. In Figure 4.7, we compare the performance of BPS and GBPS. For BPS, we set $\lambda^{\text{ref}} = 0.01, 0.1, 1$. We sample 50 paths with length 10000 for each method and take 10000 points from each path with gap 1 to form samples. Empirically, BPS is ergodic over this example. With the increase of λ^{ref}, the refreshment occurs more frequently, which reduces the performance of BPS. Even though GBPS has worse performance, compared to BPS with some refreshment rates, it is quite reliable and has no parameter to tune.

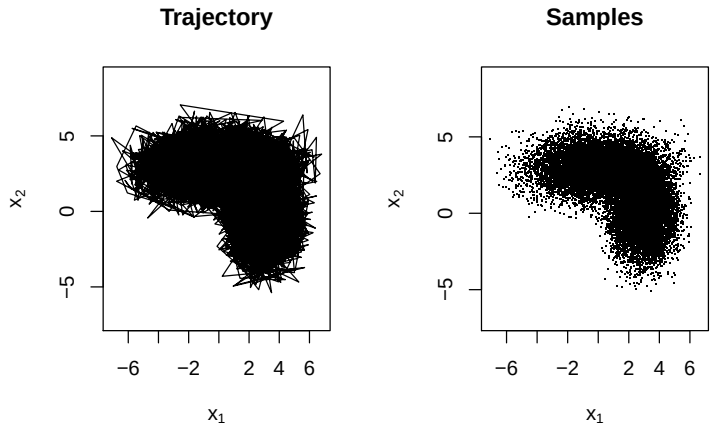

Figure 4.5: The trajectory and samples from a GBPS path.

Example 3: (Bayesian Logistic Model) For the Bayesian logistic model, we suppose $\mathbf{x} \in \mathbb{R}^d$ be the parameters and (y_i, z_i), for $i = 1, 2, \cdots, N$ be the observations, where $y_i \in \mathbb{R}^d, z_i \in \{0, 1\}$, then

$$\mathbb{P}(z_i = 1 | y_i, \mathbf{x}) = \frac{1}{1 + \exp\{-\sum_{\ell=1}^d y_i^\ell x_\ell\}}$$

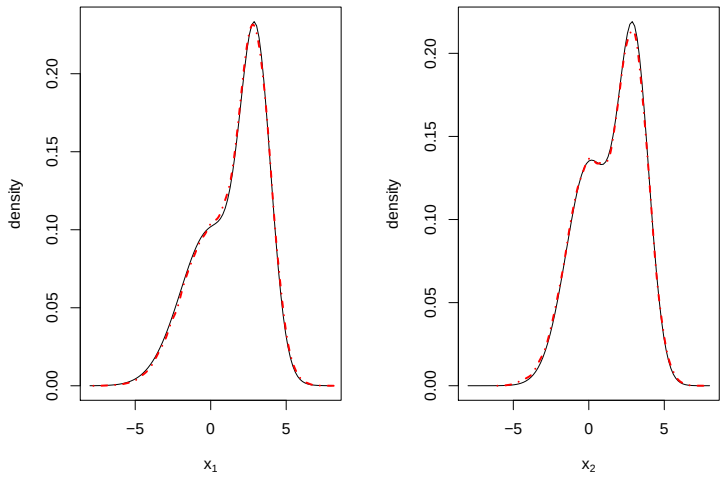

Figure 4.6: The marginal density functions: the black solid lines are true marginal density, the red dotted lines are from a GBPS path.

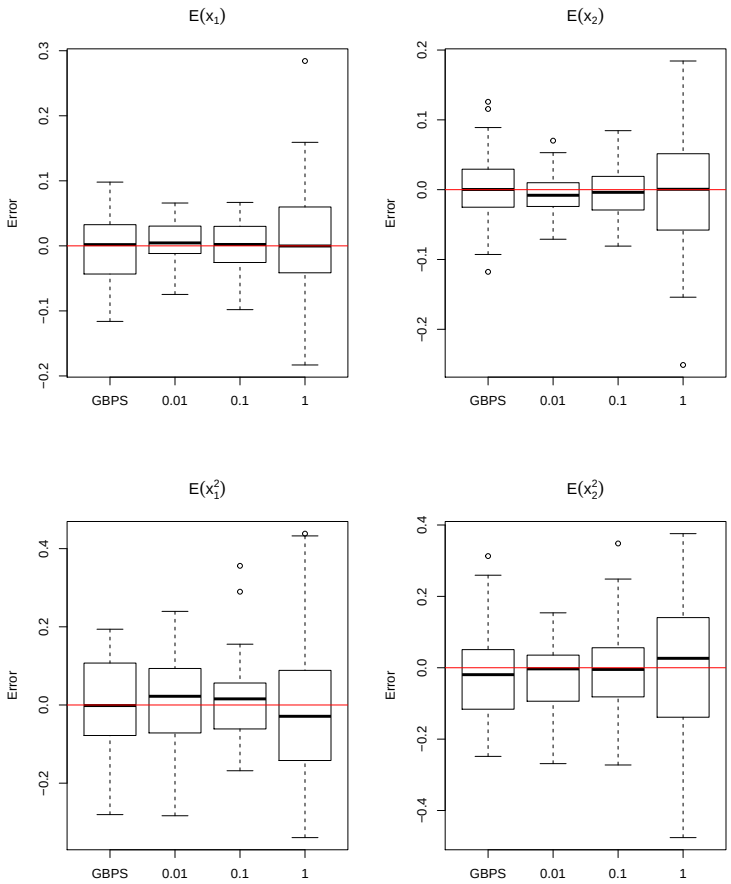

Figure 4.7: Comparison between GBPS and BPS: for each graph, the last three boxes represent BPS with $\lambda^{\text{ref}} = 0.01, 0.1, 1$.

Choosing the improper prior, then the posterior is

$$\pi(\mathbf{x}) \propto \prod_{j=1}^{N} \frac{\exp\{z_j \sum_{\ell=1}^{d} y_j^\ell x_\ell\}}{1+\exp\{\sum_{\ell=1}^{d} y_j^\ell x_\ell\}}$$

for $k = 1, \cdots, d$, the partial derivative is

$$\frac{\partial}{\partial x_k} \log \pi(\mathbf{x}) = \sum_{j=1}^{N} \left[z_j - \frac{\exp\{z_j \sum_{\ell=1}^{d} y_j^\ell x_\ell\}}{1+\exp\{\sum_{\ell=1}^{d} y_j^\ell x_\ell\}} \right] y_j^k$$

Then, they are bounded by

$$\left| \frac{\partial \log \pi(\mathbf{x})}{\partial x_k} \right| \leq \sum_{j=1}^{N} \left| y_j^k \right|$$

and the bounded rate for Poisson process is

$$\lambda^+ = \max_{1 \leq k \leq d} \sum_{j=1}^{N} \left| y_j^k \right|$$

In our experiment, we set $d = 5$, $N = 100$ and use 10 observations for subsampling at each iteration. Figure 4.8 shows the marginal density functions for each component of parameters.

4.5 Conclusion

In this article, we generalize the bouncy particle sampler in terms of its transition dynamics. Our method — Generalized Bouncy Particle Sampler (GBPS) — can be regarded as a bridge between bouncy particle sampler and zig-zag process sampler. Compared with bouncy particle sampler, GBPS changes the direction velocity according to some distribution at event time. However, compared with zig-zag process sampler, GBPS can be regarded as attaching a moving coordinate system on the state space of $(\mathbf{x}, \mathbf{v})$, instead of using a fixed one as zig-zag process sampler. One main advantage of GBPS, compared to BPS, is that it has no parameter to tune.

Throughout the whole paper, we suppose that the parameter space has no restrictions. In practice, it is often the case one encounters restricted parameter space problems. In such cases, we may transfer the restricted region into the whole Euclidean space by reparameterization techniques. Besides, Bierkens et al. (2018) shows some methods to simulate over restricted space. Another problem of implementation of these methods is how to simulate event time from Poisson process efficiently. Generally, the simulations are based on superposition and thinning theorems. The upper bound of rate function is crucial. The tighter the upper bound, the more efficient the simulation is. In Bayesian analysis for large data sets, if the upper bound is $\mathcal{O}(N^\alpha)$, then the effective sample size per likelihood computation is $\mathcal{O}(N^{-(1/2+\alpha)})$. If $\alpha < 1/2$, then both BPS and GBPS will be more efficient than traditional MCMC methods.

Exploring several simulation settings, we find that reducibility problem just appears in isotropic Gaussian distribution or in distributions who admit isotropic Gaussian distribution as their component for BPS. However, it is still an open question and needs to prove.

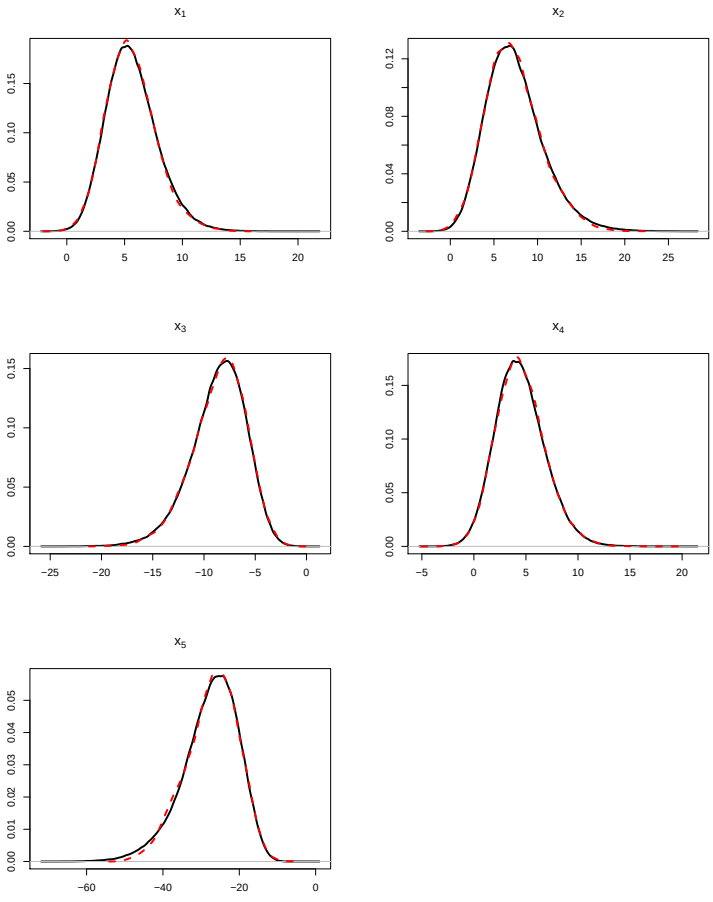

Figure 4.8: The marginal density functions: the black solid lines are marginal density of MH algorithm, which are used as benchmark. The red dotted lines are from a GBPS path.

4.6 Appendix

In this appendix, we prove the ergodicity of GBPS. For simplicity, we introduces the following notations. π denotes our target distribution, Vol denotes the Lebesgue measure over Euclidean space.

$$\mathbf{S}(\mathbf{v}) = \frac{\mathbf{v}}{\|\mathbf{v}\|_2}.$$

$$S^{\perp}(\mathbf{x}, \mathbf{v}) = \{\mathbf{x}' : \langle \mathbf{v}, \nabla \log \pi(\mathbf{x}) \rangle \times \langle \mathbf{S}(\mathbf{x}' - \mathbf{x}), \nabla \log \pi(\mathbf{x}) \rangle < 0\},$$

Assumption 1: For any two points $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^d$ and any velocity $\mathbf{v} \in \mathbb{R}^d, \|\mathbf{v}\|_2 = 1$, there exists $t > 0$, such that

$$\mathbf{x}_2 \in S^{\perp}(\mathbf{x}_1 + t\mathbf{v}, \mathbf{v})$$

Assumption 2: $\text{Vol}\left(\{\mathbf{x} : \nabla \log \pi(\mathbf{x}) = \mathbf{0}\}\right) = 0$.

Remark: Actually, Assumption 2 can be removed without influence on the correctness of Theorem 2 via a similar proof with that in the following one.

Lemma 1: The Markov chain, $\mathbf{z}'_t = (\mathbf{x}_t, \frac{\mathbf{v}_t}{\|\mathbf{v}_t\|_2})$ induced by GBPS, admits $\pi(\mathbf{x}) \times \mathcal{U}(S_{d-1})$ as its invariant distribution.

Lemma 2: For any $\mathbf{x}_0 \in \mathbb{R}^d, \mathbf{v}_0 \in \mathcal{U}(S_{d-1})$, and any open set $W \subset \mathcal{R}^d \times S_{d-1}$, there exists some positive $t > 0$ such that

$$P_t\left((\mathbf{x}_0, \mathbf{v}_0), W\right) > 0$$

Proof 1 (Proof of Lemme 2) *For any open set $W \subset \mathcal{R}^d \times S_{d-1}$, there exist $\mathbf{x}^* \in \mathcal{R}^d$, $r_1 > 0$ and $V \subset S_{d-1}$, such that $B(\mathbf{x}^*, r_1) \times V \subset \mathcal{R}^d \times S_{d-1}$ and $\nabla \log \pi(\mathbf{x}^*) \neq \mathbf{0}$. For $(\mathbf{x}_0, \mathbf{v}_0)$, according to Assumption 1, there exist $t_1 \in (0, \infty)$ and a positive constant δ_1, such that*

$$\langle \mathbf{v}_0, \nabla \log \pi(\mathbf{x}_t) \rangle < 0, \text{ for all } t \in [t_1 - \delta_1, t_1 + \delta_1],$$

$$\mathbf{x}^* \in S^{\perp}(\mathbf{x}_t, \mathbf{v}_0)$$

By a minor transition of t_1, we can suppose that

$$\langle \mathbf{x}^* - (\mathbf{x}_0 + t_1\mathbf{v}_0), \nabla \log \pi(\mathbf{x}^*) \rangle \neq 0$$

By selecting δ_1 and r_1 small enough, we can get:

(i) $\forall \mathbf{x}', \mathbf{x}'' \in B(\mathbf{x}^*, r_1), \forall t \in [t_1 - \delta_1, t_1 + \delta_1]$,

$$\langle \mathbf{x}' - (\mathbf{x}_0 + t\mathbf{v}_0), \nabla \log \pi(\mathbf{x}'') \rangle$$

is always positive or negative;

(ii) for each $t \in [t_1 - \delta_1, t_1 + \delta_1]$,

$$B(\mathbf{x}^*, r_1) \subset S^{\perp}(\mathbf{x}_t, \mathbf{v}_0)$$

(iii)

$$Vol\left(\bigcap_{t\in[t_1-\delta_1,t_1+\delta_1]}\mathbf{S}\left(B\left(\mathbf{x}^*,r_1\right)-\mathbf{x}_t\right)\right)>0$$

(iv)

$$\bigcap_{t\in[t_1-\delta_1,t_1+\delta_1]}\mathbf{S}\left(B\left(\mathbf{x}^*,r_1\right)-\mathbf{x}_t\right)\subset\bigcap_{t\in[t_1-\delta_1,t_1+\delta_1]}S^{\perp}(\mathbf{x}_t,\mathbf{v}_0)$$

As a result, by (iv), there exists a positive constant, $p_1 = p_1(\mathbf{x}_0, \mathbf{v}_0, t_1, \delta_1, \mathbf{x}^*, r_1)$*, such that,* $\forall t \in [t_1 - \delta_1, t_1 + \delta_1]$*,*

$$Q\left(\bigcap_{t'\in[t_1-\delta_1,t_1+\delta_1]}\mathbf{S}\left(B(\mathbf{x}^*,r_1)-\mathbf{x}_{t'}\right)|\mathbf{x}_t,\mathbf{v}_0\right)>p_1$$

Case 1*: If*

$$Vol\left(\bigcap_{t\in[t_1-\delta_1,t_1+\delta_1]}\mathbf{S}\left(B\left(\mathbf{x}^*,r_1\right)-\mathbf{x}_t\right)\bigcap V\right)>0,$$

then there is $t > 0$*, such that*

$$P_t\left((\mathbf{x}_0,\mathbf{v}_0),W\right)>0$$

We illustrate Case 1 in Figure 4.9.

