INSTITUT POLYTECHNIQUE DE PARIS

NNT : 20XXIPPAXXXX

Thèse de doctorat

ÉCOLE POLYTECHNIQUE

IP PARIS

Nouvelles Méthodes Variationnelles pour l'inférence et l'apprentissage

Thèse de doctorat de l'Institut Polytechnique de Paris
préparée à l'École polytechnique

École doctorale n°574 École doctorale de mathématiques Hadamard (EDMH)
Spécialité de doctorat : Mathématiques appliquées

Thèse présentée et soutenue à (TBA), le (TBA), par

ACHILLE THIN

Composition du Jury :

Gersende Fort Professeur, Institut de Mathématiques de Toulouse (UMR 5219)	Examinateur
Matthieu Jonckheere Directeur de recherche, LAAS CNRS	Rapporteur
Stéphane Mallat Professeur, École Normale Supérieure (UMR 8548)	Rapporteur
Marylou Gabrié Professeur assistant, École polytechnique (UMR 7641)	Examinateur
Arnaud Doucet Professeur, University of Oxford	Directeur de thèse
Éric Moulines Professeur, École polytechnique (UMR 7641)	Directeur de thèse

Contents

I Introduction **9**

1 Introduction et motivation **11**
1.1 Introduction générale . . . 11
1.2 Inférence bayésienne et réseaux de neurones bayésiens . . . 11
1.2.1 Introduction à l'inférence bayésienne . . . 11
1.2.2 Réseaux de neurones bayésiens . . . 13
1.3 Modèles génératifs . . . 14
1.3.1 Modèles fondés sur l'énergie et comment les apprendre . . . 14
1.3.2 Modèles à variables latentes . . . 15
1.3.3 Auto Encodeurs Variationnels . . . 17
1.4 Introduction aux chaînes de Markov . . . 20
1.5 Conclusion et plan . . . 21
1.6 Résumé des contributions . . . 22
1.6.1 Méthodes d'échantillonnages et de simulation . . . 22
1.6.2 Modèles génératifs . . . 24
1.6.3 Inférence approchée en apprentissage profond bayésien . . . 25

2 Introduction and motivation **27**
2.1 General introduction . . . 27
2.2 Bayesian Inference . . . 27
2.2.1 Bayesian neural networks . . . 29
2.3 Generative modelling . . . 29
2.3.1 Energy based models and how to learn them . . . 30
2.3.2 Latent variable models . . . 31
2.3.3 Variational Auto Encoders . . . 32
2.4 Introduction to Markov chain Monte Carlo methods . . . 35
2.5 Conclusion and plan . . . 36
2.5.1 Contributions . . . 37
2.5.2 Sampling and simulation methods . . . 38
2.5.3 Generative models . . . 40
2.5.4 Application to Bayesian Deep Learning: Efficient Approximate Inference with Gaussian Stochastic Weight Averaging . . . 41

3 General Background **43**
3.1 Classical estimators of normalizing constants . . . 43
3.2 High dimensional simulation and sampling techniques . . . 46
3.3 Generative models and approximate simulation . . . 49

II Contributions: Simulation and Sampling methods **67**

4 Non-reversible MCMC **69**
4.1 Introduction . . . 69
4.2 (π, S)-reversibility and the Generalized MH rule . . . 71
4.2.1 Generalized Metropolis-Hastings . . . 71
4.2.2 GMH for particular proposal maps . . . 72
4.3 Applications and examples . . . 74
4.3.1 Generalized Hamiltonian Dynamics . . . 74
4.3.2 Lifted kernels . . . 75
4.4 Notations, definitions and general Markov chain theory . . . 78
4.5 Standard reversible MH . . . 79
4.6 Proofs . . . 79
4.6.1 Proof of (4.4) . . . 79
4.6.2 Proof of Proposition 3 . . . 80
4.6.3 Proof of Theorem 4 . . . 81
4.6.4 Checking the GMH rule (4.7) . . . 81
4.6.5 Expressions for a and b . . . 81
4.6.6 Applications of (4.7): case with densities . . . 82
4.6.7 Proof of Theorem 6 . . . 82
4.6.8 Proofs of (4.13) and (4.14) . . . 84
4.6.9 Proof of (4.17) . . . 85
4.7 Proofs . . . 86
4.7.1 Generalized Hamiltonian Monte Carlo algorithms . . . 86
4.7.2 Proof of (4.28) . . . 92
4.7.3 Implementation details of Example 9 . . . 93
4.7.4 Proof of Lemma 10 . . . 93
4.7.5 Lifted acceptance probability with deterministic proposals . . . 94
4.7.6 L2HMC Algorithms . . . 95
4.8 Experiments . . . 99

5 NEO: Non Equilibrium Sampling **103**
5.1 Introduction . . . 103
5.2 NEO-IS algorithm . . . 104
5.3 NEO-MCMC algorithm . . . 107
5.4 Continuous-time version of NEO and NEIS . . . 110
5.5 Experiments and Applications . . . 110
5.6 Conclusion . . . 114
5.7 Proofs . . . 115
5.7.1 Additional notation . . . 115
5.7.2 Proof of (5.3) . . . 115
5.7.3 Proof of Theorem 29 . . . 115
5.7.4 Proof of Theorem 30 . . . 115
5.7.5 Proof of Lemma 31 . . . 117
5.7.6 Proofs of NEO MCMC sampler . . . 118
5.8 Continuous-time limit of NEO and NEIS . . . 120
5.8.1 Proof for the continuous-time limit . . . 120
5.8.2 NEIS algorithm after [RV19] . . . 125
5.8.3 NEO with exit times . . . 126
5.9 Iterated SIR . . . 128
5.10 Additional Experiments . . . 129
5.10.1 Normalizing constant estimation . . . 129

5.10.2 Gibbs inpainting . . . 130
5.11 NEO and VAEs . . . 130
5.11.1 Log-likelihood estimation . . . 130
5.11.2 Definition of a NEO-VAE . . . 132

6 Ex^{2}MCMC: Sampling through Exploration Exploitation 135
6.1 Introduction . . . 135
6.2 Ex^{2}MCMC . . . 136
6.2.1 From Importance Sampling to Sampling Importance Resampling . . . 136
6.2.2 From SIR to iterated Sampling Importance Resampling (i-SIR) . . . 136
6.2.3 Dependent proposals for i-SIR and Ex^{2}MCMC algorithms . . . 138
6.2.4 Dependent Gaussian proposals . . . 140
6.2.5 Related Work . . . 141
6.3 Adaptive Ex^{2}MCMC algorithm . . . 141
6.4 Experiments . . . 142
6.4.1 Sampling experiments . . . 142
6.4.2 Sampling from GAN as Energy-based model (EBM) . . . 144
6.5 Conclusions . . . 145
6.6 Sampling GANs as energy-based model on CIFAR-10 . . . 147
6.6.1 DC-GAN . . . 147
6.6.2 SN-GAN . . . 147
6.7 Proofs . . . 148
6.7.1 Proof of Lemma 46 . . . 149
6.7.2 Proof of Theorem 48 . . . 149
6.7.3 Proof of Theorem 49 . . . 152
6.7.4 Proof of Theorem 50 . . . 154
6.8 Metropolis-Adjusted Langevin rejunevation kernel . . . 154
6.9 Technical lemmas for Metropolis-Adjusted Langevin kernel . . . 156
6.10 Algorithms . . . 160
6.11 Numerical experiments . . . 160
6.11.1 Metrics . . . 161
6.11.2 Normalizing flow RealNVP . . . 161
6.11.3 Adaptive strategy for tuning the stepsize in the MALA algorithm . . . 163
6.11.4 High-dimensional Gaussian distribution sampling . . . 163
6.11.5 Mixture of 25 Gaussian distributions in $2d$ . . . 163
6.11.6 Distributions with complex geometry . . . 163
6.11.7 Bayesian Logistic regression . . . 164
6.11.8 Mixture of two Gaussian distributions . . . 165
6.11.9 Allen-Cahn equation . . . 165
6.11.10 Sampling from Ill-Conditioned Gaussian distribution . . . 167
6.11.11 Sampling from GAN as an Energy-Based Model . . . 167
6.11.12 GANs as energy-based models: artificial datasets . . . 167
6.11.13 Sampling GANs as energy-based model on CIFAR-10. . . . 168

7 Approximate Inference in Bayesian Deep Learning 169

III Contributions: Generative models 171

8 MetFlow: MCMC & VI 173
8.1 Introduction . . . 173
8.2 A New Combination Between VI and MCMC . . . 175

8.2.1 Basics of Metropolis-Hastings . . . 175
8.2.2 Variational Inference Meets Metropolis-Hastings . . . 175
8.3 MetFlow: MCMC and Normalizing Flows . . . 178
8.4 Related Work . . . 180
8.5 Experiments . . . 180
8.5.1 Synthetic data. Examples of sampling. . . . 181
8.5.2 Deep Generative Models . . . 181
8.6 Conclusions . . . 184
8.7 Proofs . . . 185
8.7.1 Proof of Proposition 65 . . . 185
8.7.2 Proof of Proposition 66 . . . 185
8.7.3 Proof of Corollary 68 . . . 186
8.7.4 Checking the Assumption of Lemma 64 for RWM and MALA algorithms . . . 187
8.8 Reparameterization trick and estimator of the gradient . . . 187
8.8.1 Expression for the reparameterization trick . . . 187
8.8.2 Unbiased estimator for the gradient of the objective . . . 188
8.8.3 Extension to Hamiltonian Monte-Carlo . . . 189
8.9 Optimization Procedure . . . 191
8.9.1 Optimization in the general case . . . 191
8.9.2 Optimization for *MetFlow* . . . 191
8.10 Experiments . . . 192
8.10.1 Mixture of Gaussians: Additional Results . . . 193
8.10.2 Funnel distribution . . . 197
8.10.3 Real-world inference - MNIST . . . 198
8.10.4 Additional setting of experiments . . . 199

9 Monte Carlo Variational Auto Encoders 203
9.1 Introduction . . . 203
9.2 Variational Inference via Sequential Importance Sampling . . . 204
9.2.1 SIS estimator . . . 204
9.2.2 AIS estimator . . . 205
9.2.3 SIS-ELBO using unadjusted Langevin . . . 206
9.3 Variational Inference via Annealed Importance Sampling . . . 207
9.3.1 Differentiating Markov kernels . . . 207
9.3.2 Differentiable AIS-based ELBO . . . 209
9.4 Experiments . . . 211
9.4.1 Methods and practical guidelines . . . 211
9.4.2 Toy 2D example and Probabilistic Principal Component Analysis . . . 212
9.4.3 Numerical results for image datasets . . . 214
9.5 Discussion . . . 216
9.6 Notations and definitions . . . 218
9.7 Experiences . . . 218
9.7.1 Toy example . . . 218
9.7.2 Probabilistic Principal Component Analysis . . . 218
9.7.3 Additional experimental results . . . 219
9.8 Proofs . . . 219
9.8.1 Proof of SIS and AIS Identities . . . 219
9.8.2 Proof of (9.14) . . . 223
9.8.3 Proof of Lemma 71 . . . 224
9.9 ELBO AIS . . . 224
9.9.1 Construction of the control variates . . . 224
9.9.2 Discussion of [WKN20b] . . . 225

A Appendix and supplementary material **249**
A.1 Notations, definitions and general Markov chain theory 249
A.1.1 Proof for the AIS estimator 250

Part I

Introduction

Chapter 1

Introduction et motivation

1.1 Introduction générale

L'objectif de l'apprentissage statistique est d'inférer des modèles complexes et en grande dimension (images, sons, ...). En particulier, nous nous concentrerons dans ce travail sur la modélisation statistique et probabiliste de ces données (souvent appelé apprentissage machine probabiliste – Probabilistic Machine Learning).

L'apprentissage machine probabiliste introduit des modèles statistiques, et donc des distributions de probabilité, qui ne sont souvent connues qu'à une constante de normalisation près. Cette constante de normalisation, inconnue donc, est aussi appelée énergie libre, par analogie à la physique statistique, ou évidence (de l'anglais evidence). En réalité, il s'agit plus précisément souvent d'une vraisemblance marginale, "preuve" du modèle, d'où le nom evidence en anglais. Ce type de modèles est utilisé pour de nombreuses tâches : de la sélection de modèles, de la quantification d'incertitude, de la génération de données, où pour inférer et traiter des modèles à variables latentes ou manquantes. Les problèmes typiques qui apparaissent dans ce cadre sont le calcul ou l'estimation de cette constante de normalisation, afin d'effectuer par exemple de la sélection de modèle, l'échantillonnage – c'est-à-dire produire des réalisations – de cette distribution, et d'une manière générale d'intégrales par rapport à cette distribution afin de calculer des quantités intéressantes, voire effectuer des prédictions. Les trois problèmes d'apprentissage statistiques considérés dans cette thèse sont

- L'estimation d'incertitude par inférence bayésienne,
- Le Auto Encodeurs Variationnels,
- les Modèles fondés sur l'énergie.

Bien que ces tâches soient très différentes, nous verrons dans l'introduction que les problèmes rencontrés ont de nombreuses similarités que nous traitons dans cette thèse.

1.2 Inférence bayésienne et réseaux de neurones bayésiens

1.2.1 Introduction à l'inférence bayésienne

L'inférence bayésienne est particulièrement utilisée pour le traitement d'incertitudes dans de nombreux problèmes d'apprentissage statistique. L'idée fondamentale est de considérer les paramètres du modèle comme des variables aléatoires dont la distribution est modifiée en fonction des observations (mise à jour de la distribution *a priori* à la distribution *a posteriori* combinant la connaissance initiale du problème et les données expérimentales).

Les avantages de cette méthode sont de pouvoir quantifier de manière explicite les incertitudes du modèle, puisque la réponse du modèle est elle-même aléatoire.

Prenons premièrement l'exemple d'une régression linéaire probabiliste. On modélise donc une réponse $y \in \mathbb{R}$ en fonction de variables explicatives $x \in \mathbb{R}^d$ et de paramètres $\theta \in \mathbb{R}^d$, en spécifiant une vraisemblance Gaussienne :

$$p(y \mid x, \theta) = (2\pi\sigma^2)^{-1/2} \mathrm{e}^{-(y - x^T\theta)/(2\sigma^2)} ,$$

où l'on peut supposer la variance σ^2 connue. De la même faccon, une réponse $y \in \{0, 1\}$ peut être modélisée avec une régression logistique, en spécifiant la vraisemblance

$$p(y \mid x, \theta) = (1 + \mathrm{e}^{-x^T\theta})^{-y} (1 + \mathrm{e}^{x^T\theta})^{y-1} .$$

Dans les deux cas, étant donné un ensemble d'observations i.i.d. $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$, on peut écrire la vraisemblance $p(\mathcal{D} \mid \theta) = \prod_{i=1}^N p(y_i; x_i, \theta)$. La spécification bayésienne du modèle requiert le choix d'une loi *a priori* de densité p_0 sur les paramètres θ du modèle, qui représente l'information disponible sur le modèle avant de recueillir les données expérimentales. L'application de la règle de Bayes permet de spécifier la distribution *a posteriori* des paramètres θ comme

$$p(\theta \mid \mathcal{D}) = \frac{p_0(\theta) p(\mathcal{D} \mid \theta)}{\mathrm{Z}} , \quad \mathrm{Z} = \int p_0(\theta) p(\mathcal{D} \mid \theta) \mathrm{d}\theta \in (0, \infty) , \tag{1.1}$$

où le terme $\mathrm{Z} = m(\mathcal{D}) = \int p_0(\theta) p(\mathcal{D} \mid \theta) \mathrm{d}\theta$ est appelé l'évidence du modèle choisi (il ne dépend que des observations et du modèle spécifié). En général, l'évidence n'est pas calculable explicitement et échantillonner la distribution *a posteriori* requiert l'utilisation de méthodes de Monte Carlo (méthodes de Monte Carlo par chaînes de Markov, méthodes de Monte Carlo Séquentielles, ou des méthodes approchées comme échantillonnage préférentiel et rééchantillonnage). Échantillonner cette distribution *a posteriori* permet en particulier d'effectuer des prédictions et des moyennes prédictives prenant en compte l'incertitude sur les paramètres du modèle. En effet, une clef de l'inférence bayésienne en apprentissage statistique est le calcul et l'approche de la distribution prédictive

$$p(y^* \mid x^*, \mathcal{D}) = \int p(y^* \mid x^*, \theta) p(\theta \mid \mathcal{D}) \mathrm{d}\theta , \tag{1.2}$$

qui moyenne les prédictions sur tous les paramètres θ distribués selon la distribution *a posteriori*.

Un autre problème central en inférence bayésienne est le choix de modèle. En effet, comme vu plus haut, la constante de normalisation de la distribution *a posteriori* peut être vue comme une évidence, "preuve" du modèle. Considérons par exemple deux familles de modèles paramétriques $\mathcal{M}_1$ et $\mathcal{M}_2$ pour le même jeu de données $\mathcal{D}$, associés respectivement aux paramètres $\theta_1 \in \mathbb{R}^{d_1}$ et $\theta_2 \in \mathbb{R}^{d_2}$, aux distributions *a priori* π_1, π_2 et aux vraisemblances $p_1(\cdot \mid \theta_1), p_2(\cdot \mid \theta_2)$. Alors, on peut comparer les modèles 1 et 2 avec le facteur de Bayes B_{12} donné par

$$B_{12} = \frac{m(\mathcal{D} \mid \mathcal{M}_1) \pi_0(\mathcal{M}_1)}{m(\mathcal{D} \mid \mathcal{M}_2) \pi_0(\mathcal{M}_2)} , \tag{1.3}$$

où π_0 est une distribution *a priori* sur les modèles 1 et 2, et

$$m(\mathcal{D} \mid \mathcal{M}_i) = \int \pi_i(\theta_i) p_i(\mathcal{D} \mid \theta_i) \mathrm{d}\theta_i , \quad i \in \{1, 2\} .$$

Ce ratio B_{12} est clef pour le choix de modèle entre 1 (si $B_{12} > 1$) et 2 ($B_{12} < 1$). Cependant, le calcul de ce ratio ne peut se faire qu'avec les constantes de normalisation des distributions *a posteriori* des modèles 1 et 2, ou une estimation fiable et consistante de ces constantes. Pour synthétiser, trois problèmes majeurs apparaissent ici :

- L'estimation fiable de constantes de normalisation,
- L'échantillonage de lois dont la constante de normalisation est inconnue,
- Le calcul d'intégrale par rapport à une loi dont la constante de normalisation est inconnue.

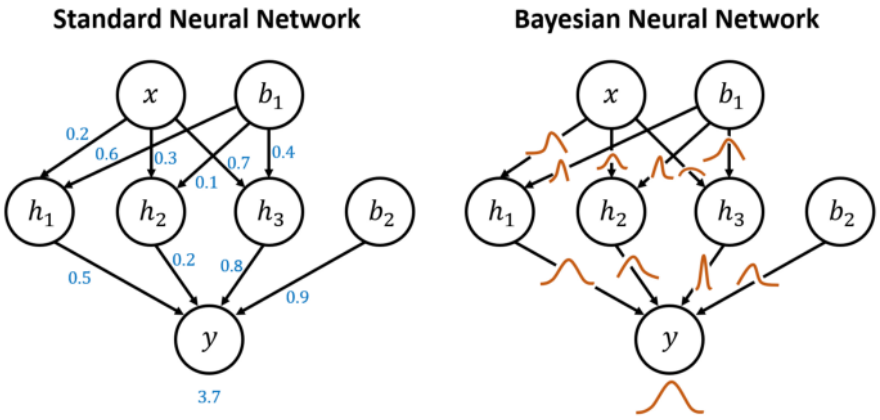


Figure 1.1: Schéma de la différence entre une réseau de neurones classique et un réseau de neurones bayésien. Source : `https://towardsdatascience.com/why-you-should-use-bayesian-neural-network-aaf76732c150`

1.2.2 Réseaux de neurones bayésiens

Les réseaux de neurones, et l'apprentissage profond, constituent aujourd'hui l'état de l'art dans une grande variété de tâches, de la régression à la classification en apprentissage supervisé par exemple, dans des domaines aussi distincts que la vision par ordinateur, la reconnaissance de langage ou le traitement automatique du langage naturel [GBC16; LBH15; MLY17; KP18; Vou+18]. Avec les nouveaux enjeux de scalabilité et d'industrialisation, les enjeux d'incertitudes sont au coeur des problématiques actuelles, et en particulier l'inférence bayésienne et son application aux réseaux de neurones pourrait apporter une solution fiable. En effet, il est connu que les réseaux de neurones, en particulier profonds, sont trop confiants en leur prédiction dans certaines tâches [NYC15; DK16; HAB19; Gaw+21]. De nombreux auteurs ont suggéré une approche bayésienne [Gal16; KG17; McA+17] pour obtenir une estimation d'incertitude plus fiable. Un réseau de neurones est une fonction complexe G_θ non linéaire, paramétrée par un vecteur de poids θ souvent de très haute dimension. Les paramètres du réseau sont typiquement les poids synaptiques et les biais des neurones formels qui constituent les éléments de base du réseau.

Étant donné un jeu de données i.i.d. $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$, un modèle de régression peut donc être construit similairement à l'exemple précédent, en définissant la vraisemblance

$$p(y \mid x, \theta) = (2\pi\sigma^2)^{-1/2} e^{-(y - G_\theta(x))/(2\sigma^2)} ,$$

et similairement pour un modèle de classification. En définissant une distribution *a priori* p_0, typiquement Gaussienne standard sur les poids du réseau G_θ, on peut effectuer une inférence bayésienne et calculer la distribution *a posteriori* des poids du réseau, donnée par

$$p(\theta \mid \mathcal{D}) \propto p_0(\theta) p(\mathcal{D} \mid \theta) .$$

On peut aussi faire un lien entre cette distribution *a posteriori* et un objectif plus "classique" d'apprentissage statistique, en écrivant l'objectif

$$\mathcal{L}(\theta; \mathcal{D}) = -\log p_0(\theta) - \sum_{i=1}^{N} \log p(y_i; x_i, \theta) . \tag{1.4}$$

Ici apparaît donc encore la nécessité d'échantillonner selon cette distribution, voire d'en calculer une constante de normalisation.

D'ailleurs, les réseaux de neurones bayésiens profonds ayant donc des paramètres vivant dans un espace d'état de très haute dimension (plusieurs centaines de millions pour certains ResNets classiques !), les méthodes d'inférence bayésienne basées sur des méthodes de simulation de type Monte Carlo par

chaîne de Markov classiques ne sont plus directement applicables (le temps de mélange des chaînes étant très négativement par la dimension de l'espace d'état du modèle, [Izm+21]). D'autre part, le calcul exact des itérations de ces chaînes est souvent prohibitif et nécessite des adaptations [WT11; CFG14; BDH17; Cha+18; BDM18], d'où l'intérêt de développer des méthodes approchées, passant à l'échelle plus facilement [AKW12; GG16; Kha+18; Izm+20; Dus+20; FSG20; Foo+20; Dax+20].

1.3 Modèles génératifs, Modèles fondés sur l'énergie et Auto encodeurs variationnels

Les modèles génératifs sont au coeur de l'apprentissage statistique probabiliste. Étant donné des observations $\mathcal{D} = \{x_i\}_{i=1}^N$, les modèles génératifs consistent à l'apprentissage de la distribution à densité $x \mapsto p^*(x)$ dont les données $\mathcal{D}$ sont issues. En particulier, le modèle génératif spécifie une classe de fonction $x \mapsto p_\theta(x)$, paramétrée par θ, et estime le paramètre θ en maximisant la vraisemblance des observations.

1.3.1 Modèles fondés sur l'énergie et comment les apprendre

Une première classe de modèles génératifs sont les modèles fondés sur l'énergie (Energy Based Models – EBM). Ce type de modèle, inspiré des distribution de Boltzmann Gibbs en physique statistique, peut être défini comme

$$p_\theta(x) = e^{-E_\theta(x)}/\mathrm{Z}_\theta \ , \quad \mathrm{Z}_\theta = \int e^{-E_\theta(x)} \mathrm{d}x \ ,$$

où E_θ est la sortie (scalaire) d'un réseau de neurones prenant en entrée une observation x. Ce modèle est très flexible puisqu'il ne requiert que la spécification de la log-probabilité non normalisée, appelée énergie par analogie avec la physique statistique et permet donc de définir très librement des densités de probabilités sur des espaces d'états potentiellement très complexes.

Choix des paramètres En analyse statistique, le critère de choix des paramètres est souvent donné par le maximum de vraisemblance. Étant donné un modèle statistique $p(x;\theta) = p_\theta(x)$ et des observations i.i.d. $x_1, \dots, x_n \in \mathbb{R}^p$ selon p_θ, la méthode du maximum de vraisemblance consiste à estimer le paramètre θ en maximisant la vraisemblance des observations $p(x_1, \dots, x_n; \theta)$, *i.e.*

$$\hat{\theta}_n^\star = \arg\max_{\theta \in \Theta} n^{-1} \sum_{i=1}^n \log p_\theta(X_i) = \arg\max_{\theta \in \Theta} L_n(\theta) \ . \tag{1.5}$$

Nous sommes donc ici confrontés à un problème d'optimisation. Cependant, nous avons plusieurs contraintes pour les modèles considérés dans cette thèse. Avec l'avancée de l'apprentissage statistique et en particulier l'usage courant des réseaux de neurones profonds, les modèles génératifs sont eux aussi devenus profonds, avec un nombre de paramètres de l'ordre de la centaine de milliers, voire du million. De plus, le nombre d'observations est très large. Dans le cas de l'application à MNIST, le jeu de données comporte 50 000 à 60 000 images. Enfin les données en elle mêmes sont de grande dimension, d'images simples (MNIST, dimension $p = 784$ par exemple – le nombre de pixels de l'image) à complexes. Toutes ces contraintes sont autant de pression sur le temps de calcul et l'énergie requise pour apprendre ces modèles, de faccon à passer à l'échelle, à la fois dans la dimension du modèle et dans le nombre d'observations. En général, cela passe par des solutions d'optimisation par gradient stochastique, c'est-à-dire que l'on va effectuer une montée de gradient pour maximiser la quantité $n^{-1} \sum_{i=1}^n \log p_\theta(x_i)$ en θ, en approchant à chaque étape le gradient de cette quantité par un estimateur

(non biaisé idéalement) de cette quantité. Pour les EBM, nous pouvons écrire

$$\begin{aligned} L_n(\theta) = &-n^{-1}\sum_{i=1}^{n}\log p_\theta(x_i)\ , \\ =&n^{-1}\sum_{i=1}^{n}E_\theta(x_i)+\log Z_\theta \\ =&n^{-1}\sum_{i=1}^{n}E_\theta(x_i)+\log\int e^{-E_\theta(x)}\mathrm{d}x \end{aligned}$$

Le gradient par rapport à θ de cette quantité peut donc être donné par

$$\begin{aligned} \nabla_\theta L_n(\theta) &= n^{-1}\sum_{i=1}^{n}\nabla_\theta E_\theta(x_i) - \nabla_\theta\left(\log\int e^{-E_\theta(x)}\mathrm{d}x\right) \\ &= n^{-1}\sum_{i=1}^{n}\nabla_\theta E_\theta(x_i) - \int \nabla_\theta E_\theta(x)e^{-E_\theta(x)}\mathrm{d}x/\mathrm{Z}_\theta\ , \\ &= n^{-1}\sum_{i=1}^{n}\nabla_\theta E_\theta(x_i) - \int \nabla_\theta E_\theta(x)\cdot p_\theta(x)\mathrm{d}x \end{aligned}$$

On voit donc ici deux termes concurrents, à approcher différemment pour l'apprentissage des paramètres θ. Le premier peut être simplement approché par une estimation stochastique, prenant des "mini-batch" des observations. Au lieu de calculer à chaque étape le gradient pour chacun des points $x_i \in \mathcal{D}$, on tire un sous ensemble de taille b (de l'ordre d'une centaine d'observations typiquement) $\mathcal{B}$ de $\mathcal{D}$ et on approche $n^{-1}\sum_{i=1}^{n}\nabla_\theta E_\theta(x_i)$ par l'estimateur $b^{-1}\sum_{x_i\in\mathcal{B}}\nabla_\theta E_\theta(x_i)$. Cela permet alors une optimisation à moindre coût et passant à l'échelle en terme de nombre de données !

En revanche, le second requiert une approximation Monte Carlo d'une intégrale prise par rapport à une distribution de probabilité dont la constante de normalisation Z_θ est inconnue. Une estimation fiable de cette quantité requiert donc ici, comme c'est le cas classiquement en EBM, un échantillonneur efficace en grande dimension. Dans ce cas, la méthode classique consiste à utiliser des algorithmes de type chaîne de Markov Monte Carlo, voir Section 1.4.

1.3.2 Modèles à variables latentes

Récemment, un autre type de modèle a été l'objet de beaucoup d'attention: Les Modèles génératifs profonds à variables latentes introduits par exemple par [Goo+14a; KW13a].

Ces modèles sont construits de la faccon suivante

$$X = F(G_\theta(Z), W)\ ,$$

ou de manière équivalente, $X \sim p_\theta(\cdot \mid Z)$. X est ici l'observation que l'on cherche à modéliser. Supposons par la suite que $X \in \mathbb{R}^p$, Z est la variable latente, spécifiée dans le modèles, sorte d'encodage de l'observation X, et l'on suppose ici $Z \in \mathbb{R}^d$. Les hypothèses de ce type de modèle est que la dimension de Z est plus petite que celle de X, et que Z suit une distribution en général très simple. G_θ est une fonction complexe, non linéaire, lourdement paramétrée par θ. Typiquement G_θ est un réseau de neurones profond. G_θ, enfin, est typiquement appelée décodeur (dans les Auto Encodeurs Variationnels) ou générateur (dans les Réseaux Adverses Génératifs – Generative Adversarial Networks). Enfin, W est un terme de bruit ajouté à notre modèle probabiliste, et F une fonction composant ce terme et la réponse déterministe $G_\theta(Z)$. D'une manière générale, le modèle statistique est donc spécifié par des paramètres θ et le modèle joint

$$p_\theta(x, z) = p_\theta(x \mid z)\cdot p_\theta(z)\ , \tag{1.6}$$

que l'on peut générer en tirant donc successivement $Z \sim p_\theta(Z)$, $X|Z \sim p_\theta(X \mid Z)$. Prenons pour

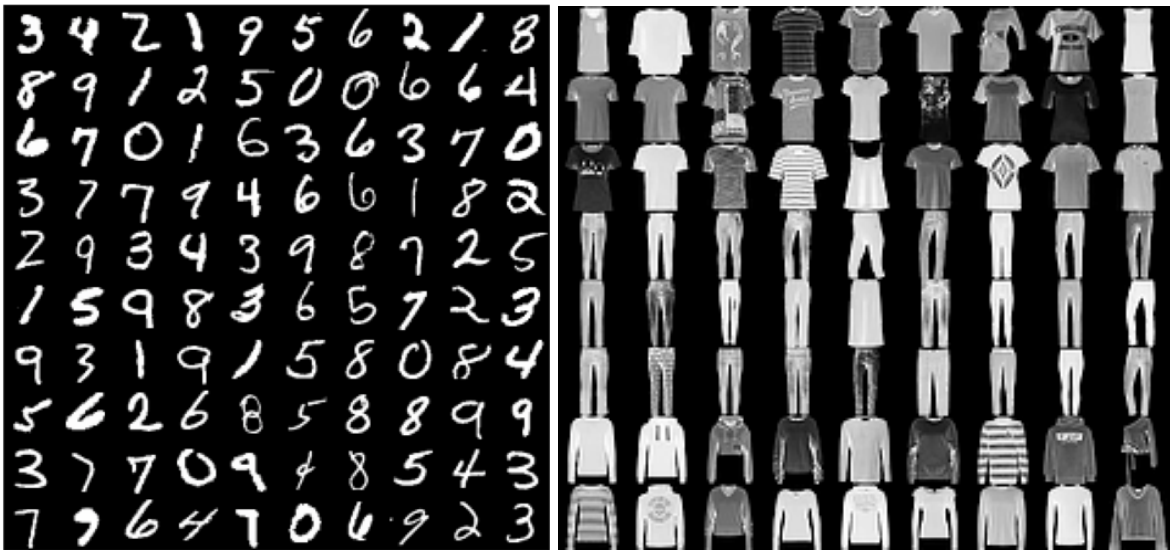


Figure 1.2: Exemples d'observations des datasets MNIST (à gauche) et Fashion MNIST (à droite).

exemple un DLGM sur le jeu de données MNIST (resp. Fashion MNIST), constitué d'images en noir et blanc binarisées de chiffres écrits à la main (resp. d'images de vêtements en noir et blanc binarisées), voir Figure 1.2. Dans ce cas, nous pouvons écrire

$$p_\theta(z) = \mathcal{N}(z; 0, \mathbf{I}) \quad \mathbf{p}(z) = (p_1(z), \dots, p_D(z)) = G_\theta(z)$$
$$\log p_\theta(x \mid z) = \sum_{j=1}^{D} \log p\left(x_j \mid \mathbf{z}\right) = \sum_{j=1}^{D} x_j \log p_j(z) + (1 - x_j) \log\left(1 - p_j(z)\right) \ ,$$

où G_θ est un réseau de neurones prenant en entrée la variable latente Z et renvoyant les paramètres de la distribution de l'observation $X \mid Z$.

Choix des paramètres Si nous appliquons maintenant le critère de maximum de vraisemblance (1.5) aux modèles génératifs à variables latentes, nous pouvons écrire la vraisemblance des observations X comme

$$p_\theta(x) = \int_{\mathbb{R}^d} p_\theta(x \mid z) p_\theta(z) \mathrm{d}z \ . \tag{1.7}$$

Dans ce cas, la vraisemblance à maximiser se trouve donc sous la forme d'une intégrale, complexe (en dimension d), qui n'admet pas d'expression explicite.

Dans le cas des modèles génératifs à variables latentes, on est confrontés à un problème supplémentaire. Le gradient, pour une observation x, $\nabla \log p_\theta(x)$ n'est pas accessible, car il est à exprimer sous la forme d'une intégrale intractable. Il est alors intéressant d'appliquer l'identité de Fisher dans ce cas, qui s'exprime comme ceci

$$\begin{aligned}
\nabla_\theta \log p_\theta(x) &= \int_{\mathbb{R}^d} \frac{\nabla_\theta p_\theta(x, z)}{p_\theta(x)} \mathrm{d}z \\
&= \int_{\mathbb{R}^d} \nabla_\theta \log p_\theta(x, z) \frac{p_\theta(x, z)}{p_\theta(x)} \mathrm{d}z \\
&= \int_{\mathbb{R}^d} \nabla_\theta \log p_\theta(x, z) p_\theta(z|x) \mathrm{d}z \ .
\end{aligned} \tag{1.8}$$

Le gradient de la log-vraisemblance marginale $\nabla \log p_\theta(x)$, intractable, s'exprime alors comme l'intégrale du gradient de la log-vraisemblance jointe $\nabla \log p_\theta(x, z)$ par la distribution conditionnelle de la variable latente z par l'observation x (soit la distribution *a posteriori* de la variable latente étant donné

l'observation). La log-vraisemblance jointe est tractable et peut se calculer facilement dans les modèles spécifiés. Si l'intégrale ci dessus est toujours intractable, il est en revanche facile maintenant de l'estimer par méthode Monte Carlo. En effet, si $Z_1 \dots, Z_m \sim p_\theta(\cdot \mid x)$, un estimateur de $\nabla \log p_\theta(x)$ est donc donné par $m^{-1} \sum_{i=1}^m \nabla \log p_\theta(x, Z_i)$. Afin d'obtenir des échantillons distribués selon $p_\theta(\cdot \mid x) \propto p_\theta(z) p_\theta(x, z)$, on peut utiliser des méthodes MCMC, comme pour les EBM.

Cependant, un problème majeur peut être soulevé ici. En effet, chaque distribution $z \mapsto p_\theta(z \mid x)$ doit être approchée indépendamment, et le nombre d'observations étant grand dans les modèles considérés, le budget computationnel grandit considérablement ! Ce n'est donc pas l'approche choisie, principalement pour des raisons computationnelles, par les Auto Encodeurs Variationnels [KW13a].

1.3.3 Auto Encodeurs Variationnels

Les Auto Encodeurs variationnels (VAE) sont un type de DLGM dont l'apprentissage est basé sur de l'inférence variationnelle. Précisons un peu l'objectif. Afin de pallier à la difficulté de l'estimation du gradient de la log-vraisemblance, les VAE introduise une famille paramétrique de distributions, appelée famille variationnelle, $\mathcal{Q} = \{q_\phi \,,\, \phi \in \Phi\}$ paramétrée par un nouveau paramètre ϕ, et l'on considère alors l'objectif auxiliaire

$$\mathcal{L}(\theta, \phi; x) = \int_{\mathbb{R}^d} \log \left(\frac{p_\theta(x, z)}{q_\phi(z \mid x)} \right) q_\phi(z \mid x) \mathrm{d}z \,. \tag{1.9}$$

Un introduction plus détaillée à l'inférence Variationnelle est donnée en Section 3.1. Cependant, notons ici que l'on peut réécrire cette quantité comme

$$\mathcal{L}(\theta, \phi; x) = \int_{\mathbb{R}^d} \log \left(\frac{p_\theta(z \mid x) p_\theta(x)}{q_\phi(z \mid x)} \right) q_\phi(z \mid x) \mathrm{d}z = \log p_\theta(x) - D_{\mathrm{KL}}(q_\phi(\cdot \mid x \| p_\theta(\cdot \mid x) \,. \tag{1.10}$$

Cet objectif étant toujours inférieur à la vraisemblance marginale $\log p_\theta(x)$, il est donc appelé borne inférieure de l'évidence, ou ELBO (Evidence Lower Bound, en anglais). Si ce nouvel objectif est tractable et optimisable facilement par des méthodes de Monte Carlo, il est différent de la véritable quantité d'intérêt ici $\log p_\theta(x)$. On peut en effet écrire le gradient par rapport à θ comme

$$\nabla_\theta \mathcal{L}(\theta, \phi; x) = \int_{\mathbb{R}^d} \nabla_\theta \log \left(p_\theta(x, z) \right) q_\phi(z \mid x) \mathrm{d}z \,, \tag{1.11}$$

ce qui est proche mais différent de l'expression obtenue avec l'identité de Fisher (1.8).

On peut voir de plus sur l'expression (2.10) que la maximisation de l'ELBO (5.67) optimisera de manière concurrente les deux quantités d'intérêt

- Cela maximisera approximativement la log-vraisemblance marginale $\log p_\theta(x)$.
- Cela minimisera la divergence $D_{\mathrm{KL}}(q_\phi(\cdot \mid x \| p_\theta(\cdot \mid x))$ entre la posterior variationnelle et la vraie posterior.

Un apport critique de ce nouveau type de modèle est donné de plus par "l'amortissement" de ces distributions variationnelles. Comme soulevé précédemment, pour chaque observation x, la distribution *a posteriori* $p_\theta(z \mid x)$ est différente. Ainsi, [KW13a] introduisent une fonction globale $x \mapsto q_\phi(\cdot \mid x) \colon z \mapsto q_\phi(z \mid x)$ en x, où les paramètres ϕ sont donc partagés. Ainsi, le budget computationnel n'augmente pas linéairement en le nombre d'observations ! La figure Section 1.3.3 présente un schéma de l'inférence variationnelle amortie. On appelle ainsi la fonction $x \mapsto q_\phi(\cdot \mid x)$ qui associe à toute observation une distribution dans l'espace latent "encodeur", et symétriquement la fonction $z \mapsto p_\theta(x \mid z)$ qui associe à tout point de l'espace latent une distribution dans l'espace des observations "décodeur".

Si l'optimisation en θ de ce modèle est donc directe avec une approximation par Monte Carlo du gradient, l'optimisation du paramètre ϕ requiert plus d'attention.

En effet, on peut en général écrire le gradient de la fonction

$$\phi \mapsto \int h(x, z) q_\phi(z \mid x) \mathrm{d}z$$

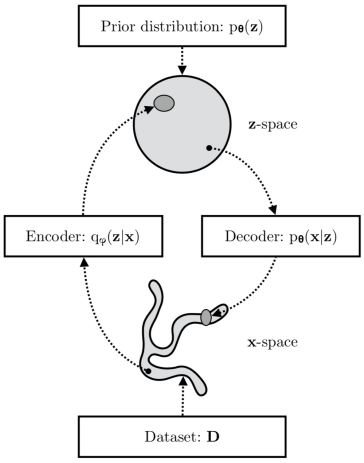


Figure 1.3: Schéma de l'inférence variationnelle amortie des Auto Encodeurs Variationnels, [KW19].

comme

$$\int h(x,z)\nabla \log q_\phi(z|x) q_\phi(z|x)\mathrm{d}z \ ,$$

que l'on peut alors estimer par Monte Carlo. Cependant, cet estimateur basé sur le log-gradient, appelé estimateur REINFORCE, a souvent une variance importante. Afin d'effectuer une optimisation par gradient stochastique plus efficace, on utilise en général une reparamétrisation du gradient. Supposons donc qu'il existe pour tout x un difféomorphisme $\epsilon \mapsto V_{\phi,x}(\epsilon)$ et une loi de densité g (facile à échantillonner), tels que

$$\epsilon \sim \mathrm{g}\,, \quad z = V_{\phi,x}(\epsilon) \sim q_\phi(\cdot \mid x)\ . \tag{1.12}$$

On peut alors reparamétriser l'ELBO de faccon à écrire

$$\mathcal{L}(\theta,\phi;x) = \int \log\left(\frac{p_\theta(x, V_{\phi,x}(\epsilon))}{q_\phi(V_{\phi,x}(\epsilon) \mid x)}\right)\mathrm{g}(\epsilon)\mathrm{d}\epsilon \ , \tag{1.13}$$

où l'intégration est maintenant faite par rapport à une distribution ne dépendant plus des paramètres π, θ à optimiser, ce qui simplifie considérablement l'estimation Monte Carlo. En général, on choisit même directement la famille variationnelle en spécifiant g et $V_{\phi,x}$. Pour l'auto encodeur variationnel classique [KW13a], par exemple, la prior $p_\theta(z)$ est alors g la Gaussienne standard sur $\mathbb{R}^d$, et $p_\theta(x \mid z)$ est la distribution paramétrée par les paramètres $G_\theta(z)$, où G_θ est un réseau de neurones. De la même faccon, $q_\phi(z \mid x) = \mathcal{N}(z; \boldsymbol{\mu}_\phi(x), \mathrm{diag}(\boldsymbol{\sigma}^2_\phi(x)))$, où $\boldsymbol{\mu}_\phi$, $\boldsymbol{\sigma}_\phi$ sont la sortie d'un réseau de neurones de paramètres ϕ, construit typiquement symétriquement à G_θ.On peut alors reparamétriser q_ϕ

$$\begin{aligned}&\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})\\ &z = \boldsymbol{\mu}_\phi(x) + \boldsymbol{\sigma}_\phi(x) \odot \boldsymbol{\epsilon} \sim q_\phi(\cdot \mid x)\ .\end{aligned}$$

Ce modèle est cependant limité, c'est à dire que l'ELBO ne propose pas forcément une approximation satisfaisante de la véritable log-vraisemblance que l'on souhaite optimiser. Une première piste d'amélioration de ce modèle est de rendre plus expressive la famille variationnelle, en proposant plus qu'une Gaussienne à covariance diagonale. Par exemple, une succession de difféomorphismes, appelés flots normalisants, voir Section 3.3, permet d'améliorer le modèle, [RM15b; Kin+16a]. D'autres stratégies sont basées sur de l'inférence variationnelle auxiliaire (AVI), où d'autres variables aléatoires auxiliaires u sont ajoutées au modèle et à la famille variationnelle afin d'en améliorer l'expressivité, [SKW15a; RTB16; Maa+16]. Dans ce cas, on étend la distribution variationnelle en définissant

$$q_\phi(u, z \mid x) = q_\phi(u \mid x) q_\phi(z \mid u, x)\ ,$$

ce qui permet alors de définir en pratique une distribution marginale potentiellement très expressive

$$q_\phi(z \mid x) = \int q_\phi(u, z \mid x)\mathrm{d}u$$

De la même faccon, on définit le modèle génératif joint

$$p_\theta(x, z, u) = p_\theta(u \mid x, z)p_\theta(x, z) \ .$$

On introduit alors un nouvel objectif auxiliaire

$$\mathcal{L}_{\mathrm{AVI}}(\theta, \phi; x) = \int \log \frac{p_\theta(x, z, u)}{q_\phi(z, u \mid x)} q_\phi(z, u \mid x)\mathrm{d}z\mathrm{d}u \leqslant \log p_\theta(x)$$

Notons que l'on peut toujours réécrire

$$\mathcal{L}_{\mathrm{AVI}}(\theta, \phi; x) + \int \left\{ \int \log \left(\frac{q_\phi(u \mid x, z)}{p_\theta(u \mid x, z)} \right) q_\phi(u \mid x, z)\mathrm{d}u \right\} q_\phi(z \mid x)\mathrm{d}z = \mathcal{L}(\theta, \phi; x)$$

Ainsi, en pratique, l'ELBO auxiliaire est encore "plus loin" de la log-vraisemblance, cependant l'expressivité augmentée des distributions variationnelles rendent en général ces modèles lus attractifs, comme présenté dans [SKW15a; RTB16; Maa+16] et dans [Thi+21b].

Enfin, une autre direction [BGS15; Mad+17; DS18] se base sur l'introduction d'autres objectifs basés sur des estimateurs de constante de normalisation. En effet, on peut construire un ELBO à partir de tout estimateur non biaisé de la constante de normalisation $p_\theta(x)$ de la distribution non normalisée $z \mapsto p_\theta(x, z)$. Notons par exemple $\widehat{\mathrm{Z}_x}(z)$ un estimateur non biaisé de $p_\theta(x)$ pour une densité de proposition $q_\phi(\cdot \mid x)$, c'est à dire $\int_{\mathbb{R}^d} \widehat{\mathrm{Z}_x}(z) q_\phi(z \mid x)\mathrm{d}x = p_\theta(x)$. Alors, la quantité

$$\mathcal{L}(\theta, \phi; x) = \int_{\mathbb{R}^d} \log(\widehat{\mathrm{Z}_x}(z)) q_\phi(z \mid x)\mathrm{d}z \leqslant \log p_\theta(x) \ ,$$

est donc bien un ELBO, par l'inégalité de Jensen et la concavité du logarithme. L'Auto Encodeur Variationnel correspond alors au cas où on a simplement $\widehat{\mathrm{Z}_x}(z) = p_\theta(x, z)/q_\phi(z \mid x)$, l'estimateur d'importance à 1 échantillon, qui est évidemment non biaisé. Cependant, on note ici que d'autres estimateurs peuvent être utilisés dans ce cadre ! De plus, des estimateurs plus performants conduisent à de "meilleurs" ELBO, dans le sens de plus proches de la log-vraisemblance. En effet, par un développement de Taylor, on peut écrire

$$\mathcal{L}(\theta, \phi; x) \approx \log p_\theta(x) - \frac{1}{2}\mathrm{var}_{q_\phi(\cdot|x)} \left[\frac{\widehat{\mathrm{Z}_x}(z)}{p_\theta(x)} \right] \ ,$$

voir [Mad+17; DS18] pour plus de détails. En particulier, citons ici l'Auto Encodeur préférentiel (Importance Weighted Auto Encoder – IWAE), qui se base plutôt sur K échantillons d'importance et donc sur l'estimateur $\widehat{\mathrm{Z}_x}(Z_{1:K}) = K^{-1} \sum_{i=1}^K p_\theta(x, Z_i)/q_\phi(Z_i \mid x)$ et donc sur l'ELBO

$$\mathcal{L}_{\mathrm{IW}}(\theta, \phi; x) = \int \log \left(K^{-1} \sum_{i=1}^K \frac{p_\theta(x, z_i)}{q_\phi(z_i \mid x)} \right) \prod_{i=1}^K q_\phi(z_i \mid x)\mathrm{d}z_i \ .$$

Un point crucial encore de cet ELBO est évidemment que l'estimateur de la constante de normalisation $p_\theta(x)$ doit être différentiable, afin de pouvoir ensuite appliquer des algorithmes d'optimisation par gradient stochastique. Le développement de tels estimateurs de constantes et des ELBO associés est aussi une part importante du travail de cette thèse.

1.4 Introduction aux chaînes de Markov

Les chaînes de Markov sont une classe de processus stochastiques utilisés couramment pour la modélisation de phénomènes aléatoires. Une chaîne de Markov $(X_k)_{k\in\mathbb{N}}$ est une séquence de variables aléatoires définies sur un espace filtré $(\mathcal{F}_k)_{k\in\mathbb{N}}$ telle que la loi de X_{n+1} conditionnellement à la filtration $\mathcal{F}_n$ est égale à la loi de X_{n+1} conditionnellement à X_n, $\mathbb{P}$-presque sûrement. En d'autres termes, cela signifie qu'un processus stochastique à temps discret possède la propriété de Markov si le passé et le futur sont indépendants étant donné le présent. [Dou+18] comprend une bibliographie très large pour les aspects théoriques de chaînes de Markov. Les chaînes de Markov sont enfin utilisées très largement dans la modélisation de processus de files d'attente ou de stockage, dans la modélisation de séries temporelles (en particulier en finance). Les chaînes de Markov peuvent enfin aussi être utilisées afin de construire un algorithme permettant d'échantillonner selon une distribution cible π, souvent connue à une constante de normalisation près. Nous rentrons alors dans le cadre des algorithmes de chaînes de Markov Monte Carlo (MCMC), devenus très populaires et largement répandus ces dernières années, et source de beaucoup d'attention. Cette thèse s'inscrit encore dans ce cadre.

Introduisons maintenant quelques notations. Dans cette thèse, nous étudions les chaînes de Markov à valeurs dans $\mathbb{R}^d$ dotées de la sigma-algèbre de Borel $\mathcal{B}(\mathbb{R}^d)$. Une chaîne de Markov homogène $(X_k)_{k\in\mathbb{N}}$ est caractérisée par son noyau $R : \mathbb{R}^d \times \mathcal{B}(\mathbb{R}^d) \to [0]\,1$ qui satisfait

1. Pour tout $x \in \mathbb{R}^d$, $R(x, \cdot)$ est une mesure de probabilité sur $\mathcal{B}(\mathbb{R}^d)$,

2. Pour tout $\mathsf{A} \in \mathcal{B}(\mathbb{R}^d)$ $x \mapsto R(x, \mathsf{A})$ est une fonction mesurable.

Pour toute mesure de probabilité μ sur $\mathcal{B}(\mathbb{R}^d)$ et $\mathsf{A} \in \mathcal{B}(\mathbb{R}^d)$, on désigne par $\mu R(\mathsf{A}) = \int_{\mathbb{R}^d} R(x, \mathsf{A})\mu(\mathrm{d}x)$ et pour tout $k \in \mathbb{N}, x \in \mathbb{R}^d$, $R^{k+1}(x, \mathsf{A}) = \int_{\mathbb{R}^d} R(x, \mathrm{d}y)R^k(y, \mathsf{A})$. Pour une mesure de probabilité μ sur $\mathcal{B}(\mathbb{R}^d)$ et f une fonction μ-intégrable, $\mu(f) = \int_{\mathbb{R}^d} f(x)\mu(\mathrm{d}x)$. Pour $x \in \mathbb{R}^d$ et f intégrable sous $R(x, \cdot)$, on dénote par $Rf(x) = \int_{\mathbb{R}^d} R(x, \mathrm{d}y)f(y)$.

Une des questions centrales en MCMC est de déterminer si le noyau R admet comme unique distribution invariante π, c'est-à-dire $\pi R = \pi$. En d'autres termes, cela signifie que l'application du noyau R à un échantillon $X \sim \pi$ résultera en un échantillon Y aussi distribué selon π. Une propriété impliquant cette invariance est la réversibilité, soit pour tout $\mathsf{A}, \mathsf{B} \in \mathcal{B}(\mathbb{R}^d)$, $\int_{\mathsf{A}} \pi(\mathrm{d}x)R(x, \mathsf{B}) = \int_{\mathsf{B}} \pi(\mathrm{d}x)R(x, \mathsf{A})$. Cette propriété est donc plus forte que la simple invariance et sera discutée en particulier dans la Section 2.5.2.

En particulier, l'invariance est une propriété clef pour assurer la convergence de la distribution des échantillons (partant d'une distribution initiale μ_0 arbitraire) $\mu_0 R^n$ produits par une chaîne de Markov vers la distribution cible π, ce qui permet alors d'utiliser directement les algorithmes MCMC pour produire des échantillons. En particulier, les algorithmes de Metropolis Hastings donnent une recette générale afin de construire de tels noyaux. Détaillons ici la Marche aléatoire de Metropolis (Random Walk Metropolis, RWM) qui est l'un des exemples les plus simples. Supposons ici que la distribution π admette une densité par rapport à la mesure de Lebesgue, aussi dénotée π par abus de notation. L'algorithme peut se définir comme ceci, pour un certain écart type $\sigma > 0$ et une distribution initiale μ_0:

- Tirer X_0 selon μ_0,

- Pour $k \geqslant 0$, définir $Y_{k+1} = X_k + \sigma W_{k+1}$, où W_k est un vecteur Gaussien standard en dimension d,

- Avec une probabilité $\min(1, \pi(Y_{k+1})/\pi(X_k))$, accepter la proposition Y_{k+1}, c'est-à-dire fixer $X_{k+1} = X_k$. Sinon, rejeter la proposition et fixer $X_{k+1} = X_k$.

Il est possible de définir le noyau R correspondant à cet algorithme, donné pour tout $x \in \mathbb{R}^d, \mathsf{A} \in \mathcal{B}(\mathbb{R}^d)$

par :

$$R(x,\mathsf{A}) = \int_{\mathsf{A}} \min\left(1, \frac{\pi(y)}{\pi(x)}\right) \frac{e^{-\|x-y\|^2/2\sigma^2}}{(2\pi\sigma^2)^{d/2}} \mathrm{d}y + \delta_x(\mathsf{A}) \int_{\mathbb{R}^d} \left\{1 - \min\left(1, \frac{\pi(y)}{\pi(x)}\right)\right\} \frac{e^{-\|x-y\|^2/2\sigma^2}}{(2\pi\sigma^2)^{d/2}} \mathrm{d}y \ . \quad (1.14)$$

On peut alors vérifier facilement la réversibilité de π par rapport à R. Si cet algorithme est très simple de compréhension et d'utilisation, beaucoup de développements et de raffinement se basant sur des idées similaires mais utilisant des propositions plus sophistiquées ont été étudiés et proposés, dans les travaux précédant et dans cette thèse.

1.5 Conclusion et plan

Les travaux de cette thèse sont donc divers et peuvent se décomposer en trois parties

- Développer des méthodes efficaces d'échantillonage et de simulation de distributions complexes, potentiellement à dimension élevée, connues à une constante de normalisation près.
- Établir de nouveaux algorithmes pour l'apprentissage de modèles génératifs complexes, pour la simulation approchée d'une distribution de probabilité, ou pour des modèles génératifs profonds à variables latentes.
- Appliquer des méthodes de simulation approchée pour l'inférence de réseaux de neurones bayésiens, proposant une approche efficace à l'apprentissage profond bayésien.

1.6 Résumé des contributions

1.6.1 Méthodes d'échantillonnages et de simulation

Trois travaux de cette thèse développent des méthodes d'échantillonage et de simulation avec des méthodes de Monte Carlo par chaîne de Markov en particulier.

MCMC non réversible avec des transformations inversibles: Une recette complète avec des garanties de convergences L'algorithme de Metropolis-Hastings (MH), le cheval de bataille de l'échantillonnage de Monte-Carlo par chaîne de Markov, fournit une recette simple pour construire des noyaux de Markov réversibles.

Cependant, si la réversibilité est une propriété facile à mettre en œuvre généralement requise dans la plupart des algorithmes MCMC utilisés (cela est directement vérifié dans les algorithmes MH), elle n'est pas nécessairement souhaitable du point de vue des performances. En effet, elle fournit une condition plus forte que l'invariance nécessaire, ce qui peut avoir un impact non négligeable sur la corrélation des échantillons produits. Cette idée a suscité un intérêt récent pour la conception de noyaux ne vérifiant cette propriété, tout en satisfaisant l'invariance. De plus, de nouveaux noyaux MH ont également été introduits, qui reposent sur des transformations déterministes inversibles complexes, généralement fortement paramétrées : [SZE17; LHS17a; Thi+20b; Nek+20]. Ce travail a deux objectifs. Le premier est de développer une recette simple et complète pour un certain type de noyaux MH non réversibles qui satisfont toujours l'invariance par rapport à une certaine distribution cible et tous les résultats classiques de convergence qui résultent de la littérature MH habituelle [Dou+18], ce qui conduit à un ensemble de conditions simples et pratiquement vérifiables. De plus, nous appliquons cette théorie et les résultats de convergence à différents types de noyaux MH basés sur ces transformations complexes, introduits dans ce papier ou dans la littérature précédente [SZE17; LHS17a].

Une illustration de notre travail avec l'échantillonneur NICE est donné sur la Figure 2.3. L'échantillonneur NICE est une généralisation de l'algorithme Hamiltonian Monte Carlo (HMC) basée sur des réseaux de neurones appris de manière adaptative. Si de manière classique cet algorithme est réversible, nous prouvons que nous pouvons le rendre irréversible de plusieurs manières différentes et en démontrons les effets sur la figure.

La réversibilité et l'irréversibilité sont ici induites par le rafraîchissement entre chaque étape du terme de quantité de mouvement de la distribution cible étendue. Le rafraîchissement complet signifie qu'à chaque étape, nous rafraîchissons complètement en rééchantillonnant une la quantité de mouvement selon la marginale $e^{-K(p)}$, le rafraîchissement complet aléatoire signifie que nous ne le rafraîchissons complètement qu'avec une certaine probabilité $\beta < 1$, et la persistance signifie que nous ne rafraîchissons la quantité de mouvement que partiellement, avec un noyau autorégressif Gaussien (l'énergie cinétique $K(p)$ étant ici choisie quadratique, de manière classique).

NEO: Echantillonnage hors équilibre sur l'orbite d'une transformation déterministe Nous proposons avec NEO un nouvel estimateur de constantes de normalisation. À partir d'une transformation déterministe et inversible T bien choisie, NEO propose d'estimer les constantes de normalisation en échantillonnant des particules puis en les transportant par T vers les régions qui sont les plus significatives pour la distribution cible π. Pour garantir l'absence de biais de l'estimateur, nous dérivons une formule permettant de combiner la contribution de chaque point le long de l'orbite. Ceci fournit donc un algorithme pour un nouvel estimateur. Nous donnons aussi quelques critères afin d'évaluer la performance de cet estimateur et guider son réglage, en particulier avec un homologue de la variance d'un estimateur d'importance. La transformation dans les expériences est un opérateur Hamiltonien dissipatif, permettant d'informer les trajectoires avec la distribution cible.
Un échantillonneur NEO-MCMC est aussi dérivé de cet estimateur. Cet échantillonneur est directement comparable à la méthode HMC en termes de complexité et se montre compétitif sur plusieurs exemples jouets et réels.

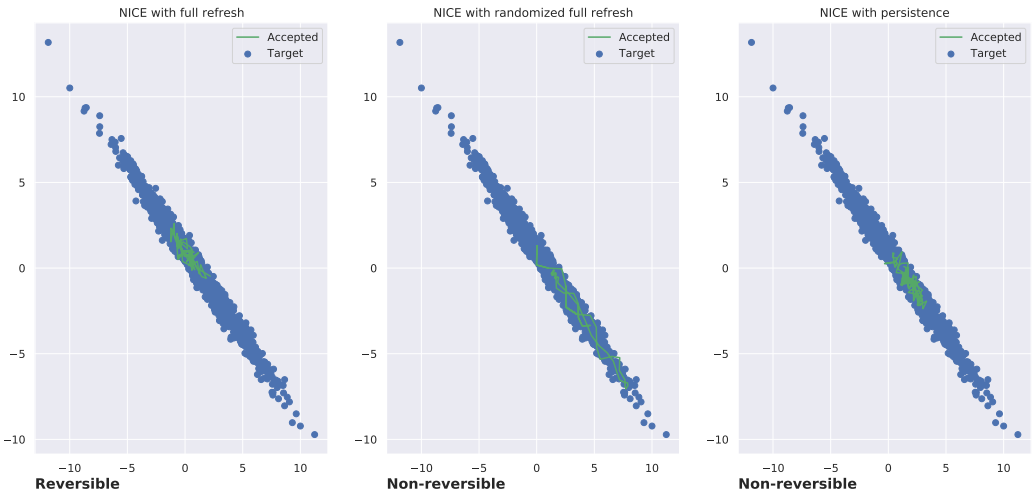


Figure 1.4: Effet de la non réversibilité sur un algorithme MCMC.