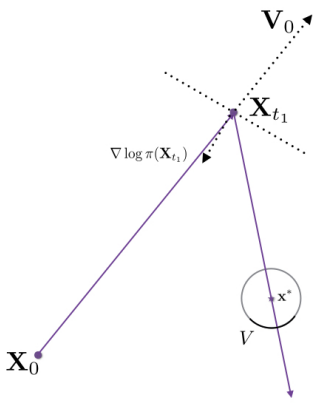

Figure 4.9: Case1

Case 2*: If*

$$Vol\left(\bigcap_{t\in[t_1-\delta_1,t_1+\delta_1]}\mathbf{S}\left(B\left(\mathbf{x}^*,r_1\right)-\mathbf{x}_t\right)\bigcap V\right)=0$$

and

$$\langle\mathbf{S}(\mathbf{x}^*-\mathbf{x}_{t_1}),\nabla\log\pi(\mathbf{x}^*)\rangle<0$$

By (i), we have $\forall \mathbf{x}', \mathbf{x}'' \in B(\mathbf{x}^*, r_1), \forall t \in [t_1 - \delta_1, t_1 + \delta_1]$*,*

$$\langle\mathbf{x}'-(\mathbf{x}_0+t\mathbf{v}_0),\nabla\log\pi(\mathbf{x}'')\rangle<0.$$

Denote

$$\mathbf{M}_1 \quad = \bigcap_{\substack{\mathbf{x}\in B(\mathbf{x}^*,r_1),\mathbf{v}\in\mathbf{S}(\mathbf{x}-\mathbf{x}_t),\\ t\in[t_1-\delta_1,t_1+\delta_1]}} S^{\perp}(\mathbf{x},\mathbf{v})$$

By decreasing δ_1, r_1*, we can have*

$$Vol\left(\mathbf{M}_1 \bigcap \mathbf{V}\right) > 0$$

or

$$Vol\left(\mathbf{M}_1 \bigcap \mathbf{V}\right) = 0, \quad Vol\left(\mathbf{M}_1 \bigcap \mathbf{V}'\right) > 0, \text{ where } \mathbf{V}' = \mathbf{V}$$

Case 2.1*: If*

$$Vol\left(\mathbf{M}_1 \bigcap \mathbf{V}\right) > 0$$

By selecting $r_2 < r_1$*, such that the length of segment of the line across* $\mathbf{x}_t$ *and* $\mathbf{x}' \in B(\mathbf{x}^*, r_2)$ *in the ball* $B(\mathbf{x}^*, r_1)$ *is larger than some positive constant* c_1*. Then event occurs on each line across* $\mathbf{x}_t$ *and* $\mathbf{x}'$ *during the segment in the ball* $B(\mathbf{x}^*, r_1)$ *and the changed velocity traverses* $\mathbf{V}$ *with positive probability which is larger than some positive constant* p_2*. As a result, there exists some* $t > 0$ *such that*

$$P_t\left((\mathbf{x}_0, \mathbf{v}_0), W\right) > 0$$

See Figure 4.10 for illustration.

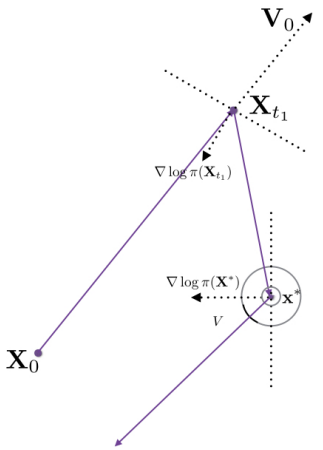

Figure 4.10: Case 2.1

Case 2.2*: If*

$$Vol\left(\mathbf{M}_1 \bigcap \mathbf{V}\right) = 0, \quad Vol\left(\mathbf{M}_1 \bigcap \mathbf{V}'\right) > 0, \text{ where } \mathbf{V}' = -\mathbf{V}$$

With the same treatment as that in Case 2.1, there exists $\mathbf{x}^{**}$ *such that*

$$\mathbf{S}(\mathbf{x}^{**} - \mathbf{x}^*) \in \mathbf{V}', \quad \text{and } \langle S(\mathbf{x}^{**} - \mathbf{x}^*), \nabla \log \pi(\mathbf{x}^{**}) \rangle < 0$$

As a result, there exist two positive constants $r_3 > r_4$, such that, if necessary, decreasing r_1 to be small enough,

$$\mathbf{S}(\mathbf{x}'' - \mathbf{x}') \in \mathbf{V}', \text{ where } \mathbf{x}' \in B(\mathbf{x}^*, r_1), \quad \mathbf{x}'' \in B(\mathbf{x}^{**}, r_3)$$

and

$$Vol\left(\mathbf{M}_1 \bigcap \mathbf{M}_2 \bigcap \mathbf{V}'\right) > 0,$$

where

$$\mathbf{M}_2 = \bigcup_{\substack{\mathbf{x}' \in B(\mathbf{x}^*, r_1), \\ \mathbf{x}'' \in B(\mathbf{x}^{**}, r_4)}} \mathbf{S}(\mathbf{x}'' - \mathbf{x}').$$

Then, the event, that a particle begins from any point $\mathbf{x}' \in B(\mathbf{x}^*, r_1)$ with any velocity $\mathbf{v}' \in \mathbf{S}(B(\mathbf{x}^{**}, r_4) - \mathbf{x}')$, changes velocity in the region $B(\mathbf{x}^{**}, r_4)$ to $-\mathbf{M}_2$ and passes the area $B(\mathbf{x}^*, r_1)$, occurs with positive probability. As a result, there exists $t > 0$ such that

$$P_t\left((\mathbf{x}_0, \mathbf{v}_0), W\right) > 0$$

See Figure 4.11 for illustration.

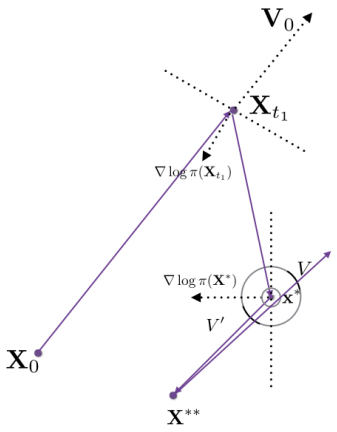

Figure 4.11: Case 2.2

***Case 3**: If*

$$Vol\left(\bigcap_{t \in [t_1 - \delta_1, t_1 + \delta_1]} \mathbf{S}\left(B\left(\mathbf{x}^*, r_1\right) - \mathbf{x}_t\right) \bigcap V\right) = 0$$

and

$$\langle \mathbf{S}(\mathbf{x}^* - \mathbf{x}_{t_1}), \nabla \log \pi(\mathbf{x}^*) \rangle > 0$$

By Assumption 1, there exists $\mathbf{x}^{**}$ on the line which passes $\mathbf{x}_{t_1}$ and $\mathbf{x}^*$, such that

$$\langle \mathbf{S}(\mathbf{x}^* - \mathbf{x}_{t_1}), \nabla \log \pi(\mathbf{x}^{**}) \rangle < 0.$$

Then there exists r_5, such that, if necessary, decreasing δ_1, r_1 to be small enough,

$$Vol\left(\mathbf{M}_3 \bigcap \mathbf{V}\right) > 0$$

or

$$Vol\left(\mathbf{M}_3 \bigcap \mathbf{V}\right) = 0, \text{ and } Vol\left(\mathbf{M}_3 \bigcap \mathbf{V}'\right) > 0, \text{ where } \mathbf{V}' = -\mathbf{V}$$

where

$$\mathbf{M}_3 = \bigcap_{\substack{\mathbf{x}\in B(\mathbf{x}^*, r_1),\\ \mathbf{v}\in \mathbf{S}(\mathbf{x}-B(\mathbf{x}^{**}, r_5))}} S^{\perp}(\mathbf{x}, \mathbf{v})$$

Case 3.1*: If*

$$Vol\left(\mathbf{M}_3 \bigcap \mathbf{V}\right) > 0$$

Then, the event that a particle begins from any point $\mathbf{x}_t, t \in [t_1 - \delta_1, t1 + \delta_1]$ *with velocity* $\mathbf{v}$*, passes the region* $B(\mathbf{x}^*, r_1)$*, changes velocity in the region* $B(\mathbf{x}^{**}, r_5)$ *to* $\mathbf{v}' \in \mathbf{S}(B(\mathbf{x}^*, r_1) - B(\mathbf{x}^{**}, r_5))$*, reaches the area* $B(\mathbf{x}^*, r_1)$ *and changes velocity to* $\mathbf{V}$ *occurs with positive probability. As a result, we obtain the desired result in this case. See Figure 4.12 for illustration.*

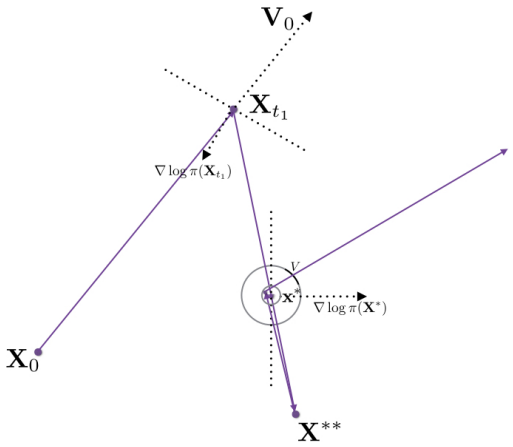

Figure 4.12: Case 3.1

Case 3.2*: If*

$$Vol\left(\mathbf{M}_3 \bigcap \mathbf{V}\right) = 0, \text{ and } Vol\left(\mathbf{M}_3 \bigcap \mathbf{V}'\right) > 0, \text{ where } \mathbf{V}' = -\mathbf{V}$$

By the similar treatment of Case 3.1 and Case 2.2, we can get the admired result. We give an illustration of Case 3.2 in Figure 4.13.

With the help of Lemma 1 and Lemma 2, we can prove the ergodicity of GBPS and the way of proof is similar to that of Theorem 1 of Bouchard-Côté et al. (2018)

Proof 2 (Proof of Theorem 2) *Suppose GBPS is not ergodic, then according to Theorem 7.1 of Hairer (2010), there exist two measures* μ_1 *and* μ_2 *such that* $\mu_1 \perp \mu_2$ *and* μ_1 *and* μ_2 *both are the invariant distribution of GBPS. Thus, there is a measurable set* $A \subset \mathbb{R}^d \times S_{d-1}$ *such that*

$$\mu_1(A) = \mu_2(A^c) = 0$$

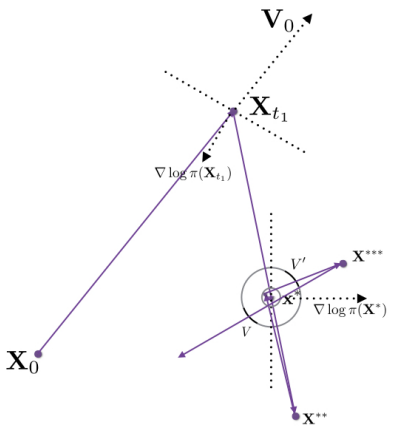

Figure 4.13: Case 3.2

Let $A_1 = A, A_2 = A^c$. By Lemma 2 and Lemma 2.2 of Hairer (2010), the support of μ_i is $\mathbb{R}^d \times S_{d-1}$. For any open set $B \subset \mathbb{R}^d \times S_{d-1}$, $\mu_i(B) > 0$. As a result, at least one of $A_1 \cap B$ or $A_2 \cap B$ has a positive volume by the measure on $\mathbb{R}^d \times S_{d-1}$ which is induced by Lebesgue measure on $\mathbb{R}^d \times \mathbb{R}^d$. Hence, we denote by $i^* \in \{1, 2\}$ an index satisfying $Vol(A_{i^*} \cap B) > 0$. From Lemma 2, we know there exists a z in the interior in $A_{i*} \cap B$ and $t > 0$ such that $P_t(z, A_{i^*}) > 0$. As a result, there exist $r_0 > 0$ and $\delta > 0$ such that $P_t(z', A_{i^*}) > \delta$ for all $z' \in B(z, r_0) \subset A_{i*} \cap B$. Hence,

$$\begin{aligned}
\mu_{i^*}(A_{i^*}) &= \int \mu_{i^*}(dz'')P_t(z'', A_{i^*}) \\
&\geq \int_B \mu_{i^*}(dz'')P_t(z'', A_{i^*}) \\
&\geq \int_{B \cap A_{i^*}} \mu_{i^*}(dz'')\delta \\
&\geq \delta \mu_{i^*}(B \cap A_{i^*}) > 0
\end{aligned}$$

This contradicts that $\mu_i(A_i) = 0$ for $i \in \{1, 2\}$. As a result, GBPS has as most one invariant measure. By Lemma 1, we complete the proof.

Bibliography

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.

Bierkens, J. (2016). Non-reversible Metropolis-Hastings. *Statistics and Computing*, 26(6):1213–1228.

Bierkens, J., Bouchard-Côté, A., Doucet, A., Duncan, A. B., Fearnhead, P., Lienart, T., Roberts, G., and Vollmer, S. J. (2018). Piecewise deterministic Markov processes for scalable Monte Carlo on restricted domains. *Statistics & Probability Letters*, 136:148–154.

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*.

Bierkens, J., Roberts, G., et al. (2017). A piecewise deterministic scaling limit of lifted Metropolis–Hastings in the Curie–Weiss model. *The Annals of Applied Probability*, 27(2):846–882.

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.

Chen, T.-L. and Hwang, C.-R. (2013). Accelerating reversible Markov chains. *Statistics & Probability Letters*, 83(9):1956–1962.

Davis, M. H. (1984). Piecewise-deterministic Markov processes: A general class of non-diffusion stochastic models. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 353–388.

Davis, M. H. (1993). *Markov Models & Optimization*, volume 49. CRC Press.

Fearnhead, P., Bierkens, J., Pollock, M., and Roberts, G. O. (2016). Piecewise Deterministic Markov Processes for Continuous-Time Monte Carlo. *arXiv preprint arXiv:1611.07873*.

Hairer, M. (2010). Convergence of Markov processes. *http://www.hairer.org/notes/Convergence.pdf*.

Hastings, W. K. (1970). Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109.

Hwang, C.-R., Hwang-Ma, S.-Y., and Sheu, S.-J. (1993). Accelerating gaussian diffusions. *The Annals of Applied Probability*, pages 897–913.

Kingman, J. F. C. (1993). *Poisson processes*. Wiley Online Library.

Lewis, P. A. and Shedler, G. S. (1979). Simulation of nonhomogeneous Poisson processes by thinning. *Naval research logistics quarterly*, 26(3):403–413.

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.

Minsker, S., Srivastava, S., Lin, L., and Dunson, D. B. (2017). Robust and scalable Bayes via a median of subset posterior measures. *The Journal of Machine Learning Research*, 18(1):4488–4527.

Neiswanger, W., Wang, C., and Xing, E. (2013). Asymptotically exact, embarrassingly parallel MCMC. *arXiv preprint arXiv:1311.4780*.

Peters, E. A. et al. (2012). Rejection-free Monte Carlo sampling for general potentials. *Physical Review E*, 85(2):026703.

Quiroz, M., Kohn, R., Villani, M., and Tran, M.-N. (2018). Speeding up MCMC by efficient data subsampling. *Journal of the American Statistical Association*, (2018):1–35.

Scott, S. L., Blocker, A. W., Bonassi, F. V., Chipman, H. A., George, E. I., and McCulloch, R. E. (2016). Bayes and big data: The consensus Monte Carlo algorithm. *International Journal of Management Science and Engineering Management*, 11(2):78–88.

Sun, Y., Schmidhuber, J., and Gomez, F. J. (2010). Improving the asymptotic performance of Markov chain Monte-Carlo by inserting vortices. In *Advances in Neural Information Processing Systems*, pages 2235–2243.

Wang, X. and Dunson, D. B. (2013). Parallelizing MCMC via weierstrass sampler. *arXiv preprint arXiv:1312.4605*.

Parallelizing MCMC via Random Forest

Joint work with Christian P. ROBERT

5.1 Introduction

Markov chain Monte Carlo (MCMC) algorithm, a generic sampling method, is ubiquitous in modern statistics, especially in Bayesian fields. MCMC algorithms only require to evaluate the target pointwisely up to a multiple constant in order to asymptotically sample from it. For instance, in Bayesian analysis, the main object of interest is the posterior distribution, which most often does not have a closed form, and MCMC has become a standard workhorse in this field. However, MCMC is quite delicate to scale and its applications are limited when the observation size gets very large, for it need sweep over the entire observation set (or sample) in order to evaluate the likelihood function at each iteration. Recently, several methods have been proposed to scale MCMC algorithm against so-called "big data" and they can be roughly classified into two groups (Bardenet et al., 2017): divide-and-conquer methods and subsampling-based methods.