Nous présentons dans les figures suivantes une illustration de ce travail. La Figure 5.1 montre comment l'estimateur évolue en termes de longueur de trajectoire par rapport à un estimateur d'importance classique. Nous montrons, pour différentes valeurs des hyperparamètres, comment la variance effective diminue avec la longueur de la trajectoire par rapport à la variance de l'estimateur d'importance.

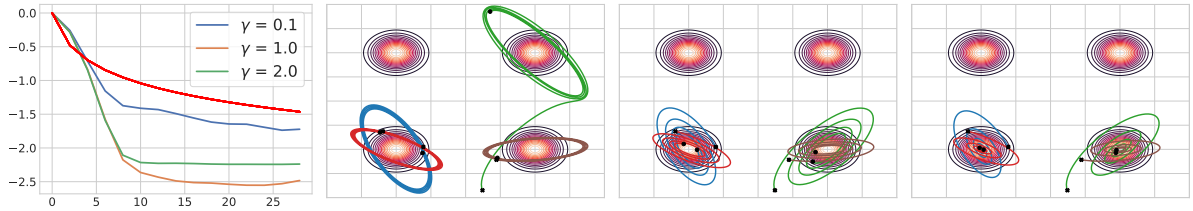


Figure 1.5: À gauche: Variance de NEO en fonction de la longueur de la trajectoire K sur l'orbite comparé à la variance (en rouge) de l'échantillonneur préférentiel avec $(K+1)$ échantillons en échelle logarithmique (la variance la plus basse est à privilégier). De gauche à droite après: Quatre exemples de trajectoires permettant de calculer l'estimateur de la constante de normalisation selon le réglage du terme de friction γ (de gauche à droite, $\gamma = 0.1, 1, 2$), sur une distribution cible mélange de 4 Gaussiennes.

Ex^{2}2MCMC: Échantillonnage avec Exploration/Exploitation Une critique commune envers les algorithmes MCMC est l'importance de l'auto-corrélation des échantillons produits par la chaîne est élevée et difficile à réduire, de par la proposition de mouvements seulement "locaux". D'un autre côté, d'autres méthodes, comme l'échantillonnage préférentiel et rééchantillonnage (i-SIR – et d'une manière générale les méthodes pseudo-marginales) ne sont basées sur des propositions de mouvements "globaux". Cependant, ces méthodes ont beaucoup de mal lorsque la dimension augmente. L'algorithme Ex^{2}MCMC est un nouveau type de noyau MCMC, qui combine plusieurs propositions globales et des déplacements locaux, visant à retenir le meilleur des deux. À cet effet, plusieurs idées sont développées. La première est de proposer un algorithme combinant le i-SIR composé avec quelques mouvements locaux du noyau pour "rafraîchir" la particule échantillonnée, typiquement des noyaux de Langevin Monte Carlo. Dans ce cas, nous montrons que l'hypothèse nécessaire aux résultats d'ergodicité et d'ergodicité géométrique de l'échantillonneur i-SIR s'affaiblit.

De plus, en haute dimension, l'algorithme i-SIR est souvent mis en difficulté car il ne propose

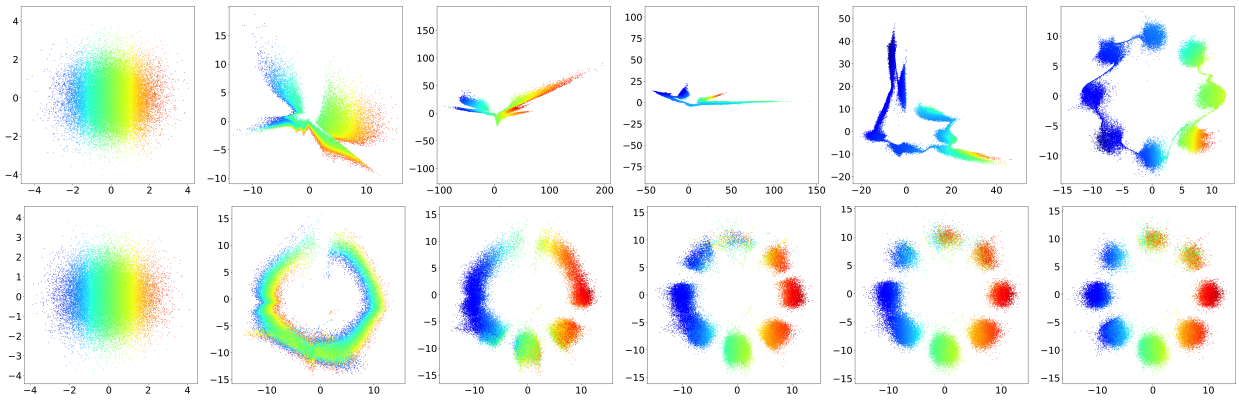


Figure 1.6: Sortie consécutive des différentes étapes d'un flot normalisant (en haut) par rapport aux étapes de MetFlow construit avec la même architecture de flot normalisant. A gauche : prior Gaussienne standard, puis l'effet successifs des cinq transformations qui constituent le flot.

que des mouvements globaux non informés, ce qui réduit drastiquement le temps de mixage. Afin d'essayer d'atténuer ce comportement, une recette complète sur la faccon d'échantillonner avec des propositions i-SIR corrélées est développée. Elle permet notamment de n'échantillonner que quelques points d'exploration globaux et de rafraîchir également en explorant localement la région, tout en bénéficiant de la parallélisation massive possible avec le noyau i-SIR. Enfin, nous proposons également une contrepartie adaptative du noyau Ex^{2}MCMC appelée FlEx^{2}MCMC. FlEx^{2}MCMC est basé sur des flots normalisants, appris de manière adaptative durant l'échantillonnage, qui permet donc de d'améliorer les propositions.

1.6.2 Modèles génératifs

Nous développons dans cette partie deux modèles génératifs, liés aux auto encodeurs ou aux flots normalisants.

MetFlow: Une nouvelle approche pour combiner Inférence Variationnelle, flots normalisants et Monte Carlo par chaînes de Markov Nous développons ici une méthode pour combiner à la fois l'inférence variationnelle et les algorithmes MCMC, en particulier les algorithmes MCMC avec des étapes de rejet de Metropolis-Hastings, qui sont cruciales pour satisfaire les conditions d'invariance par rapport à la distribution cible. Cela permet de construire des familles variationnelles basées sur la distribution obtenue après quelques étapes MCMC appliquées sur une distribution initiale dont les paramètres peuvent aussi être appris, ce qui augmente considérablement leur expressivité.

De plus, nous montrons que les paramètres du noyau MCMC peuvent être optimisés à l'aide d'un nouveau critère de type ELBO. Ce critère permet ainsi d'introduire de nouveaux noyaux MCMC basés sur des propositions de flots normalisants, appelés *Metropolized Flows*, ou MetFlow. MetFlow propose donc une nouvelle faccon d'échantillonner en utilisant flots normalisants et MCMC avec une faccon automatique d'optimiser les paramètres des noyaux MCMC par différenciation automatique avec un critère d'inférence variationnelle. Ces deux cas peuvent être illustrés sur la Figure 8.6, qui démontre clairement comment les noyaux MetFlow sont capables de passer outre certaines des contraintes topologiques auxquelles sont soumis les flots normalisants et à quel point une famille variationnelle basée sur MetFlow peut être expressive.

Monte Carlo Auto Encodeur Variationnel Nous développons ici une nouvelle manière d'apprendre les DLGM, en particulier en se basant sur des estimateurs de constante de normalisation état-de-l'art pour les modèles génératifs, Annealed Importance Sampling et Sequential Importance Sampling [GBC16]. Afin de pouvoir différencier l'ELBO basé sur ces estimateurs performants, nous développons dans

ce cadre une forme particulière d'Inférence Variationnelle Auxiliaire, avec des noyaux de Markov implémentés en particulier avec des dynamiques de Langevin. Cela mène à la définition des Monte Carlo Auto Encodeurs Variationnels, qui se montrent compétitifs sur de nombreux exemples par rapport à des modèles état-de-l'art.

1.6.3 Inférence approchée en apprentissage profond bayésien

Cette partie développe la solution proposée pour de l'inférence approchée dans des réseaux de neurones profonds dans le cadre de la compétition NeurIPS *Inférence approchée en apprentissage bayésien profond* (*Approximate Inference in Bayesian Deep Learning*)[1].

En particulier, nous développons une généralisation de la méthode de moyenne de poids stochastique avec des Gaussiennes, en utilisant en particulier des dynamiques de Langevin avec gradient stochastique, ce qui nous permet d'approcher la distribution *a posteriori* des poids du réseau par une mixture de distributions Gaussiennes qui se montre compétitive pour approcher la véritable distribution *a posteriori* des poids et la distribution prédictive correspondante par un étalon fourni par de très longues chaînes de Markov produite par l'algorithme de Hamiltonian Monte Carlo.

[1] `https://izmailovpavel.github.io/neurips_bdl_competition/`

Chapter 2

Introduction and motivation

2.1 General introduction

The objective of machine learning is to propose models for data in large dimensions (images, sounds, ...). In particular, we will focus in this work on the statistical and probabilistic modeling of these complex data (often called probabilistic machine learning).Generally speaking, probabilistic machine learning introduces statistical models, and therefore probability distributions, which are often known only to up to some normalizing constant. This normalizing constant is also called free energy, by analogy with statistical physics, or evidence. More precisely, it is often a marginal likelihood, the "proof" of the model, hence the name evidence. This type of model is used for many tasks: model selection, uncertainty quantification, data generation, or to infer and process models with latent or missing variables. Typical problems that arise in this context are the computation or estimation of this normalization constant (in order to perform for example model selection), the sampling (i.e. extracting samples) of this distribution, and in a general way the computation of averages, integrals, with respect to this distribution (in order to compute interesting quantities from our model, or even to perform predictions). In particular, let us introduce here three typical tasks in machine learning that we focus on in this thesis:

- Uncertainty quantification *via* Bayesian inference,
- Variational Auto Encoders,
- Energy based models.

Although theses tasks may be quite different, we will see that the problems that they raise are very similar. This thesis aims at contributing to develop novel methods suitable for those problems.

2.2 Bayesian Inference

Bayesian inference is particularly used for uncertainty quantification in machine learning. The core idea is to trat the parameters of the model as random variables whose distribution will be modified by the observations collected (update of the distribution *a priori* to a distribution *a posteriori* combining prior knowledge on the problem and experimental data).

The advantages of this method are to be able to quantify explicitly the uncertainties of the model, since the response of the model is itself random.

Let us first consider a coin, which we ignore if it is biased or not. We could estimate the probability of falling on heads directly by making n throws and calculating the empirical proportion, or consider the probability of falling on heads θ as a random variable between 0 and 1, with a uniform π distribution for example. The probability that a throw lands on heads, given the probability θ, is thus a Bernoulli

of parameter θ. We can compute the distribution after n throws $Y_1, \dots, Y_n$ using Bayes' formula:

$$\pi(\theta \mid y_1, \dots, y_n) = \frac{\pi(\theta) p(y_1, \dots, y_n \mid \theta)}{\int \pi(\theta) p(y_1, \dots, y_n \mid \theta) \mathrm{d}\theta} \,. \tag{2.1}$$

The term $\mathrm{Z} = m(y_1, \dots, y_n) = \int \pi(\theta) p(y_1, \dots, y_n \mid \theta) \mathrm{d}\theta$ is called the evidence of the chosen model (it depends only on the observations and the specified model). In general, the evidence is not explicitly computable and sampling the distribution *a posteriori* is only possible with Markov chain Monte Carlo methods. Sampling this distribution allows in particular to make predictions and predictive averages with all the uncertainty remaining in the model parameters.

The Bayesian approach is also crucial in machine learning applications. Typically, let $\mathcal{D} = \{(x_i, y_i\}_{i=1}^N$ be some observations, where $x_i \in \mathbb{R}^d$ are the covariates associated y_i. Suppose we are given as well a statistical model and a likelihood $p(y \mid x, \theta) = l(y; x, \theta)$ "explaining" the observations. Typically, for the Gaussian regression model when y is continuous in $\mathbb{R}$

$$p(y \mid x, \theta) = (2\pi\sigma^2)^{-1/2} e^{-(y - x^T\theta)^2/(2\sigma^2)} \,,$$

where we can suppose the variance σ^2 known, or for binary data $y \in \{0, 1\}$, the logistic regression specifies the likelihood

$$p(y \mid x, \theta) = (1 + \mathrm{e}^{-x^T\theta})^{-y} (1 + \mathrm{e}^{x^T\theta})^{y-1} \,.$$

In both cases, given a set of observations i.i.d. $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$, we can write the likelihood $p(\mathcal{D} \mid \theta) = \prod_{i=1}^N p(y_i \mid x_i, \theta)$. We specify the bayesian model by the choice of a prior distribution with density p_0 on the parameters θ, which represents the available knowledge on the model before collecting data. Applying Bayes rule allows to computethe posterior distribution of the parameters θ as

$$p(\theta \mid \mathcal{D}) = \frac{p_0(\theta) p(\mathcal{D} \mid \theta)}{\mathrm{Z}} \,, \quad \mathrm{Z} = \int p_0(\theta) p(\mathcal{D} \mid \theta) \mathrm{d}\theta \in (0, \infty) \,, \tag{2.2}$$

where $\mathrm{Z} = m(\mathcal{D}) = \int p_0(\theta) p(\mathcal{D} \mid \theta) \mathrm{d}\theta$ is called the evidence of the model (depends only on the observations and the model chosen – prior and likelihood).

In general, the evidence is not computable expicitly and sampling according to the posterior distribution has to rely on Monte Carlo methods (Markov chain Monte Carlo, Sequential Monte Carlo, or approximate methods, such as sampling importance resampling). Moreover, sampling according to this posterior distribution allows in particular to compute prediction or predictive means taking into account the uncertainty onthe parameters of the model. Indeed, a key of Bayesian inference in machine learning is the computation of the predictive distribution

$$p(y^* \mid x^*, \mathcal{D}) = \int p(y^* \mid x^*, \theta) p(\theta \mid \mathcal{D}) \mathrm{d}\theta \,. \tag{2.3}$$

This predictive distribution obtained *via* the marginalization of the posterior thus takes into account the uncertainty linked to the model and the data.

Moreover, an important part of Bayesian analysis is the model choice problem. Indeed, as seen above, the normalization constant of the *a posteriori* distribution is interpretable as evidence of the model, specified for example in the previous coinage case by the uniform a priori distribution on the parameter θ and the choice of the Bernoulli likelihood of the observations given the parameter θ. However, a reasonable approach could consider different parametric models $\mathcal{M}_1$ and $\mathcal{M}_2$ for the same data set $\mathcal{D}$, associated respectively to the parameters $\theta_1 \in \mathbb{R}^{d_1}$ and $\theta_2 \in \mathbb{R}^{d_2}$, to the distributions π_1, π_2 and to the likelihoods $p_1(\cdot \mid \theta_1), p_2(\cdot \mid \theta_2)$. Then, we can compare models 1 and 2 with the Bayes factor B_{12} given by

$$B_{12} = \frac{m(\mathcal{D} \mid \mathcal{M}_1) \pi_0(\mathcal{M}_1)}{m(\mathcal{D} \mid \mathcal{M}_2) \pi_0(\mathcal{M}_2)}, \tag{2.4}$$

where pi_0 is an a priori distribution on models 1 and 2, and

$$m(\mathcal{D} \mid \mathcal{M}_i) = \int \pi_i(\theta_i) p_i(\mathcal{D} \mid \theta_i) \mathrm{d}\theta_i \,, \quad i \in \{1, 2\} \,.$$

Above 1, the Bayes factor will tend to lean towards model 1, and below 1 towards model 2. Finally, it is important to note here that the calculation of this ration can only be done with the normalization constants of the distributions specified by the models. Particular attention is given in this thesis to the computation of normalization constants, a motivation for which is therefore given above. to synthesize, three major problems arise here

- The estimation of normalizing constants,
- Sampling distributions known up to a normalizing constant
- Computing expectations w.r.t. distributions known up to a normalizing constant

2.2.1 Bayesian neural networks

Neural networks, and deep learning in particular, have been state-of-the-art in numerous regression and classification tasks in the recent years and in many different fields, such as computer vision, or natural language porcessing [GBC16; LBH15; MLY17; KP18; Vou+18]. However, a proper uncertainty quantification for epistemic uncertainty and generalization is still crucial for a safe deployment of deep learning. Indeed, it has been put forward that neural networks, especially when deep, might be overconfident in their predictions in some types of tasks [NYC15; DK16; HAB19; Gaw+21]. This is of course problematic in a large scale deployment of deep learning. Many authors have suggested a bayesian approach [GG16; Gal16; KG17; McA+17] for a more trustable uncertainty quantification.

A neural network is a complex non linear function G_θ, parameterized by a vector of weights θ often high dimensional. Typically the parameters are the synaptic weights and the bias of the neurons which constitutes the base elements of the network.

Given a dataset $\mathcal{D} = \{(x_i, y_i\}_{i=1}^N$, a regression model may be built similarly than before, specifying the likelihood

$$p(y \mid x, \theta) = (2\pi\sigma^2)^{-1/2} \mathrm{e}^{-(y - G_\theta(x))^2/(2\sigma^2)} ,$$

and similarly for a binary classification model. Defining a prior distribution with weights p_0, typically standard normal, on the wiehgts of the network G_θ, one can then perfomr bayesian inference and compute the posterior istribution given by

$$p(\theta \mid \mathcal{D}) \propto p_0(\theta) p(\mathcal{D} \mid \theta) .$$

We can also link this distribution with a more classical objective of machine learning by writing the equivalent loss

$$\mathcal{L}(\theta; \mathcal{D}) = -\log p(\theta) - \sum_{i=1}^{N} \log p(y_i; x_i, \theta) . \tag{2.5}$$

Here rises again the problem of sampling according to this distribution.

The parameters of deep neural networks living in state spaces of high dimensions (hendreds of millions for some classical ResNets !), classical Bayesian methods, based on simulation methods such as MCMC are not directly applicable [Izm+21]. Moreover, the exact computation of the iteration of these chains is prohivitive and requires adaptations [WT11; CFG14; BDH17; Cha+18; BDM18], hence tge need of approximate methods, more scalable [AKW12; GG16; Kha+18; Izm+20; Dus+20; FSG20; Foo+20; Dax+20].

2.3 Generative modelling, Energy based models and Variational Auto Encoders

Generative models are at the heart of statistical modeling. Given a dataset $\mathcal{D} = \{x_i\}_{i=1}^N$, generative models aim at learnong the distribution p^* from which the data $\mathcal{D}$ were sampled. Practically, generative model specifies some parametric distribution with density $x \mapsto p_\theta(x)$, parameterized by θ, and estimates θ, typically by maximizing the likelihood f the observations, as we will see underneath.

2.3.1 Energy based models and how to learn them

A first class of generative models are the Energy based models (EBM). These types of models, inpired by Boltzmann Gibbs distribution from statistical physics, can be defined as

$$p_\theta(x) = e^{-E_\theta(x)}/\mathrm{Z}_\theta \ , \quad \mathrm{Z}_\theta = \int e^{-E_\theta(x)}\mathrm{d}x \ ,$$

where E_θ is the scalar output of a neural network taking as input an observation x. This model can be very flexible in its specification as it only requiers the unnormalized log-probability, called energy by analogy with statistical physics, and thus allow the definition of probability densities on potentially very complex state spaces.

How to tune the parameters of the model In statistical analysis, the choice of the parameters is often given by the maximum likelihood estimator. Given a statistical model $p(x;\theta) = p_\theta(x)$ and independent observations identically distributed $x_1, \dots, x_n \in \mathbb{R}^p$ according to p_θ, the idea is to choose the θ parameters maximizing the likelihood $p(x_1, \dots, x_n \mid \theta)$, i.e.

$$\hat{\theta}_n^\star = \arg\max_{\theta\in\Theta} n^{-1} \sum_{i=1}^n \log p_\theta(x_i) = \arg\max_{\theta\in\Theta} L_n(\theta) \ , \tag{2.6}$$

where we defined

$$L_n(\theta) = -n^{-1} \sum_{i=1}^n \log p_\theta(x_i) \tag{2.7}$$

We are thus confronted here with an optimization problem. However, we have several constraints for the models considered in this thesis. On the one hand, the models considered will generally be specified by deep neural networks, thus with a number of parameters of the order of hundreds of thousands, or even a million. Moreover, the number of observations is very large. In the case of the application to MNIST dataset, the data set contains 50,000 to 60,000 images. Finally, the data itself is very large, from simple images (MNIST, dimension $p = 784$ for example – the number of pixels in the image) to complex images for example. All these constraints impact the computation time and energy required to learn these models, in the most scalable way possible. In general, ne opts for stochastic gradient optimization, that is gradient ascent of the quantity $L_n(\theta)$, by approximating the gadient at each step by an (unbiased) estimator. For EBM, we can write

$$\begin{aligned} L_n(\theta) = &-n^{-1} \sum_{i=1}^n \log p_\theta(x_i) \ , \\ =&n^{-1} \sum_{i=1}^n E_\theta(x_i) + \log Z_\theta \\ =&n^{-1} \sum_{i=1}^n E_\theta(x_i) + \log \int e^{-E_\theta(x)}\mathrm{d}x \end{aligned}$$

The gradient w.r.t. θ of this quantity can be given by

$$\begin{aligned} \nabla_\theta L_n(\theta) &= n^{-1} \sum_{i=1}^n \nabla_\theta E_\theta(x_i) - \nabla_\theta \left(\log \int e^{-E_\theta(x)}\mathrm{d}x \right) \\ &= n^{-1} \sum_{i=1}^n \nabla_\theta E_\theta(x_i) - \int \nabla_\theta E_\theta(x) e^{-E_\theta(x)}\mathrm{d}x/\mathrm{Z}_\theta \ , \\ &= n^{-1} \sum_{i=1}^n \nabla_\theta E_\theta(x_i) - \int \nabla_\theta E_\theta(x) \cdot p_\theta(x)\mathrm{d}x \end{aligned}$$

We can thus see here two concurrent terms, to approximate independently for learning the parameters θ. The first one can be approximated by taking "mini-batches" of data: Instead of, at each step, computing the gradient of $\log p_\theta(x_i)$ for each of the points $x_i \in \mathcal{D}$, we draw a subset of size b (of the order of a hundred observations typically) $\mathcal{B}$ from the dataset and we approach the gradient $n^{-1}\sum_{i=1}^n \nabla_\theta E_\theta(x_i)$ by the estimator $b^{-1}\sum_{x_i\in\mathcal{B}}^n \nabla_\theta E_\theta(x_i)$. This allows a low cost optimization that can be scaled up in terms of number of data!

On the other hand, the second term requires a Monte Carlo approximation of an integral taken w.r.t. to a density known up to some normalizing constant only. A trustable approximation of this quantity thus requires a sampler in high dimensions. In thus case, EBM usually rely on Markov chain Monte Carlo algorithms, see Section 5.3.

2.3.2 Latent variable models

With the progress of machine learning and in particular the common use of deep neural networks, generative models have also become deep. Lately, another type of model has focused much attention: Deep latent generative models (DLGM) introduced for example by [KB14; Goo+14a].

These models are constructed in the following way

$$X = F(G_\theta(Z), W) \ ,$$

or equivalently, $X \sim p_\theta(\cdot \mid Z)$. X is here the observation that we want to model. Let us assume thereafter that $X \in \mathbb{R}^p$, Z is the latent variable, specified in the model, a kind of encoding of the observation X, and we assume here $Z \in \mathbb{R}^d$. The assumptions of this type of model are that the dimension of Z is smaller than that of X, and that Z follows a distribution which is in general very simple. G_θ is a complex function, non-linear, heavily parameterized by θ. Typically G_θ is a deep neural network. G_θ, finally, is typically called a decoder (in Variational Autoencoders) or generator (in Generative Adversarial Networks). Finally, W is a noise term added to our probabilistic model, and F a function composing this term and the deterministic response $G_\theta(Z)$. In a general way, the statistical model is thus specified by parameters θ and the joint model

$$p_\theta(x, z) = p_\theta(x \mid z) \cdot p_\theta(z) \ , \tag{2.8}$$

that we can generate by drawing successively $Z \sim p_\theta(z)$, $X|Z \sim p_\theta(X \mid Z)$.

For example, a DLGM for the MNIST (resp. FashionMNIST) dataset of binarized images of handwritten digits (resp. binarized fashion items) – see Figure 2.1 – could be defined as follows:

$$\begin{aligned} p_\theta(z) = \mathcal{N}(z; 0, \mathbf{I}) \quad \mathbf{p}(z) = (p_1(z), \dots, p_D(z)) = g_\theta(z) \\ \log p_\theta^{X|Z}(x \mid z) = \sum_{j=1}^D \log p\left(x_j \mid \mathbf{z}\right) = \sum_{j=1}^D x_j \log p_j(z) + (1 - x_j) \log\left(1 - p_j(z)\right) \ , \end{aligned}$$

where G_θ is the neural network with input the latent variable Z and output the parameters for the distribution of the observation given the latent $X \mid Z$.

How to tune the parameters of the model Let us now apply the maximum likelihood criterion to DLGMs. Note that we can write the likelihood of the observations X as

$$p_\theta(x) = \int_{\mathbb{R}^d} p_\theta(x \mid z) p_\theta(z) \mathrm{d}z \ . \tag{2.9}$$

In this case, the likelihood to be maximized is thus in the form of a complex integral (in d dimension), which does not admit of an explicit expression.

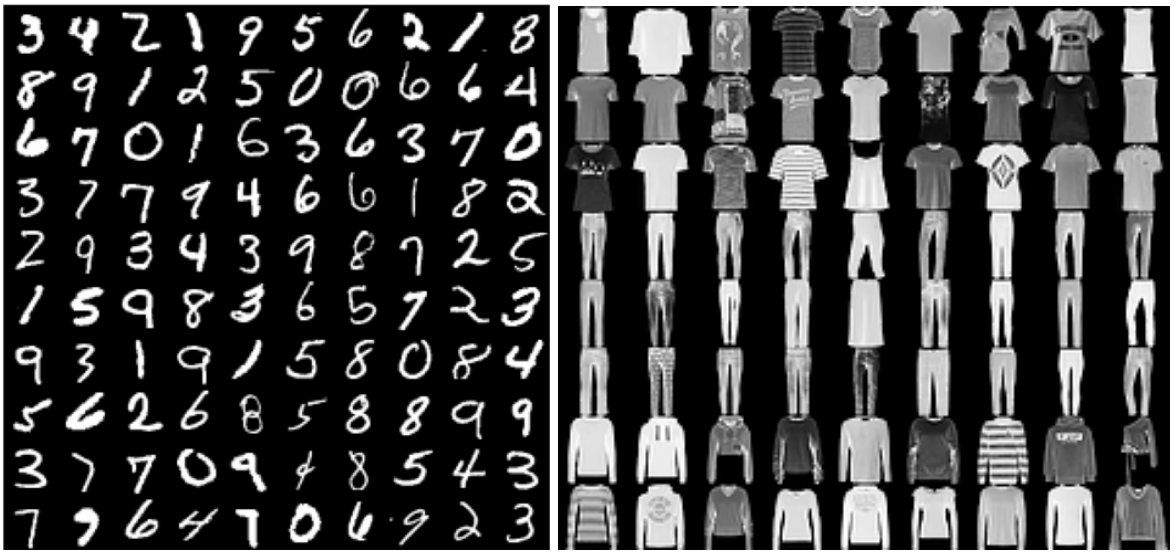


Figure 2.1: MNIST (left) and Fashion MNIST (right) datasets.

Second, in the case of DLGMs, we are faced with an additional problem. The gradient, for an observation x, $\nabla \log p_\theta(x)$ is not accessible, because it has to be expressed as an intractable integral. It is then interesting to apply Fisher's identity in this case, which is expressed as follows

$$\begin{aligned}\nabla_\theta \log p_\theta(x) &= \int_{\mathbb{R}^d} \frac{\nabla_\theta p_\theta(x,z)}{p_\theta(x)} \mathrm{d}z \\ &= \int_{\mathbb{R}^d} \nabla_\theta \log p_\theta(x,z) \frac{p_\theta(x,z)}{p_\theta(x)} \mathrm{d}z \\ &= \int_{\mathbb{R}^d} \nabla_\theta \log p_\theta(x,z) p_\theta(z|x) \mathrm{d}z \ .\end{aligned}$$

The gradient of the marginal log-likelihood $\nabla \log p_\theta(x)$, intractable, is then expressed as the integral of the gradient of the joint log-likelihood $\nabla \log p_\theta(x,z)$ by the conditional distribution of the latent variable z by the observation x (i.e., the distribution *a posteriori* of the latent variable given the observation). The joint log-likelihood is tractable and can be easily computed in the specified models. If the above integral is still intractable, it is now easy to estimate it by Monte Carlo method. Indeed, if $Z_1 \ldots, Z_m \sim p_\theta(\cdot \mid x)$, an estimator of $\nabla \log p_\theta(x)$ is thus given by $m^{-1} \sum_{i=1}^m \nabla \log p_\theta(x, Z_i)$. In order to obtain samples distributed according to $p_\theta(\cdot \mid x) \propto p_\theta(z) p_\theta(x,z)$, we can use Markov chain Monte Carlo algorithms, as for EBMs.

However, this approach should be repeated as many times as there are observations x, as for each x comes a different posterior $p_\theta(z \mid x)$. Thus, the need for more scalable and efficient methods, even at the expense of some theoretical convergence results. This is the base idea of amortized Variational Inference introduced in [KW13a].

2.3.3 Variational Auto Encoders

Variational Auto Encoders (VAE) [KW13a] are a type of DLGMs based on Variational Inference. VAEs introduce a parametric family of distributions, called variational family $\mathcal{Q} = \{q_\phi \, , \, \phi \in \Phi\}$, parameterized by another parameter ϕ, which will help for the estimation of the gradient of the loglikelihood. [KW13a] introduce the auxiliary criterion

$$\mathcal{L}(\theta, \phi; x) = \int_{\mathbb{R}^d} \log \left(\frac{p_\theta(x,z)}{q_\phi(z \mid x)} \right) q_\phi(z \mid x) \mathrm{d}z \ ,$$

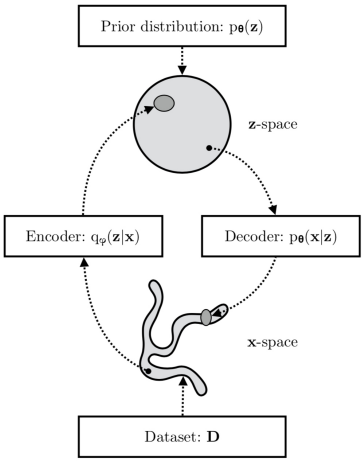


Figure 2.2: Amortized inference scheme, [KW19].

A more detailed introduction to Variational infernece is given in Section 3.1. However, note that one can express this quantity as

$$\mathcal{L}(\theta, \phi; x) = \int_{\mathbb{R}^d} \log\left(\frac{p_\theta(z \mid x) p_\theta(x)}{q_\phi(z \mid x)}\right) q_\phi(z \mid x) \mathrm{d}z = \log p_\theta(x) - D_{\mathrm{KL}}(q_\phi(\cdot \mid x \| p_\theta(\cdot \mid x) \ . \tag{2.10}$$

This objecrive is thus always lower than the marginal loglikelihood $\log p_\theta(x)$ and is thus referred to as the Evidence Lower Bound (ELBO). If this objective is straightfowrad to estimate *via* Monte Carlo, it differs from the loglikelihood $\log p_\theta(x)$, real quantity of interest here. Indeed, one can write

$$\begin{aligned}
\nabla_\theta \mathcal{L}(\theta, \phi; x) &= \int_{\mathbb{R}^d} \nabla_\theta \log p_\theta(x, z) q_\phi(z \mid x) \mathrm{d}z \ , \\
\widehat{\nabla_\theta \mathcal{L}}(\theta, \phi; x) &= K^{-1} \sum_{k=1}^{K} \nabla_\theta \log p_\theta(x, Z_k) \ , \quad Z_{1:K} \sim q_\phi(\cdot \mid x) \ .
\end{aligned}$$

and we can thus easily optimize the ELBO w.r.t. the parameters θ, although the expression of the gradient is quite different from Fisher's identity (1.8).

Moreover, we can see on (2.10) that the maximization of the ELBO will optimize concurrently the two quantities of interest

- Maximizing the marginal loglikelihood $\log p_\theta(x)$,
- Minimizing the divergence $D_{\mathrm{KL}}(q_\phi(\cdot \mid x \| p_\theta(\cdot \mid x)$ between the variational posterior and the true posterior.

A key contribution of those models is also given by Amortized Variational Inference [KW13a]. Indeed, we saw above that a critical point of the optimization is that the posterior distribution $p_\theta(\cdot \mid x)$ is different for each obserbation x. [KW13a] proposes to learn one global mapping $x \mapsto q_\phi(\cdot \mid x) \colon z \mapsto q_\phi(z \mid x)$ in x, where the parameters ϕ are thus shared. Section 2.3.3 shows schematically the idea of amortized variational inference. The Function which maps an observation x to a distribution in the latent space $x \mapsto q_\phi(\cdot \mid x)$ is called the encoder, while the counterpart which maps a latent point z to a distribution in the observation space $z \mapsto p_\theta(\cdot \mid z)$ is called the decoder, hence the name auto encoder.

If optimizing w.r.t. the parameters θ is now easy, taking the gradient w.r.t. the parameters ϕ of this model is however not so straightforward. Indeed, in general, one may write the gradient of the function

$$\phi \mapsto \int h(x, z) q_\phi(z \mid x) \mathrm{d}z$$

as

$$\int h(x,z)\nabla \log q_\phi(z|x) q_\phi(z|x)\mathrm{d}z \ ,$$

which we an estimate via Monte Carlo. However, this log gradient trick, called the REINFORCE estimator, has a very large variance. In order to make the optimization *via* stochastic gradient ascent easier, one might use the reparameterization trick, crucial in this type of models. Assume that for all x there exists a diffeomorphism $\epsilon \mapsto V_{\phi,x}(\epsilon)$ and a distribution g easy to sample from such that

$$\epsilon \sim \mathrm{g}\,, \quad z = V_{\phi,x}(\epsilon) \sim q_\phi(\cdot \mid x)\ . \tag{2.11}$$

Then, we can reparameterize the ELBO to write it as

$$\mathcal{L}(\theta,\phi;x) = \int \log\left(\frac{p_\theta(x, V_{\phi,x}(\epsilon))}{q_\phi(V_{\phi,x}(\epsilon)\mid x)}\right)\mathrm{g}(\epsilon)\mathrm{d}\epsilon \ , \tag{2.12}$$

where the expectation is now taken w.r.t. a function independent of the parameters ϕ, θ, which allows for an easy Monte Carlo estimation. Moreover, the variational distribution is specified often by choosing g and the diffeomorphism $V_{\phi,x}$. Let us present here the Gaussian Mean-Field Variational Auto Encoder [KW13a] as an example. The prior $p_\theta(z)$ is chosen as g the standard normal Gaussian on $\mathbb{R}^d$, and $p_\theta(x \mid z)$ is a distribution parameterized by the parameters $g_\theta(z)$, where g_θ is a neural network. Symmetrically, we build $q_\phi(z \mid x) = \mathcal{N}(z; \boldsymbol{\mu}_\phi(x), \mathrm{diag}(\boldsymbol{\sigma}^2_\phi(x)))$ in the case of Mean-Field VI, where $\boldsymbol{\mu}_\phi$, $\boldsymbol{\sigma}_\phi$ are both neural networks, typically symmetrically built w.r.t. g_θ. In this case, one can write the reparameterization trick associated with $q_\phi(\cdot \mid x)$:

$$\begin{aligned}&\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})\\ &z = \boldsymbol{\mu}_\phi(x) + \boldsymbol{\sigma}_\phi(x) \odot \boldsymbol{\epsilon} \sim q_\phi(\cdot \mid x)\ .\end{aligned}$$

This model is however limited, that is the ELBO does not necessarily give a satisfying approximation of the true marginal log-likelihood. Numerous ways have been introduced in the literature to enhance this classical Mean-Field VAE. Enhancing the variational family (reduced here to Gaussian distributions with diagonal covariances), for example using normalizing flows – a composition of parametric diffeomorphisms [RM15b; Kin+16a] is an effective way to imporve the results. But other strategies have been presented as well. In Auxiliary Variational Inference [SKW15a; RTB16; Maa+16], auxiliary random variables u are added to the model to i,crease expressivity. In this case one writes the variational distribution as

$$q_\phi(u, z \mid x) = q_\phi(u \mid x) q_\phi(z \mid u, x)\ ,$$

which can then allow to define a marginal variational distribution potentially much more expressive

$$q_\phi(z \mid x) = \int q_\phi(u, z \mid x)\mathrm{d}u$$

Similarly, we define the joint generative model

$$p_\theta(x, z, u) = p_\theta(u \mid x, z) p_\theta(x, z)\ .$$

We can then define the novel auxiliary objective as

$$\mathcal{L}_{\mathrm{AVI}}(\theta,\phi;x) = \int \log \frac{p_\theta(x,z,u)}{q_\phi(z,u\mid x)} q_\phi(z,u\mid x)\mathrm{d}z\mathrm{d}u \leqslant \log p_\theta(x)\ .$$

Note that we can always write

$$\begin{aligned}\mathcal{L}_{\mathrm{AVI}}(\theta,\phi;x) + \int \left\{\int \log\left(\frac{q_\phi(u\mid x,z)}{p_\theta(u\mid x,z)}\right) q_\phi(u\mid x,z)\mathrm{d}u\right\} q_\phi(z\mid x)\mathrm{d}z&\\ = \mathcal{L}(\theta,\phi;x)&\end{aligned}$$

Thus if in practice the auxilaty ELBO is "further" away from the marginal lg-likelihood, these models can outperform classical VAEs due to the the increased expressivity of the variational distributions, as presented in [SKW15a; RTB16; Maa+16] and in [Thi+21b].

Another direction [BGS15; Mad+17; DS18] focuses on the introduction on novel objectives based on estimators of the quantity $\log p_\theta(x)$. Indeed, we can build an ELBO from any unbiased estimator of the normalizing constant $p_\theta(x)$ of the unnormalized distribution with density $z \mapsto p_\theta(x, z)$.

Note for example $\widehat{\mathrm{Z}_x}(z)$ an unbiased estimator of $p_\theta(x)$ for a proposal density $q_\phi(\cdot \mid x)$, *i.e.* $\int_{\mathbb{R}^d} \widehat{\mathrm{Z}_x}(z) q_\phi(z \mid x) \mathrm{d}x = p_\theta(x)$. Then,

$$\mathcal{L}(\theta, \phi; x) = \int_{\mathbb{R}^d} \log(\widehat{\mathrm{Z}_x}(z)) q_\phi(z \mid x) \mathrm{d}z \leqslant \log p_\theta(x) \ ,$$

defines an ELBO, by Jensen's inequality and the concavity of the logarithm. The classical VAE implements the case where $\widehat{\mathrm{Z}_x}(z) = p_\theta(x, z)/q_\phi(z \mid x)$, the importance estimate with 1 sample, which is of course unbiased. However, many other estimators could be used here ! Indeed, more efficient estimators yield "better" ELBO, *i.e.* closer to the marginal log-likelihood. Indeed, a Taylor expansion shows that

$$\mathcal{L}(\theta, \phi; x) \approx \log p_\theta(x) - \frac{1}{2} \mathrm{var}_{q_\phi(\cdot|x)} \left[\frac{\widehat{\mathrm{Z}_x}(z)}{p_\theta(x)} \right] \ ,$$

see [Mad+17; DS18] for more details. Let us cite here the Importance Weighted Auto Encoder (IWAE), based on the K-samples importance estimator $\widehat{\mathrm{Z}_x}(Z_{1:K}) = K^{-1} \sum_{i=1}^K p_\theta(x, Z_i)/q_\phi(Z_i \mid x)$,which leads to the ELBO

$$\mathcal{L}_{\mathrm{IW}}(\theta, \phi; x) = \int \log \left(K^{-1} \sum_{i=1}^K \frac{p_\theta(x, z_i)}{q_\phi(z_i \mid x)} \right) \prod_{i=1}^K q_\phi(z_i \mid x) \mathrm{d}z_i \ .$$

A crucial characteristic of the estimator and the corresponding ELBO is that the estimator of $p_\theta(x)$ has to be differentiable, in order to apply stochastic gradient optimization afterwards. Developping such estimators and novel ELBOs is an important part of this thesis.

2.4 Introduction to Markov chain Monte Carlo methods

Markov chains are a class of stochastic processes commonly used to model random phenomena. A Markov chain $(X_k)_{k\in\mathbb{N}}$ is a sequence of random variables defined on a filtered space $(\mathcal{F}_k)_{k\in\mathbb{N}}$ such that the law of X_{n+1} conditional on the $\mathcal{F}_n$ filtration is equal to the law of X_{n+1} conditional on X_n, $\mathbb{P}$-almost certainly. In other words, this means that a discrete-time stochastic process has the Markov property if the past and the future are independent given the present. [Dou+18] includes a very large bibliography for the theoretical aspects of Markov chains. Markov chains are also widely used in the modeling of queuing or storage processes, and in the modeling of time series (in particular in finance). Markov chains can also be used to build algorithms aiming at sampling according to a target distribution π, often known up to some normalizing constant. We then enter the framework of Markov chain Monte Carlo (MCMC) algorithms, which are the source of much attention in the past years. This thesis concentrates on the latter.

Let us now introduce some notations. In this thesis, we study Markov chains with values in $\mathbb{R}^d$ endowed with the Borel sigma-algebra $\mathcal{B}(\mathbb{R}^d)$. A homogeneous Markov chain $(X_k)_{k\in\mathbb{N}}$ is characterized by its kernel $R : \mathbb{R}^d \times \mathcal{B}(\mathbb{R}^d) \to [0]\,1$ which satisfies

1. For any $x \in \mathbb{R}^d$, $R(x, \cdot)$ is a probability measure on $\mathcal{B}(\mathbb{R}^d)$,

2. For all $\mathsf{A} \in \mathcal{B}(\mathbb{R}^d)$ $x \mapsto R(x, \mathsf{A})$ is a measurable function.

For any probability measure μ on $\mathcal{B}(\mathbb{R}^d)$ and $\mathsf{A} \in \mathcal{B}(\mathbb{R}^d)$, we denote by $\mu R(\mathsf{A}) = \int_{\mathbb{R}^d} R(x, \mathsf{A})\mu(\mathrm{d}x)$ and for any $k \in \mathbb{N}, x \in \mathbb{R}^d$, $R^{k+1}(x, \mathsf{A}) = \int_{\mathbb{R}^d} R(x, \mathrm{d}y)R^k(y, \mathsf{A})$. For a probability measure μ on $\mathcal{B}(\mathbb{R}^d)$ and f an μ-integrable function, $\mu(f) = \int_{\mathbb{R}^d} f(x)\mu(\mathrm{d}x)$. For $xin\mathbb{R}^d$ and f integrable under $R(x, \cdot)$, we denote by $Rf(x) = \int_{\mathbb{R}^d} R(x, \mathrm{d}y)f(y)$.

One of the central questions in MCMC is to determine whether the kernel R admits as unique invariant distribution π, i.e. $\pi R = \pi$. In other words, this means that applying the kernel R to a sample X will result in a sample Y also distributed according to π. A property implying this invariance is reversibility, i.e. for any $\mathsf{A}, \mathsf{B} \in \mathcal{B}(\mathbb{R}^d)$, $\int_{\mathsf{A}} \pi(\mathrm{d}x)R(x, \mathsf{B}) = \int_{\mathsf{B}} \pi(\mathrm{d}x)R(x, \mathsf{A})$. This property is therefore stronger than simple invariance and will be discussed in particular in the Section 2.5.2.

In particular, invariance is a key property to ensure the convergence of the distribution of samples (starting from an initial μ_0 arbitrary distribution) $\mu_0 R^n$ produced by a Markov chain after n iterations of the kernel R to the target π , which then allows to use directly MCMC algorithms to produce samples from π.

In particular, the Metropolis-Hastings algorithms give a general recipe to build such kernels. Let us detail here the Random Walk Metropolis (RWM) which is one of the simplest examples. Let us suppose here that the distribution π admits a density with respect to the Lebesgue measure, also denoted π by abuse of notation. The RWM algorithm can be defined, for a certain standard deviation $\sigma > 0$ and an initial distribution μ_0, by:

- Draw X_0 according to μ_0,
- For $k \geqslant 0$, define the proposal $Y_{k+1} = X_k + \sigma W_{k+1}$, where W_k is a standard Gaussian vector in dimension d,
- With probability $\min(1, \pi(Y_{k+1})/\pi(X_k))$, accept the proposal Y_{k+1}, i.e. set $X_{k+1} = X_k$. Otherwise, reject the proposal and set $X_{k+1} = X_k$.

It is possible to define the kernel R corresponding to this algorithm, given for any $x \in \mathbb{R}^d, \mathsf{A} \in \mathcal{B}(\mathbb{R}^d)$ by

$$R(x, \mathsf{A}) = \int_{\mathsf{A}} \min\left(1, \frac{\pi(y)}{\pi(x)}\right) \frac{e^{-\|x-y\|^2/2\sigma^2}}{(2\pi\sigma^2)^{d/2}} \mathrm{d}y + \delta_x(\mathsf{A}) \int_{\mathbb{R}^d} \left\{1 - \min\left(1, \frac{\pi(y)}{\pi(x)}\right)\right\} \frac{e^{-\|x-y\|^2/2\sigma^2}}{(2\pi\sigma^2)^{d/2}} \mathrm{d}y \ . \quad (2.13)$$

We can then easily check the reversibility of π with respect to R. If this algorithm is very simple to understand and to use, many developments and refinements based on similar ideas but using more sophisticated proposals have been studied and proposed, in previous works and in this thesis.

2.5 Conclusion and plan

This thesis focuses thus on different tasks which can be decomposed in three main parts.

- The development of efficient sampling and simulation methods for potentially high dimensional distributions, known up to some normalizing constant.
- The derivation of algorithms for learning generative models for an approximate simulation of a target distribution, or for deep latent generative models.
- The application of approximate inference methods for Bayesian deep learning.

Summary of our contributions This thesis is based on the following articles and preprints:

- Thin, A., Kotelevskii, N., Denain, J.S., Grinsztajn, L., Durmus, A., Panov, M. & Moulines, E., 2020. *Metflow: A new efficient method for bridging the gap between Markov chain Monte Carlo and variational inference.* arXiv preprint arXiv:2002.12253.

- Thin, A., Kotelevskii, N., Durmus, A., Panov, M. & Moulines, E., 2020. *Metropolized flow: from invertible flow to MCMC.* ICML Workshop on Invertible Neural Networks, Normalizing Flows, and Explicit Likelihood Models, 2020.

- Thin, A., Kotelevskii, N., Andrieu, C., Durmus, A., Moulines, E. & Panov, M., 2020. *Non-reversible MCMC from conditional invertible transforms: a complete recipe with convergence guarantees.* arXiv preprint arXiv:2012.15550.

- Thin, A., Kotelevskii, N., Doucet, A., Durmus, A., Moulines, E., & Panov, M. (2021, July). *Monte Carlo variational auto-encoders.* In International Conference on Machine Learning (pp. 10247-10257). PMLR.

- Thin, A., Janati El Idrissi, Y., Le Corff, S., Ollion, C., Moulines, E., Doucet, A., Durmus, A. & Robert, C. (2021). *NEO: Non Equilibrium Sampling on the Orbits of a Deterministic Transform.* Advances in Neural Information Processing Systems, 34.

- Lagutin, E., Selikhanovych, D., Thin, A., Samsonov, S., Naumov, A., Belomestny, D., Panov, M. & Moulines, E., 2021. *Ex* ^{2} *MCMC: Sampling through Exploration Exploitation.* arXiv preprint arXiv:2111.02702.

2.5.1 Contributions

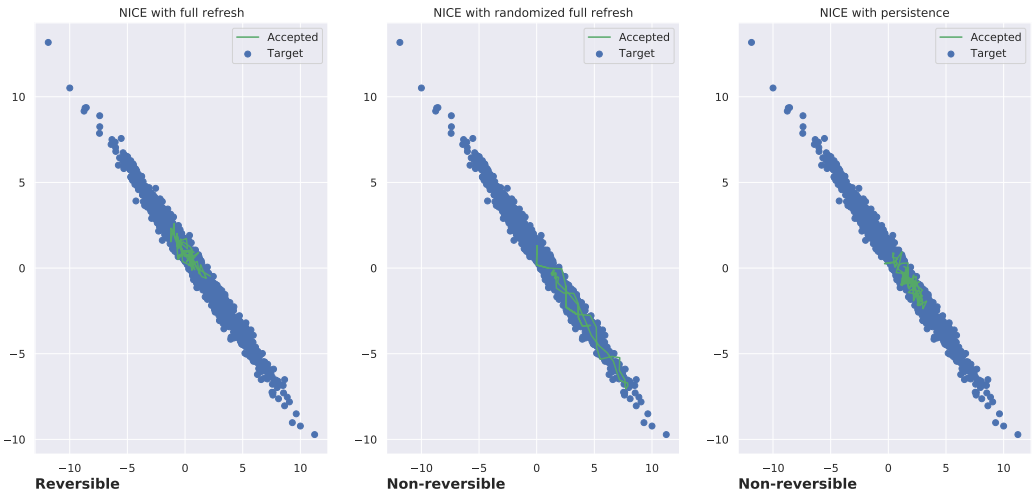


Figure 2.3: Effect of the irreversibility of an MCMC sampler.

2.5.2 Sampling and simulation methods

We develop three main works here for novel sampling and simulation methods, in particular using Markov chain Monte Carlo algorithms.

Non-reversible MCMC from conditional invertible transforms: a complete recipe with convergence guarantees The Metropolis-Hastings (MH) algorithm, the workhorse of Markov Chain Monte Carlo sampling, provides a simple recipe to construct reversible Markov kernels.

However, reversibility is a tractable property which is usually considered in most of the MCMC algorithms we work with (as directly verified in MH algorithms), it is however not necessarily desirable when considering performance. Indeed, it provides a stronger condition than the necessary invariance of the Markov kernel which can have a non negligible impact on correlation of the produced samples. This has prompted recent interest in designing kernels breaking this detailed balance property while satisfying invariance. Moreover, novel MH kernels have also been introduced which rely on complex invertible deterministic transforms, usually heavily parameterized [SZE17; LHS17a; Thi+20b; Nek+20]. This work has two objectives. The first one is to develop a simple and complete recipe for a certain type of non reversible MH kernels which still satisfy invariance with respect to a certain target distributions and all the classical results of convergence that result in the usual MH literature [Dou+18], which lead to a set of simple and practically verifiable conditions. Moreover, we apply this theory and the convergence results to different types of MH kernels based on these complex transforms, introduced in this paper or the previous literature [SZE17; LHS17a].

We illustrate on this figure our work with NICE sampler. NICE sampler is a generalization of Hamiltonian Monte Carlo based on neural networks learnt during the process. If classically, this algorithm is reversible, we prove that we can make it non reversible with two different ways, we we express in this figure.

The reversibility and irreversibility here are induced by the refreshment between each step of the momentum term of the extended target distribution. Full refresh means that at each step, we reresh completely by redrawing a momentum form the marginal $e^{-K(p)}$, randomized full refresh means that we only refresh it fully with some probability $\beta < 1$, and persistence means that we only partially refresh it with a Gaussian autoregression kernel (the kinetic energy being chosen here quadratic, as it often is).

NEO: Non Equilibrium Sampling on the Orbit of a Deterministic Transform Non Equilibrium sampling along the Orbits of a deterministic transform (NEO) proposes a novel estimator for

normalizing constants. Based on a well chosen deterministic and invertible transform T, NEO proposes to estimate evidence by sampling particles according to some prior, not necessarily informative, and transporting those particles by T to the regions that are the most significative for the target distribution π. To ensure unbiasedness of the estimator, we derive a formula to combine the contribution of each point along the orbit. This provides thus an algorithm for a novel estimator. A few criteria are derived to assess the performance of this estimator and guide its tuning, in particular a counterpart of the variance of an IS estimator, showing how powerful the effect of the transform T can be. This transform is chosen as a conformal Hamiltonian operator, inspired by the MCMC literature, allowing to inform the trajectories with the target distribution.

Moreover, from this estimator, a NEO-MCMC sampler is derived. This sampler is directly comparable to HMC in terms of complexity and shows competitiveness on several toy and real world examples.

We show in the following figures an illustration of this work. Figure 5.1 shows how the estimator scales in term of trajectory length compared to a classical IS counterpart. We display, for different values of hyperparameters, how the effective variance decreases with the length of the trajectory compared to how the variance of the IS estimator decreases as the number of samples increases.

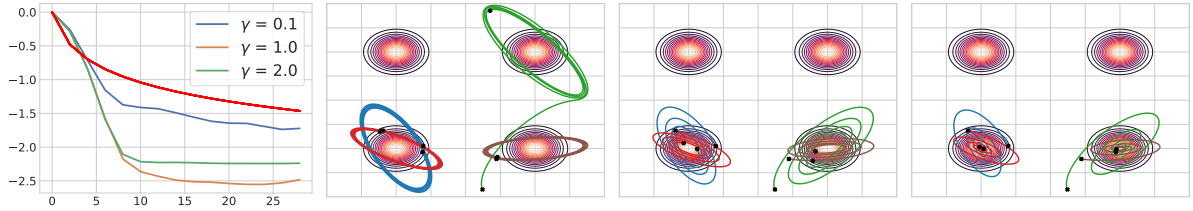


Figure 2.4: Left: Variance of NEO as a function of the lengtt K of the trajectory vs Variance of IS estimator with K samples (red) in $\log_{10}$-scale (the lower the better). Second left to right: Four examples of the corresponding trajectories with the same random seed for different values of the friction term of the conformal Hamiltonian γ (from left to right, $\gamma = 0.1, 1, 2$).

Ex^{2}2MCMC: Sampling through Exploration Exploitation A common criticism towards MCMC is that as they propose only "local" moves, auto-correlation of the samples produced by the chain is high and hard to decrease. On the other hand, pseudo marginal methods presented above, and in particular i-SIR algorithm describe methods which are based on "global" proposal moves. However, these methods struggle a lot when dimension increases. Explore-Exploit Markov chain Monte Carlo algorithm (Ex^{2}MCMC) is a novel type of MCMC kernel, which combines multiple global proposals and local moves and aims at retaining the best of both. To this effect, several ideas are developed. The first is to propose an algorithm combining i-SIR composed with a few local kernel moves to "rejuvenate" the sampled particle, typically MALA. In that case, we show that the hypothesis necessary for ergodicity and geometrical ergodicity results of the i-SIR sampler weakens.

Moreover, one can also argue that in high dimension, i-SIR struggles because it only proposes global uninformed moves. In order to try and alleviate such behaviour, a complete recipe of how to sample with correlated i-SIR proposals is developed. It allows in particular to only sample a few global exploratory points and refresh as well by exploring locally the region, still enjoying the embarrassingly parallel character of the i-SIR kernel. On the other hand, we also propose an adaptive counterpart of the Ex^{2}MCMC kernel, called FlEx^{2}MCMC. FlEx^{2}MCMC is based on normalizing flow proposals, learnt adaptively during the sampling stage, which will approximate better and better the target distribution to propose more effectively points , especially when dimension increases. The normalizing flow can be learnt with forward or backward KL, or a mixture or both.

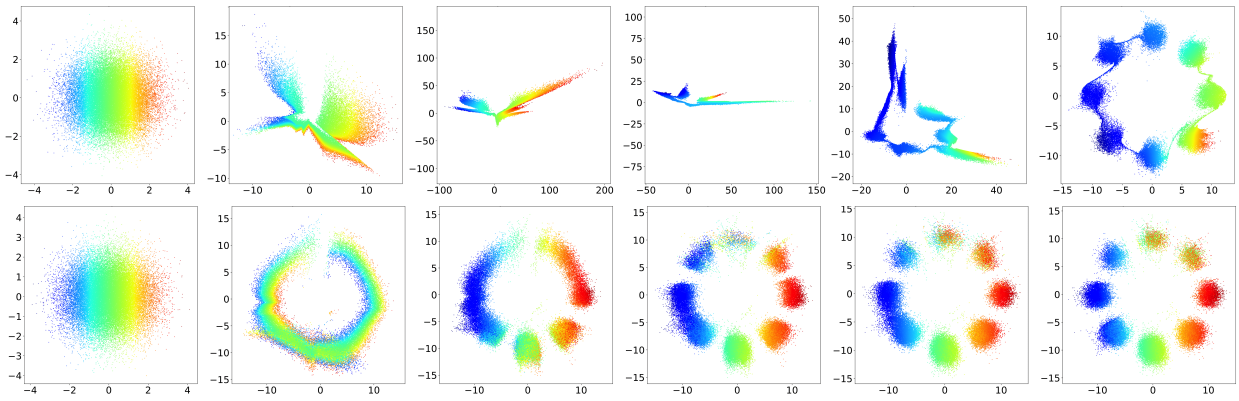


Figure 2.5: Consecutive outputs of normalizing flows (top) or metropolized flows (MetFlow, bottom). Left: prior normal distribution, then successive effect of the 5 transformations.

2.5.3 Generative models

We develop in this part two types of generative models, linked to deep latent generative models or normalizing flows.

MetFlow: A New Efficient Method for Bridging the Gap between Markov Chain Monte Carlo and Variational Inference MCMC and Variational Inference are typically opposed as their weaknesses and strengths are very complementary (asymptotic and non asymptotic convergence results for MCMC; scalability and automatic tuning of the parameters for VI). In *MetFlow: A New Efficient Method for Bridging the Gap between Markov Chain Monte Carlo and Variational Inference*, a method is developed to propose a way of combining both Variational Inference and MCMC algorithms, in particular MCMC algorithms with Metropolis Hastings rejection steps, which are crucial to satisfy invariance conditions with respect to the distribution of interest. This allows to construct variational families based on distribution obtained after a few MCMC steps applied on an initial learnable distribution, which increases greatly their expressivity.

Moreover, the parameters of the MCMC kernel can be optimized in this framework using a custom Evidence Lower Bound (ELBO). To unleash the full potential of this ELBO, new MCMC kernels based on normalizing flows proposals are introduced, called *Metropolized Flows*, or MetFlow. The resulting method thus proposes a novel way to sample using normalizing flows and MCMC with an automatic way of tuning (through automatic differentiation) the parameters of the MCMC kernels. Both those cases can be displayed on the following toy examples. The first example shows the benefits of being able to learn an initial distribution before applying MCMC steps afterwards, and the second showcases the effect of MetFlow with transitions provided by normalizing flows on a mixture of 8 Gaussian distributions examples. It displays clearly how MetFlow kernels are able to override some of the topological constraints normalizing flows are subject to and how expressive a MetFlow based variational family can be.