Divide-and-conquer methods split the data set into subsets or batches, runs MCMC over each subset to generate parameter samples, and combines them to produce an approximation of the posterior distribution. Depending on how an MCMC algorithm is applied to these subsets, the methods can be further classified into two sub-categories. First, let $\mathcal{X}$ be the whole data set and $\mathcal{X}_1, \cdots, \mathcal{X}_K$ be the subsets. Denote π_0 as the prior distribution over the parameter θ. Then, one category of proposal (Neiswanger et al., 2013; Nemeth et al., 2017; Scott et al., 2016; Wang and Dunson, 2013; Wang et al., 2015) is to run MCMC over

$$\pi_k(\theta|\mathcal{X}_k) \propto (\pi_0(\theta))^{1/K} \prod_{x \in \mathcal{X}_k} p(x|\theta).$$

A second category (Minsker et al., 2014; Srivastava et al., 2015) instead targets

$$\pi_k(\theta|\mathcal{X}_k) \propto \pi_0(\theta) \left(\prod_{x \in \mathcal{X}_k} p(x|\theta) \right)^K.$$

Subsampling-based methods use a partition of the whole data set to estimate the MH acceptance ratio at each iteration, towards accelerating the resulting MCMC

algorithms. These techniques can also be sub-dvivided into two finer classes: exact versus approximate subsampling methods, depending on their outputs. Exact subsampling approaches typically require to explore an augmented space, treating the target distribution as its invariant marginal distribution. One direction (Quiroz et al., 2016) is to take advantage of pseudo-marginal MCMC (Andrieu and Roberts, 2009) via the construction of unbiased estimators of the target density based on subsets of the data. Another one is to resort to piecewise deterministic Markov processes (Bierkens et al., 2018, 2016; Bouchard-Côté et al., 2018; Davis, 1984, 1993; Fearnhead et al., 2016; Sherlock and Thiery, 2017; Vanetti et al., 2017), which take the targets as the marginal of their corresponding invariant distributions. Approximate subsampling approaches aim at constructing an approximation of the target distributions. One approach (Bardenet et al., 2014, 2017) is to determine the acceptance of proposals with high probability via subsets of the data. Another approach Chen et al. (2014); Ding et al. (2014); Welling and Teh (2011) is based on direct modifications of exact methods. The seminal work in this direction is stochastic gradient Langevin dynamics (SGLD) (Welling and Teh, 2011).

In this chapter, we propose two new methods to scale MCMC algorithms, based on the divide-and-conquer methodology. Unlike former divide-and-conquer solutions, we run MCMC over a weighted version of the partial posteriors

$$\pi_k(\theta|\mathcal{X}_k) \propto \left\{ (\pi_0(\theta))^{1/K} \prod_{x\in\mathcal{X}_k} p(x|\theta) \right\}^{\lambda},$$

where λ is not necessary restricted to the values 1 or K. In addition, we use random forests (Breiman, 2001) to learn fast approximations of these subposteriors and we take advantage of the solutions to deduce an approximation of the true posterior though an additional MCMC step or by importance sampling. Section 2 describes the proposed methods in details and several numerical examples are presented in Section 3. Section 4 discusses the limitations of these methods and concludes the paper.

5.2 Methodology

Denote by $\mathcal{X} = \{x_1, x_2, \cdots, x_N\}$ the whole set of observations, where $x_i \sim p_\theta(\cdot)$ i.i.d. and $\theta \in \Theta \subset \mathbb{R}^d$. Let $\pi_0(\theta)$ denote the prior distribution on Θ. Splitting the data set $\mathcal{X}$ into subsets $\mathcal{X}_1, \cdots, \mathcal{X}_K$, each with the same size $m = \frac{N}{K}$. In Bayesian analysis, the target of interest is the posterior distribution:

$$\pi(\theta|\mathcal{X}) \propto \pi_0(\theta) \prod_{i=1}^{N} p(x_i|\theta)$$

For each subset $\mathcal{X}_k$, $k = 1, \cdots, K$, we define the associated λ_k-subposterior as

$$\pi_k^{\lambda_k}(\theta|\mathcal{X}_k) = \frac{(\gamma_k(\theta|\mathcal{X}_k))^{\lambda_k}}{Z_{k,\lambda_k}}, \quad \gamma_k(\theta|\mathcal{X}_k) = \pi_0(\theta)^{\frac{1}{K}} \prod_{x\in\mathcal{X}_k} p(x|\theta),$$

where Z_{k,λ_k} is the normalizing constant of $(\gamma_k(\theta|\mathcal{X}_k))^{\lambda_k}$.

In divide-and-conquer strategies, one applies MCMC algorithms to each subposterior, thus generating samples from these distributions, and combines these samples to approximate the true posterior distribution. Existing divide-and-conquer MCMC methods treats the powers λ_k in two possible ways: one is to set $\lambda_k = 1$ for all $k = 1, \cdots, K$, as in consensus Monte Carlo and Weierstrass samplers; the other is to set $\lambda_k = K$, as in WASP (Srivastava et al., 2015) and M-posterior (Minsker et al., 2014). But there is no reason to adopt this restriction. Besides, the above mentioned methods, with the exception of Nemeth et al. (2017), ignore a valuable byproduct—namely, the value of the target distribution computed at proposed sample points—of running these MCMC algorithms. When compared with Nemeth et al. (2017), our methods differ in two ways: a minor one is that we use the scaled subposterior and the other one is that the learning algorithm we implement is cheap at both training and prediction stages.

As mentioned above, the core idea in our approach is to embed a machine-learning regression procedure within the divide-and-conquer methodology towards constucting approximations of all subposterior density functions and derive from these approximations of the true posterior. Specifically, we run some MCMC algorithms, such as random walk Metropolis–Hastings, HMC, MALA, or other versions, targetting $\pi_k^{\lambda_k}(\theta|\mathcal{X}_k)$ to obtain MCMC samples $\{\theta_1^k, \theta_2^k, \cdots, \theta_T^k\}$ that are (approximately) from these targets. In the Metropolis–Hastings acceptance ratio, we therefore evaluate $\log \gamma_k(\theta|\mathcal{X}_k)$ for each proposal, rather than the more challenging $\pi_k^{\lambda_k}(\theta|\mathcal{X}_k)$, bypassing normalizing constant derivation and subsequent numerical problems. As a byproduct, we thus obtain evaluations of $\log \gamma_k(\theta|\mathcal{X}_k)$ at some parameter values $\{\vartheta_1^k, \vartheta_1^k, \cdots, \vartheta_{T_k}^k\}$, which are the quantities proposed by the MCMC algorithms. Thus, calling in a random forests or another regression machine-learning algorithm on the learning set

$$\left\{ \left(\vartheta_1^k, \log \gamma_k(\vartheta_1^k|\mathcal{X}_k) \right), \left(\vartheta_2^k, \log \gamma_k(\vartheta_2^k|\mathcal{X}_k) \right), \cdots, \left(\vartheta_{T_k}^k, \log \gamma_k(\vartheta_{T_k}^k|\mathcal{X}_k) \right) \right\}$$

provides an estimator, f_k, of the value of the unnormalised partial log-likelihoods, $\log\{\gamma_k(\theta|\mathcal{X}_k)\}$. These estimators f_k, $k = 1, \cdots, K$, can subsequently been exploited in two way. First, they provide an approximation of $\pi(\theta|\mathcal{X})$ by

$$f(\theta) = \exp\left\{ \sum_{k=1}^{K} f_k(\theta) \right\}.$$

This approximation can then be used as a substitute to run a new MCMC algorithm targetting f, so as to derive a parameter sample that is approximately generated from the posterior distribution. Second, the approximation of $\pi(\theta|\mathcal{X})$ by $f(\theta)$ can also be recycled into an importance sampling step using simulations from $\gamma_k^{\lambda_k}(\theta|\mathcal{X}_k)$ and target f.

Remark 1: (Scale factors) The scale factor λ_k is used to control the additional uncertainty resulting from using subposteriors. Informaly, this term controls the range of the region on which the embedded regression algorithm will learn each subposterior. When the scale factor gets too large, for instance, when $\lambda_k = 100K$, $k = 1, \cdots, K$, each scaled subposterior is endowed with little uncertainty, which may lead to their respective MCMC samples not to overlap. When this situation occurs, the approximations f_k cannot provide information about the regions where $\gamma_j(\theta|\mathcal{X}_j)$

is high for $j \neq k$ and as a result neither the additional MCMC method nor the importance sampling version work. In the opposite situation, namely, when the scale factor is too small, for example $\lambda_k = 0.001$, $k = 1, \cdots, K$, the approach requires more pairs of parameters and corresponding subposteriors to be able to train a good approximation, even though when some or all subposteriors are overlapping. In cases where we can obtain adequate and cheap approximations of the means and covariances of both the true posterior and the subposteriors (where $\lambda_k = 1, k = 1, \cdots, K$), we can choose λ_k such that a sufficiently high posterior probability region is charged by the subposterior π_k. Specifically, we proceed as follows in the one-dimensional case ($d = 1$): take $\hat{\theta}$, $\hat{\theta}_k$, $\hat{\sigma}$ and $\hat{\sigma}_k$ as the estimates of the means and standard deviations of the true posterior and the subposterior π_k, respectively. We denote

$$\delta_k = \max\{|\hat{\theta}_k - \hat{\theta} - 2\hat{\sigma}|, |\hat{\theta}_k - \hat{\theta} + 2\hat{\sigma}|\}$$

and we choose $\lambda_k = (\delta_k/\hat{\sigma}_k)^{-2}$. By Markov inequality, $\pi_k^{\lambda_k}$ covers the region which has large probability under the true posterior. In multi-dimension cases, i.e., when $d > 1$, we select the scale factors as the minimal ones accross marginal components. If such cheap estimations are not available, we instead choose the scale factors by running our method several times over pre-defined scale factors, like $\lambda_k = 0.1, 0.5, 1, 2, 5, 10$ and choose the ones that bring the subposteriors to overlap.

Remark 2: (Regression algorithms) Considering random forests enjoy the appealing features of an easy implementation, a strong learning ability of non-linear relations, and robustness, we make the choice of applying this method to build our random forest, we apply it to learn approximations of the subposterior density functions. It goes without saying that other regression algorithms are available and potential competitors to random forests. However, when compared with random forest, these other machine learning techniques, like support vector machines and neural networks, require extra hyper-parameters to be tuned. Besides, the prediction abilities of random forests are scalable, that is, given a training set of size T, the cost of predicting the output of a new input is of order $\mathcal{O}(\log(T))$. When we call these approximations of the subposteriors f_k to run an additional MCMC, the evaluation of a new proposal comes at a cost of $\mathcal{O}(K \log(T))$, which corresponds to a cost of $\mathcal{O}(KT)$ for memory-based algorithms, such as local linear or polynomial regression, nonparametric kernel density estimation, and Gaussian processes.

Remark 3: (Combination of importance sampling steps) For our second method, we use importance sampling over each subposterior and combine the resulting samples to approximate the true posterior, with no need for an additional MCMC run. However, considering that the subposteriors are spread farther than the true posterior, a large portion of these samples have extremely low weights and need be discarded to lead to a better approximation of the true posterior. More specifically, suppose $\left\{\theta_1^k, \cdots, \theta_T^k\right\}$ are the samples from $\pi_k^{\lambda_k}$, with weights ($t = 1, \cdots, T$)

$$w_t^k \propto \exp\left\{\sum_{j=1}^{K} f_j(\theta_t^k) - \lambda_k f_k(\theta_t^k)\right\}$$

Let σ be a permutation of $\{1, \cdots, T\}$ such that $w_{\sigma(1)}^k \geq w_{\sigma(2)}^k \geq \cdots \geq w_{\sigma(T)}^k$. Having selected a truncation probability p, we then choose $i_k = \min\{i : \sum_{\ell=1}^{i} w_{\sigma(\ell)}^k \geq p\}$ and consider instead $\tilde{\pi}_k = \sum_{\ell=1}^{i_k} \tilde{w}_\ell^k \delta_{\theta_{\sigma\ell}^k}$, where $\tilde{w}_\ell^k = w_\ell^k / \sum_{r=1}^{i_k} w_r^k$. By default, we

choose p within $0.99, 0.999, 0.9999$. Considering that the i_ks may differ a lot across samples by virtue of mere by stochasticity, some $\tilde{\pi}_k$ will enjoy more or even much more atoms than othes. We thus weight the approximations $\tilde{\pi}_k$ proportional to their effective sample sizes (ESS) (Liu, 2008), that is, we approximate the true posterior by

$$\hat{\pi} \propto \sum_{k=1}^{K} ESS_k \tilde{\pi}_k, \qquad ESS_k = i_k / \{1 + \mathbb{V}_k\}$$

where $\mathbb{V}_k$ is the variance of $\{i_k w^k_{\sigma(1)}, \cdots, i_k w^k_{\sigma(i_k)}\}$.

Remark 4: (Computation complexity) The computing budget of our approach is made of three components

- At the divide-and-conquer stage, the computating cost is $\mathcal{O}(T_k N/K)$ on each subsample and we generate a total of T samples points, where T may differ from T_k according to the techniques of burn-in and thinning of MCMC.
- At the regression training stage, the cost of each random forest is $\mathcal{O}(T_k \log T_k)$.
- At the combination stage
    1. The cost of an additional MCMC is is $\mathcal{O}(K \log T_k)$ per proposed point, while
    2. the cost of importance sampling is $\mathcal{O}(KT \log T_k)$ for weighting all samples over all subposteriors.

5.3 Numerical Experiments

For simplicity's sake, we now call our first proposal RF-MH and our second proposal RF-IS. In this section, we apply our second method to several examples, from Gaussian posterior to strongly non-Gaussian posteriors and to a model misspecification case. We compare our method with consensus Monte Carlo (CMC) (Scott et al., 2016), nonparametric KDE (Nonpara) (Neiswanger et al., 2013), and Weierstrass sampler (Weierstrass) (Wang and Dunson, 2013). In practice, apart from toy examples, we could not implement the RF-MH version due to its extremely intense cost at the stage of the additional MCMC run, since we need to call the trained random forests at each MCMC iteration and since we could not keep them active in permanence in the computer memory. Each such recall is quite expensive, even though the theoretic cost of RF-MH is the same as the one of RF-IS.

5.3.1 A Bimodal Posterior

In this example, taken from Welling and Teh (2011), for a parameter $\theta = (\theta_1, \theta_2)$, the model is

$$X \sim \frac{1}{2}\mathcal{N}(\theta_1, 2) + \frac{1}{2}\mathcal{N}(\theta_1 + \theta_2, 2),$$

under a flat prior $\pi_0(\theta_1, \theta_2) \propto 1$. We draw $n = 200$ observations $\{X_1, X_2, \cdots, X_{200}\}$ from the model with $\theta_1 = 0, \theta_2 = 1$. In order to implement our divide-and-conquer

algorithm, we consider the cases $K = 10$ and 20, under two values of the parameter, $(0, 1)$ and $(1, -1)$. When the sample size is moderate, both configurations can be detected and the posterior is bimodal. While this size is larger, the simulated Markov chains are often stuck at one mode, an issue that turns the posterior into a unimodal distribution.

A first experiment aims at showing that our method can handle non-Gaussian posterior distributions, in which we choose a moderate sample size $n = 200$. We generate $500,000$ parameter values for each subposterior, discard the first $100,000$ points and thin the rest at a rate of one selection per 100 sample points. In this case, we set $\lambda_k = 1, k = 1, \cdots, K$ and train each random forest with 10 random partition trees on random subsets of those sample sets, of size $50,000$. In a second experiment, we generate $n = 10,000$ observations from the model, and set $K = 50$.

Figure 1 compares the contours of the pdf's of the samples from each method for $K = 10$ over $n = 200$ observations. It is clear that consensus Monte Carlo cannot capture the bimodal feature of the posterior, while a nonparametric kernel density estimation (KDE) gives very unequal weights on the two modes, while our method and Weierstrass sampler perform quite satisfactorily. Figure 2 corresponds to $K = 20$ and $n = 200$ observations. There, in contrast with our method, the other methods under examination deviate from the true posteriors. In Table 1, we compare the time budgets of all methods. In this example, the MCMC stage dominates in the computation cost, while all but the nonparametric KDE have the same order of time requirement. Since the KDE is memory-based, its prediction cost is of order $\mathcal{O}(T^2)$, while our method exhibits scalable prediction costs. Figure 3 compares the performances of all method for $n = 10,000$ observations for $K = 50$. In this case, we train each random forest over $100,000$ sampled parameters, with 20 random partition trees. Since most MCMC algorithms will be stuck at one mode of the posterior, depending on its starting point, we use importance sampling to generate the posterior distribution. It is clear from this figure that only our method performs well, Weierstrass sampler giving unequal weights for two modes, consensus Monte Carlo failing irremediably, and KDE similarly stuck at one mode of the posterior.