Monte Carlo Variational Auto Encoders We develop here a novel way to learn DLGMs and VAEs, in particular relying on state-of-the-art estimators of the normalizing constant for generative and decoder based models, Annealed Importance Sampling and Sequential Importance Sampling, see [GBC16]. In order to build a differentiable ELBO based on thos normalizing constant estimators, we develop a particular type of Auxiliary Variational Inference, using Markov kernels relying in particular on Langevin dynamics. If using classical Sequential Importance Sampling with Langevin proposal densities bears no technical difficulty to design a novel VAE, the case of AIS is more particular. Indeed, enforcing reversibility (3.3) implicates the use of Metropolis Hastings kernels which are not differentiable per say. A similar development to the one obtained in MetFlow allows to write however an extension of

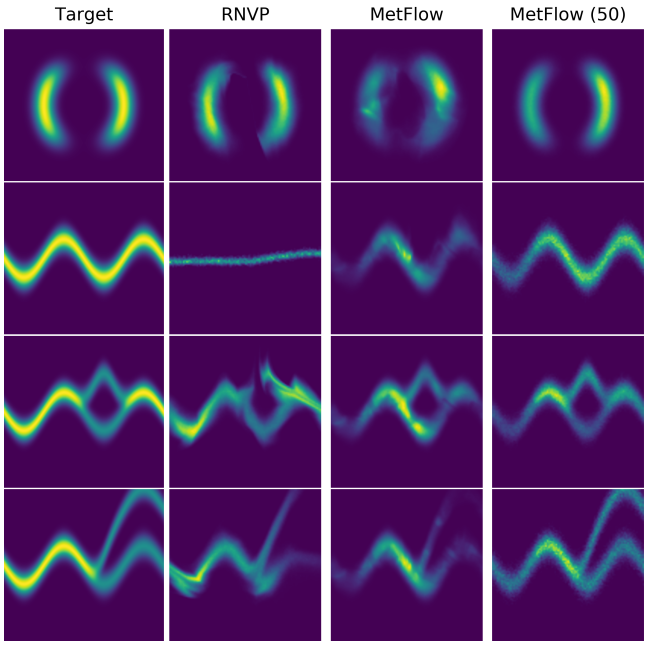


Figure 2.6: Comparison of Metropolized flow against the same normalizing flow learnt classically on difficult target distributions in $\mathbb{R}^2$.

the VAE based on AIS estimator of the normalizing constant $p_\theta(x)$. This results in the Monte Carlo Variational Auto Encoder, a VAE competitive to other state-of-the-art methods for learning VAEs.

2.5.4 Application to Bayesian Deep Learning: Efficient Approximate Inference with Gaussian Stochastic Weight Averaging

This part develops the proposed solution for approximate inference in bayesian deep neural networks in the context of the NeurIPS competition *Approximate Inference in Bayesian Deep Learning* [1]

In particular, we develop here a generalization of the stochastic weight averaging method with Gaussians, using in particular Langevin dynamics with stochastic gradient, which allows us to approximate the distribution of the network weights by a mixture of Gaussian distributions. The resulting method is competitive to approximate the true posterior distribution of the weights and the corresponding predictive distribution compared to a standard provided by very long Markov chains produced by the Hamiltonian Monte Carlo algorithm.

[1] `https://izmailovpavel.github.io/neurips_bdl_competition/`

Chapter 3

General Background

3.1 Classical estimators of normalizing constants

The estimation of the normalizing constant, as seen above, is a fundamental problem in probabilistic machine learning, and many estimators have been considered [FW12; MFR20]. The most straightforward is of course given by Importance Sampling (IS). Consider a distribution π, known only up to some normalizing constant Z, (we note in the following $\pi = \tilde{\pi}/\mathrm{Z}$, where $\tilde{\pi}$ is the unnormalized version of π to which we have access), and an auxiliary distribution, normalized and from which sampling is easy. Suppose as well that they are absolutely continuous w.r.t. each other, that is, there exists a non-negative function $w = \mathrm{d}\tilde{\pi}/\mathrm{d}q$. In the case where π and q have a density w.r.t. the Lebesgue measure, also noted π and q here, as long as $\pi(x) > 0 \Rightarrow q(x) > 0$, we can write $w = \tilde{\pi}/q$ directly.

Then, an estimator $\hat{\mathrm{Z}}_{\mathrm{IS}}(X_{1:n})$ of Z is given by sampling

$$X_1, \dots, X_n \overset{\text{iid}}{\sim} q \ , \quad \hat{\mathrm{Z}}_{\mathrm{IS}}(X_{1:n}) = n^{-1} \sum_{i=1}^{n} w(X_i) \ . \tag{3.1}$$

The IS estimator $\hat{\mathrm{Z}}_{\mathrm{IS}}(X_{1:n})$ is unbiased and converges almost surely to Z as n goes to ∞. However, the choice of the importance distribution q is crucial and can be difficult. Indeed, it has been put forward for example in [Aga+17]that when the dimension d increases, the number of samples n should grow exponentially in d in order to the performance of this estimator to maintain a given accuracy. This can be known as an effect of the curse of dimensionality.

Moreover, finding a good importance distribution (or tuning its parameters when q is learnable) automatically is not necessarily straightforward.

Many improvement of the vanilla IS algorithm have been proposed, see for example [OZ00b; Aga+17; AM21]. Of particular interest is Annealed Importance Sampling (AIS) estimator, [Nea01b; Wu+16; DF19] which can be seen as a special case of Sequential Monte Carlo (SMC) [DDJ06b].

Annealed Importance Sampling [Nea01b] and in general Sequential Monte Carlo are based on the intuition that if the distributions q and π are quite different, one might benefit from creating a annealing (or tempering) path from q to π. Let us present first the outline of an SMC algorithm.

Introduction to Sequential Monte Carlo Suppose now generally that we have access to some target unnormalized pdf $\{p_k(x^{0:k})\}_{0 \leqslant k \leqslant K}$. We could sample from a sequence of proposal pdf $\{q_k(x^{0:k})\}_{0 \leqslant k \leqslant K}$, or let $q_k(x^{0:k}) = m_k(z^{k-1}, z^k) q_{k-1}(x^{1:k-1})$, where $m_k(z^{k-1}, \cdot)$ is a transition density functions so as to *re-use* our samples. In this case, we can write the importance weights on this sequence as

$$\begin{aligned} w_k(x^{0:k}) &= \frac{p_k(x^{0:k})}{q_k(x^{0:k})} = \frac{p_k(x^{0:k})}{m_k(x^{k-1}, x^k) q_{k-1}(x^{0:k-1})} \\ &= \frac{p_k(x^{0:k})}{m_k(x^{k-1}, x^k) p_{k-1}(x^{0:k-1})} w_{k-1}(x^{0:k-1}) \ . \end{aligned}$$

Thus, the Sequential Importance Sampling algorithms boils down to

- Sample $X_1^0, \dots, X_n^0 \sim q_0$ the first particles according to the proposal distributions q_0.
- Compute the importance weights $W_i^0 = p_0(X_i^0)/q_0(X_i^0)$.
- For steps $1 \leqslant k \leqslant K$,
  - Sample $X_i^k \sim m_k(X_i^{k-1}, \cdot)$ the next step particle.
  - Compute the incremental importance weight

$$w_k(X_i^{k-1}, X_i^k) = \frac{p_k(x_{0:k})}{m_k(x_{k-1}, x_k) p_{k-1}(x_{0:k-1})} w_{k-1}(x_{0:k-1}) \ ,$$

  and the weight $W_i^k \propto w_k(X_i^{k-1}, X_i^k) W_i^{k-1}$

Some methods add as well resampling steps between each iteration, but we omit them here for simplicity and as in many of the applications and extensions of this method we consider, we do not add this resampling step. This framework is useful as we can actually use it to build estimators and samplers from any sequence of distributions [DDJ06b]. Indeed, given a sequence $\{\gamma_k\}_{k \leqslant K}$ of target distributions on $\mathbb{R}^d$, we may construct a *auxiliary* sequence $\{p_k\}_{k \leqslant K}$ of distributions by introducing Markov transition densities $\{\ell_k\}_{k \leqslant K}$

$$p_k(x^{0:k}) = \gamma_k(z^k) \prod_{p=0}^{k-1} \ell_k(z^{p+1}, z^p) \ .$$

This form of distributions is particularly useful because each p_k has the correct marginal on x^k, and also has a convenient form for the use of the techniques introduced above.

SMC algorithm now boils down to The choice of the kernel density ℓ_k is discussed in [DDJ06b] and should be optimally chosen to

$$\ell_{k-1}(x^k, x^{k-1}) = \frac{\gamma_{k-1}(x^{k-1}) m_k(x^{k-1}, x^k)}{\int \gamma_{k-1}(x^{k-1}) m_k(x^{k-1}, x^k) \mathrm{d}x^{k-1}} \tag{3.2}$$

Annealed Importance Sampling implements a special case of an SMC algorithm for computing the normalizing constant of the target distribution π. To compute the AIS estimator, build a sequence of intermediate distribution which will bridge the simple, normalized, easy-to-sample distribution q to the complex unnormalized π, that is, consider $\{\gamma_k\}_{0 \leqslant k \leqslant K}$ such that $\gamma_0 = q$, $\gamma_K = \tilde{\pi}$. Typically, one might choose a sequence $\{\beta_k\}_{0 \leqslant k \leqslant K}$ such that $0 = \beta_0 < \beta_1 < \cdots < \beta_K = 1$ and for $k \in \{0, \dots, K\}$, $\gamma_k = q^{1-\beta_k} \tilde{\pi}^{\beta_k}$, where $\{\beta_k\}_{k=0}^K$ is called the annealing schedule. The AIS estimator is given directly by the SMC algorithm by computing

$$\begin{aligned} &X_1^0, \dots, X_n^0 \overset{\text{iid}}{\sim} q \ , \quad X_i^k \sim m_k(X_i^{k-1}, \cdot) \text{ for } i \in \{1, \dots, n\}, \ k \in \{1, \dots, K\} \ , \\ &\hat{\mathrm{Z}}_{\mathrm{AIS}}(X_{1:N}^{0:K}) = \frac{1}{N} \sum_{k=1}^{N} W_K(X_i^{0:K}) \ , \end{aligned}$$

where for $k \in \{1, \dots, K\}$

$$W_k(X_i^{0:k}) = w_k(X_i^{0:k}, X_i^k) W_{k-1}(X_i^{0:k-1}) \ .$$

In particular, the AIS algorithm approximates the optimal expression (3.2) of the kernel ℓ_k by specifying the constraint

$$\gamma_k(x^{k-1}) m_k(x^{k-1}, x^k) = \gamma_k(x^k) \ell_k(x^k, x^{k-1}) \ . \tag{3.3}$$

We express for simplicity this expression with density here. However, in order to satisfy this detailed balance condition, one usually has to rely on Metropolis-Hastings kernels, which do not have a density

w.r.t. the Lebesgue measure. In this case, the expression of the weights and the incremental weights simplifies, and we can easily show that in this case,

$$w_k(x^{k-1}, x^k) = w_k(x^{k-1}) = \frac{\gamma_k(x^{k-1})}{\gamma_{k-1}(x^{k-1})}, \quad W_K(x^{0:K}) = \prod_{k=1}^{K} \frac{\gamma_k(x^{k-1})}{\gamma_{k-1}(x^{k-1})} .$$

This estimator can also be shown to be unbiased and provides in particular a state-of-the-art method for estimating the normalizing constants of generative models, [Wu+16]. However, it still remains unclear in this type of case how to derive a good initial distribution q, and how the parameters of this refined algorithm should be tuned (Markov kernels and their own parameters, length of the annealing schedule, parameters for the annealing schedule $\{\beta_k\}_{0 \leqslant k \leqslant K}$, ...).

Variational Inference Variational Inference (VI) is a method that approximates probability densities through optimization. This method can be used widely and is usually faster and more scalable than classical methods such as MCMC. The idea is to build a variational family of distributions, $\mathcal{Q}$, and then re formulate the sampling problem as a minimization of some divergence $D(q\|\pi)$ between a distribution $q \in \mathcal{Q}$ and the target π. The classical divergence used is the Kullback–Leibler divergence D_{KL} , and in this case we can thus express the VI problem as

$$q^* = \arg\min_{q \in \mathcal{Q}} D(q\|\pi) = \arg\min_{q \in \mathcal{Q}} \int \log\left(\frac{q}{\pi}\right) \mathrm{d}q .$$

From this problem, we can build a novel objective, referred to as the Evidence Lower Bound (ELBO). Indeed, suppose that π, q here have a density w.r.t. to the Lebesgue measure, and that π is only accessible up to some (unknown) normalizing constant Z, $\pi = \tilde{\pi}/\mathrm{Z}$. Then, we introduce the ELBO

$$\mathcal{L} = \int_{\mathbb{R}^d} \log\left(\frac{\tilde{\pi}(x)}{q(x)}\right) q(x)\mathrm{d}x = \log \mathrm{Z} - \int_{\mathbb{R}^d} \log\left(\frac{q(x)}{\pi(x)}\right) q(x)\mathrm{d}x , \tag{3.4}$$

which is thus a lower bound of the evidence Z by nonnegativity of the KL divergence. The VI problem is thus equivalent to the maximization of the ELBO above. Usually, the variational family $\mathcal{Q} = \{q_\phi \,;\, \phi \in \Phi\}$ is parameterized by some parameter ϕ, and the ELBO $\mathcal{L}(\phi)$ is thus optimized *via* stochastic gradient ascent on the parameter ϕ.

Note that an ELBO can also be built from any unbiased estimator of the normalizing constant Z. Indeed, if in classical VI, the estimator considered is just a one sample importance sampling estimator: $\hat{\mathrm{Z}}(X) = \tilde{\pi}(X)/q(X),\ X \sim q$, this could be generalized. Let $\hat{\mathrm{Z}}(X)$ denote an unbiased estimator of Z for a density q, that is $\int_{\mathbb{R}^d} \hat{\mathrm{Z}}(x)q(x)\mathrm{d}x = \mathrm{Z}$. Then, the quantity

$$\mathcal{L} = \int_{\mathbb{R}^d} \log(\hat{\mathrm{Z}}(x))q(x)\mathrm{d}x \leqslant \log \mathrm{Z}$$

is also an Evidence Lower Bound, by Jensen's inequality and the concavity of the logarithm. In particular, note that Importance Weighted Variational Inference [BGS15] considers rather K importance samples and the estimator $\hat{\mathrm{Z}}_{\mathrm{IS}}(X_{1:K}) = K^{-1} \sum_{i=1}^{K} \tilde{\pi}(X_i)/q(X_i)$ and thus the ELBO

$$\mathcal{L}_{\mathrm{IW}} = \int \log\left(K^{-1} \sum_{i=1}^{K} \frac{\tilde{\pi}(x_i)}{q(x_i)}\right) \prod_{i=1}^{K} q(x_i)\mathrm{d}x_i .$$

Building efficient normalizing constant estimators thus allow to consider different ELBOs. Moreover, a Taylor expansion shows that

$$\mathcal{L} \approx \log \mathrm{Z} - \frac{1}{2}\mathrm{var}_q\left[\frac{\hat{\mathrm{Z}}(x)}{\pi(x)}\right] ,$$

see [Mad+17; DS18] for more details. This shows that changing the estimator for a more efficient one (less variance) will increase the ELBO performance (closer to the actual evidence).

This becomes crucial in DLGM trained *via* Maximum likelihood estimation (or a proxy of maximum likelihood) as in that case the normalizing constant $\log \mathrm{Z} = \log p_\theta(x)$ the actual quantity to optimize in θ, as seen in Section 2.3.3.

Usually, variational inference builds on a simple mean-field approximation, that is the family $\mathcal{Q}$ is composed of normal distributions with diagonal covariances, the parameters ϕ thus being the mean and the diagonal of the covariance of the distribution. However, this struggles easily with a lack of expressivity. Usual attempts at extending the classical mean-field distribution usually involve the use of normalizing flows [RM15b; Pap+21], *i.e.* a composition of deterministic diffeomorphisms (C^1 bijective mappings) which map a simple initial distribution to an arbitrarily complex one depending on the number and the expressivity of the composed diffeomorphisms, see Section 3.3. The optimization of such models can be done easily with the automatic differentiation packages present for example in Pytorch [Fal19] and provide scalable inference models.

In the recent works, IS based on neural networks has been subject to an increasing attention; see e.g. [EM12; Mül+19; Pap+19; Pra19a; Wir+20; WKN20b]. In particular, Neural IS is an adaptive IS relying on an importance distributions obtained using normalizing flows to a reference initial distribution. We detail in Section 3.3 normalizing flows. The advantage of these methods is that they provide an automatic way of tuning the importance distribution parameterized *via* the normalizing flow, by minimizing a divergence between the proposal and the target (such as the Kullback–Leibler [Mül+19] or the χ^2-divergence [Aga+17]). Other works suggests to add stochasticity in the process to improve the normalizing flow; see for example [WKN20b; Cor+19]. For learning importance distributions, as well as in a normalizing flow context, literature suggests to optimize the parameters of the importance distribution using a VI loss.

Other methods estimation methods non covered here can be given (non exhaustive list) by nested sampling [Ski04], bridge and path sampling [GM98], and are discussed for example in [CS97].

3.2 High dimensional simulation and sampling techniques

MCMC is classically used for simulating high dimensional distribution. However, when dimension and complexity increases, the Random Walk Metropolis algorithm presented in Section 5.3 is not necessarily favored. Indeed, gradient-based algorithms are widely considered to be state-of-the-art in MCMC, and are rather used in these more complex settings. This type of algorithms, starting with Langevin algorithms [Par81], use gradient of the target distribution π to inform the proposals. Several measures of performance have been developed to quantify this argument. In particular, high-dimensional scaling arguments compare how the method is affected by the dimension d of the target π. To give some benchmark, for RWM, the decay of performance is of order d^{-1} [GGR97], for the Langevin algorithm, this decay is of order $d^{-1/3}$ [RR98], while for Hamiltonian Monte Carlo, [Bes+13] showed that it was of order $d^{-1/4}$. Let us now present these different algorithms, and assume thus in the following that the target π is continuously log-differentiable.

The overdamped Langevin dynamics associated to a target π log-differentiable can be written as the solution of the stochastic differentiable equation

$$\mathrm{d}X_t = \mathrm{d}\log\pi(X_t)\mathrm{d}t + \sqrt{2}\mathrm{d}B_t \,, \tag{3.5}$$

where B_t is a d-dimensional Brownian motion. This continuous dynamics is π-reversible. In a MCMC sampling context, we rather consider the Euler-Maruyama discretization of this dyamics, with stepsize η, which can be simulated, given a starting point X_0, by the update for $k \in \mathbb{N}$

$$X_{k+1} = X_k + \eta \mathrm{d}\log\pi(X_k) + \sqrt{2\eta} W_{k+1} \,, \tag{3.6}$$

where $\{W_k\}_{k\in\mathbb{N}}$ is a sequence of d-dimensional i.i.d. standard Gaussian variables. This describes the Unadjusted Langevin Algorithm (ULA). This discretization is however not exact, and the distribution sampled by this dynamics is then slightly off the right target π, see [DM17] for a detailed discussion.

In order to recover the reversibility w.r.t. to the target π, one has to add a Metropolis–Hastings correction step, in the spirit of the RWM algorithm. Define the proposal density of this algorithm, which can be defined as $q(x,y) = \mathrm{N}(y; x + \eta \mathrm{d}\log\pi(x), 2\eta)$, where $\mathrm{N}(\cdot;\mu,\Sigma)$ denotes the d-dimensional Gaussian density with mean μ and covariance matrix Σ. Given a starting point X_0, and a sequence $\{W_k\}_{k\in\mathbb{N}}$ of d dimensional standard Gaussian vectors, the adjusted algorithm repeats for $k \in \mathbb{N}$

- Define $Y_{k+1} = X_k + \eta \mathrm{d}\log\pi(X_k) + \sqrt{2\eta}W_{k+1}$,
- With probability $\min(1, [\pi(Y_{K+1})q(Y_{k+1}, X_k)]/[\pi(X_k)q(X_k, Y_{k+1})]$, accept proposal Y_{k+1}, and set $X_{k+1} = Y_{k+1}$. Else, reject proposal and set $X_{k+1} = X_k$.

This is called the Metropolis Adjusted Langevin Algorithm (MALA) [Bes94a].

More recently, the Hamiltonian (or Hybrid) Monte Carlo (HMC) algorithm provides an even more efficient sampler [Bes+13]. HMC starts by extending the target distribution π to $\bar{\pi}$ distribution in $\mathbb{R}^{2d}$ as follows: $\bar{\pi}(q,p) \propto e^{\log\tilde{\pi}(q) - K(p)}$, where $K(p) = 1/2p^T M^{-1}p$, with M a positive definite mass matrix and $p \in \mathbb{R}^d$. By analogy with statistical physics, we refer to q (resp. p) as the position (resp. momentum) and let us denote $H(q,p) = U(q) + K(p)$, with $U(q) = -\log\tilde{\pi}(q)$ referred to as the potential energy, $K(p)$ referred to as the kinetic energy. It is worth noting here that choices of kinetic energies other than the Gaussian term which we wrote here have been proposer, see for example [Lu+17; LFR19]. Let us follow the presentation given in [DMS17]. This facorization implies moreover that if we can sample according to this joint distribution, discarding the momentum variable would give us a sample from the distribution of interest π.

Hamilton's equations which control the evolution of the system are defined now

$$\begin{aligned} \frac{\mathrm{d}q}{\mathrm{d}t} &= \frac{\partial H(q,p)}{\partial p} \\ \frac{\mathrm{d}p}{\mathrm{d}t} &= -\frac{\partial H(q,p)}{\partial q}\ , \end{aligned} \tag{3.7}$$

or equivalently

$$\begin{aligned} &\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix} q(t) \\ p(t) \end{bmatrix} = J^{-1}\nabla H(q(t),p(t)), \\ &\text{where} \quad J = \begin{bmatrix} 0_{d\times d} & -\mathrm{Id}_{d\times d} \\ \mathrm{Id}_{d\times d} & 0_{d\times d} \end{bmatrix}, \quad \nabla H(q,p) = \begin{bmatrix} \nabla U(q) \\ p \end{bmatrix}. \end{aligned}$$

Denote by $(\varphi_t)_{t\geqslant 0}$ the flow associated to (3.7), that is, for $t \geqslant 0$, φ_t is the function $\mathbb{R}^{2d} \to \mathbb{R}^{2d}$ which associates to each (q_0,p_0) the value at time t of the (unique) solution of (3.7) equal to (q_0,p_0) at time $t = 0$.

An important feature of these dynamics is that they preserve the Hamiltonian. Indeed, any solution $(q(t),p(t))$ of this equation verifies

$$\frac{\mathrm{d}}{\mathrm{d}t}H(q(t),p(t)) = \nabla H(q(t),p(t))J^{-1}\nabla H(q(t),p(t)) = 0\ ,$$

which is equivalent to say that for each $t \geqslant 0$, $H \circ \varphi_t = H$. A mapping $\Phi\colon \mathbb{R}^{2d} \to \mathbb{R}^{2d}$ is said to be symplectic if, at each point $(p,q) \in \mathbb{R}^{2d}$, $\mathrm{J}_\Phi(q,p)^T J \mathrm{J}_\phi(q,p) = J$, where $\mathrm{J}_\phi(q,p)$ denotes the $2d \times 2d$ Jacobian of Φ. Symplectiness is crucial because it implies volume preservation on $\mathbb{R}^{2d}$. In particular, for $t \in \mathbb{R}$, φ_t is symplectic. Denote by S the momentum flip involution, $S\colon (q,p) \mapsto (q,-p)$. A mapping $\Phi\colon \mathbb{R}^{2d} \to \mathbb{R}^{2d}$ is said to be *reversible* w.r.t.S (S-reversible) if $\Phi \circ S = S \circ \Phi^{-1}$. By uniqueness of solutions of (3.7), for all $t \in \mathbb{R}$, the flow φ_t is a S-reversible mapping. More precisely, with an initial state (q_0,p_0) and $(q(t),p(t)) = \varphi_t(q_0,p_0)$ the state at time t, then

$$\varphi_t(q(t),-p(t)) = \varphi_t\left(S \circ \varphi_t\left(q_0,p_0\right)\right) = (q_0,-p_0) = S\left(q_0,p_0\right)\ .$$

Thus, as φ_t preserves the Hamiltonian and oriented volume, its preserves as well the probability measure $\bar{\pi}$, that is for $t \in \mathbb{R}, \mathsf{A} \in \mathcal{B}(\mathbb{R}^{2d})$, $\bar{\pi}(\varphi_t(\mathsf{A})) = \bar{\pi}(\mathsf{A})$, where we have written for $\mathsf{A} \in \mathcal{B}(\mathbb{R}^{2d})$, $\bar{\pi}(\mathsf{A}) = \int_{\mathbb{R}^{2d}} \bar{\pi}(q,p)\mathbb{1}_{\mathsf{A}}(q,p)\mathrm{d}q\mathrm{d}p$.

The Hamiltonian Monte Carlo aims at using samples produces using a solution of (3.7). However, as an explicit solution is usually not accessible, a discretization must be used instead. A key point is that the discretization must be symplectic and S-reversible as well, which can be satisfied using the Störmer Verlet integrator, or Leap-frog integrator. It is interesting to note that if the Hamiltonian is not exactly preserved through the discretization, it is "approximately" preserved, and this difference will be corrected using a Metropolis-Hastings step, similarly to MALA.

A single leapfrog step $\mathrm{F}_\eta(q_0, p_0) = (q_1, p_1)$ is a combination of 3 shear transforms:

$$p_{1/2} = p_0 - \eta/2\nabla U(q_0) \tag{3.8}$$

$$q_1 = q_0 + \eta p_{1/2}, \tag{3.9}$$

$$p_1 = p_{1/2} - \eta/2\nabla U(q_1)\ . \tag{3.10}$$

In general, one chooses a discretization step η and a number of steps K (thus equivalent to integrating the Hamiltonian system (3.7) during a physical time $K\eta$). The sequence $(q_\ell, p_\ell)_{l=0}^K$ is an approximation of the solution of (3.7) at times $\{\ell\eta \colon \ell \in \{0, \dots, K\}\}$ started at (q_0, p_0).

In the end, the HMC algorithm implements a special case of a Metropolis Hastings algorithm, in which we sample a proposal, and then accept or reject it with some probability designed to make the kernel reversible w.r.t. the target distribution. The HMC algorithm with K LeapFrog steps, a stepsize η goes as follows:

- Refresh the momentum, *i.e.* draw $p \sim \mathrm{N}(0, M)$.
- Compute propose $(q', p') = \mathrm{F}_\eta^K(q, p)$.
- With probability $\alpha^{\mathrm{HMC}}((q,p),(q',p')) = 1 \wedge \frac{\bar{\pi}(q',p')}{\bar{\pi}(q,p)}$ accept and set current point to (q', p'). Else, reject and set current point to $(q, -p)$.

Note in particular hat setting $K = 1$ reduces the algorithm to MALA.

The HMC algorithm has been very popular and considered state-of-the-art in the recent literature, along with its extensions , for example [LHS17a; GC11], and in particular [HG+14a].

Pseudo-marginal MCMC Interestingly enough, from an unbiased estimator of the normalising constant Z of $\tilde{\pi}$, it is possible to build an MCMC procedure that samples from π. Let $\hat{\mathrm{Z}}(X)$ denote an unbiased estimator of Z for a density q, that is $\int_{\mathbb{R}^d} \hat{\mathrm{Z}}(x)q(x)\mathrm{d}x = \mathrm{Z}$. Build the procedure as follows:

- Draw $X_{1:K} \sim q$ [Note here that $X_{1:K}$ do not necessarily need to be independent, but more on that later; see [Lag+21; Thi+21a]].
- Compute the estimators $\{\hat{\mathrm{Z}}(X_k)\}_{1 \leqslant k \leqslant K}$.
- Sample $Y \sim \mathrm{Cat}(\hat{\mathrm{Z}}(X_k)/\{\sum_{i=1}^K \hat{\mathrm{Z}}(X_i))\})$, where $\mathrm{Cat}(\omega_k)$ is the categorical distribution over $\{1, \dots, K\}$ associated to the probabilities $\{\omega_k\}_{1 \leqslant k \leqslant K}$.

In the case where $\hat{\mathrm{Z}}(x) = \tilde{\pi}(x)/q(x)$ the importance estimate with 1 sample, this procedure describes exactly the Sampling Importance Resampling procedure [Rub87]. This procedure, unfortunately, is only valid when K goes to ∞, see [SG92; SBH03]. However, this inexact procedure can be turned into a invariant MCMC procedure w.r.t. π, see in the Sampling Importance Resampling case [ADH10; ALV+18]. It goes as follows. Given the current point Y,

- Draw $i \sim \mathcal{U}\{1 \dots, K\}$ and draw $X_{1:K\setminus\{i\}} \sim q$ [Note here that $X_{1:K\setminus\{i\}}$ do not necessarily need to be independent, but more on that later], and set $X_1 = Y$.

- Compute the estimators $\{\hat{\mathrm{Z}}(X_k)\}_{1\leqslant k\leqslant K}$.
- Sample $Y \sim \mathrm{Cat}(\hat{\mathrm{Z}}(X_k)/\{\sum_{i=1}^K \hat{\mathrm{Z}}(X_i))\})$.

This procedure designs a MCMC kernel invariant w.r.t. π, defined for $x \in \mathbb{R}^d, \mathsf{A} \in \mathcal{B}(\mathbb{R}^d)$ by

$$P_K(x, \mathsf{A}) = \frac{1}{N} \int \sum_{\ell=1}^N \delta_x(\mathrm{d}x_\ell) \prod_{j \mathrm{ne} \ell} q(\mathrm{d}x_j) \sum_{i=1}^N \frac{\hat{\mathrm{Z}}(x_i)}{\sum_{\ell=1}^N \hat{\mathrm{Z}}(x_\ell)} \mathbb{1}_{\mathsf{A}}(x^i) \ .$$

Moreover, we can compute

$$\int \pi(\mathrm{d}x) P_K(x, \mathsf{A}) = N^{-1} \int \pi(\mathrm{d}x) \sum_{\ell=1}^N \delta_x(\mathrm{d}x_\ell) \prod_{j \mathrm{ne} \ell} q(\mathrm{d}x_j) \sum_{i=1}^N \frac{\hat{\mathrm{Z}}(x_i)}{\sum_{\ell=1}^N \hat{\mathrm{Z}}(x_\ell)} \mathbb{1}_{\mathsf{A}}(x_i) \tag{3.11}$$

$$= N^{-1} \int (\sum_{\ell=1}^N \hat{\mathrm{Z}}(x_\ell)) \prod_{j=1}^N q(\mathrm{d}x_j) \sum_{i=1}^N \frac{\hat{\mathrm{Z}}(x_i)}{\sum_{\ell=1}^N \hat{\mathrm{Z}}(x_\ell)} \mathbb{1}_{\mathsf{A}}(x_i) \tag{3.12}$$

$$= N^{-1} \int \prod_{j=1}^N q(\mathrm{d}x_j) \sum_{i=1}^N \hat{\mathrm{Z}}(x_i) \mathbb{1}_{\mathsf{A}}(x_i) = \pi(\mathsf{A}) \tag{3.13}$$

The relationship to pseudo-marginal MCMC lays in the fact that one can do these computations by defining an extending target $\bar{\pi}(x, k)$ in the case of ISIR for example, where k denotes the index of the chosen particle in $\{1 \ldots, K\}$.