K	RF-IS	CMC	Nonpara	Weierstrass
$K = 10$	MCMC 54.8	MCMC 54.8	MCMC 54.8	MCMC 54.8
	Training 2.8			
	Weighting 0.3	Combination 0.1	Combination 109.3	Combination 0.8
	Total 57.9	Total 54.9	Total 164.1	Total 55.6
$K = 20$	MCMC 52.8	MCMC 52.8	MCMC 52.8	MCMC 52.8
	Training 2.8			
	Weighting 0.6	Combination 0.2	Combination 403.2	Combination 1.6
	Total 56.2	Total 53.0	Total 456.0	Total 54.4

Table 5.1: Time requirements of the approaches used in Example 1 (in seconds).

5.3.2 A Moon-shaped Posterior

This example uses a toy model that is unidentifiable in some parameters. We denote the parameters as $\boldsymbol{\theta} = (\theta_1, \theta_2) \in [0, \infty)^2$, while the sample, $X_1, \cdots, X_N$, is generated from

$$X \sim \mathcal{N}(\sqrt{\theta_1} + \sqrt{\theta_2}, 2), \quad \theta_1 \geq 0, \theta_2 \geq 0$$

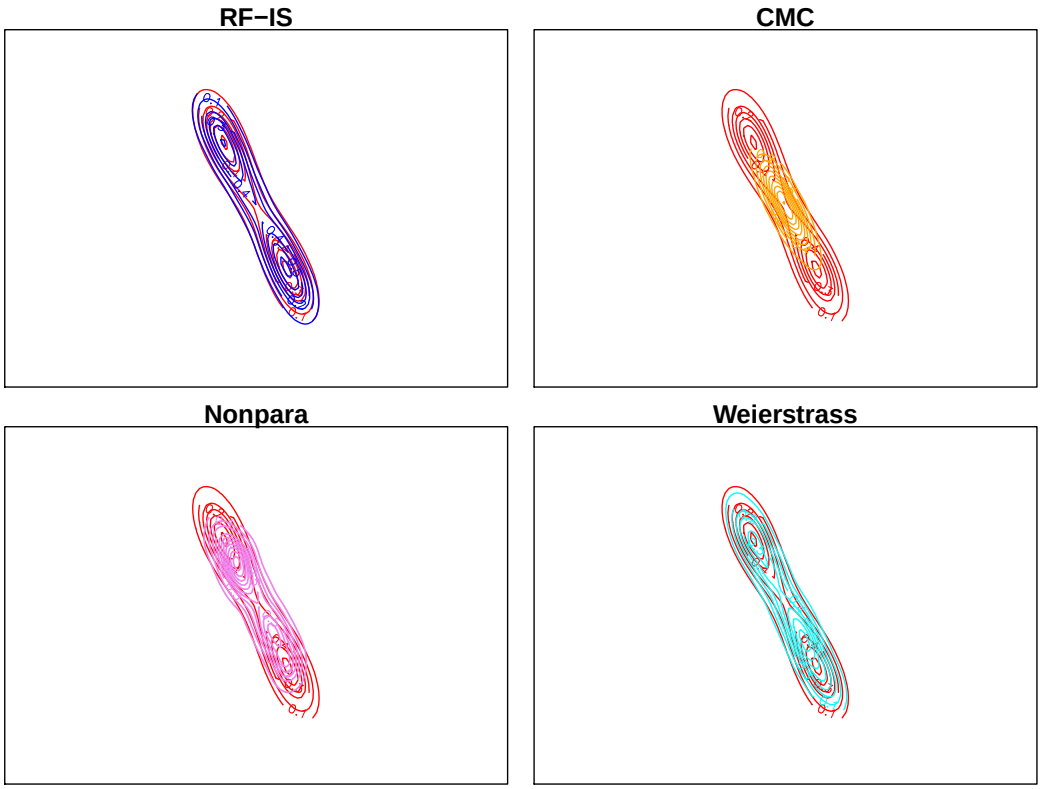

Figure 5.1: Example 1: Comparison of the contours of the true posterior (red), RF-IS (blue), consensus Monte Carlo (orange), KDE (violet) and Weierstrass sampler (cyan) for $K = 10$ subsamples.

In this model, due to non-identifiability, the posterior is moon-shaped and thus strongly non-Gaussian. Our experiment consists in generating observations from a standard normal distribution, $\mathcal{N}(0, 1)$, with $N = 1000$, $K = 10, 20$, $\lambda_k = 1$, $k = 1, \cdots, K$, and draw $500,000$ parameters from each subposterior, exclude the first 10^5 simulations, and thin the rest at the same rate of one per 100 simulations. Given this larger number of proposal points for each subposterior, we now train each random forest with 10 random partition trees on random subsets of the parameter subsamples set of size $50,000$. Figure 4 compares the performances of each method for $K = 10$. In this strongly non-Gaussian case, consensus Monte Carlo and nonparametric KDE produce Gaussian posteriors. To some extent, Weierstrass sampler keeps its ability to learn a non-Gaussian posterior, and our method has the best performances of all. In Figure 5, corresponding to $K = 20$, the Weierstrass performances degrade while our method still approximate the true target closely. Table 2 details the time requirements for each method. We find that KDE is the most costly, with the cost of a new prediction being $\mathcal{O}(T^2)$. Since the MCMC step over subsets dominates the other steps, our methods has an order of time requirement that compares with consensus Monte Carlo and Weierstrass samplers.

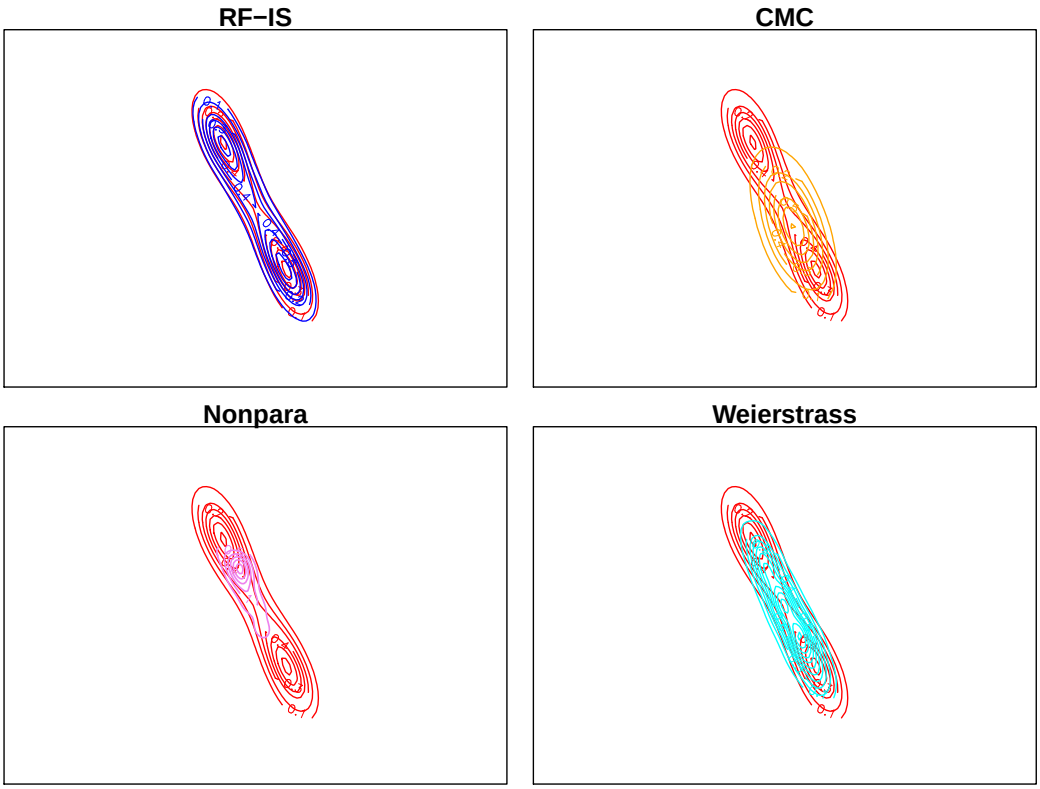

Figure 5.2: Example 1: Comparison of the contours of the true posterior (red), RF-IS (blue), consensus Monte Carlo (orange), KDE (violet) and Weierstrass sampler (cyan) for $K = 20$ subsamples.

K	RF-IS	CMC	Nonpara	Weierstrass
$K = 10$	MCMC 67.6	MCMC 67.6	MCMC 67.6	MCMC 67.6
	Training 2.8			
	Weighting 0.3	Combination 0.1	Combination 101.7	Combination 0.7
	Total 70.7	Total 67.7	Total 169.3	Total 68.3
$K = 20$	MCMC 58.7	MCMC 58.7	MCMC 58.7	MCMC 58.7
	Training 2.7			
	Weighting 0.7	Combination 0.2	Combination 371.5	Combination 1.5
	Total 62.1	Total 58.9	Total 430.2	Total 60.2

Table 5.2: Time requirements of the methods of Example 2 (in seconds).

5.3.3 A Misspecification Example

This example is taken from (Bardenet et al., 2017) in order to show that with a suitable selection of the scale factors, λ_k, our methods are robust to a misspecification of the model. In this setting, the model is

$$X \sim \mathcal{N}(\mu, \sigma^2),$$

the parameter is $\boldsymbol{\theta} = (\mu, \sigma^2)$. We generate two data sets, each having $N = 10,000$ points $X_1, \cdots, X_N$, from a log-normal distribution, $\mathcal{LN}(0, 1)$ and from a standard normal distribution, $\mathcal{N}(0, 1)$, respectively, and use a flat prior, $p(\mu, \sigma^2) \propto 1$. For $K = 10$, we determine the λ_k's according to the method drafted in Remark 1.

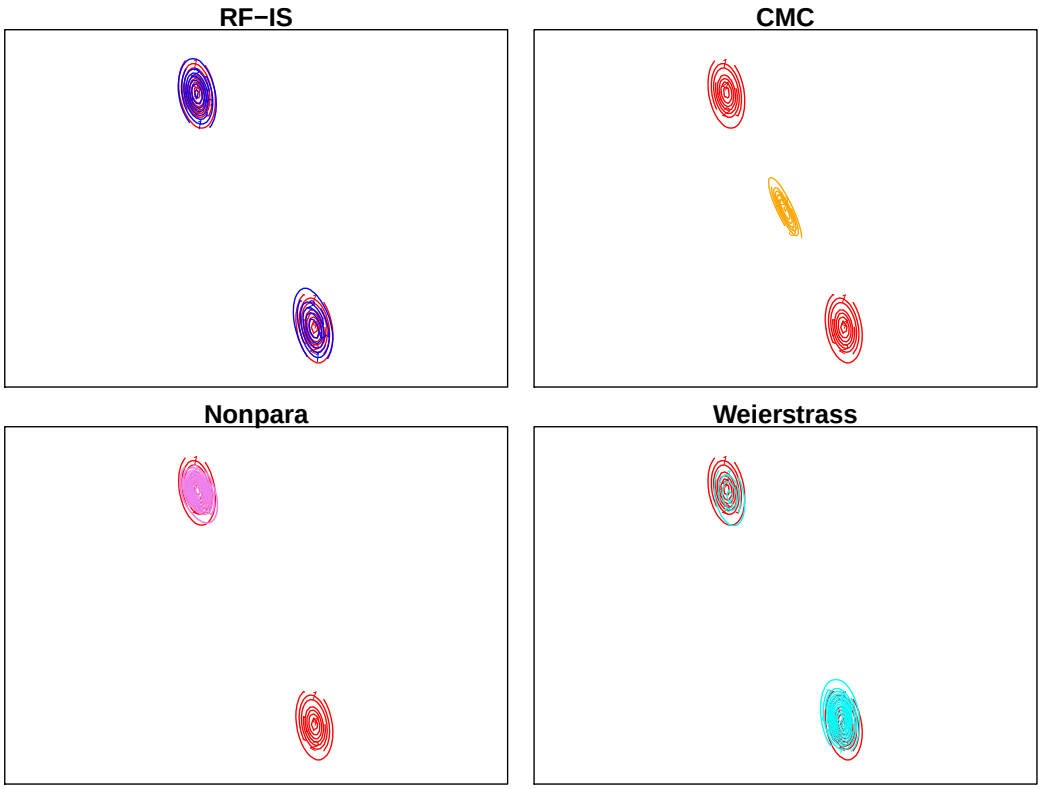

Figure 5.3: Example 1: Comparison of the contours of the true posterior (red), RF-IS (blue), consensus Monte Carlo (orange), KDE (violet) and Weierstrass sampler (cyan) for $K = 50$ subsamples of 10^4 observations.

Specifically, the maximum likelihood estimators over the entire data set are

$$\hat{\boldsymbol{\theta}} = \left(\frac{\sum_{i=1}^{N} x_i}{N}, \frac{\sum_{i=1}^{N} (x_i - \bar{x})^2}{N} \right)$$

$$\hat{\boldsymbol{\sigma}}_{\boldsymbol{\theta}} = \left(\sqrt{\frac{\hat{\theta}_2}{N}}, \sqrt{\frac{2\hat{\theta}_2^2}{N}} \right)$$

For each subset $\mathcal{X}_k$, we obtain $\hat{\boldsymbol{\theta}}^{(k)}$ and $\hat{\boldsymbol{\sigma}}_{\boldsymbol{\theta}^{(k)}}$ similarly. Set

$$\lambda_{k,1} = \left(\frac{\max\{|\hat{\theta}_1^{(k)} - \hat{\theta}_1 - 2\hat{\sigma}_{\theta_1}|, |\hat{\theta}_1^{(k)} - \hat{\theta}_1 + 2\hat{\sigma}_{\theta_1}|\}}{\hat{\sigma}_{\theta_1^{(k)}}} \right)^{-2}$$

$$\lambda_{k,2} = \left(\frac{\max\{|\hat{\theta}_2^{(k)} - \hat{\theta}_2 - 2\hat{\sigma}_{\theta_2}|, |\hat{\theta}_2^{(k)} - \hat{\theta}_2 + 2\hat{\sigma}_{\theta_2}|\}}{\hat{\sigma}_{\theta_2^{(k)}}} \right)^{-2}$$

$$\lambda_k = \min\{\lambda_{k,1}, \lambda_{k,2}\}$$

As above, we generate $500,000$ parameter values from each subposterior, exclude the first 10^5 values and thin out the rest at a rate of one per 100 points in the Gaussian case and one per 10 points in the Log-Normal case. Each random forest is trained with 20 random partition trees over 50,000 training samples. Figure 6 and Figure 7 present the performances of each method under the normal and log-normal distributions, respectively. In each case, results for KDE are far away

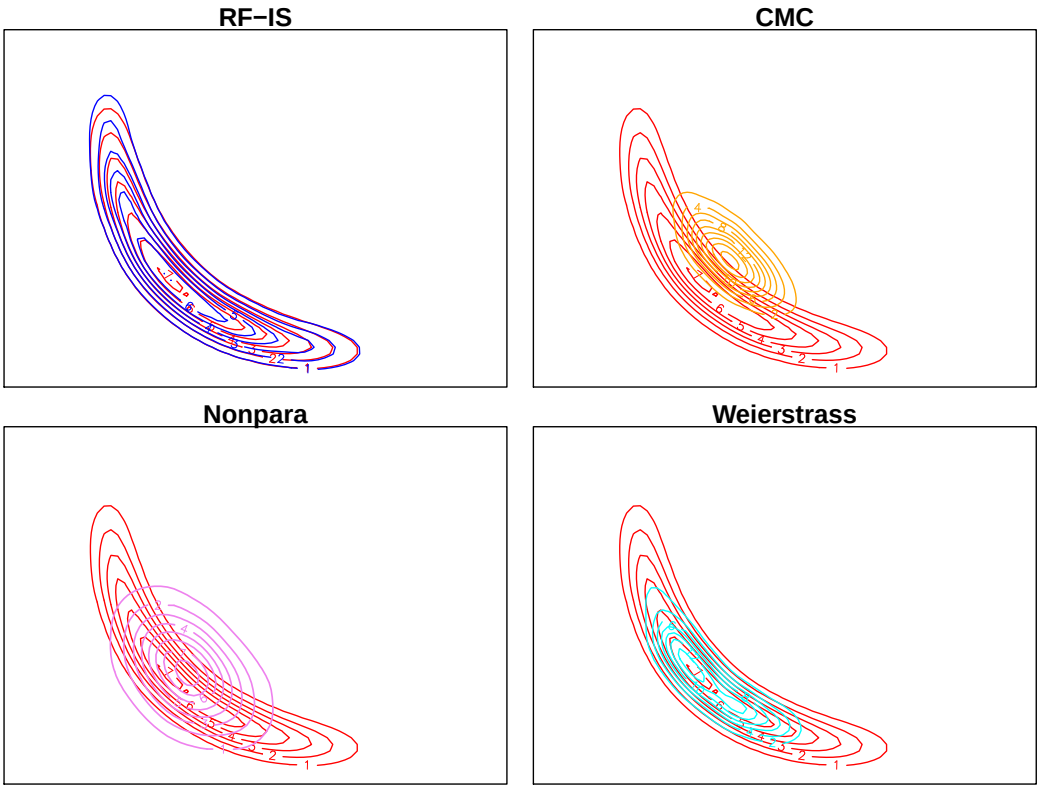

Figure 5.4: Example 2: Comparison of the contours of true posterior (red), RF-IS (blue), consensus Monte Carlo (orange), KDE (violet) and Weierstrass sampler (cyan) for $K = 10$ subsamples.

from the true posterior, as some subposteriors do not cover the high probability region of the true posterior. In the normal case, RF-IS, consensus Monte Carlo and Weierstrass sampler all perform very well. However, only our method is robust to model misspecification, as the others deviate to some extend from the true posteriors in the log-normal case. In Table 3, we compare the time requirements for all methods where again KDE is the most costly approach in log-normal case.