3.3 Generative models and approximate simulation

We have covered in the previous section algorithms allowing to sample *exactly* (or at least *asymptotically exactly*) according to some target distribution π. These methods provide a way with convergence guarantees to obtain samples from π. However, other directions have been considered in the recent literature.

Normalizing Flows Normalizing flows [TV10; TT13; Pap+19; KPB20] are a way of building expressive probability distributions that can serve as a proxy of the target π and provide after training a cheap way of producing samples. Normalizing flows are built from a simple density "pushed" through a sequence of invertible and differentiable transformations to produce a richer and more expressive distribution. The idea is that the repeated application of even very simple transformations (thus easily tractable) can produce a very flexible distribution in the end.

The idea is based on the change of variables formula. Let T denote a C^1 diffeomorphism, that is a continuously differentiable invertible mapping whose inverse is also continuously differentiable, and p be a density on $\mathbb{R}^d$.

Then, we can write the density of the pushforward of p by T, noted $\mathrm{T}_{\#}p$ or p_{T}, as

$$p_{\mathrm{T}}(x) = p(\mathrm{T}^{-1}(x)) \mathrm{J}_{\mathrm{T}^{-1}}(x) \ , \tag{3.14}$$

where $\mathrm{J}_T(y)$ denotes the absolute value of the Jacobian determinant of the C^1 function T applied at y. The pushforward density of p by T is the density of the random variable $x = \mathrm{T}(u)$, where $u \sim p$. Thus, as long as the Jacobian determinant is tractable and easy to evaluate, we can build pushforward densities from a simple density p easy to sample from and any C^1 diffeomorphism. The expressivity of normalizing flows comes from the fact that one can compose many of those transformations. That is, $\mathrm{T} = \mathrm{T}_K \circ \cdots \circ \mathrm{T}_1$, and T_k are all C^1 diffeomorphisms with tractable Jacobian. The name "flow" stems from the fact that a sample u from p is gradually transported by the sequence of mappings $\mathrm{T}_1, \ldots, \mathrm{T}_K$, while the name "normalizing" stems from the fact that a sample x from the pushforward (potentially

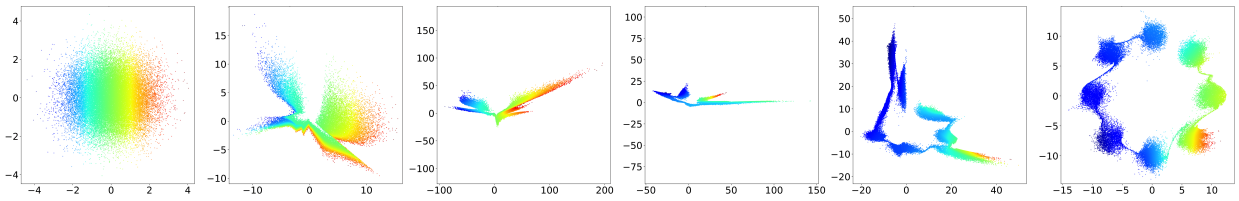


Figure 3.1: Consecutive outputs of normalizing flows. Left: prior normal distribution, then successive effect of the 5 transformations.

complex) p_{T} goes through the inverse flow and the mappings $\mathrm{T}_K^{-1}, \dots, \mathrm{T}_1^{-1}$ to the simple reference density p often taken as a multivariate normal (hence are "normalized" in some way), see Figure 8.6 for a multivariate normal transformed into a mixture of 8 Gaussian distributions. Many works in the recent literature have introduced different ways of building such transformations T, trading off between expressiveness and tractability of the Jacobian and the inverse T^{-1}, see [Pap+19] for a full review. These models however are limited. Indeed, as they build on a deterministic structure, normalizing flows can struggle to cover and represent difficult distributions with topological constraints (see [Cor+20]) or by essence stochastic transformations, such as diffusion processes or convolutions.

To that effect, stochastic processes linked to normalizing flows have been introduced lately in the literature, [WKN20c; Hod+20; Thi+20b; HHS21] in order to increase efficiently the expressiveness of such transformations without dramatic increase in the numbers of parameters.

Deep Latent Generative Models Other type of generative models considered in this thesis include of course Deep Latent Generative Models, introduced in Section 2.3.

References

[AB04] Felix V Agakov and David Barber. "An auxiliary variational method". In: *International Conference on Neural Information Processing*. Springer. 2004, pp. 561–566.

[AC75] Samuel M Allen and John W Cahn. "Coherent and incoherent equilibria in iron-rich iron-aluminum alloys". In: *Acta Metallurgica* 23.9 (1975), pp. 1017–1026. ISSN: 0001-6160. DOI: https://doi.org/10.1016/0001-6160(75)90106-6. URL: https://www.sciencedirect.com/science/article/pii/0001616075901066.

[ACB17] Martin Arjovsky, Soumith Chintala, and Léon Bottou. "Wasserstein generative adversarial networks". In: *International conference on machine learning*. PMLR. 2017, pp. 214–223.

[ADH10] Christophe Andrieu, Arnaud Doucet, and Roman Holenstein. "Particle Markov chain Monte Carlo methods". In: *Journal of the Royal Statistical Society: Series B* 72.3 (2010), pp. 269–342.

[Aga+17] Sergios Agapiou, Omiros Papaspiliopoulos, Daniel Sanz-Alonso, and AM Stuart. "Importance sampling: Intrinsic dimension and computational cost". In: *Statistical Science* (2017), pp. 405–431.

[AKS19] MS Albergo, G Kanwar, and PE Shanahan. "Flow-based generative models for Markov chain Monte Carlo in lattice field theory". In: *Physical Review D* 100.3 (2019), p. 034515.

[AKW12] Sungjin Ahn, Anoop Korattikara, and Max Welling. "Bayesian posterior sampling via stochastic gradient Fisher scoring". In: *arXiv preprint arXiv:1206.6380* (2012).

[AL19a] C. Andrieu and S. Livingstone. "Peskun-Tierney ordering for Markov chain and process Monte Carlo: beyond the reversible scenario". In: *arXiv preprint arXiv:1906.06197* (2019).

[AL19b] Christophe Andrieu and Samuel Livingstone. "Peskun-Tierney ordering for Markov chain and process Monte Carlo: beyond the reversible scenario". In: *arXiv preprint arXiv:1906.06197* (2019).

[ALV+18] Christophe Andrieu, Anthony Lee, Matti Vihola, et al. "Uniform ergodicity of the iterated conditional SMC and geometric ergodicity of particle Gibbs samplers". In: *Bernoulli* 24.2 (2018), pp. 842–872.

[AM21] Ömer Deniz Akyildiz and Joaquín Mıéguez. "Convergence rates for optimised adaptive importance samplers". In: *Statistics and Computing* 31.2 (2021), pp. 1–17.

[And+18] Christophe Andrieu, Arnaud Doucet, Sinan Yıldırım, and Nicolas Chopin. "On the utility of Metropolis-Hastings with asymmetric acceptance ratio". In: *arXiv preprint arXiv:1803.09527* (2018).

[Ard+19] Lynton Ardizzone, Carsten Lüth, Jakob Kruse, Carsten Rother, and Ullrich Köthe. *Guided Image Generation with Conditional Invertible Neural Networks*. 2019. arXiv: 1907.02392 [cs.CV].

[ASW14] Sungjin Ahn, Babak Shahbaba, and Max Welling. "Distributed stochastic gradient MCMC". In: *International conference on machine learning*. PMLR. 2014, pp. 1044–1052.

[AT08] Christophe Andrieu and Johannes Thoms. "A tutorial on adaptive MCMC". In: *Statistics and computing* 18.4 (2008), pp. 343–373.

[Aza+18] Samaneh Azadi, Catherine Olsson, Trevor Darrell, Ian Goodfellow, and Augustus Odena. "Discriminator Rejection Sampling". In: *International Conference on Learning Representations*. 2018.

[Aza+19] Samaneh Azadi, Catherine Olsson, Trevor Darrell, Ian Goodfellow, and Augustus Odena. "Discriminator Rejection Sampling". In: *arXiv:1810.06758* (2019). eprint: 1810.06758 (stat.ML).

[BDH17] Rémi Bardenet, Arnaud Doucet, and Christopher C Holmes. "On Markov chain Monte Carlo methods for tall data". In: *Journal of Machine Learning Research* 18.47 (2017).

[BDM12] Mylène Bédard, Randal Douc, and Eric Moulines. "Scaling analysis of multiple-try MCMC methods". In: *Stochastic Processes and their Applications* 122.3 (2012), pp. 758–786.

[BDM18] Nicolas Brosse, Alain Durmus, and Eric Moulines. "The promises and pitfalls of stochastic gradient Langevin dynamics". In: *Advances in Neural Information Processing Systems* 31 (2018).

[Bes+13] Alexandros Beskos, Natesh Pillai, Gareth Roberts, Jesus-Maria Sanz-Serna, and Andrew Stuart. "Optimal tuning of the hybrid Monte Carlo algorithm". In: *Bernoulli* 19.5A (2013), pp. 1501–1534.

[Bes94a] J.E. Besag. "Comments on "Representations of knowledge in complex systems" by U. Grenander and M. Miller". In: *J. Roy. Statist. Soc. Ser. B* 56 (1994), pp. 591–592.

[Bes94b] J.E. Besag. "Comments on "Representations of knowledge in complex systems" by U. Grenander and M. Miller". In: *J. Roy. Statist. Soc. Ser. B* 56 (1994), pp. 591–592.

[BGS15] Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. "Importance weighted autoencoders". In: *arXiv preprint arXiv:1509.00519* (2015).

[Bin+18] Eli Bingham, Jonathan P. Chen, Martin Jankowiak, Fritz Obermeyer, Neeraj Pradhan, Theofanis Karaletsos, Rohit Singh, Paul Szerlip, Paul Horsfall, and Noah D. Goodman. *Pyro: Deep Universal Probabilistic Programming*. 2018. arXiv: 1810.09538 [cs.LG].

[Bin+19] Eli Bingham, Jonathan P Chen, Martin Jankowiak, Fritz Obermeyer, Neeraj Pradhan, Theofanis Karaletsos, Rohit Singh, Paul Szerlip, Paul Horsfall, and Noah D Goodman. "Pyro: Deep universal probabilistic programming". In: *Journal of Machine Learning Research* 20.1 (2019), pp. 973–978.

[BJ18] N. Bou-Rabee and S.-S. Jesús Marıéa. "Geometric Integrators and the Hamiltonian Monte Carlo method". In: *Acta Numerica* (2018), pp. 1–92.

[Bon+11] Nicolas Bonneel, Michiel Van De Panne, Sylvain Paris, and Wolfgang Heidrich. "Displacement interpolation using Lagrangian mass transport". In: *Proceedings of the 2011 SIGGRAPH Asia Conference*. 2011, pp. 1–12.

[BR17] Joris Bierkens and Gareth Roberts. "A piecewise deterministic scaling limit of lifted Metropolis–Hastings in the Curie–Weiss model". In: *Ann. Appl. Probab.* 27.2 (Apr. 2017), pp. 846–882. DOI: 10.1214/16-AAP1217. URL: https://doi.org/10.1214/16-AAP1217.

[Bro+11] Steve Brooks, Andrew Gelman, Galin Jones, and Xiao-Li Meng. *Handbook of Markov chain Monte Carlo*. CRC press, 2011.

[BZM20] Ricardo Baptista, Olivier Zahm, and Youssef Marzouk. "An adaptive transport framework for joint and conditional density estimation". In: *arXiv preprint arXiv:2009.10303* (2020).

[CDO+11] Su Chen, Josef Dick, Art B Owen, et al. "Consistency of Markov chain quasi-Monte Carlo on continuous state spaces". In: *The Annals of Statistics* 39.2 (2011), pp. 673–701.

[CDS18a] Anthony L Caterini, Arnaud Doucet, and Dino Sejdinovic. "Hamiltonian variational auto-encoder". In: *Advances in Neural Information Processing Systems*. 2018, pp. 8167–8177.

[CDS18b] Anthony L Caterini, Arnaud Doucet, and Dino Sejdinovic. "Hamiltonian variational auto-encoder". In: *Advances in Neural Information Processing Systems*. 2018, pp. 8167–8177.

[CFG14] Tianqi Chen, Emily Fox, and Carlos Guestrin. "Stochastic gradient Hamiltonian Monte Carlo". In: *International conference on machine learning*. PMLR. 2014, pp. 1683–1691.

[Cha+18] Niladri Chatterji, Nicolas Flammarion, Yian Ma, Peter Bartlett, and Michael Jordan. "On the theory of variance reduction for stochastic gradient Monte Carlo". In: *International Conference on Machine Learning*. PMLR. 2018, pp. 764–773.

[Che+20a] Tong Che, Ruixiang ZHANG, Jascha Sohl-Dickstein, Hugo Larochelle, Liam Paull, Yuan Cao, and Yoshua Bengio. "Your GAN is Secretly an Energy-based Model and You Should Use Discriminator Driven Latent Sampling". In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin. Vol. 33. Curran Associates, Inc., 2020, pp. 12275–12287. URL: https://proceedings.neurips.cc/paper/2020/file/90525e70b7842930586545c6f1c9310c-Paper.pdf.

[Che+20b] Tong Che, Ruixiang Zhang, Jascha Sohl-Dickstein, Hugo Larochelle, Liam Paull, Yuan Cao, and Yoshua Bengio. "Your GAN is Secretly an Energy-based Model and You Should use Discriminator Driven Latent Sampling". In: *arXiv preprint arXiv:2003.06060* (2020).

[CJ21] Adam D Cobb and Brian Jalaian. "Scaling Hamiltonian Monte Carlo inference for Bayesian neural networks with symmetric splitting". In: *Uncertainty in Artificial Intelligence*. PMLR. 2021, pp. 675–685.

[CL07a] Radu V Craiu and Christiane Lemieux. "Acceleration of the multiple-try Metropolis algorithm using antithetic and stratified sampling". In: *Statistics and Computing* 17.2 (2007), p. 109.

[CL07b] Radu V. Craiu and Christiane Lemieux. "Acceleration of the Multiple-Try Metropolis algorithm using antithetic and stratified sampling". In: *Statistics and Computing* 17.2 (2007), p. 109. ISSN: 1573-1375. DOI: 10.1007/s11222-006-9009-4. URL: https://doi.org/10.1007/s11222-006-9009-4.

[CLP99] F. Chen, L. Lovász, and I. Pak. "Lifting Markov chains to speed up mixing". In: *Annual ACM Symposium on Theory of Computing (Atlanta, GA, 1999)*. ACM, New York, 1999, pp. 275–281. DOI: 10.1145/301250.301315. URL: https://doi.org/10.1145/301250.301315.

[CMD17] Chris Cremer, Quaid Morris, and David Duvenaud. "Reinterpreting importance-weighted autoencoders". In: *arXiv preprint arXiv:1704.02916* (2017).

[Cor+19] Rob Cornish, Anthony L Caterini, George Deligiannidis, and Arnaud Doucet. "Relaxing bijectivity constraints with continuously indexed normalising flows". In: *arXiv preprint arXiv:1909.13833* (2019).

[Cor+20] Rob Cornish, Anthony Caterini, George Deligiannidis, and Arnaud Doucet. "Relaxing bijectivity constraints with continuously indexed normalising flows". In: *International conference on machine learning*. PMLR. 2020, pp. 2133–2143.

[CR10] N. Chopin and C. P. Robert. "Properties of nested sampling". In: *Biometrika* 97.3 (2010), pp. 741–755.

[Cro98] Gavin E Crooks. "Nonequilibrium measurements of free energy differences for microscopically reversible Markovian systems". In: *Journal of Statistical Physics* 90.5-6 (1998), pp. 1481–1487.

[CS97] Ming-Hui Chen and Qi-Man Shao. "On Monte Carlo Methods for Estimating Ratios of Normalizing Constants". In: *Annals of Statistics* 25 (Aug. 1997). DOI: 10.1214/aos/1031594732.

[CTA19] Nicola De Cao, Ivan Titov, and Wilker Aziz. "Block Neural Autoregressive Flow". In: *UAI*. 2019.

[Dal17] Arnak S. Dalalyan. "Theoretical guarantees for approximate sampling from smooth and log-concave densities". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 79.3 (2017), pp. 651–676. DOI: 10.1111/rssb.12183. eprint: https://rss.onlinelibrary.wiley.com/doi/pdf/10.1111/rssb.12183. URL: https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/rssb.12183.

[Dax+20] Erik Daxberger, Eric Nalisnick, James Allingham, Javier Antorán, and José Miguel Hernández-Lobato. "Expressive yet tractable Bayesian deep learning via subnetwork inference". In: (2020).

[DB+16] Kushal Kr Dey, Sourabh Bhattacharya, et al. "On geometric ergodicity of additive and multiplicative transformation-based Markov Chain Monte Carlo in high dimensions". In: *Brazilian Journal of Probability and Statistics* 30.4 (2016), pp. 570–613.

[DB14] Somak Dutta and Sourabh Bhattacharya. "Markov chain Monte Carlo based on deterministic transformations". In: *Statistical Methodology* 16 (Jan. 2014), pp. 100–116. ISSN: 1572-3127. DOI: 10.1016/j.stamet.2013.08.006.

[DDJ06a] P. Del Moral, A. Doucet, and A. Jasra. "Sequential Monte Carlo samplers". In: *Journal of the Royal Statistical Society: Series B* 68.3 (2006), pp. 411–436.

[DDJ06b] Pierre Del Moral, Arnaud Doucet, and Ajay Jasra. "Sequential Monte Carlo samplers". In: *Journal of the Royal Statistical Society: Series* B 68.3 (2006), pp. 411–436.

[DF19] Xinqiang Ding and David J Freedman. "Learning Deep Generative Models with Annealed Importance Sampling". In: *arXiv preprint arXiv:1906.04904* (2019).

[DHN00] Persi Diaconis, Susan Holmes, and Radford M. Neal. "Analysis of a nonreversible Markov chain sampler". In: *Ann. Appl. Probab.* 10.3 (2000), pp. 726–752. ISSN: 1050-5164. DOI: 10.1214/aoap/1019487508. URL: https://doi.org/10.1214/aoap/1019487508.

[Din+14] Nan Ding, Youhan Fang, Ryan Babbush, Changyou Chen, Robert D Skeel, and Hartmut Neven. "Bayesian sampling using stochastic gradient thermostats". In: *Advances in neural information processing systems* 27 (2014).

[DK16] Samuel Dodge and Lina Karam. "Understanding how image quality affects deep neural networks". In: *2016 eighth international conference on quality of multimedia experience (QoMEX)*. IEEE. 2016, pp. 1–6.

[DK19] Arnak S Dalalyan and Avetik Karagulyan. "User-friendly guarantees for the Langevin Monte Carlo with inaccurate gradient". In: *Stochastic Processes and their Applications* 129.12 (2019), pp. 5278–5311.

[DM17] Alain Durmus and Eric Moulines. "Nonasymptotic convergence analysis for the unadjusted Langevin algorithm". In: *The Annals of Applied Probability* 27.3 (2017), pp. 1551–1587.

[DMS17] Alain Durmus, Eric Moulines, and Eero Saksman. "On the convergence of Hamiltonian Monte Carlo". In: *Accepted for publication in Ann. Statist.* (2017).

[Dou+11] Randal Douc, Aurélien Garivier, Eric Moulines, Jimmy Olsson, et al. "Sequential Monte Carlo smoothing for general state space hidden Markov models". In: *Annals of Applied Probability* 21.6 (2011), pp. 2109–2145.

[Dou+15] Arnaud Doucet, Michael K Pitt, George Deligiannidis, and Robert Kohn. "Efficient implementation of Markov chain Monte Carlo when using an unbiased likelihood estimator". In: *Biometrika* 102.2 (2015), pp. 295–313.

[Dou+18] R. Douc, E. Moulines, P. Priouret, and P. Soulier. *Markov chains*. Springer Series in Operations Research and Financial Engineering. Springer, Cham, 2018, pp. xviii+757. ISBN: 978-3-319-97703-4; 978-3-319-97704-1. DOI: 10.1007/978-3-319-97704-1. URL: https://doi.org/10.1007/978-3-319-97704-1.

[DS18] Justin Domke and Daniel R Sheldon. "Importance weighting and variational inference". In: *Advances in neural information processing systems*. 2018, pp. 4470–4479.

[DSB16] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. "Density estimation using Real NVP". In: *arXiv preprint arXiv:1605.08803* (2016).

[DSB17] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. *Density estimation using Real NVP*. 2017. arXiv: 1605.08803 [cs.LG].

[Dua+87] Simon Duane, A.D. Kennedy, Brian J. Pendleton, and Duncan Roweth. "Hybrid Monte Carlo". In: *Physics Letters B* 195.2 (1987), pp. 216–222. ISSN: 0370-2693. DOI: https://doi.org/10.1016/0370-2693(87)91197-X. URL: http://www.sciencedirect.com/science/article/pii/037026938791197X.

[Dus+20] Michael Dusenberry, Ghassen Jerfel, Yeming Wen, Yian Ma, Jasper Snoek, Katherine Heller, Balaji Lakshminarayanan, and Dustin Tran. "Efficient and scalable Bayesian neural nets with rank-1 factors". In: *International conference on machine learning*. PMLR. 2020, pp. 2782–2792.

[EM12] Tarek A El Moselhy and Youssef M Marzouk. "Bayesian inference with optimal maps". In: *Journal of Computational Physics* 231.23 (2012), pp. 7815–7850.

[Fal19] WA Falcon. "PyTorch Lightning". In: *GitHub. Note: https://github.com/PyTorchLightning/pytorch-lightning* 3 (2019).

[Foo+19] Andrew YK Foong, Yingzhen Li, José Miguel Hernández-Lobato, and Richard E Turner. "'In-Between'Uncertainty in Bayesian Neural Networks". In: *arXiv preprint arXiv:1906.11537* (2019).

[Foo+20] Andrew Foong, David Burt, Yingzhen Li, and Richard Turner. "On the expressiveness of approximate inference in Bayesian neural networks". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 15897–15908.

[Fra+20] Guilherme Franca, Jeremias Sulam, Daniel P Robinson, and René Vidal. "Conformal symplectic and relativistic optimization". In: *Journal of Statistical Mechanics: Theory and Experiment* 2020.12 (2020), p. 124008.

[FSG20] Sebastian Farquhar, Lewis Smith, and Yarin Gal. "Liberty or depth: Deep Bayesian neural nets do not need complex weight posterior approximations". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 4346–4357.

[FSS14] Youhan Fang, Jesus-Maria Sanz-Serna, and Robert D Skeel. "Compressible generalized hybrid Monte Carlo". In: *The Journal of chemical physics* 140.17 (2014), p. 174108.

[FW12] Nial Friel and Jason Wyse. "Estimating the evidence–a review". In: *Statistica Neerlandica* 66.3 (2012), pp. 288–308.

[Gal16] Yarin Gal. "Uncertainty in Deep Learning". In: 2016.

[Gaw+21] Jakob Gawlikowski, Cedrique Rovile Njieutcheu Tassi, Mohsin Ali, Jongseok Lee, Matthias Humt, Jianxiang Feng, Anna Kruspe, Rudolph Triebel, Peter Jung, Ribana Roscher, et al. "A survey of uncertainty in deep neural networks". In: *arXiv preprint arXiv:2107.03342* (2021).

[GBC16] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[GC11] Mark Girolami and Ben Calderhead. "Riemann manifold Langevin and Hamiltonian Monte Carlo methods". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 73.2 (2011), pp. 123–214.

[GF21] Adrià Garriga-Alonso and Vincent Fortuin. "Exact Langevin dynamics with stochastic gradients". In: *arXiv preprint arXiv:2102.01691* (2021).

[GG16] Yarin Gal and Zoubin Ghahramani. "Dropout as a Bayesian approximation: Representing model uncertainty in deep learning". In: *international conference on machine learning*. PMLR. 2016, pp. 1050–1059.

[GGA15] Roger B Grosse, Zoubin Ghahramani, and Ryan P Adams. "Sandwiching the marginal likelihood using bidirectional Monte Carlo". In: *arXiv preprint arXiv:1511.02543* (2015).

[GGR97] Andrew Gelman, Walter R Gilks, and Gareth O Roberts. "Weak convergence and optimal scaling of random walk Metropolis algorithms". In: *The annals of applied probability* 7.1 (1997), pp. 110–120.

[GHK17] Yarin Gal, Jiri Hron, and Alex Kendall. "Concrete dropout". In: *Advances in neural information processing systems* 30 (2017).

[GM98] Andrew Gelman and Xiao-Li Meng. "Simulating normalizing constants: From importance sampling to bridge sampling to path sampling". In: *Statistical science* (1998), pp. 163–185.

[Goo+14a] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. "Generative adversarial nets". In: *Advances in neural information processing systems* 27 (2014).

[Goo+14b] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. "Generative Adversarial Nets". In: *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*. NIPS'14. Montreal, Canada: MIT Press, 2014, pp. 2672–2680.

[Goy+17] Anirudh Goyal Alias Parth Goyal, Nan Rosemary Ke, Surya Ganguli, and Yoshua Bengio. "Variational walkback: Learning a transition operator as a stochastic recurrent net". In: *Advances in Neural Information Processing Systems*. 2017, pp. 4392–4402.

[Gra11] Alex Graves. "Practical variational inference for neural networks". In: *Advances in neural information processing systems* 24 (2011).

[GRV21] Marylou Gabrié, Grant M. Rotskoff, and Eric Vanden-Eijnden. "Adaptive Monte Carlo augmented with normalizing flows". In: *arXiv preprint arXiv:2105.12603* (2021).

[Gus98] Paul Gustafson. "A guided walk Metropolis algorithm". In: *Statistics and computing* 8.4 (1998), pp. 357–364.

[HA15] José Miguel Hernández-Lobato and Ryan Adams. "Probabilistic backpropagation for scalable learning of Bayesian neural networks". In: *International conference on machine learning*. PMLR. 2015, pp. 1861–1869.

[HAB19] Matthias Hein, Maksym Andriushchenko, and Julian Bitterwolf. "Why relu networks yield high-confidence predictions far away from the training data and how to mitigate the problem". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 41–50.

[Har82] P. Hartman. *Ordinary Differential Equations: Second Edition*. Classics in Applied Mathematics. Society for Industrial and Applied Mathematics (SIAM, 3600 Market Street, Floor 6, Philadelphia, PA 19104), 1982. ISBN: 9780898719222. URL: `https://books.google.fr/books?id=NEkkJ93O9okC`.

[Hen+20] Jeremy Heng, Adrian N Bishop, George Deligiannidis, and Arnaud Doucet. "Controlled sequential Monte Carlo". In: *The Annals of Statistics* 48.5 (2020), pp. 2904–2929.

[Heu+17] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. "GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett. Vol. 30. Curran Associates, Inc., 2017. URL: `https://proceedings.neurips.cc/paper/2017/file/8a1d694707eb0fefe65871369074926d-Paper.pdf`.

[HG+14a] Matthew D Hoffman, Andrew Gelman, et al. "The No-U-Turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo." In: *J. Mach. Learn. Res.* 15.1 (2014), pp. 1593–1623.

[HG+14b] Matthew D Hoffman, Andrew Gelman, et al. "The No-U-Turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo." In: *J. Mach. Learn. Res.* 15.1 (2014), pp. 1593–1623.

[HH80] P. Hall and C. Heyde. *Martingale Limit Theory and Its Application*. Academic Press, 1980.

[HHS21] Paul Hagemann, Johannes Hertrich, and Gabriele Steidl. "Stochastic normalizing flows for inverse problems: a Markov Chains viewpoint". In: *arXiv preprint arXiv:2109.11375* (2021).

[HM20] Matthew Hoffman and Yian Ma. "Black-box variational inference as a parametric approximation to Langevin dynamics". In: *International Conference on Machine Learning*. PMLR. 2020, pp. 4324–4341.

[Ho+19] Jonathan Ho, Xi Chen, Aravind Srinivas, Yan Duan, and Pieter Abbeel. "Flow++: Improving flow-based generative models with variational dequantization and architecture design". In: *arXiv preprint arXiv:1902.00275* (2019).

[Hod+20] Liam Hodgkinson, Chris van der Heide, Fred Roosta, and Michael W Mahoney. "Stochastic normalizing flows". In: *arXiv preprint arXiv:2002.09547* (2020).

[Hof+19] Matthew Hoffman, Pavel Sountsov, Joshua V Dillon, Ian Langmore, Dustin Tran, and Srinivas Vasudevan. "NeuTra-lizing Bad Geometry in Hamiltonian Monte Carlo Using Neural Transport". In: *arXiv preprint arXiv:1903.03704* (2019).

[Hof17] Matthew D. Hoffman. "Learning Deep Latent Gaussian Models with Markov Chain Monte Carlo". In: *Proceedings of the 34th International Conference on Machine Learning*. Ed. by Doina Precup and Yee Whye Teh. Vol. 70. Proceedings of Machine Learning Research. International Convention Centre, Sydney, Australia: PMLR, June 2017, pp. 1510–1519.

[Hor91] Alan M Horowitz. "A generalized guided Monte Carlo algorithm". In: *Physics Letters B* 268.2 (1991), pp. 247–252.

[HS13] K Hukushima and Y Sakai. "An irreversible Markov-chain Monte Carlo method with skew detailed balance conditions". In: *Journal of Physics: Conference Series*. Vol. 473. 1. IOP Publishing. 2013, p. 012012.

[HST99] Heikki Haario, Eero Saksman, and Johanna Tamminen. "Adaptive proposal distribution for random walk Metropolis algorithm". English. In: *Computational Statistics* 14.3 (1999), pp. 375–395. ISSN: 0943-4062.