Model	RF-IS	CMC	Nonpara	Weierstrass
$\mathcal{N}(0,1)$	MCMC 63.9 Training 6.1 Weighting 0.6	MCMC 65.2 Combination 0.1	MCMC 65.2 Combination 99.6	MCMC 65.2 Combination 0.8
	Total 70.6	Total 65.3	Total 164.8	Total 66.0
$\mathcal{LN}(0,1)$	MCMC 67.4 Training 6.1 Weighting 4.4	MCMC 69.2 Combination 0.9	MCMC 69.2 Combination 1114.4	MCMC 69.2 Combination 7.5
	Total 77.9	Total 70.1	Total 1183.6	Total 76.7

Table 5.3: Time requirements of Example 3 (in seconds).

5.3.4 Logistic Model

In this benchmark example of a Bayesian logistic model, the data set *covtype.binary*[1] is analysed. It contains 54 attributes, of which the first ten are quantitative, and the

[1]available at: `https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html`

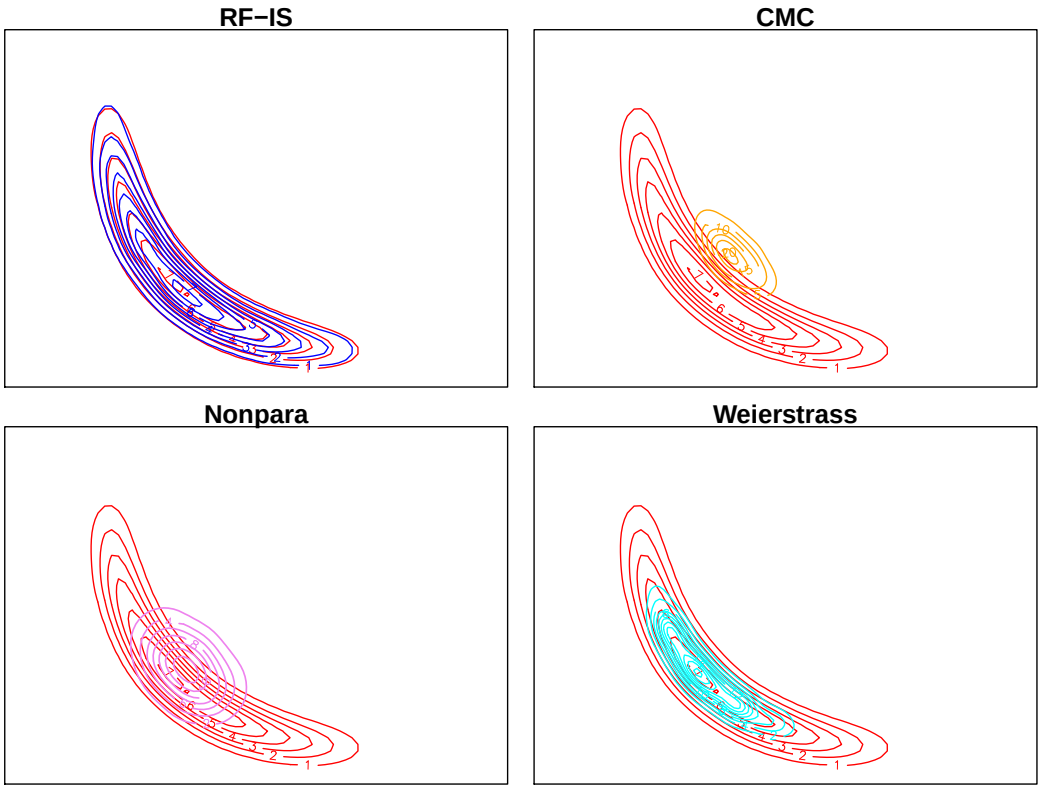

Figure 5.5: Example 2: Comparison of the contours of true posterior (red), RF-IS (blue), consensus Monte Carlo (orange), KDE (violet) and Weierstrass sampler (cyan) for $K = 20$ subsamples.

others are binary. In this final experiment, we find our method loses its appeal and efficiency, due to the increase in the dimension of the parameter. When picking the first ten attributes, the parameter space is of dimension 11 (including the intercept). For $K = 50$, Figure 8 shows the mean squared error of the mean associated with our method, consensus Monte Carlo and Weierstrass samplers. In this case, our method perform worst of all since, in a larger dimension setting, random forests suffer from a curse of dimensionality. Besides, in this logistic setting, where all subposteriors are nearly Gaussian, consensus Monte Carlo shows its appeal. Since the Weierstrass sampler is an refinement of consensus Monte CArlo via a KDE, it also suffers from a curse of dimensionality, which is due to the general poor adequation of KDE's in high dimensional cases.

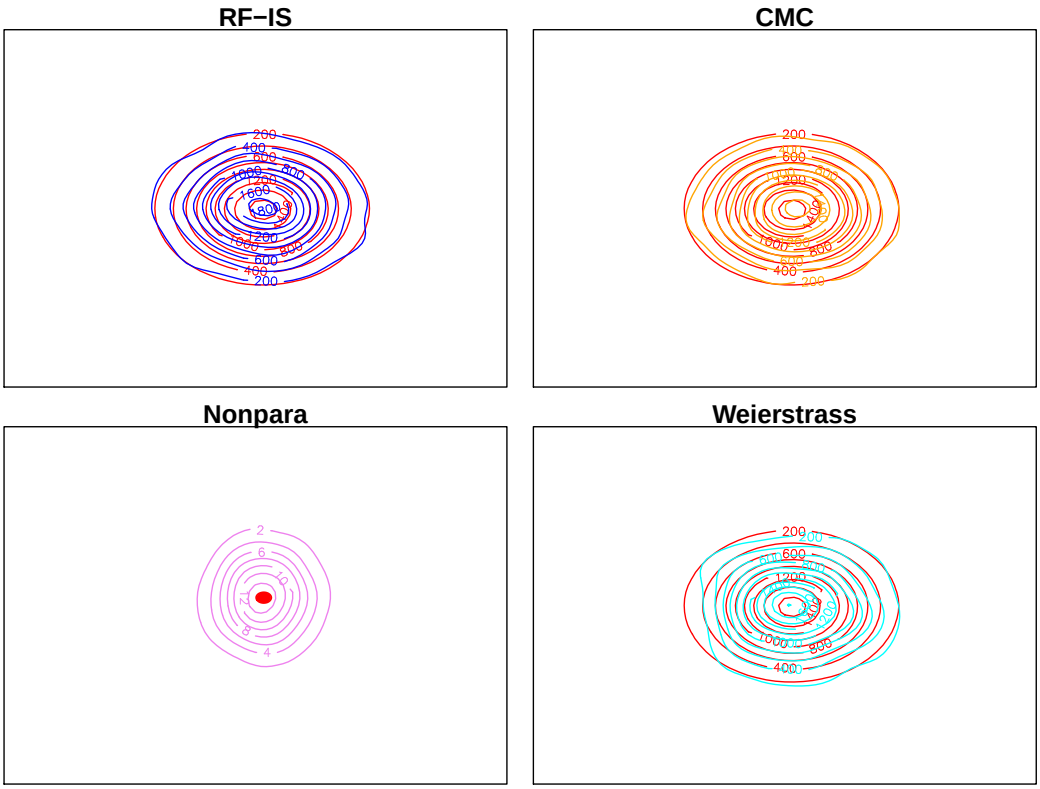

Figure 5.6: Example 3: For normal observations, comparison of the contours of true posterior (red), RF-IS (blue), consensus Monte Carlo (orange), KDE (violet) and Weierstrass sampler (cyan).

5.4 Conclusion

Our approach combines divide-and-conquer MCMC methods, random forest learning, and importance sampling simulation, to scale MCMC algorithms. Unlike existing divide-and-conquer MCMC methods, we propose to scale partial posteriors with penalty power factors that are neither 1 nor the number of subsets, and not necessarily all equal. Given such suitable scale factors, the resulting subposteriors may overlapping one another, which is a very important feature for the recombination stage. The choice of random forests as a learning method naturally follows from its robust and computing features. Several numerical experiments exhibit good performances compared with existing divide-and-conquer scalable MCMC methods.

However, the main limitation of our method is the curse of dimensionality in random forest training. In high dimensional parameter spaces, learning requires larger parameter samples to train each random forest. A second shortcoming is the necessity to select of scale factors. We cannot propose a generic method to tune them, except in some cases where we can obtain cheap estimations of the means and covariances of both the true posteriors and the subposteriors.

Nonetheless, if we can roughly evaluate a high probability region, E, of the complete data posterior, we can discard the earlier stage of running MCMC simulations over the K subsets, and instead generate points from E uniformly, in order to directly train the random forests with these points or to implement the importance sampling. Besides, the result of our methods can also be used to construct such E's.

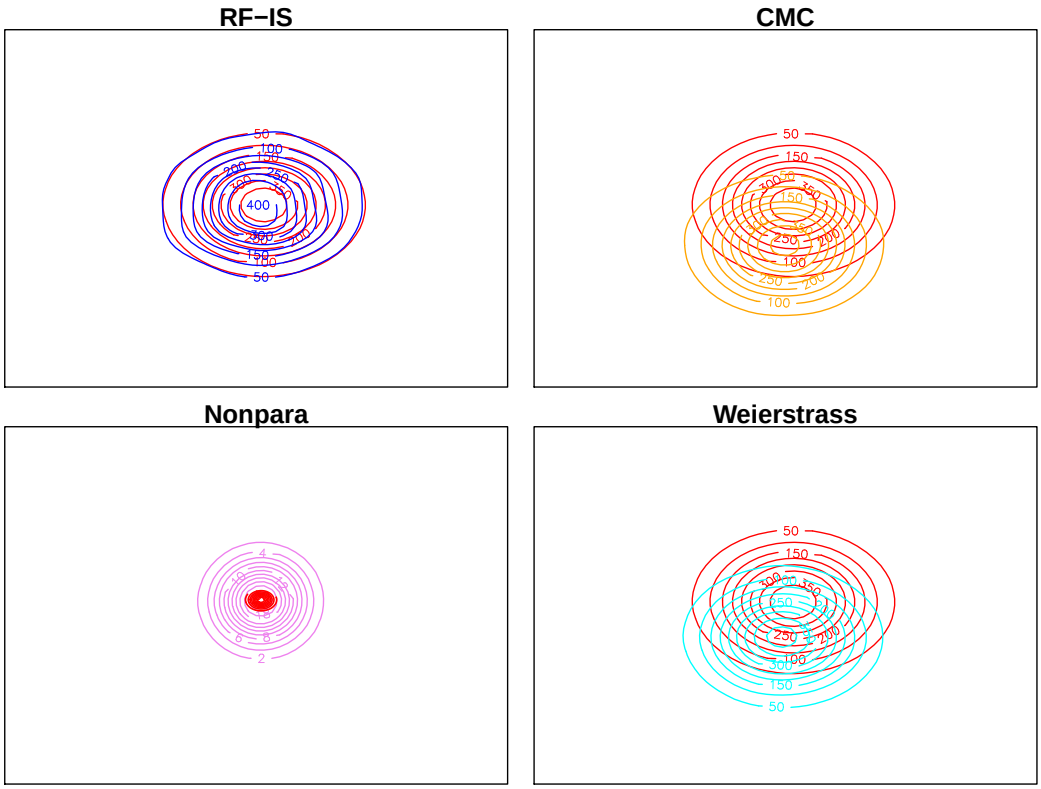

Figure 5.7: Example 3: For log-normal observations, comparison of the contours of true posterior (red), RF-IS (blue), consensus Monte Carlo (orange), KDE (violet) and Weierstrass sampler (cyan).

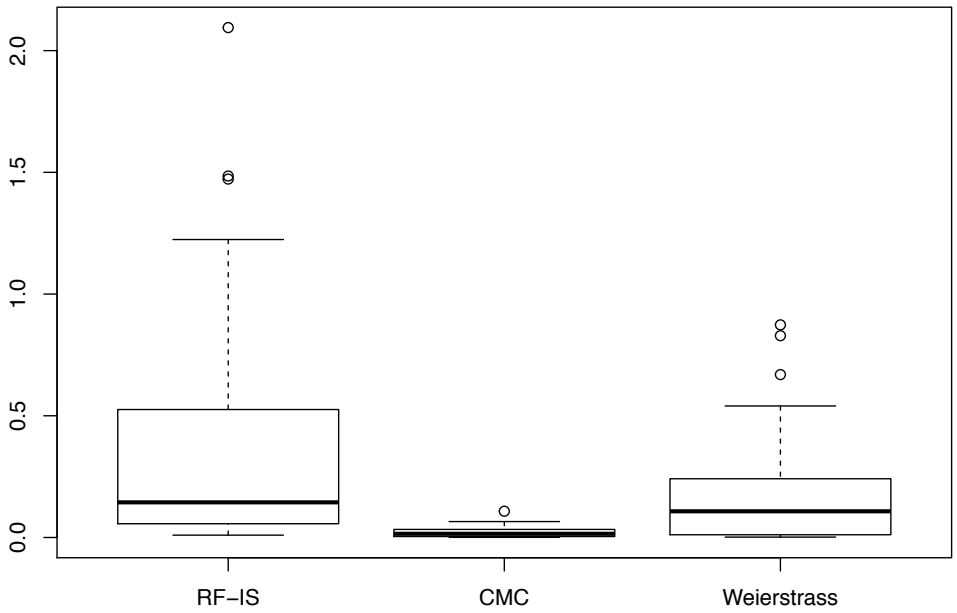

Figure 5.8: Example 4: boxplots of mean squared errors for RF-IS, consensus Monte Carlo and Weierstrass samplers, based on 30 replications.

Bibliography

Andrieu, C. and Roberts, G. O. (2009). The pseudo-marginal approach for efficient Monte Carlo computations. *The Annals of Statistics*, pages 697–725.

Bardenet, R., Doucet, A., and Holmes, C. (2014). Towards scaling up Markov chain Monte Carlo: an adaptive subsampling approach. *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 405–413.

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.

Bierkens, J., Bouchard-Côté, A., Doucet, A., Duncan, A. B., Fearnhead, P., Lienart, T., Roberts, G., and Vollmer, S. J. (2018). Piecewise deterministic Markov processes for scalable Monte Carlo on restricted domains. *Statistics & Probability Letters*, 136:148–154.

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*.

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.

Breiman, L. (2001). Random forests. *Machine learning*, 45(1):5–32.

Chen, T., Fox, E., and Guestrin, C. (2014). Stochastic gradient Hamiltonian Monte Carlo. *International Conference on Machine Learning*, pages 1683–1691.

Davis, M. H. (1984). Piecewise-deterministic Markov processes: A general class of non-diffusion stochastic models. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 353–388.

Davis, M. H. (1993). *Markov Models & Optimization*, volume 49. CRC Press.

Ding, N., Fang, Y., Babbush, R., Chen, C., Skeel, R. D., and Neven, H. (2014). Bayesian sampling using stochastic gradient thermostats. *Advances in neural information processing systems*, pages 3203–3211.

Fearnhead, P., Bierkens, J., Pollock, M., and Roberts, G. O. (2016). Piecewise Deterministic Markov Processes for Continuous-Time Monte Carlo. *arXiv preprint arXiv:1611.07873*.

Liu, J. S. (2008). *Monte Carlo strategies in scientific computing*. Springer Science & Business Media.

Minsker, S., Srivastava, S., Lin, L., and Dunson, D. (2014). Scalable and robust Bayesian inference via the median posterior. *International Conference on Machine Learning*, pages 1656–1664.

Neiswanger, W., Wang, C., and Xing, E. (2013). Asymptotically exact, embarrassingly parallel MCMC. *arXiv preprint arXiv:1311.4780*.

Nemeth, C., Sherlock, C., et al. (2017). Merging MCMC subposteriors through Gaussian-process approximations. *Bayesian Analysis*.

Quiroz, M., Villani, M., and Kohn, R. (2016). Exact subsampling MCMC. *arXiv preprint arXiv:1603.08232*.

Scott, S. L., Blocker, A. W., Bonassi, F. V., Chipman, H. A., George, E. I., and McCulloch, R. E. (2016). Bayes and big data: The consensus Monte Carlo algorithm. *International Journal of Management Science and Engineering Management*, 11(2):78–88.

Sherlock, C. and Thiery, A. H. (2017). A Discrete Bouncy Particle Sampler. *arXiv preprint arXiv:1707.05200*.

Srivastava, S., Cevher, V., Dinh, Q., and Dunson, D. (2015). WASP: Scalable Bayes via barycenters of subset posteriors. *Artificial Intelligence and Statistics*, pages 912–920.

Vanetti, P., Bouchard-Côté, A., Deligiannidis, G., and Doucet, A. (2017). Piecewise Deterministic Markov Chain Monte Carlo. *arXiv preprint arXiv:1707.05296*.

Wang, X. and Dunson, D. B. (2013). Parallelizing MCMC via Weierstrass sampler. *arXiv preprint arXiv:1312.4605*.

Wang, X., Guo, F., Heller, K. A., and Dunson, D. B. (2015). Parallelizing MCMC with random partition trees. *Advances in Neural Information Processing Systems*, pages 451–459.

Welling, M. and Teh, Y. W. (2011). Bayesian learning via stochastic gradient Langevin dynamics. *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 681–688.

Average of Recentered Parallel MCMC for Big Data

Joint work with Christian P. ROBERT

6.1 Introduction

Due to the massive influx of data, the power of traditional MCMC algorithms is inhibited for Bayesian inference, for MCMC algorithms are difficult to scale. Indeed, MCMC algorithms, such as Metropolis-Hastings (MH) algorithms (see e.g. Robert and Casella (2004)), require at each iteration to sweep over the whole data set, which is very expensive on large data sets. In order to overcome this shortcoming and rescue MCMC algorithms for big data, a lot of efforts have been devoted over recent years to develop scalable MCMC algorithms. These approaches can be classified into two classes (Angelino et al. (2016), Bardenet et al. (2017)): divide-and-conquer approaches (Gelman et al. (2014),Minsker et al. (2014),Neiswanger et al. (2013),Srivastava et al. (2015),Scott et al. (2016),Wang and Dunson (2013)) and subsampling approaches(Bardenet et al. (2014), Chen et al. (2014), Korattikara et al. (2014), Welling and Teh (2011)). In this article, we propose a new method belonging to the divide-and-conquer category. Specifically, we divide the whole data set into batches and repeat each batch a certain amount of times, run MCMC over repeated batches, recenter all subposteriors thus obtained and take their average as an approximation of the true posterior.

Our article extends the traditional parallel MCMC algorithms in three directions. First, we scale each likelihood of the subposterior with a factor such that it could be regarded as an approximation of the true likelihood, by which we mean turning each subposterior covariance matrix into the same scale with that of the true posterior. Second, our combination method is simple enough, has solid mathematical justifications and is efficient. Third, even though our method is justified in parametric framework, it can be extend to non-parametric Bayesian without modification.

The organization of this paper is as follows. Section 2 outlines the methodology and provides a mathematical justification of its validation. Section 3 applies the method to four numerical experiments and shows its power and efficiency. Section 4 concludes and discusses further research.

6.2 Averaging and Recentering Subposterior Distributions

Let $\mathcal{X} = \{X_1, \cdots, X_N\}$ denote the data set and suppose that X_i be i.i.d. observations from a common distribution P_θ possessing a density $f(x|\theta)$ where $\theta \in \Theta$, an open set of $\mathbb{R}^d$. We fix $\theta_0 \in \Theta$, which may be regarded as the "true value" of the parameter. Suppose the whole data set $\mathcal{X}$ be divided into K subsets $\mathcal{X}_1, \cdots, \mathcal{X}_K$ with same size $M = N/K$. Denote

$$\ell(\theta, x) = \log f(x|\theta)$$

$$\ell_i(\theta) = \sum_{j=1}^{M} \log f(x_{ij}|\theta)$$

$$L_N(\theta) = \sum_{i=1}^{K}\sum_{j=1}^{M} \log f(x_{ij}|\theta) = \sum_{i=1}^{K} \ell_i(\theta)$$

$$\hat{\theta}_i = \arg\max_{\theta\in\Theta} \ell_i(\theta), \quad \hat{\theta} = \arg\max_{\theta\in\Theta} L_N(\theta), \quad \bar{\theta} = \frac{1}{K}\sum_{i=1}^{K} \hat{\theta}_i$$

for each $i \in \{1, \cdots, K\}$, $\mathcal{X}_i = \{x_{ij}\}_{j=1}^{M}$. In classical parallel approaches, one decomposes the overall posterior into a product of subposteriors:

$$\pi(\theta|\mathcal{X}) \propto \prod_{i=1}^{K} \left(\pi(\theta)^{1/K} \exp(\ell_i(\theta))\right)$$

Even though this decomposition is correct mathematically, it is not reasonable in statistics. In Bayesian analysis, the type of prior should not change with the size of data set. Hence, using a prior that depends on the observation size is not appropriate. In order to overcome this shortcoming, we can create an artificial data set for each subset, which just repeats each data point K times for each subset. Hence, we can apply the overall prior on these artificial data sets. That is, we regard the following rescaled subposteriors as approximations to the overall posterior:

$$\pi_i(\theta|\mathcal{X}_i) \propto \exp\{K\ell_i(\theta)\}\pi(\theta)$$

This idea has also appeared in Minsker et al. (2014), Srivastava et al. (2015). Denote

$$\theta_i^* = \mathbb{E}_{\pi_i}(\theta), \quad \bar{\theta}^* = \frac{1}{K}\sum_{i=1}^{K} \theta_i^*$$

In these approximations, the factor K rescales the variance of each subset posterior $\pi_i(\theta|\mathcal{X}_i)$ to be roughly of the same order as that of the overall posterior $\pi(\theta|\mathcal{X}) \propto \exp(L_N(\theta))\pi(\theta)$. Inspired by this phenomenon, we recenter each subset posterior to their common mean and then average them to approximate the true posterior. That is, the overall posterior $\pi(\theta|\mathcal{X})$ is approximated by

$$\frac{1}{K}\sum_{i=1}^{K} \pi_i(\theta - \bar{\theta}^* + \theta_i^*|\mathcal{X}_i)$$

In order to proceed the theoretical analysis, we make some mild assumptions on the

Algorithm 6.1 Average of Recentered Subposterior

Input: K subsets of data $\mathcal{X}_1, \cdots, \mathcal{X}_K$, each with size M.
Output: Samples to approximate the true posterior.
for $i = 1, \cdots, K$ (in parallel) **do**
  for $t = 1, \cdots, T$ **do**
    Draw θ_i^t from $\pi_i(\theta|\mathcal{X}_i)$ via MCMC.
  end for
  Calculate $\theta_i^* = \frac{1}{T}\sum_{t=1}^{T}\theta_i^t$.
end for
Calculate $\bar{\theta}^* = \frac{1}{K}\sum_{i=1}^{K}\theta_i^*$
for $i = 1, \cdots, K$ (in parallel) **do**
  for $t = 1, \cdots, T$ **do**
    $\theta_i^t \leftarrow \theta_i^t - \theta_i^* + \bar{\theta}^*$.
  end for
end for
Result: $\{\theta_i^t | i = 1, \cdots; K, t = 1, \cdots, T\}$ approximates the overall posterior.

likelihood and the prior $\pi(\theta)$. These assumption are standard for the Bernstein-von Mises theorem (Ghosh et al. (2007)).

Assumption 1: The support set $\{x : f(x|\theta) > 0\}$ is the same for all $\theta \in \Theta$.

Assumption 2: $\ell(\theta, x)$ is three times differentiable with respect to θ in a neighbourhood $\{\theta : ||\theta - \theta_0|| \leq \delta_0\}$ of θ_0. The expectation of $\mathbb{E}_{\theta_0}\nabla\ell(\theta_0, X_1)$ and $\mathbb{E}_{\theta_0}\nabla^2\ell(\theta_0, X_1)$ are both finite and for any x and $p, q, r \in \{1, \cdots, d\}$,

$$\sup_{\theta:||\theta-\theta_0||\leq\delta_0} \frac{\partial^3}{\partial\theta_p\partial\theta_q\partial\theta_r}\ell(\theta, x) \leq M(x) \quad \text{and} \quad \mathbb{E}_{\theta_0}M(X_1) < \infty$$

Assumption 3: Interchange of the order of integrating with respect to P_{θ_0} and differentiation at θ_0 is justified, so that

$$\mathbb{E}_{\theta_0}\nabla\ell(\theta_0, X_1) = 0 \quad \text{and} \quad \mathbb{E}_{\theta_0}\nabla^2\ell(\theta_0, X_1) = -\mathbb{E}_{\theta_0}\nabla\ell(\theta_0, X_1)[\nabla\ell(\theta_0, X_1)]^T$$

Also the Fisher information $I(\theta_0) = \mathbb{E}_{\theta_0}\nabla\ell(\theta_0, X_1)[\nabla\ell(\theta_0, X_1)]^T$ is positive definitely.

Assumption 4: For any $\delta > 0$, there exists an $\epsilon > 0$, with $P_{\theta_0}-$probability one, such that

$$\sup_{\theta:||\theta-\theta_0||>\delta} \frac{1}{N}(L_N(\theta) - L_N(\theta_0)) < -\epsilon$$

for all sufficiently large N.

Theorem 1: If Assumptions 1 -4 holds, then as $N \to \infty$ and $M \to \infty$,

$$\left|\frac{1}{K}\sum_{i=1}^{K}\pi_i(\theta - \bar{\theta} + \hat{\theta}_i|\mathcal{X}_i) - \pi(\theta|\mathcal{X})\right|_{TV} \to 0$$

$$\left|\frac{1}{K}\sum_{i=1}^{K}\pi_i(\theta - \bar{\theta}^* + \theta_i^*|\mathcal{X}_i) - \pi(\theta|\mathcal{X})\right|_{TV} \to 0$$

The proof of Theorem 1 can be found in Appendix.

Remark 1: For the center $\bar{\theta}^*$, we have $\bar{\theta}^* - \hat{\theta} = \mathcal{O}_P(\frac{1}{\sqrt{N}})$. In order to improve its performance, we can resort to the Newton-Raphson method Kaw et al. (2008). Under mild conditions, Newton-Raphson method can converge quadratically. Given its impact, the Newton-Raphon method only needs to be called for a few steps.

Remark 2: In Lemma 1, we can replace $I(\theta_0)$ with $I_i(\hat{\theta}_i)$. Then $\sqrt{KM}(\theta - \hat{\theta}_i)$ has the same limit distribution with $\mathcal{N}(0, I^{-1}(\hat{\theta}_i))$ as $M \to \infty$, where, $\theta \sim \pi_i(\theta|\mathcal{X}_i)$. As such, if $\theta \sim \frac{1}{K}\sum_{i=1}^{K} \pi_i(\theta - \bar{\theta} + \hat{\theta}_i|\mathcal{X}_i)$,

$$\mathrm{Var}(\sqrt{N}(\theta - \bar{\theta})) \asymp \frac{1}{K}\sum_{i=1}^{K} I_i^{-1}(\hat{\theta}_i)$$

Because, for each $i = 1, \cdots, K$, $I_i(\hat{\theta}_i) - I(\theta_0) \to \mathcal{N}(0, \frac{1}{M}\Sigma_1)$, as a result, with Delta method, we obtain

$$\sqrt{N}\left(\frac{1}{K}\sum_{i=1}^{K} I_i^{-1}(\hat{\theta}_i) - I^{-1}(\theta_0)\right) \to \mathcal{N}(0, \Sigma_1)$$

This means that the covariance matrix of our sample converges to the true one with $\mathcal{O}_P(\frac{1}{\sqrt{N}})$.

Remark 3: M is a user-specified parameter, which determines the gap between $\bar{\theta}$ and $\hat{\theta}$. Actually, M is not necessary $\mathcal{O}(N^\gamma)$, as long as each $\bar{\theta}_i$ is reasonable approximation to θ_0, which means CLT works well for M.

6.3 Numerical Experiments

We illustrate the accuracy of our method in the first three examples and compared its performance with three other methods: Consensus Monte Carlo (CMC, Scott et al. (2016)), Weierstrass Sampler (WS, Wang and Dunson (2013)), Mposterior (MP, Minsker et al. (2014)) in L_2 distance,

$$L_2(p, q) = \int_{\mathbb{R}^d} (p(x) - q(x))^2 dx$$

where p, q are two probability density functions on $\mathbb{R}^d$. Our method is denoted by AR in the table of results of L_2 distances. In Example 4, we show that our method can be applied to data augmentation cases.

Example 1: (Gaussian Model) In this example, the model is assumed as follows:

$$X_i|\theta \sim \mathcal{N}(\mu, \sigma^2), \quad \theta = (\mu, \sigma^2)$$

we sampled $X_i \sim \mathcal{N}(0, 10), i = 1, \cdots, 10^6$ and chose $p(\mu, \log(\sigma)) \propto 1$ as the prior. The data set was split into K subsets, where we set $K = 20, 50, 100$. In Table 1 we compare the performance of the four methods.

Example 2: (Bayesian Logistic Model) In the Bayesian logistic model, we have observations $\{(x_i, y_i)\}_{i=1}^N$, where $x_i \in \mathbb{R}^p$ and $y_i \in \{0, 1\}$, and

$$\mathbb{P}(y_i = 1|x_i, \theta) = \frac{1}{1 + e^{-x_i^T\theta}}$$

K	CMC	AR	WS	MP
20	1.03×10^{-4}	1.00×10^{-4}	1.71×10^{-4}	1.06×10^{-3}
50	1.54×10^{-4}	1.33×10^{-4}	2.87×10^{-4}	1.53×10^{-3}
100	1.48×10^{-4}	2.24×10^{-4}	1.58×10^{-4}	7.25×10^{-4}

Table 6.1: The L_2 distances of our method versus others for Example 1. CMC is Consensus Monte Carlo, AR is our method, WS is Weierstrass sampler and MP is Mposterior.

We applied this model both on synthetic and on real data sets.

Synthetic dataset: The dataset $\{(x_i, y_i)\}_{i=1}^N$ consists of $N = 10^5$ observations and $p = 5$. We set $\theta = (0.3, 5, -7, 2.4, -20), x_{i1} \equiv 1$ and draw $x_{ij} \sim \mathcal{U}(0, 1), j = 2, \cdots, 5$. We set $K = 50, 100$. In this example, we apply the Newton-Raphon method to correct the center. Because the original center of our method is quite close to the MAP, the Netwon-Raphon method converges in this example after 5 iterations. The results are shown in Table 2.

Real Dataset: We consider the *Covtype* dataset Wang et al. (2015), which consists of 581,012 observations in 54 dimensions. We consider a Bayesian logistic classification using the first $p = 3$ attributes only and taking $N = 5 \times 10^5$ for simplicity. During the simulations, we set $K = 100$ and $K = 500$ and call the Newton-Raphon method for 5 times. The results are shown in Table 2.

K	CMC	AR	WS	MP
(Synthetic) 50	1.08×10^{-2}	7.56×10^{-3}	1.03×10^{-2}	2.34×10^{-1}
(Synthetic) 100	1.78×10^{-2}	8.83×10^{-3}	4.71×10^{-2}	2.54×10^{-1}
(Real) 100	3.77×10^{-4}	3.05×10^{-4}	9.04×10^{-4}	6.06×10^{-3}
(Real) 500	7.28×10^{-4}	3.35×10^{-4}	2.00×10^{-3}	$- - - - -$

Table 6.2: The L_2 distances of our method versus others for Example 2.

Example 3: (Beta-Bernoulli Model) In this example, the dimension of the parameter is one and the posterior has an analytical expression, which means we can use the true posterior directly, instead of MCMC approximation. We simulated 100,000 samples from Bernoulli distribution $B(p)$ and the prior is $Beta(0.01, 0.01)$. We applied our method in two cases: $p = 0.1$, corresponding to a common scenario, and $p = 0.01$, corresponding to the rare event case. We simulated 10^5 samples from posterior or subposteriors. The L_2 distances are shown in Table 3 and the marginal density functions are in Figure 1. Compared with the other three methods, our method is more accurate in the cases where each subset contains only small part of information compared with the whole data set.