[Hua+18a] Chin-Wei Huang, David Krueger, Alexandre Lacoste, and Aaron Courville. "Neural autoregressive flows". In: *arXiv preprint arXiv:1804.00779* (2018).

[Hua+18b] Chin-Wei Huang, Shawn Tan, Alexandre Lacoste, and Aaron C Courville. "Improving Explorability in Variational Inference with Annealed Variational Objectives". In: *Advances in Neural Information Processing Systems*. 2018, pp. 9701–9711.

[Izm+18a] Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. "Averaging weights leads to wider optima and better generalization". In: *34th Conference on Uncertainty in Artificial Intelligence 2018, UAI 2018*. Association For Uncertainty in Artificial Intelligence (AUAI). 2018, pp. 876–885.

[Izm+18b] Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. "Averaging weights leads to wider optima and better generalization". In: *arXiv preprint arXiv:1803.05407* (2018).

[Izm+20] Pavel Izmailov, Wesley J Maddox, Polina Kirichenko, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. "Subspace inference for Bayesian deep learning". In: *Uncertainty in Artificial Intelligence*. PMLR. 2020, pp. 1169–1179.

[Izm+21] Pavel Izmailov, Sharad Vikram, Matthew D Hoffman, and Andrew Gordon Gordon Wilson. "What are Bayesian neural network posteriors really like?" In: *International Conference on Machine Learning*. PMLR. 2021, pp. 4629–4640.

[JS20] He Jia and Uros Seljak. "Normalizing constant estimation with Gaussianized bridge sampling". In: *Symposium on Advances in Approximate Bayesian Inference*. PMLR. 2020, pp. 1–14.

[KB14] Diederik P Kingma and Jimmy Ba. "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980* (2014).

[KG17] Alex Kendall and Yarin Gal. "What uncertainties do we need in Bayesian deep learning for computer vision?" In: *Advances in neural information processing systems* 30 (2017).

[Kha+18] Mohammad Khan, Didrik Nielsen, Voot Tangkaratt, Wu Lin, Yarin Gal, and Akash Srivastava. "Fast and scalable Bayesian deep learning by weight-perturbation in adam". In: *International Conference on Machine Learning*. PMLR. 2018, pp. 2611–2620.

[Kin+16a] Diederik P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. "Improving variational inference with inverse autoregressive flow". In: *arXiv preprint arXiv:1606.04934* (2016).

[Kin+16b] Durk P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. "Improved variational inference with inverse autoregressive flow". In: *Advances in neural information processing systems*. 2016, pp. 4743–4751.

[Kis65] Leslie Kish. *Survey sampling*. English. Chichester : Wiley New York, 1965, xvi, 643 p. : ISBN: 0471109495.

[Kol+19] Soheil Kolouri, Kimia Nadjahi, Umut Simsekli, Roland Badeau, and K Gustavo. "Generalized Sliced Wasserstein Distances". In: *NeurIPS 2019*. 2019.

[KP18] Andreas Kamilaris and Francesc X Prenafeta-Boldú. "Deep learning in agriculture: A survey". In: *Computers and electronics in agriculture* 147 (2018), pp. 70–90.

[KPB19] Ivan Kobyzev, Simon Prince, and Marcus A Brubaker. "Normalizing flows: Introduction and ideas". In: *arXiv preprint arXiv:1908.09257* (2019).

[KPB20] Ivan Kobyzev, Simon JD Prince, and Marcus A Brubaker. "Normalizing flows: An introduction and review of current methods". In: *IEEE transactions on pattern analysis and machine intelligence* 43.11 (2020), pp. 3964–3979.

[KW13a] Diederik P Kingma and Max Welling. "Auto-encoding variational bayes". In: *arXiv preprint arXiv:1312.6114* (2013).

[KW13b] Diederik P Kingma and Max Welling. "Auto-encoding variational bayes". In: *arXiv preprint arXiv:1312.6114* (2013).

[KW14] Diederik Kingma and Max Welling. "Auto-Encoding Variational Bayes". In: *International Conference on Learning Representations*. 2014.

[KW19] Diederik P Kingma and Max Welling. "An introduction to variational autoencoders". In: *arXiv preprint arXiv:1906.02691* (2019).

[Lag+21] Evgeny Lagutin, Daniil Selikhanovych, Achille Thin, Sergey Samsonov, Alexey Naumov, Denis Belomestny, Maxim Panov, and Eric Moulines. "Ex²MCMC: Sampling through Exploration Exploitation". In: *arXiv preprint arXiv:2111.02702* (2021).

[Law+19] Dieterich Lawson, George Tucker, Bo Dai, and Rajesh Ranganath. "Energy-inspired models: Learning with sampler-induced distributions". In: *arXiv preprint arXiv:1910.14265* (2019).

[LBH15] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. "Deep learning". In: *nature* 521.7553 (2015), pp. 436–444.

[LDM15] Fredrik Lindsten, Randal Douc, and Éric Moulines. "Uniform ergodicity of the particle Gibbs sampler". In: *Scandinavian Journal of Statistics* 42.3 (2015), pp. 775–797.

[Lee+10] Anthony Lee, Christopher Yau, Michael B Giles, Arnaud Doucet, and Christopher C Holmes. "On the utility of graphics cards to perform massively parallel simulation of advanced Monte Carlo methods". In: *Journal of computational and graphical statistics* 19.4 (2010), pp. 769–789.

[Lee11] Anthony Lee. "On auxiliary variables and many-core architectures in computational statistics". PhD thesis. University of Oxford, 2011.

[LFR19] Samuel Livingstone, Michael F Faulkner, and Gareth O Roberts. "Kinetic energy choice in Hamiltonian/hybrid Monte Carlo". In: *Biometrika* 106.2 (2019), pp. 303–319.

[LHS17a] Daniel Levy, Matthew D Hoffman, and Jascha Sohl-Dickstein. "Generalizing Hamiltonian Monte Carlo with neural networks". In: *arXiv preprint arXiv:1711.09268* (2017).

[LHS17b] Daniel Levy, Matthew D Hoffman, and Jascha Sohl-Dickstein. "Generalizing Hamiltonian Monte Carlo with neural networks". In: *arXiv preprint arXiv:1711.09268* (2017).

[LHS18] Daniel Levy, Matthew D. Hoffman, and Jascha Sohl-Dickstein. "Generalizing Hamiltonian Monte Carlo with Neural Networks". In: *International Conference on Learning Representations*. 2018.

[Li+16] Chunyuan Li, Changyou Chen, David Carlson, and Lawrence Carin. "Preconditioned stochastic gradient Langevin dynamics for deep neural networks". In: *Thirtieth AAAI Conference on Artificial Intelligence*. 2016.

[Liu+18] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. "Large-scale celebfaces attributes (celeba) dataset". In: *Retrieved August* 15.2018 (2018), p. 11.

[LLC11] Faming Liang, Chuanhai Liu, and Raymond Carroll. *Advanced Markov chain Monte Carlo methods: learning from past samples*. Vol. 714. John Wiley & Sons, 2011.

[LLW00] Jun S Liu, Faming Liang, and Wing Hung Wong. "The multiple-try method and local optimization in Metropolis sampling". In: *Journal of the American Statistical Association* 95.449 (2000), pp. 121–134.

[LM00] B. Laurent and P. Massart. "Adaptive estimation of a quadratic functional by model selection". In: *Ann. Statist.* 28.5 (Oct. 2000), pp. 1302–1338. DOI: 10.1214/aos/1015957395. URL: https://doi.org/10.1214/aos/1015957395.

[LS13] Fredrik Lindsten and Thomas B Schön. "Backward simulation methods for Monte Carlo statistical inference". In: *Foundations and Trends® in Machine Learning* 6.1 (2013), pp. 1–143.

[Lu+17] Xiaoyu Lu, Valerio Perrone, Leonard Hasenclever, Yee Whye Teh, and Sebastian Vollmer. "Relativistic Monte Carlo". In: *Artificial Intelligence and Statistics*. PMLR. 2017, pp. 1236–1245.

[LW17] Christos Louizos and Max Welling. "Multiplicative normalizing flows for variational Bayesian neural networks". In: *International Conference on Machine Learning*. PMLR. 2017, pp. 2218–2227.

[Ma+16] Yi-An Ma, Tianqi Chen, Lei Wu, and Emily B Fox. "A unifying framework for devising efficient and irreversible MCMC samplers". In: *arXiv preprint arXiv:1608.05973* (2016).

[Maa+16] Lars Maaløe, Casper Kaae Sønderby, Søren Kaae Sønderby, and Ole Winther. "Auxiliary deep generative models". In: *International conference on machine learning*. PMLR. 2016, pp. 1445–1453.

[Mad+17] Chris J Maddison, Dieterich Lawson, George Tucker, Nicolas Heess, Mohammad Norouzi, Andriy Mnih, Arnaud Doucet, and Yee Whye Teh. "Filtering variational objectives". In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. 2017, pp. 6576–6586.

[Mad+18] Chris J Maddison, Daniel Paulin, Yee Whye Teh, Brendan O'Donoghue, and Arnaud Doucet. "Hamiltonian descent methods". In: *arXiv preprint arXiv:1809.05042* (2018).

[Mad+19] Wesley J Maddox, Pavel Izmailov, Timur Garipov, Dmitry P Vetrov, and Andrew Gordon Wilson. "A simple baseline for Bayesian uncertainty in deep learning". In: *Advances in Neural Information Processing Systems* 32 (2019).

[McA+17] Rowan McAllister, Yarin Gal, Alex Kendall, Mark Van Der Wilk, Amar Shah, Roberto Cipolla, and Adrian Weller. "Concrete problems for autonomous vehicle safety: Advantages of Bayesian deep learning". In: *International Joint Conferences on Artificial Intelligence, Inc.* IEEE. 2017.

[MCF15] Yi-An Ma, Tianqi Chen, and Emily Fox. "A complete recipe for stochastic gradient MCMC". In: *Advances in neural information processing systems* 28 (2015).

[MFR20] Gael M Martin, David T Frazier, and Christian P Robert. "Computing Bayes: Bayesian computation from 1763 to the 21st century". In: *arXiv preprint arXiv:2004.06425* (2020).

[MHB17] Stephan Mandt, Matthew D Hoffman, and David M Blei. "Stochastic gradient descent as approximate Bayesian inference". In: *arXiv preprint arXiv:1704.04289* (2017).

[Mic16] Manon Michel. "Irreversible Markov chains by the factorized Metropolis filter : algorithms and applications in particle systems and spin models". 2016PSLEE039. PhD thesis. 2016. URL: `http://www.theses.fr/2016PSLEE039/document`.

[Miy+18] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. "Spectral Normalization for Generative Adversarial Networks". In: *arXiv:1802.05957* (2018). eprint: `1802.05957` (cs.LG).

[MLY17] Seonwoo Min, Byunghan Lee, and Sungroh Yoon. "Deep learning in bioinformatics". In: *Briefings in bioinformatics* 18.5 (2017), pp. 851–869.

[MMT16] Chris J Maddison, Andriy Mnih, and Yee Whye Teh. "The concrete distribution: A continuous relaxation of discrete random variables". In: *arXiv preprint arXiv:1611.00712* (2016).

[Mon20] Pierre Monmarché. "High-dimensional MCMC with a standard splitting scheme for the underdamped Langevin". In: *arXiv preprint arXiv:2007.05455* (2020).

[Mon81] Gaspard Monge. "Mémoire sur la théorie des déblais et des remblais". In: *Histoire de l'Académie Royale des Sciences de Paris* (1781).

[MR16] Andriy Mnih and Danilo Rezende. "Variational inference for Monte Carlo objectives". In: *International Conference on Machine Learning*. PMLR. 2016, pp. 2188–2196.

[MT96] K. Mengersen and R. L. Tweedie. "Rates of Convergence of the Hastings and Metropolis Algorithms". In: *Ann. Statist.* 24 (1996), pp. 101–121.

[Mül+19] Thomas Müller, Brian McWilliams, Fabrice Rousselle, Markus Gross, and Jan Novák. "Neural importance sampling". In: *ACM Transactions on Graphics* 38.145 (2019).

[Nea01a] R. M. Neal. "Annealed importance sampling". In: *Statistics and Computing* 11 (2001), pp. 125–139.

[Nea01b] Radford M Neal. "Annealed importance sampling". In: *Statistics and Computing* 11.2 (2001), pp. 125–139.

[Nea03] R. M. Neal. "Slice sampling". In: *Ann. Statist.* 31.3 (June 2003), pp. 705–767. DOI: `10.1214/aos/1056562461`.

[Nea11] R. M. Neal. "MCMC Using Hamiltonian Dynamics". In: *Handbook of Markov Chain Monte Carlo* (2011), pp. 113–162.

[Nea12] Radford M Neal. *Bayesian learning for neural networks*. Vol. 118. Springer Science & Business Media, 2012.

[Nek+20] Kirill Neklyudov, Max Welling, Evgenii Egorov, and Dmitry Vetrov. "Involutive MCMC: a Unifying Framework". In: *arXiv preprint arXiv:2006.16653* (2020).

[Nij+20] Erik Nijkamp, Mitch Hill, Tian Han, Song-Chun Zhu, and Ying Nian Wu. "On the anatomy of MCMC-based maximum likelihood learning of energy-based models". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34. 04. 2020, pp. 5272–5280.

[NYC15] Anh Nguyen, Jason Yosinski, and Jeff Clune. "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 427–436.

[Oor+17] Aaron van den Oord et al. *Parallel WaveNet: Fast High-Fidelity Speech Synthesis*. 2017. arXiv: `1711.10433 [cs.LG]`.

[Ott16] Michela Ottobre. "Markov chain Monte Carlo and irreversibility". In: *Reports on Mathematical Physics* 77.3 (2016), pp. 267–292.

[OZ00a] Art Owen and Yi Zhou. "Safe and Effective Importance Sampling". In: *Journal of the American Statistical Association* 95.449 (2000), pp. 135–143.

[OZ00b] Art Owen and Yi Zhou. "Safe and effective importance sampling". In: *Journal of the American Statistical Association* 95.449 (2000), pp. 135–143.

[Pap+19] George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. "Normalizing flows for probabilistic modeling and inference". In: *arXiv preprint arXiv:1912.02762* (2019).

[Pap+21] George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. "Normalizing flows for probabilistic modeling and inference". In: *Journal of Machine Learning Research* 22.57 (2021), pp. 1–64.

[Par81] Giorgio Parisi. "Correlation functions and computer simulations". In: *Nuclear Physics B* 180.3 (1981), pp. 378–384.

[Pas+17] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. "Automatic differentiation in PyTorch". In: (2017).

[Pas+19] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. "Pytorch: An imperative style, high-performance deep learning library". In: *arXiv preprint arXiv:1912.01703* (2019).

[Pes73] Peter H Peskun. "Optimum Monte Carlo sampling using Markov chains". In: *Biometrika* 60.3 (1973), pp. 607–612.

[Pra19a] Dennis Prangle. "Distilling importance sampling". In: *arXiv preprint arXiv:1910.03632* (2019).

[Pra19b] Dennis Prangle. "Distilling importance sampling". In: *arXiv preprint arXiv:1910.03632* (2019).

[RC13a] C. Robert and G. Casella. *Monte Carlo statistical methods*. Springer Science & Business Media, 2013.

[RC13b] Christian Robert and George Casella. *Monte Carlo statistical methods*. Springer Science & Business Media, 2013.

[RM15a] Danilo Rezende and Shakir Mohamed. "Variational Inference with Normalizing Flows". In: *Proceedings of the 32nd International Conference on Machine Learning*. Ed. by Francis Bach and David Blei. Vol. 37. Proceedings of Machine Learning Research. Lille, France: PMLR, July 2015, pp. 1530–1538.

[RM15b] Danilo Rezende and Shakir Mohamed. "Variational inference with normalizing flows". In: *International Conference on Machine Learning*. PMLR. 2015, pp. 1530–1538.

[RMC16] Alec Radford, Luke Metz, and Soumith Chintala. "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks". In: *arXiv:1511.06434* (2016). eprint: 1511.06434 (cs.LG).

[RMW14] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. "Stochastic backpropagation and approximate inference in deep generative models". In: *arXiv preprint arXiv:1401.4082* (2014).

[Rob07] C. Robert. *The Bayesian choice: from decision-theoretic foundations to computational implementation*. Springer Science & Business Media, 2007.

[RR04] Gareth O Roberts and Jeffrey S Rosenthal. "General state space Markov chains and MCMC algorithms". In: *Probability surveys* 1 (2004), pp. 20–71.

[RR98] Gareth O Roberts and Jeffrey S Rosenthal. "Optimal scaling of discrete approximations to Langevin diffusions". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 60.1 (1998), pp. 255–268.

[RT19] Francisco Ruiz and Michalis Titsias. "A Contrastive Divergence for Combining Variational Inference and MCMC". In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. Long Beach, California, USA: PMLR, Sept. 2019, pp. 5537–5545.

[RT96] G. O. Roberts and R. L. Tweedie. "Geometric convergence and central limit theorems for multidimensional Hastings and Metropolis algorithms". In: *Biometrika* 83.1 (Mar. 1996), pp. 95–110. ISSN: 0006-3444. DOI: 10.1093/biomet/83.1.95. eprint: https://academic.oup.com/biomet/article-pdf/83/1/95/709644/83-1-95.pdf. URL: https://doi.org/10.1093/biomet/83.1.95.

[RTB16] Rajesh Ranganath, Dustin Tran, and David Blei. "Hierarchical variational models". In: *International Conference on Machine Learning*. PMLR. 2016, pp. 324–333.

[Rub87] Donald B Rubin. "Comment: A noniterative Sampling/Importance Resampling alternative to the data augmentation algorithm for creating a few imputations when fractions of missing information are modest: The SIR algorithm". In: *Journal of the American Statistical Association* 82.398 (1987), pp. 542–543.

[Rui+21] Francisco JR Ruiz, Michalis K Titsias, Taylan Cemgil, and Arnaud Doucet. "Unbiased Gradient Estimation for Variational Auto-Encoders using Coupled Markov Chains". In: *Uncertainty in Artificial Intelligence*. 2021.

[RV19] G.M. Rotskoff and E. Vanden-Eijnden. "Dynamical Computation of the Density of States and Bayes Factors Using Nonequilibrium Importance Sampling". In: *Physical Review Letters* 122.15 (2019), p. 150602.

[SBH03] Øivind Skare, Erik Bølviken, and Lars Holden. "Improved sampling-importance resampling and reduced bias importance sampling". In: *Scandinavian Journal of Statistics* 30.4 (2003), pp. 719–737.

[SC18] Tobias Schwedes and Ben Calderhead. "Quasi Markov chain Monte Carlo methods". In: *arXiv preprint arXiv:1807.00070* (2018).

[SFM20] Span Spanbauer, Cameron Freer, and Vikash Mansinghka. "Deep involutive generative models for neural MCMC". In: *arXiv preprint arXiv:2006.15167* (2020).

[SG92] Adrian FM Smith and Alan E Gelfand. "Bayesian statistics without tears: a sampling–resampling perspective". In: *The American Statistician* 46.2 (1992), pp. 84–88.

[SK21] Yang Song and Diederik P Kingma. "How to train your energy-based models". In: *arXiv preprint arXiv:2101.03288* (2021).

[Ski04] John Skilling. "Nested sampling". In: *AIP Conference Proceedings*. Vol. 735. 1. American Institute of Physics. 2004, pp. 395–405.

[Ski06] John Skilling. "Nested sampling for general Bayesian computation". In: *Bayesian Analysis* 1.4 (2006), pp. 833–859.

[SKW15a] Tim Salimans, Diederik Kingma, and Max Welling. "Markov chain Monte Carlo and variational inference: Bridging the gap". In: *International Conference on Machine Learning*. 2015, pp. 1218–1226.

[SKW15b] Tim Salimans, Diederik Kingma, and Max Welling. "Markov chain Monte Carlo and variational inference: Bridging the gap". In: *International Conference on Machine Learning*. 2015, pp. 1218–1226.

[SM08a] R. Salakhutdinov and I. Murray. "On the quantitative analysis of deep belief networks". In: *Proceedings of the 25th international conference on Machine Learning*. 2008, pp. 872–879.

[SM08b] Ruslan Salakhutdinov and Iain Murray. "On the quantitative analysis of deep belief networks". In: *Proceedings of the 25th international conference on Machine learning*. 2008, pp. 872–879.

[SMD14] Jascha Sohl-Dickstein, Mayur Mudigonda, and Michael R DeWeese. "Hamiltonian Monte Carlo without detailed balance". In: *arXiv preprint arXiv:1409.5191* (2014).

[Smi17] Leslie N Smith. "Cyclical learning rates for training neural networks". In: *2017 IEEE winter conference on applications of computer vision (WACV)*. IEEE. 2017, pp. 464–472.

[SN+18] Alexander Y Shestopaloff, Radford M Neal, et al. "Sampling latent states for high-dimensional non-linear state space models with the embedded HMM method". In: *Bayesian Analysis* 13.3 (2018), pp. 797–822.

[SN18] Alexander Y Shestopaloff and Radford M Neal. "Sampling latent states for high-dimensional non-linear state space models with the embedded HMM method". In: *Bayesian Analysis* 13.3 (2018), pp. 797–822.

[So06] Mike KP So. "Bayesian analysis of nonlinear and non-Gaussian state space models via multiple-try sampling methods". In: *Statistics and Computing* 16.2 (2006), pp. 125–141.

[Sri+14] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. "Dropout: a simple way to prevent neural networks from overfitting". In: *The journal of machine learning research* 15.1 (2014), pp. 1929–1958.

[ST19] Chris Sherlock and Alexandre H. Thiery. "A Discrete Bouncy Particle Sampler". In: *arXiv preprint 1707.05200* (2019).

[SZE17] Jiaming Song, Shengjia Zhao, and Stefano Ermon. "A-NICE-MC: Adversarial training for MCMC". In: *Advances in Neural Information Processing Systems*. 2017, pp. 5140–5150.

[Tan19] Akinori Tanaka. "Discriminator optimal transport". In: *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. 2019, pp. 6816–6826.

[TCV11] Konstantin S. Turitsyn, Michael Chertkov, and Marija Vucelja. "Irreversible Monte Carlo algorithms for efficient sampling". In: *Physica D: Nonlinear Phenomena* 240.4 (2011), pp. 410–414. ISSN: 0167-2789. DOI: https://doi.org/10.1016/j.physd.2010.10.003. URL: http://www.sciencedirect.com/science/article/pii/S0167278910002782.

[Thi+20a] Achille Thin, Nikita Kotelevskii, Christophe Andrieu, Alain Durmus, Eric Moulines, and Maxim Panov. "Nonreversible MCMC from conditional invertible transforms: a complete recipe with convergence guarantees". In: *arXiv preprint arXiv:2012.15550* (2020).

[Thi+20b] Achille Thin, Nikita Kotelevskii, Jean-Stanislas Denain, Leo Grinsztajn, Alain Durmus, Maxim Panov, and Eric Moulines. "MetFlow: A New Efficient Method for Bridging the Gap between Markov Chain Monte Carlo and Variational Inference". In: *arXiv preprint arXiv:2002.12253* (2020).

[Thi+21a] Achille Thin, Yazid Janati El Idrissi, Sylvain Le Corff, Charles Ollion, Eric Moulines, Arnaud Doucet, Alain Durmus, and Christian Robert. "NEO: Non Equilibrium Sampling on the Orbits of a Deterministic Transform". In: *Advances in Neural Information Processing Systems* 34 (2021).

[Thi+21b] Achille Thin, Nikita Kotelevskii, Arnaud Doucet, Alain Durmus, Eric Moulines, and Maxim Panov. "Monte Carlo variational auto-encoders". In: *International Conference on Machine Learning*. PMLR. 2021, pp. 10247–10257.

[Tie94] Luke Tierney. "Markov Chains for Exploring Posterior Distributions". In: *The Annals of Statistics* 22.4 (1994), pp. 1701–1728.

[Tie98] Luke Tierney. "A note on Metropolis-Hastings kernels for general state spaces". In: *Ann. Appl. Probab.* 8.1 (Feb. 1998), pp. 1–9. DOI: 10.1214/aoap/1027961031. URL: https://doi.org/10.1214/aoap/1027961031.

[Tje04] Hakon Tjelmeland. *Using all Metropolis–Hastings proposals to estimate mean values*. Tech. rep. 2004.

[TK10] Surya T Tokdar and Robert E Kass. "Importance sampling: a review". In: *Wiley Interdisciplinary Reviews: Computational Statistics* 2.1 (2010), pp. 54–60.

[TT13] Esteban G Tabak and Cristina V Turner. "A family of nonparametric density estimation algorithms". In: *Communications on Pure and Applied Mathematics* 66.2 (2013), pp. 145–164.

[Tur+19a] Ryan Turner, Jane Hung, Eric Frank, Yunus Saatchi, and Jason Yosinski. "Metropolis–Hastings generative adversarial networks". In: *International Conference on Machine Learning*. PMLR. 2019, pp. 6345–6353.

[Tur+19b] Ryan Turner, Jane Hung, Eric Frank, Yunus Saatchi, and Jason Yosinski. "Metropolis-Hastings generative adversarial networks". In: *International Conference on Machine Learning*. PMLR. 2019, pp. 6345–6353.

[TV10] Esteban G Tabak and Eric Vanden-Eijnden. "Density estimation by dual ascent of the log-likelihood". In: *Communications in Mathematical Sciences* 8.1 (2010), pp. 217–233.

[Vou+18] Athanasios Voulodimos, Nikolaos Doulamis, Anastasios Doulamis, and Eftychios Protopapadakis. "Deep learning for computer vision: A brief review". In: *Computational intelligence and neuroscience* 2018 (2018).

[VVN03] Jakob Verbeek, Nikos Vlassis, and Jan Nunnink. "A variational EM algorithm for large-scale mixture modeling". In: *9th Annual Conference of the Advanced School for Computing and Imaging (ASCI '03)*. Ed. by S. Vassiliades, L.M.J. Florack, J.W.J. Heijnsdijk, and A. van der Steen. Heijen, Netherlands, June 2003, pp. 136–143.

[Wai19] Martin J. Wainwright. *High-Dimensional Statistics: A Non-Asymptotic Viewpoint*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 2019. DOI: 10.1017/9781108627771.

[Wen+20] Florian Wenzel, Kevin Roth, Bastiaan S Veeling, Jakub Świątkowski, Linh Tran, Stephan Mandt, Jasper Snoek, Tim Salimans, Rodolphe Jenatton, and Sebastian Nowozin. "How good is the bayes posterior in deep neural networks really?" In: *arXiv preprint arXiv:2002.02405* (2020).

[WI20] Andrew G Wilson and Pavel Izmailov. "Bayesian deep learning and a probabilistic perspective of generalization". In: *Advances in neural information processing systems* 33 (2020), pp. 4697–4708.

[Wil92] Ronald J Williams. "Simple statistical gradient-following algorithms for connectionist reinforcement learning". In: *Machine Learning* 8.3-4 (1992), pp. 229–256.

[Wir+20] Peter Wirnsberger, Andrew J Ballard, George Papamakarios, Stuart Abercrombie, Sébastien Racanière, Alexander Pritzel, Danilo Jimenez Rezende, and Charles Blundell. "Targeted free energy estimation via learned mappings". In: *The Journal of Chemical Physics* 153.14 (2020), p. 144112.

[WJ+08] Martin J Wainwright, Michael I Jordan, et al. "Graphical models, exponential families, and variational inference". In: *Foundations and Trends® in Machine Learning* 1.1–2 (2008), pp. 1–305.

[WKN20a] Hao Wu, Jonas Köhler, and Frank Noe. "Stochastic Normalizing Flows". In: *Advances in Neural Information Processing Systems*. Vol. 33. 2020.

[WKN20b] Hao Wu, Jonas Köhler, and Frank Noé. "Stochastic Normalizing Flows". In: *Advances in Neural Information Processing Systems* (2020).

[WKN20c] Hao Wu, Jonas Köhler, and Frank Noé. "Stochastic normalizing flows". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 5933–5944.

[WKS16] Christopher Wolf, Maximilian Karl, and Patrick van der Smagt. "Variational inference with Hamiltonian Monte Carlo". In: *arXiv preprint arXiv:1609.08203* (2016).

[WL19] Antoine Wehenkel and Gilles Louppe. "Unconstrained monotonic neural networks". In: *Advances in Neural Information Processing Systems*. 2019, pp. 1543–1553.

[WT11] Max Welling and Yee W Teh. "Bayesian learning via stochastic gradient Langevin dynamics". In: *Proceedings of the 28th international conference on machine learning (ICML-11)*. Citeseer. 2011, pp. 681–688.

[Wu+16] Yuhuai Wu, Yuri Burda, Ruslan Salakhutdinov, and Roger Grosse. "On the quantitative analysis of decoder-based generative models". In: *arXiv preprint arXiv:1611.04273* (2016).

[Xie+18] Jianwen Xie, Yang Lu, Ruiqi Gao, and Ying Nian Wu. "Cooperative learning of energy-based model and latent variable model via MCMC teaching". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 32. 1. 2018.

[YSN21] Jiancheng Yang, Rui Shi, and Bingbing Ni. "Medmnist classification decathlon: A lightweight automl benchmark for medical image analysis". In: *2021 IEEE 18th International Symposium on Biomedical Imaging (ISBI)*. IEEE. 2021, pp. 191–195.

[Zha+19] Ruqi Zhang, Chunyuan Li, Jianyi Zhang, Changyou Chen, and Andrew Gordon Wilson. "Cyclical stochastic gradient MCMC for Bayesian deep learning". In: *arXiv preprint arXiv:1902.03932* (2019).