(p,K)	CMC	AR	WS	MP
(0.1,50)	1.13×10^{-4}	2.88×10^{-5}	1.35×10^{-4}	2.66×10^{-4}
(0.1,100)	2.32×10^{-4}	3.66×10^{-5}	3.56×10^{-4}	8.42×10^{-5}
(0.001,50)	1.58×10^{-4}	3.49×10^{-6}	1.85×10^{-4}	2.74×10^{-5}
(0.001,100)	4.78×10^{-4}	4.02×10^{-6}	3.37×10^{-4}	1.15×10^{-4}

Table 6.3: The L_2 distances of our method versus others for Example 3.

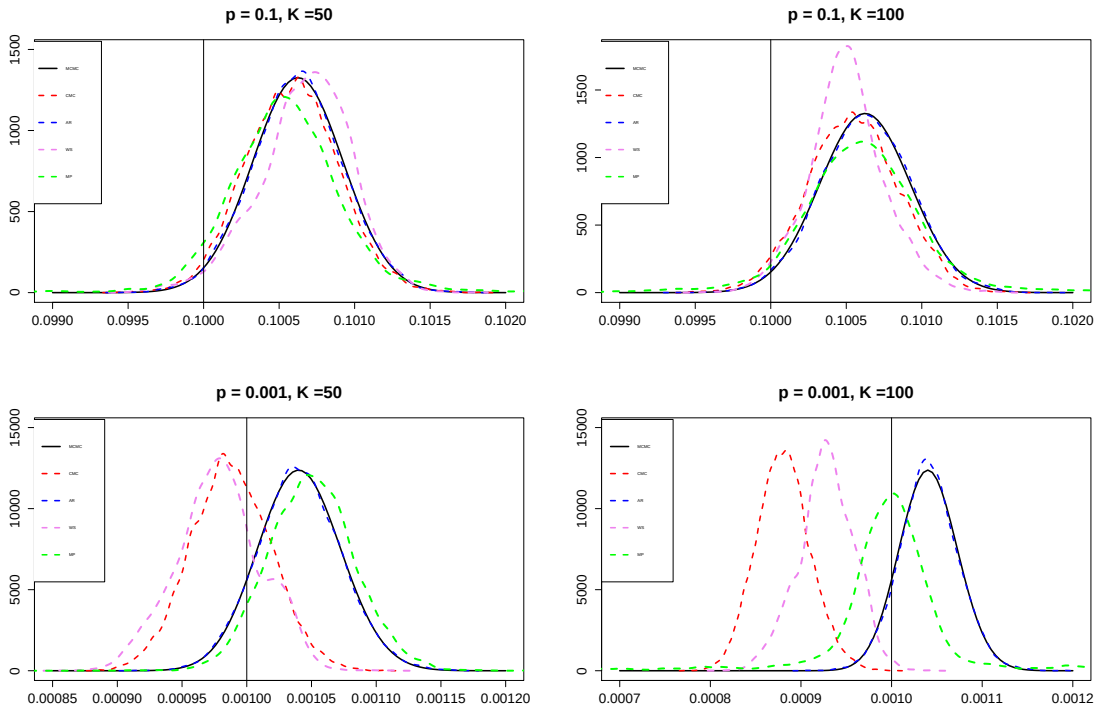

Figure 6.1: The graphs of probability density functions of each type of parameter. The black solid line corresponds to the true posterior, the blue dashed line is our method (AR), the red dashed line is Consensus Monte Carlo (CMC), the violet dashed line is Weierstrass sampler (WS) and the green dashed line is Mposterior (MP).

Example 4: (Gaussian Mixture Model) In this example, we extended our method to latent variable cases. The model is

$$Y_i|X_i \sim (1-p)\mathcal{N}(Y_i; \alpha X_{i1} + \beta X_{i2}, \sigma^2) + p\mathcal{N}(Y_i; 0, \psi^2)$$

The parameter is $\theta = (\alpha, \beta, \sigma^2, \psi^2, p)$. Adding latent variables Z_i and running Gibbs Sampling is the traditional way to conduct parameter inference for this model. Here,

$$\begin{aligned} Y_i|Z_i = 0, X_i &\sim \mathcal{N}(Y_i; \alpha X_{i1} + \beta X_{i2}, \sigma^2) \\ Y_i|Z_i = 1, X_i &\sim \mathcal{N}(Y_i; 0, \psi^2) \end{aligned}$$

The posterior of θ, Z given the observations $\{(X_i, Y_i)\}$ is

$$\begin{aligned} p(\theta, Z|Y, X) \propto &\left(\prod_{i=1}^{N} \left(\phi(y_i|\alpha x_{i1} + \beta x_{i2}, \sigma^2)\right)^{1-Z_i} \left(\phi(y_i|0, \psi^2)\right)^{Z_i} \pi(Z_i|p) \right) \\ &\times \pi(\alpha)\pi(\beta)\pi(\sigma^2)\pi(\psi^2)\pi(p) \end{aligned}$$

The priors are chosen to be

$$\begin{aligned} \alpha \sim \mathcal{N}(m_\alpha, \sigma_\alpha^2), \quad \beta \sim \mathcal{N}(m_\beta, \sigma_\beta^2), \\ \sigma^2 \sim \mathcal{IG}(\alpha_\sigma, \beta_\sigma), \quad \psi^2 \sim \mathcal{IG}(\alpha_\psi, \beta_\psi), \\ Z_i|p \sim Bernoulli(p), \quad p \sim Beta(\lambda, \eta). \end{aligned}$$

These priors are conjugate for the model. That is,

$$z_i|X_i, Y_i, \theta \sim Bernoulli(p_i^*) \quad p_i^* = \frac{p\phi(y_i|0, \psi^2)}{(1-p)\phi(y_i|\alpha x_{i1} + \beta x_{i2}, \sigma^2) + p\phi(y_i|0, \psi^2)}$$

$$\alpha|X,Y,Z,\theta_{-\alpha} \sim \mathcal{N}(m_\alpha^*, {\sigma_\alpha^*}^2),$$
$$m_\alpha^* = \frac{\sum_{i=1}^N \frac{(y_i-\beta x_{i2})x_{i1}(1-z_i)}{\sigma^2} + \frac{m_\alpha}{\sigma_\alpha^2}}{\sum_{i=1}^N \frac{x_{i1}^2(1-z_i)}{\sigma^2} + \frac{1}{\sigma_\alpha^2}} \quad {\sigma_\alpha^*}^2 = \frac{1}{\sum_{i=1}^N \frac{x_{i1}^2(1-z_i)}{\sigma^2} + \frac{1}{\sigma_\alpha^2}}$$

$$\beta|X,Y,Z,\theta_{-\beta} \sim \mathcal{N}(m_\beta^*, {\sigma_\beta^*}^2),$$
$$m_\beta^* = \frac{\sum_{i=1}^N \frac{(y_i-\alpha x_{i1})x_{i2}(1-z_i)}{\sigma^2} + \frac{m_\beta}{\sigma_\beta^2}}{\sum_{i=1}^N \frac{x_{i2}^2(1-z_i)}{\sigma^2} + \frac{1}{\sigma_\beta^2}} \quad {\sigma_\beta^*}^2 = \frac{1}{\sum_{i=1}^N \frac{x_{i2}^2(1-z_i)}{\sigma^2} + \frac{1}{\sigma_\beta^2}}$$

$$\sigma^2|X,Y,Z,\theta_{-\sigma^2} \sim \mathcal{IG}(\alpha_\sigma^*, \beta_\sigma^*),$$
$$\alpha_\sigma^* = \alpha_\sigma + \frac{1}{2}\sum_{i=1}^N (1-z_i), \quad \beta_\sigma^* = \beta_\sigma + \frac{1}{2}\sum_{i=1}^N (1-z_i)(\alpha x_{i1} + \beta x_{i2} - y_i)^2$$

$$\psi^2|X,Y,Z,\theta_{-\psi^2} \sim \mathcal{IG}(\alpha_\psi^*, \beta_\psi^*), \quad \alpha_\psi^* = \alpha_\psi + \frac{1}{2}\sum_{i=1}^N z_i, \quad \beta_\psi^* = \beta_\psi + \frac{1}{2}\sum_{i=1}^N z_i y_i^2$$

$$p|X,Y,Z,\theta_{-p} \sim Beta\left(\lambda + \sum_{i=1}^N z_i, \eta + \sum_{i=1}^N (1-z_i)\right)$$

Even though this model is conjugate, simulating a label Z_i for each data point (x_i, y_i) at each iteration is too expensive to use the Gibbs Sampling in big data context. In our simulation, we set $\theta = (2, 5, 1, 10, 0.05)$, $N = 10^6$ and $K = 50, M = N/K$. For each subset $\mathcal{X}_i = \{x_{i1}, \cdots, x_{iM}\}$, imagine that we have a sequence of artificial observations $\{(x_i^*, y_i^*)\}_{i=1}^N$ by repeating $\mathcal{X}_i$ with K times, that is,

$$x_t^* = x_{ij}, \qquad \text{for} \quad (j-1)K + 1 \le t \le jK$$

For each x_{ij}, it appears K times, which means its corresponding labels follow binomial distribution. The Gibbs updating procedure should be changed:

$$z_{ij}|x_{ij}, y_{ij}, \theta = \text{Bernoulli}(K, p_{ij}^*),$$
$$p_{ij}^* = \frac{p\phi(y_{ij}|0,\psi^2)}{(1-p)\phi(y_{ij}|\alpha x_{ij1} + \beta x_{ij2}, \sigma^2) + p\phi(y_{ij}|0,\psi^2)};$$
$$\alpha|\mathcal{X}_i, \mathcal{Z}_i, \theta_{-\alpha} \sim \mathcal{N}(m_\alpha^*, {\sigma_\alpha^*}^2),$$
$$m_\alpha^* = \frac{\sum_{j=1}^M \frac{(y_{ij}-\beta x_{ij2})x_{ij1}(K-z_{ij})}{\sigma^2} + \frac{m_\alpha}{\sigma_\alpha^2}}{\sum_{j=1}^M \frac{x_{ij1}^2(K-z_{ij})}{\sigma^2} + \frac{1}{\sigma_\alpha^2}},$$
$${\sigma_\alpha^*}^2 = \frac{1}{\sum_{j=1}^M \frac{x_{ij1}^2(K-z_{ij})}{\sigma^2} + \frac{1}{\sigma_\alpha^2}};$$

$$\beta|\mathcal{X}_i, \mathcal{Z}_i, \theta_{-\beta} \sim \mathcal{N}(m_\beta^*, \sigma_\beta^{*2}),$$

$$m_\beta^* = \frac{\sum_{j=1}^M \frac{(y_{ij}-\alpha x_{ij1})x_{ij2}(K-z_{ij})}{\sigma^2} + \frac{m_\beta}{\sigma_\beta^2}}{\sum_{j=1}^M \frac{x_{ij2}^2(K-z_{ij})}{\sigma^2} + \frac{1}{\sigma_\beta^2}},$$

$$\sigma_\beta^{*2} = \frac{1}{\sum_{j=1}^M \frac{x_{ij2}^2(K-z_{ij})}{\sigma^2} + \frac{1}{\sigma_\beta^2}};$$

$$\sigma^2|\mathcal{X}_i, \mathcal{Z}_i, \theta_{-\sigma^2} \sim \mathcal{IG}(\alpha_\sigma^*, \beta_\sigma^*),$$

$$\alpha_\sigma^* = \alpha_\sigma + \frac{1}{2}\sum_{j=1}^M (K - z_{ij}),$$

$$\beta_\sigma^* = \beta_\sigma + \frac{1}{2}\sum_{j=1}^M (K - z_{ij})(\alpha x_{ij1} + \beta x_{ij2} - y_{ij})^2;$$

$$\psi^2|\mathcal{X}_i, \mathcal{Z}_i, \theta_{-\psi^2} \sim \mathcal{IG}(\alpha_\psi^*, \beta_\psi^*),$$

$$\alpha_\psi^* = \alpha_\psi + \frac{1}{2}\sum_{j=1}^M z_{ij},$$

$$\beta_\psi^* = \beta_\psi + \frac{1}{2}\sum_{j=1}^M z_{ij} y_{ij}^2;$$

$$p|\mathcal{X}_i, \mathcal{Z}_i, \theta_{-p} \sim Beta\left(\lambda + \sum_{j=1}^M z_{ij}, \eta + \sum_{j=1}^M (K - z_{ij})\right)$$

In this example, the Figure 2 shows that our method is quite appealing in accuracy.

6.4 Conclusion

In this article, we proposed a new combination of samples from rescaled subposteriors to approximate the overall posterior and gave its mathematical justification. In order to show its validation in practice, we applied it on several common models. Compared with classical parallel approaches, our method is more reasonable at a statistical level, shares the same computation cost in parallel stage and the combination stage is very cheap, without the necessity of running an additional MCMC. At the same time, according to the simulations, our method is quite accurate and satisfactory.

6.5 Appendix

In this section, we give a proof of Theorem 1. First, let's introduce some useful lemmas.

Lemma 1: The distribution of $\sqrt{N}(\theta - \hat{\theta}_i)$, where $\theta \sim \pi_i(\theta|\mathcal{X}_i)$ converges to the normal distribution $\mathcal{N}(\mathbf{0}, I^{-1}(\theta_0))$ under the total variation metric, i.e.

$$\int_{\mathcal{R}^d} \left| \pi_i(\theta|\mathcal{X}_i) - \phi(\theta; \hat{\theta}_i, I^{-1}(\theta_0)) \right| d\theta \to 0 \tag{6.1}$$

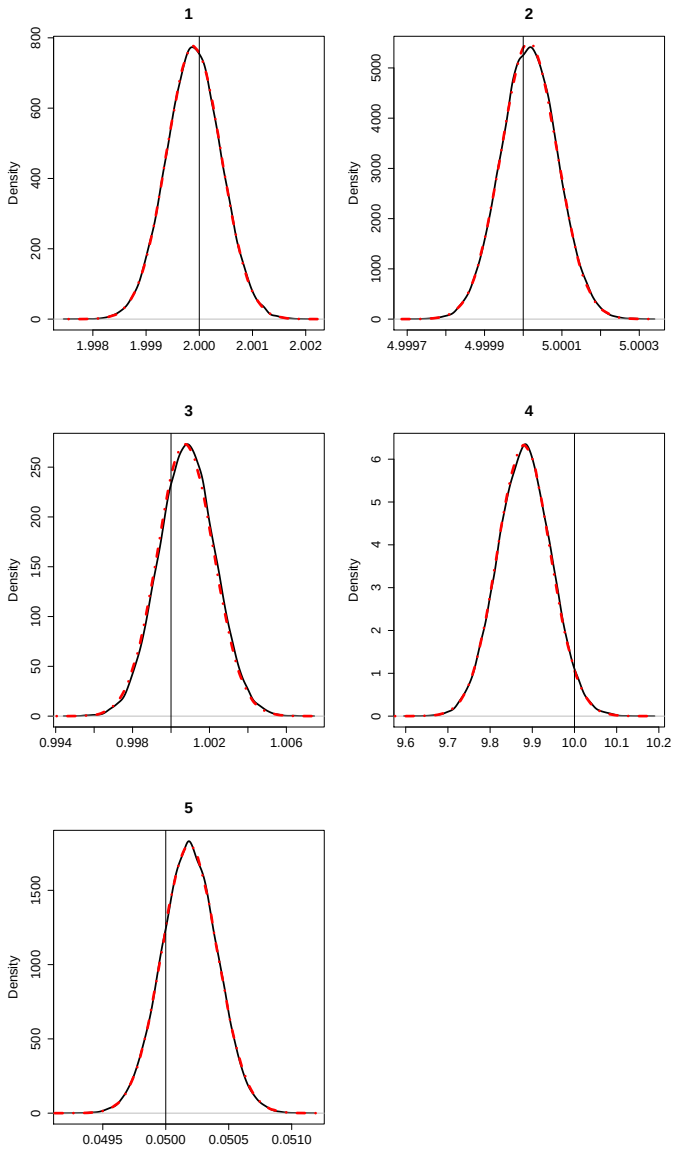

Figure 6.2: The graphs of probability density functions of each type of parameter. Vertical lines mark the parameter values from which the data set is generated. Black solid lines represent the marginal pdf of samples from full MCMC while red dashed lines represent the ones from our method

Proof: Denote $t = \sqrt{N}(\theta - \hat{\theta}_i)$, $\theta \sim \pi_i(\theta|\mathcal{X}_i)$, then

$$t \sim C_i^{-1}\pi(\hat{\theta}_i + \frac{t}{\sqrt{N}}) \exp\left[K\ell_i(\hat{\theta}_i + \frac{t}{\sqrt{N}}) - K\ell_i(\hat{\theta}_i)\right]$$

where $C_i = \int_{\mathbb{R}^d} \pi(\hat{\theta}_i + \frac{t}{\sqrt{N}}) \exp\left[K\ell_i(\hat{\theta}_i + \frac{t}{\sqrt{N}}) - K\ell_i(\hat{\theta}_i)\right] dt$. Denote

$$g_i(t) = \pi(\hat{\theta}_i + \frac{t}{\sqrt{N}}) \exp\left[K\ell_i(\hat{\theta}_i + \frac{t}{\sqrt{N}}) - K\ell_i(\hat{\theta}_i)\right] - \pi(\theta_0) \exp\left[-\frac{t^T I(\theta_0) t}{2}\right]$$

On $A_1 = \{t : ||t|| > \delta_0\sqrt{N}\}$, we have

$$K\ell_i(\hat{\theta}_i + \frac{t}{\sqrt{N}}) - K\ell_i(\hat{\theta}_i) < -N\epsilon$$

Then,

$$\int_{A_1} g_i(t)dt \to 0$$

On $A_2 = \{t : ||t|| \leq \delta_0\sqrt{N}\}$, we have

$$K\ell_i(\hat{\theta}_i + \frac{t}{\sqrt{N}}) - K\ell_i(\hat{\theta}_i) = -\frac{1}{2}t^T\hat{I}_i t + R_i(t)$$

where $R_i(t) = \frac{K}{6}(\frac{1}{\sqrt{N}})^3 \sum_{j=1}^{M}\sum_{p=1}^{d}\sum_{q=1}^{d}\sum_{r=1}^{d} t_p t_q t_r \frac{\partial^3}{\partial\theta_p\partial\theta_q\partial\theta_r}\log f(x_{ij}|\theta')$, θ' lies in the line segment between $\hat{\theta}_i$ and $\hat{\theta}_i + \frac{t}{\sqrt{N}}$, and

$$(\hat{I}_i)_{pq} = \frac{1}{M}\sum_{j=1}^{M}\left[-\frac{\partial^2}{\partial\theta_p\partial\theta_q}\log f(x_{ij}|\theta)\right]\Big|_{\hat{\theta}_i}$$

For each $t \in A_2$, we obtain $R_i(t) \to 0$ and $\hat{I}_i \to I(\theta_0)$ as $M \to \infty$. Hence, $g_i(t) \to 0$. Besides,

$$|R_i(t)| \leq \frac{1}{6}\delta_0\frac{t^2}{N}d^3K\sum_{j=1}^{M}M(x_{ij}) \leq \frac{1}{4}t^T\hat{I}_i t$$

As a result,

$$\exp\left[K\ell_i(\hat{\theta}_i + \frac{t}{\sqrt{N}}) - K\ell_i(\hat{\theta}_i)\right] \leq \exp\left[-\frac{1}{4}t^T\hat{I}_i t\right] \leq \exp\left[-\frac{t^T I(\theta_0)t}{8}\right]$$

Therefore, $|g_i(t)|$ is dominated by an integrable function on A_2. Thus, we obtain $\int_{\mathbb{R}^d}|g_i(t)|dt \to 0$ as $M \to \infty$. Hence, we have immediately

$$C_i \to \pi(\theta_0)\sqrt{(2\pi)^d/\det I(\theta_0)}$$

$$\int\left|C_i^{-1}\pi(\hat{\theta}_i + \frac{t}{\sqrt{N}})\exp\left[K\ell_i(\hat{\theta}_i + \frac{t}{\sqrt{N}}) - K\ell_i(\hat{\theta}_i)\right] - \sqrt{\frac{\det I(\theta_0)}{(2\pi)^d}}\exp\left[-\frac{1}{2}t^T I(\theta_0)t\right]\right|dt$$

$$\leq C_i^{-1}\int|g_i(t)|dt + \int\left|C_i^{-1}\pi(\theta_0)\exp\left[-\frac{1}{2}t^T I(\theta_0)t\right] - \sqrt{\frac{\det I(\theta_0)}{(2\pi)^d}}\exp\left[-\frac{1}{2}t^T I(\theta_0)t\right]\right|dt$$

$$\leq C_i^{-1}\int|g_i(t)|dt + \left|C_i^{-1}\pi(\theta_0)/\sqrt{\frac{\det I(\theta_0)}{(2\pi)^d}} - 1\right| \to 0$$

Corollary 1: Denote $\bar{\theta} = \frac{1}{K}\sum_{i=1}^{K}\hat{\theta}_i$. Then

$$\int_{\mathcal{R}^d}\left|\frac{1}{K}\sum_{i=1}^{K}\pi_i(\theta - \bar{\theta} + \hat{\theta}_i|\mathcal{X}_i) - \phi(\theta;\bar{\theta}, I^{-1}(\theta_0))\right|d\theta \to 0$$

Corollary 2: Denote $\theta_i^* = \int\theta\pi_i(\theta|\mathcal{X}_i)d\theta$, then $\sqrt{N}(\theta_i^* - \hat{\theta}_i) \to 0$.

Proof: Proceeding as in the proof of Lemma 1 and using the assumption of finite expectation of the prior, we can have

$$\int|t|\Bigg|C_i^{-1}\pi(\hat{\theta}_i + \frac{t}{\sqrt{N}})\exp\left[K\ell_i(\hat{\theta}_i + \frac{t}{\sqrt{N}}) - K\ell_i(\hat{\theta}_i)\right]$$
$$- \sqrt{\frac{\det I(\theta_0)}{(2\pi)^d}}\exp\left[-\frac{1}{2}t^T I(\theta_0)t\right]\Bigg|dt \to 0$$

This implies

$$\int tC_i^{-1}\pi(\hat{\theta}_i + \frac{t}{\sqrt{N}}) \exp\left[K\ell_i(\hat{\theta}_i + \frac{t}{\sqrt{N}}) - K\ell_i(\hat{\theta}_i)\right] dt$$
$$\to \int t\sqrt{\frac{\det I(\theta_0)}{(2\pi)^d}} \exp\left[-\frac{1}{2}t^T I(\theta_0)t\right] dt = 0$$

Therefore,

$$\sqrt{N}(\theta_i^* - \hat{\theta}_i) = \int tC_i^{-1}\pi(\hat{\theta}_i + \frac{t}{\sqrt{N}}) \exp\left[K\ell_i(\hat{\theta}_i + \frac{t}{\sqrt{N}}) - K\ell_i(\hat{\theta}_i)\right] dt \to 0$$

Lemma 2: For two multivariate normal distributions $P = \mathcal{N}(\mu_1, \Sigma)$ and $Q = \mathcal{N}(\mu_2, \Sigma)$, their Kullback-Leibler divergence is

$$KL(P|Q) = KL(Q|P) = \frac{1}{2}(\mu_1 - \mu_2)^T\Sigma^{-1}(\mu_1 - \mu_2)$$

Lemma 3: For two probability measures P and Q, we have following inequality

$$\left|P - Q\right|_{TV} \leq 2\sqrt{KL(P|Q)}$$

Lemma 4: For each $i \in \{1, \cdots, K\}$, we have $\hat{\theta}_i - \theta_0 \to \mathcal{N}(0, \frac{1}{M}I^{-1}(\theta_0))$, then $\bar{\theta} - \theta_0 \to \mathcal{N}(0, \frac{1}{N}I^{-1}(\theta_0))$ and $\bar{\theta} - \hat{\theta} = O_p(\frac{1}{\sqrt{N}})$.

Proof: Based on the above lemmas, we have

$$\sqrt{N}(\bar{\theta} - \theta_0) = \frac{1}{\sqrt{K}}\sum_{i=1}^{K}\sqrt{M}(\hat{\theta}_i - \theta_0) \to \mathcal{N}(0, I^{-1}(\theta_0))$$

$$||\bar{\theta} - \hat{\theta}|| \leq ||\bar{\theta} - \theta_0|| + ||\hat{\theta} - \theta_0|| = O_P(\frac{1}{\sqrt{N}})$$

Theorem 1: If the Assumptions 1 -4 holds, then as $N \to \infty$ and $M \to \infty$,

$$\left|\frac{1}{K}\sum_{i=1}^{K}\pi_i(\theta - \bar{\theta} + \hat{\theta}_i|\mathcal{X}_i) - \pi(\theta|\mathcal{X})\right|_{TV} \to 0$$

$$\left|\frac{1}{K}\sum_{i=1}^{K}\pi_i(\theta - \bar{\theta}^* + \theta_i^*|\mathcal{X}_i) - \pi(\theta|\mathcal{X})\right|_{TV} \to 0$$

Proof:

$$\left|\frac{1}{K}\sum_{i=1}^{K}\pi_i(\theta - \bar{\theta} + \hat{\theta}_i|\mathcal{X}_i) - \pi(\theta|\mathcal{X})\right|_{TV}$$
$$\leq \frac{1}{K}\sum_{i=1}^{K}\left|\pi_i(\theta - \bar{\theta} + \hat{\theta}_i|\mathcal{X}_i) - \mathcal{N}(\theta; \bar{\theta}, \frac{1}{\sqrt{N}}I^{-1}(\theta_0))\right|_{TV}$$
$$+ \left|\mathcal{N}(\theta; \bar{\theta}, \frac{1}{\sqrt{N}}I^{-1}(\theta_0)) - \mathcal{N}(\theta; \hat{\theta}, \frac{1}{\sqrt{N}}I^{-1}(\theta_0))\right|_{TV}$$
$$+ \left|\mathcal{N}(\theta; \hat{\theta}, \frac{1}{\sqrt{N}}I^{-1}(\theta_0)) - \pi(\theta|\mathcal{X})\right|_{TV}$$
$$\to 0$$

$$
\begin{aligned}
&\left|\pi(\theta-\bar{\theta}^*+\theta_i^*|\mathcal{X}_i)-\pi_i(\theta-\bar{\theta}+\hat{\theta}_i|\mathcal{X}_i)\right|_{TV}\\
&\leq\left|\pi(\theta-\bar{\theta}^*+\theta_i^*|\mathcal{X}_i)-\mathcal{N}(\theta;\bar{\theta}^*-\theta_i^*+\hat{\theta}_i,\frac{1}{N}I^{-1}(\theta_0))\right|_{TV}\\
&+\left|\mathcal{N}(\theta;\bar{\theta}^*-\theta_i^*+\hat{\theta}_i,\frac{1}{N}I^{-1}(\theta_0))-\mathcal{N}(\theta;\bar{\theta},\frac{1}{N}I^{-1}(\theta_0))\right|_{TV}\\
&+\left|\mathcal{N}(\theta;\bar{\theta},\frac{1}{N}I^{-1}(\theta_0))-\pi_i(\theta-\bar{\theta}+\hat{\theta}_i|\mathcal{X}_i)\right|_{TV}\\
&\to 0
\end{aligned}
$$

$$
\begin{aligned}
&\left|\frac{1}{K}\sum_{i=1}^{K}\pi_i(\theta-\bar{\theta}^*+\theta_i^*|\mathcal{X}_i)-\pi(\theta|\mathcal{X})\right|_{TV}\\
&\leq\frac{1}{K}\sum_{i=1}^{K}\left|\pi_i(\theta-\bar{\theta}^*+\theta_i^*|\mathcal{X}_i)-\pi_i(\theta-\bar{\theta}+\hat{\theta}_i|\mathcal{X}_i)\right|_{TV}\\
&+\left|\frac{1}{K}\sum_{i=1}^{K}\pi_i(\theta-\bar{\theta}+\hat{\theta}_i|\mathcal{X}_i)-\pi(\theta|\mathcal{X})\right|_{TV}\\
&\to 0
\end{aligned}
$$

Bibliography

Angelino, E., Johnson, M. J., Adams, R. P., et al. (2016). Patterns of scalable Bayesian inference. *Foundations and Trends® in Machine Learning*, 9(2-3):119–247.

Bardenet, R., Doucet, A., and Holmes, C. (2014). Towards scaling up Markov chain Monte Carlo: an adaptive subsampling approach. In *International Conference on Machine Learning*, pages 405–413.

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.

Chen, T., Fox, E., and Guestrin, C. (2014). Stochastic gradient Hamiltonian Monte Carlo. In *International Conference on Machine Learning*, pages 1683–1691.

Gelman, A., Vehtari, A., Jylänki, P., Robert, C., Chopin, N., and Cunningham, J. P. (2014). Expectation propagation as a way of life. *arXiv preprint arXiv:1412.4869*.

Ghosh, J. K., Delampady, M., and Samanta, T. (2007). *An introduction to Bayesian analysis: theory and methods*. Springer Science & Business Media.

Kaw, A., Kalu, E., and Ngyen, D. (2008). Numerical Methods with Applications.

Korattikara, A., Chen, Y., and Welling, M. (2014). Austerity in MCMC land: Cutting the Metropolis-Hastings budget. In *International Conference on Machine Learning*, pages 181–189.

Minsker, S., Srivastava, S., Lin, L., and Dunson, D. (2014). Scalable and robust Bayesian inference via the median posterior. In *International Conference on Machine Learning*, pages 1656–1664.

Neiswanger, W., Wang, C., and Xing, E. (2013). Asymptotically exact, embarrassingly parallel MCMC. *arXiv preprint arXiv:1311.4780*.

Robert, C. P. and Casella, G. (2004). Monte Carlo statistical methods. *Springer New York*.

Scott, S. L., Blocker, A. W., Bonassi, F. V., Chipman, H. A., George, E. I., and McCulloch, R. E. (2016). Bayes and big data: The consensus Monte Carlo algorithm. *International Journal of Management Science and Engineering Management*, 11(2):78–88.

Srivastava, S., Cevher, V., Dinh, Q., and Dunson, D. (2015). WASP: Scalable Bayes via barycenters of subset posteriors. In *Artificial Intelligence and Statistics*, pages 912–920.

Wang, X. and Dunson, D. B. (2013). Parallelizing MCMC via Weierstrass sampler. *arXiv preprint arXiv:1312.4605*.

Wang, X., Guo, F., Heller, K. A., and Dunson, D. B. (2015). Parallelizing MCMC with random partition trees. In *Advances in neural information processing systems*, pages 451–459.

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.

Résumé

Cette thèse porte sur l'accélération des algorithmes de Monte Carlo par chaîne de Markov pour le calcul Bayésien; les contributions consistent à concevoir de nouveaux échantillonneurs MCMC et à adapter l'algorithme MCMC classique à des données massives.

En ne mettant à jour qu'une seule coordonnée du paramètre entre deux temps d'événement, nous dérivons un nouvel échantillonneur MCMC non-réversible, à temps continu, de type de Gibbs, comme l'échantillonneur de coordonnées, et nous montrons empiriquement son accélération sur échantillonneur de zigzag. En outre, nous montrons que la chaîne de Markov induite par l'échantillonneur de coordonnées présente une ergodicité géométrique pour les distributions qui se désintègrent au moins aussi vite qu'une distribution exponentielle et tout au plus aussi rapidement qu'une distribution Gaussienne.

En gardant la qualité de mouvement à travers les itérations inchangées, nous dérivons une variante efficace de l'Hamiltonien Monte Carlo pour recycler les propositions intermédiaires sur le chemin du leapfrog. En remplaçant la dynamique de transition déterministe dans un échantillonneur de bouncy particle par un dynamique aléatoire, nous surmontons le problème de réductibilité dont souffre l'échantillonneur de bouncy particle canonique. De plus, en incorporant des forêts aléatoires dans des méthodes de diviser pour régner, nous réduisons MCMC pour les données massives et montrons que cet algorithme fonctionne bien, dans les configurations à faible dimension, du cas Gaussien au cas fortement non Gaussien, à cause de aide de forêt aléatoire et de sous-postérieurs à l'échelle.

Mots Clés

Chaîne de Markov Monte Carlo, processus de Markov déterministe par morceaux, diviser pour régner, données massives

Abstract

This thesis focuses on accelerating Markov chain Monte Carlo algorithms for Bayesian computation; the contributions consist of devising novel MCMC samplers and scaling classic MCMC algorithm to massive data.

Through updating only one coordinate of the parameter between two event times, we derive a novel non-reversible, continuous-time, Gibbs-like MCMC sampler, Coordinate Sampler, and show its acceleration over zigzag sampler empirically. Furthermore, we prove that the Markov chain induced by coordinate sampler exhibits geometrical ergodicity for distributions which tails decay at least as fast as an exponential distribution and at most as fast as a Gaussian distribution.

By keeping the momentum across iterations unchanged, we derive an efficient variant of Hamiltonian Monte Carlo to recycle the intermediate proposals along leapfrog path. By replacing the deterministic transition dynamic in bouncy particle sampler with a random one, we overcome the reducibility problem suffered from by canonical bouncy particle sampler. Besides, by embedding random forest in divide-and-conquer methods, we scale MCMC for big data and show that this novel algorithm works well, in low-dimensional setting, from Gaussian case to strongly non Gaussian case, and to model misspecification with the help of random forest and scaled sub-posteriors.

Keywords

Markov chain Monte Carlo, piecewise deterministic Markov process, divide-and-conquer, big data