Part II

Contributions: Simulation and Sampling methods

Chapter 4

Non-reversible MCMC from conditional invertible transforms: a complete recipe with convergence guarantees

Achille Thin[1] , Nikita Kotelevskii[2], Christophe Andrieu[3], Alain Durmus[4], Eric Moulines[1] , Maxim Panov[5]

Abstract

Markov Chain Monte Carlo (MCMC) is a class of algorithms to sample complex and high-dimensional probability distributions. The Metropolis-Hastings (MH) algorithm, the workhorse of MCMC, provides a simple recipe to construct reversible Markov kernels. Reversibility is a tractable property which implies a less tractable but essential property here, invariance. Reversibility is however not necessarily desirable when considering performance. This has prompted recent interest in designing kernels breaking this property. At the same time, an active stream of research has focused on the design of novel versions of the MH kernel, some nonreversible, relying on the use of complex invertible deterministic transforms. While standard implementations of the MH kernel are well understood, aforementioned developments have not received the same systematic treatment to ensure their validity. This paper fills the gap by developing general tools to ensure that a class of nonreversible Markov kernels, possibly relying on complex transforms, has the desired invariance property and lead to convergent algorithms. This leads to a set of simple and practically verifiable conditions.

4.1 Introduction

Being able to simulate from a probability distribution, say π defined on a measurable space $(\mathbb{Z}, \mathcal{Z})$ and referred to as the target distribution hereafter, is a ubiquitous task. Markov chain Monte Carlo methods (MCMC) is an important body of versatile techniques to sample from π. They consist of simulating realisations of time-homogeneous Markov chains $(Z_k)_{k\in\mathbb{N}}$ of invariant distribution π which possess the property that their realised states can be used to mimic samples from π, that is $Z_k \sim \pi$ approximately, but with arbitrary precision, and approximate expectations with respect to π – more precise statements are provided in Theorem 1 and we refer to these, for now, lose concepts as “convergence". We denote by P the Markov kernel associated with $(Z_k)_{k\in\mathbb{N}}$.

[1]Centre de Mathématiques Appliquées, UMR 7641, Ecole polytechnique, France
[2]CDISE, Skolkovo Institute of Science and Technology, Moscow, Russian Federation
[3]School of Mathematics, University of Bristol, UK.
[4]Université Paris-Saclay, ENS Paris-Saclay, CNRS, Centre Borelli, F-91190 Gif-sur-Yvette, France
[5]CS Departement, HSE University, Russian Federation

Metropolis-Hastings (MH) is a popular strategy to design such a Markov kernel. In its most common form, the "textbook" MH kernel samples the $(k+1)$-th state Z_{k+1} of $(Z_k)_{k\in\mathbb{N}}$ as follows: (1) sample a proposal $Y_{k+1} \sim Q(Z_k, \cdot)$; (2) set $Z_{k+1} = Y_{k+1}$ with probability $\alpha(Z_k, Y_{k+1})$; otherwise, set $Z_{k+1} = Z_k$, where $Q\colon \mathbb{Z} \times \mathcal{Z} \to [0,1]$ is a Markov kernel and $\alpha\colon \mathbb{Z} \times \mathbb{Z} \to [0,1]$ is the acceptance probability. General conditions on π, Q and α in order to ensure invariance and convergence of $(Z_k)_{k\in\mathbb{N}}$ have been known for some time. In the particular situation where π and $\{Q(z,\cdot), z \in \mathbb{Z}\}$ have densities π and $\{q(z,\cdot), z \in \mathbb{Z}\}$ with respect to a common dominating measure and are positive everywhere one can choose $\alpha(z,z') = \min\{1, \pi(z')q(z',z)/[\pi(z)q(z,z')]\}$ and define a convergent algorithm.

Contribution #1: a complete recipe for $(\pi, S)-$reversible kernels. In the context of MCMC the $\pi-$invariance property of P is traditionally the consequence of a stronger property, $\pi-$reversibility (related to detailed balance [FSS14]), which is however more tractable in practice. The MH Markov kernel is designed to satisfy this property. However, there has been a re-kindled interest in the development of "nonreversible" algorithms [TCV11; HS13; Ma+16; Ott16; BR17; Nek+20; ST19; Gus98] which come with the promise of removing the backtracking behaviour of reversible algorithms, and hence speed-up convergence. Our first contribution (Section 4.2) is **(a)** a review of $(\pi, S)-$reversibility, related to the modified detailed balance condition [FSS14], a generalisation of reversibility behind most so-called "nonreversible" MCMC algorithms and **(b)** a method generalizing the MH rule to obtain (π, S)-reversible kernels from arbitrary proposal kernels Q. This is a generalisation of [Tie98] which aims to provide a unifying and firm theoretical footing to recent and future contributions. The framework encompasses, for example, both the scenarios where π and Q have common dominating measure or when Q corresponds to a deterministic mapping.

New challenges. Novel applications have led to the development of highly sophisticated extensions of this basic scheme, prompted in particular by recent developments in the context of probability density representation with normalising flows [BZM20; Pra19b; Pap+19], invertible neural networks [Ard+19]. For example, following the realisation that the textbook MH can be generalised by combining deterministic invertible mappings of the current state and a source of randomness in the proposal stage, some authors have proposed using complex mappings involving both non-linearities and the composition of multiple layers [AKS19; Thi+20b; SFM20], while [ST19; Gus98] explore the use of nonreversible Markov kernels. However it is not always clear that the resulting algorithms are convergent. In particular application of Markov chain theory may seem difficult at first sight given the new levels of complexity involved. Our aim in this paper is to provide users with simple to use theoretical guarantees ensuring validity of the algorithms.

Contribution #2: easy ready-made convergence results. Proving convergence of MH methods can be delicate in general. However, in the π-reversible case, [MT96] and [Tie94] have derived simple conditions ensuring convergence of P in the case where π and Q share a common dominating measure μ, for example the Lebesgue measure when $\mathbb{Z} = \mathbb{R}^d$.

Theorem 1 (Convergence of textbook MH)**.** *Assume that π is not a Dirac mass function and has common $\sigma-$finite dominating measure μ with $\{Q(z,\cdot), z \in \mathbb{Z}\}$. Denote π and $\{q(z,\cdot), z \in \mathbb{Z}\}$ the resulting densities. Suppose in addition that π is not a Dirac mass and $Q(z, \mathsf{Z}^+) = 1$ for any $z \notin \mathbb{Z}^+$ with $\mathbb{Z}^+ = \{z \in \mathsf{Z} : \pi(z) > 0\}$. If for any $z' \in \mathbb{Z}$ such that $\pi(z') > 0$ we have $q(z,z') > 0$ for any $z \in \mathsf{Z}$, then for any $f\colon \mathbb{Z} \to \mathbb{R}$ such that $\pi(|f|) < \infty$, almost surely it holds that*

$$\lim_{n\to\infty} n^{-1} \sum_{i=1}^{n} f(Z_i) = \pi(f) \; . \tag{4.1}$$

In addition, for all $z \in \mathbb{Z}$

$$\lim_{n\to\infty} \|P^n(z,\cdot) - \pi(\cdot)\|_{\mathrm{TV}} = 0 \; . \tag{4.2}$$

Our second contribution is to provide similarly simple and easy to use conditions to establish convergence for (π, S)-reversible kernel. We translate these conditions to cover complex proposal mechanisms based on conditional invertible neural transform ensuring that basic convergence properties

hold in these novel settings. As we shall see some of the results lead to simple implementation suggestions ensuring that conclusions similar to those of Theorem 1 hold. Establishing these properties is often overlooked and a necessary prerequisite to any more refined analysis characterising their performance, such as quantitative finite time convergence bounds.

Contribution #3: application to particular MCMC algorithms. We show how our conditions and construction can be used in practice to design (π, S)-reversible kernels which come with convergence guarantees. We first work out a generalization of the Hamiltonian Monte Carlo algorithm in which the gradients of the log-density in the leap frog steps are replaced by general neural transforms [Nea11; SMD14]. Next, we derive and analyse two lifted Markov kernels [DHN00; CLP99; TCV11; Nek+20] covering obtained using conditional invertible transforms on an augmented space. Our experimental results (postponed to Supplementary paper) show numerically the benefits of nonreversibility in several sampling experiments.

The proofs of the main results and some facts, followed by a $*$, can be found in the supplementary material and, for example, (S#) refers to the #-th equation in the supplement. The standard notation and definitions used are precisely described in the supplementary Appendix A.1 for the reader's convenience.

4.2 (π, S)-reversibility and the Generalized MH rule

There has recently been renewed interest in the design of π-invariant Markov kernels which are nonreversible. In many scenarios, departing from reversibility can both improve the mixing time and reduce the asymptotic variance of resulting estimators. It has been shown in [AL19b] that these nonreversible Markov kernels fall under the same common framework of (π, S)-reversibility (introduced below) which encompasses the modified (or skew) detailed balance conditions. Before proceeding further, additional notations are needed. Let s be an involution on $\mathbb{Z}$, $s \circ s = \mathrm{Id}$ and S be the associated kernel $S(z, A) = \mathbb{1}_A\big(s(z)\big)$, $z \in \mathbb{Z}$, $A \in \mathcal{Z}$. Let $\check{\mu}$ be a σ-finite measure on the product space $(\mathbb{Z}^2, \mathcal{Z}^{\otimes 2})$ (the diacritic $\check{}$ is used to denote measures on the product space $\mathbb{Z}^2$). Denote by $\check{\mu}^s = (F_s)_{\#}\check{\mu}$ the pushforward of $\check{\mu}$ by the transform $F_s(z, z') = \big(s(z'), s(z)\big)$:

$$\check{\mu}^s(C) = \int \mathbb{1}_C\big(s(z'), s(z)\big)\check{\mu}\big(\mathrm{d}(z, z')\big) \ , C \in \mathcal{Z}^{\otimes 2} \ . \tag{4.3}$$

Note that F_s is an involution $F_s \circ F_s = \mathrm{Id}$ which implies that $(\check{\mu}^s)^s = \check{\mu}$.

Definition 2 (after [AL19b]). *The measure $\check{\mu}$ is s-symmetric if $\check{\mu} = \check{\mu}^s$. The sub-Markovian kernel P is (π, S)-reversible if the measure $\check{\mu}_P$ defined as $\check{\mu}_P\big(\mathrm{d}(z, z')\big) = \pi(\mathrm{d}z)P(z, \mathrm{d}z')$ is s-symmetric.*

It is established in Section 4.6.1 that P is (π, S)-reversible if it satisfies the **skew detailed balance** condition,

$$\pi(\mathrm{d}z)P(z, \mathrm{d}z') = s_{\#}\pi(\mathrm{d}z')SPS(z', \mathrm{d}z) \ . \tag{4.4}$$

In particular, if $s_{\#}\pi = \pi$ and P is a Markov kernel, then π is invariant for P. We assume that the condition $s_{\#}\pi = \pi$ is in force in the rest of the paper. Note that for $s = \mathrm{Id}$ we recover the standard detailed balance condition (see Section 4.5).

4.2.1 Generalized Metropolis-Hastings

The MH algorithm gives a method to transform any proposal Markov kernel Q into a π-reversible Markov kernel. We derive a Generalized Metropolis-Hastings (GMH) rule to turn Q into a (π, S)-reversible Markov kernel. We then apply this condition to the case where $\pi(\mathrm{d}z')$ and $Q(z, \mathrm{d}z')$ have a density w.r.t. to a common dominating measure, and to the case where $Q(z, \mathrm{d}z') = \delta_{\Phi(z)}(\mathrm{d}z')$ for $\Phi\colon \mathbb{Z} \to \mathbb{Z}$. We first establish a simple necessary and sufficient condition on the proposal kernel Q and the acceptance probability function $\alpha\colon \mathsf{Z}^2 \to [0, 1]$ for the resulting (sub-Markovian) kernel

$$Q_\alpha(z, \mathrm{d}z') := \alpha(z, z')Q(z, \mathrm{d}z') \ , \tag{4.5}$$

to be (π, S)-reversible. A subset $A \subset \mathbb{Z}^2$ is said to be s-symmetric if $(z, z') \in A$ if and only if $\big(s(z'), s(z)\big) \in A$. We denote

$$\check{\nu}\big(\mathrm{d}(z, z')\big) := \pi(\mathrm{d}z)Q(z, \mathrm{d}z') \; . \tag{4.6}$$

The following result provides us with a key instrument to work with the densities of $\check{\nu}$ and its pushforward $\check{\nu}^s$ in full generality.

Proposition 3. *Set $\check{\lambda} = \check{\nu} + \check{\nu}^s$, $h = \mathrm{d}\check{\nu}/\mathrm{d}\check{\lambda}$ and $A_{\check{\nu}} = \{h \times h \circ F_s > 0\} \in \mathcal{Z}^{\otimes 2}$. Then, the restrictions $\check{\nu}_A(\cdot) = \check{\nu}(\cdot \cap A_{\check{\nu}})$ and $\check{\nu}^s_A(\cdot) = \check{\nu}^s(\cdot \cap A_{\check{\nu}})$ are equivalent and $\check{\nu}_{A,\mathrm{c}}(\cdot) = \check{\nu}(\cdot \cap A^{\mathrm{c}}_{\check{\nu}})$ and $\check{\nu}^s_{A,\mathrm{c}}(\cdot) = \check{\nu}^s(\cdot \cap A^{\mathrm{c}}_{\check{\nu}})$ are mutually singular. In addition, define, for $(z, z') \in A_{\check{\nu}}$, $r(z, z') = h(z, z')/h\big(s(z'), s(z)\big)$. Then, r is a version of the density of $\check{\nu}_A$ w.r.t. $\check{\nu}^s_A$, i.e. $r = \mathrm{d}\check{\nu}_A/\mathrm{d}\check{\nu}^s_A$ and $r(z, z') = 1/r \circ F_s(z, z')$ for all $(z, z') \in A_{\check{\nu}}$.*

The following result applies Proposition 3 and extends the seminal result [Tie98, Theorem 2] to the (π, S)-reversible case.

Theorem 4. *The sub-Markovian kernel Q_α in (4.5) is (π, S)-reversible if and only if the following conditions hold.*
(i) The function α is zero $\check{\nu}$-a.e.on $A^{\mathrm{c}}_{\check{\nu}}$.
(ii) The function α satisfies $\alpha(z, z')r(z, z') = \alpha(s(z'), s(z))$ $\check{\nu}$-a.e.on $A_{\check{\nu}}$.

Similarly to the π-reversible case, we define the generalized Metropolis-Hastings (GMH) rejection probability by

$$\alpha(z, z') = \begin{cases} \mathsf{a}\left(\frac{h(s(z'), s(z))}{h(z, z')}\right) & h(z, z')\mathrm{ne}0, \\ 1 & h(z, z') = 0, \end{cases} \tag{4.7}$$

where $\mathsf{a} \colon \mathbb{R}^*_+ \to [0, 1]$ satisfies $\mathsf{a}(0) = 0$ and for $t \in \mathbb{R}^*_+$,

$$t\mathsf{a}(1/t) = \mathsf{a}(t) \; . \tag{4.8}$$

Then α satisfies the conditions (i)-(ii) of Theorem 4, see Section 4.6.4. We may take for example $\mathsf{a}(t) = \min(1, t)$ or $\mathsf{a}(t) = t/(1 + t)$ which correspond to the classical Metropolis-Hastings and Barker ratio respectively.

We can obtain the GMH Markov kernel P which is (π, S)-reversible by adding Dirac masses:

$$P(z, \mathrm{d}z') = Q_\alpha(z, \mathrm{d}z') + a(z)\delta_z(\mathrm{d}z') + b(z)\delta_{s(z)}(\mathrm{d}z') \tag{4.9}$$

with a, b nonnegative, measurable satisfying $a(z) = a(s(z))$ and $a(z) + b(z) = 1 - Q_\alpha(z, \mathsf{Z})$; see Section 4.6.5. In the sequel, we focus on the case $a(z) = 0$ and $b(z) = 1 - Q_\alpha(z, \mathsf{Z})$.

4.2.2 GMH for particular proposal maps

We now specialize (4.7) to the case where π and Q admit a common dominating measure and the case where Q is deterministic.

Proposal with densities. Suppose there is a common dominating measure μ on $(\mathbb{Z}, \mathcal{Z})$ such that $\pi(\mathrm{d}z) = \pi(z)\mu(\mathrm{d}z)$, $Q(z, \mathrm{d}z') = q(z, z')\mu(\mathrm{d}z')$ and that μ is invariant by s, *i.e.* $s_\#\mu = \mu$. In this scenario, we have (see Section 4.6.6)

$$A_{\check{\nu}} = \big\{\pi(z)q(z, z') \times \pi(z')q\big(s(z'), s(z)\big) > 0\big\} \; . \tag{4.10}$$

In addition, we obtain using (4.7) that

$$\alpha(z, z') = \begin{cases} \mathsf{a}\left[\frac{\pi(z')q(s(z'), s(z))}{\pi(z)q(z, z')}\right] & \pi(z)q(z, z')\mathrm{ne}0, \\ 1 & \pi(z)q(z, z') = 0 \; . \end{cases} \tag{4.11}$$

Theorem 1 exploits the fact that in the π-reversible scenario the MH kernel is π-irreducible if the condition $\pi(z') > 0$ implies that $q(z, z') > 0$ [MT96]. This result can be extended to the (π, S)-reversible case as follows.

Lemma 5. *The GMH Markov kernel P in (4.9) is π-irreducible if, $\pi(z') > 0$ implies that, for all $z \in \mathbb{Z}$, $q(z, z') > 0$ and $q\big(s(z), s(z')\big) > 0$.*

In the π-reversible case, [Tie94, Corollary 2] shows that the π-irreducibility condition implies that the GMH Markov kernel P (4.9) is Harris recurrent and aperiodic. These two properties have consequences that are very important in practice: the convergence in total variation of the iterates of the kernel to the invariant distribution and the ergodic theorem become valid also for all the initial conditions. These results extend to (π, S)-Markov kernels (see Section 4.6.7).

Theorem 6. *Let P be defined as in (4.9), with $a(z) = 0$ and $b(z) = 1 - Q_\alpha(z, \mathsf{Z})$. Assume that for any $z' \in \mathsf{Z}$, $\pi(z') > 0$ implies $q(z, z') \times q\big(s(z), s(z')\big) > 0$ for any $z \in \mathsf{Z}$. Suppose in addition that π is not a Dirac mass and $Q(z, \mathsf{Z}^+) = 1$ for any $z \notin \mathbb{Z}^+$ with $\mathbb{Z}^+ = \{z \in \mathsf{Z} : \pi(z) > 0\}$. The conclusions of Theorem 1 hold.*

Deterministic proposal. Suppose now that Φ is a one-to-one mapping from $\mathbb{Z}$ onto $\mathbb{Z}$ such that

$$\Phi^{-1} = s \circ \Phi \circ s \; . \tag{4.12}$$

We consider the deterministic proposal kernel $Q(z, \mathrm{d}z') = \delta_{\Phi(z)}(\mathrm{d}z')$: when the current state is z, the proposal is $\Phi(z)$. Condition (4.12) implies that $F = s \circ \Phi$ is an involution. Our setting covers involutive MCMC – corresponding to the case $s = \mathrm{Id}$ introduced in [Tie98, Section 2] and more recently in [Nek+20].

In this scenario we have that (see Section 4.6.8) $\check{\nu}\big(\mathrm{d}(z, z')\big) = \pi(\mathrm{d}z)\delta_{\Phi(z)}(\mathrm{d}z')$ and $\check{\nu}^s\big(\mathrm{d}(z, z')\big) = \pi(\mathrm{d}z')\delta_{\Phi^{-1}(z')}(\mathrm{d}z)$. The function h defined in Proposition 3 is given by $h(z, z') = \mathbb{1}_{\Phi(z)}(z')k(z)$ with

$$k(z) = \frac{\mathrm{d}\pi}{\mathrm{d}\lambda}(z), \quad \lambda = \pi + (\Phi^{-1})_{\#}\pi \; . \tag{4.13}$$

Theorem 4 is satisfied with the acceptance probability given by $\alpha\big(z, \Phi(z)\big) = \bar{\alpha}(z)$ with

$$\bar{\alpha}(z) = \mathsf{a}\left(\frac{k\big(s \circ \Phi(z)\big)}{k(z)}\right) \tag{4.14}$$

if $k(z) > 0$ and $\bar{\alpha}(z) = 1$, otherwise. Of course, there is no need to define $\alpha(z, z')$ for $z' \mathrm{ne} \Phi(z)$. A special case of interest is when $\mathbb{Z} = \mathbb{R}^d$ and the target distribution $\pi(\mathrm{d}z) = \pi(z)\mathrm{Leb}_d(\mathrm{d}z)$ has a density w.r.t. the Lebesgue measure on $\mathbb{R}^d$. Here the dominating measure λ is given by

$$\lambda(\mathrm{d}z) = \{\pi(z) + \pi \circ \Phi(z)\mathrm{J}_\Phi(z)\}\mathrm{Leb}_d(\mathrm{d}z) \; , \tag{4.15}$$

where J_f denotes the absolute value of the Jacobian determinant of f. Then, the density $k(z)$ is given by

$$k(z) = \frac{\pi(z)}{\pi(z) + \pi \circ \Phi(z)\mathrm{J}_\Phi(z)} \tag{4.16}$$

and the acceptance ratio $\bar{\alpha}(z)$ takes the simple form

$$\bar{\alpha}(z) = \mathsf{a}\left(\frac{\pi \circ \Phi(z)\mathrm{J}_\Phi(z)}{\pi(z)}\right) \tag{4.17}$$

if $\pi(z)\mathrm{ne}0$ and $\bar{\alpha}(z) = 1$ otherwise (see Section 4.6.9). We obtain the same acceptance ratio given by [Nek+20, Eq. (5)], which derive this expression in the special case $s = \mathrm{Id}$ and thus $\Phi^{-1} = \Phi$ is an involution. It is perhaps striking that the acceptance ratio **does not depend** on s: this comes from the fact the target distribution π is invariant by s. This setting encompasses many algorithms, HMC [Dua+87; Nea11], and NICE-MC [SZE17] – see below, [Nek+20] and the references therein. Of course, in most cases, (π, S)-reversible deterministic Markov kernels are not π-irreducible and Harris recurrent. They can nevertheless be important building blocks of Markov kernels as in the HMC construction.

4.3 Applications and examples

4.3.1 Generalized Hamiltonian Dynamics

We first consider generalizations of the Hamiltonian Monte Carlo algorithm (see [Nea11; SMD14]). These methods might also be seen as a special case of NICE (Non-linear Independent Components Estimation) MCMC methods [SZE17; Nek+20]. The objective is to sample a distribution on $\mathbb{R}^d$ of density π_0 w.r.t. the Lebesgue measure. We use a data augmentation approach which consists of adding a "momentum" variable with stationary distribution admitting a symmetric density φ on $\mathbb{R}^d$ w.r.t. the Lebesgue measure, e.g. $\varphi(-p) = \varphi(p)$. More precisely, on the extended state space $\mathbb{Z} = \mathbb{R}^{2d}$, we consider the extended target density defined by $\pi(x,p) = \pi_0(x)\varphi(p)$ and the Markov chain $\big(Z_i = (X_i, P_i)\big)_{i\in\mathbb{N}}$. The involution is taken to be $s(x,p) = (x,-p)$. By construction, $s_{\#}\pi = \pi$.

We first show how to construct a (π, S)-reversible Markov kernel on $\mathbb{R}^{2d}$ using modified leap-frog integrators. Let $m \in \mathbb{N}$ and $\{M_i, N_i\}_{i=1}^m$ be C^1 functions on $\mathbb{R}^d$. We define a mapping $\Phi(x,p) = \mathrm{F}_m \circ \cdots \circ \mathrm{F}_1(x,p)$ on $\mathbb{R}^{2d}$ where F_i is given by $(x_{i+1}, p_{i+1}) = \mathrm{F}_i(x_i, p_i)$ where for $h > 0$

$$\begin{cases} p_{i+1/2} &= p_i + hM_i(x_i)\ , \\ x_{i+1} &= x_i + hp_{i+1/2}\ , \\ p_{i+1} &= p_{i+1/2} + hN_i(x_{i+1})\ . \end{cases} \tag{4.18}$$

It is easily seen that F_i is a C^1 diffeomorphism on $\mathbb{R}^{2d}$ to $\mathbb{R}^{2d}$ with $\mathrm{J}_{\mathrm{F}_i}(x,p) = 1$. Moreover, if for any $i \in \{1,\dots,m\}$, $M_i = N_{m+1-i}$, then $s \circ \Phi \circ s = \Phi^{-1}$; see Section 4.7.1. We assume in the sequel that this condition holds. Consider now the Markov kernel

$$P((x,p), \mathrm{d}(y,q)) = \bar\alpha(x,p)\delta_{\Phi(x,p)}(\mathrm{d}(y,q)) \tag{4.19}$$

$$+ (1 - \bar\alpha(x,p))\delta_{(x,-p)}(\mathrm{d}(y,q))\ , \tag{4.20}$$

$$\text{with} \qquad \bar\alpha(x,p) = \mathsf{a}\left(\pi \circ \Phi(x,p)/\pi(x,p)\right)\ , \tag{4.21}$$

if $\pi(x,p) > 0$ and $\bar\alpha$ is equal to 1 otherwise. Using (4.17), P is (π, S)-reversible, but is deterministic and therefore not ergodic. A standard approach to address this issue, used in the context of HMC algorithms, is to refresh the momentum between two successive moves according to a Markov transition preserving the distribution φ. A particular choice consists of sampling the velocity afresh from φ before applying the kernel (4.101). More precisely, we define the Markov chain $(X_i)_{i\in\mathbb{N}}$ by the following recursion. From a state X_k, the $k+1$-th iterate is defined by: 1. sample P_{k+1} from φ and set $(Y_{k+1}, Q_{k+1}) = \Phi(X_k, P_{k+1})$; accept $X_{k+1} = Y_{k+1}$ with probability $\bar\alpha(X_k, P_{k+1})$ and reject $X_{k+1} = X_k$ otherwise. In this case one can check that $(X_i)_{i\in\mathbb{N}}$ is a Markov chain on $\mathbb{R}^d$ of kernel, obtained by marginalisation of (4.9) w.r.t. the momentum distribution,

$$K(x, \mathrm{d}y) = K_\alpha(x, \mathrm{d}y) + \{1 - \bar\alpha(x)\}\delta_x(\mathrm{d}y)\ , \tag{4.22}$$

where $\bar\alpha(x) = K_\alpha(x, \mathbb{R}^d)$ and denoting $G_x(p) = \mathrm{proj}_1 \circ \Phi(x,p)$, $\mathrm{proj}_1(x,p) = x$,

$$K_\alpha(x, \mathrm{d}y) = \int \bar\alpha(x,p)\varphi(p)\delta_{G_x(p)}(\mathrm{d}y)\mathrm{d}p \tag{4.23}$$

If for any $x \in \mathbb{R}^d$, $p \mapsto G_x(p)$ is a diffeomorphism on $\mathbb{R}^d$, then Theorem 6 can be applied. In such case, $K_\alpha(x, \mathrm{d}y) = \alpha(x,y)q(x,y)$ with

$$\alpha(x,y) = \mathsf{a}\left(\frac{\pi_0(y)\varphi\big\{H_x\big(G_x^{-1}(y)\big)\big\}}{\pi_0(x)\varphi\big(G_x^{-1}(y)\big)}\right)\ , \tag{4.24}$$

$$q(x,y) = \varphi\big(G_x^{-1}(y)\big)\mathrm{J}_{G_x^{-1}}(y)\,, \tag{4.25}$$

and $H_x(p) = \mathrm{proj}_2 \circ \Phi(x,p)$ and $\mathrm{proj}_2(x,p) = p$. The expression of $\alpha(x,y)$ is only of theoretical interest and is not needed to implement the algorithm. Of course, requiring that G_x is a diffeomorphism imposes conditions on F_i, $i \in \{1,\dots,m\}$ and Theorem 24 (see Section 4.7.1).

Theorem 7. *Assume that $\varphi > 0$, $\varphi(-p) = \varphi(p)$ for all $p \in \mathbb{R}^d$ and for any $i \in \{1, \dots, m\}$, M_i and N_i are* L*-Lipschitz and $h \leqslant c_0/[\mathrm{L}^{1/2}m]$, where $c_0 \approx 0.3$ (see Theorem 24). Then for any $x \in \mathbb{R}^d$, $p \mapsto G_x(p)$ is a C^1-diffeomorphism.*

The proof of this result is along the same lines as the proof of [DMS17, Theorem 1] which focuses on the standard HMC algorithm.

A by-product of the proof of Theorem 7, is that, perhaps surprisingly (see (4.7.1))

$$q(y,x)/q(x,y) = \varphi\big(H_x \circ G_x^{-1}(y)\big)/\varphi\big(G_x^{-1}(y)\big) \;, \tag{4.26}$$

implying that α (4.24) is the textbook MH acceptance ratio corresponding to q in (4.25), and the Markov kernel K (4.22) is therefore π-reversible. It easily checked that this kernel satisfies the conditions of Theorem 1 and the convergence results apply.

The π_0-reversibility of K has the disadvantage of loosing the potentially advantageous non-backtracking (or persistency) features of P. It is possible to recover persistency by considering the mixture of kernels on the extended space $\mathbb{R}^{2d}$ $\omega P + (1-\omega)L$ where P is the deterministic kernel (4.101) and $L((x,p), \mathrm{d}(y,q)) = K(x, \mathrm{d}y)\varphi(q)\mathrm{d}q$. In words, we refresh independently the position and the momentum. The amount of persistency is controlled by ω. Theorem 7 establishes π_0-irreducibility of K (see its proof), which immediately implies π-irreducibility of L; see Section 4.7.1 for a more detailed discussion.

4.3.2 Lifted kernels

In this section, we apply the results of Section 4.2 to lifted kernels introduced in [DHN00; CLP99; TCV11; Mic16]. As above, let Π be a target probability density on $\mathbb{R}^d$ w.r.t. the Lebesgue measure. We extend the state space with a direction, *i.e.* we consider $\mathsf{Z} = \mathbb{R}^d \times \mathsf{V}$ with $\mathsf{V} = \{-1, 1\}$ and the extended target distribution $\pi = \Pi \otimes [\{\delta_{-1} + \delta_1\}/2]$. In this scenario the involution is $s(x,v) = (x,-v)$.

Proposal with densities. Let $q_{-1}(x,\cdot), q_1(x,\cdot)$ be two transition densities w.r.t. the Lebesgue measure on $\mathbb{R}^d$. Consider a proposal kernel $Q((x,v), \mathrm{d}(y,w))$ with density $q((x,v),(y,w))$ with respect to $\mathrm{Leb}_d(\mathrm{d}y) \otimes \{\delta_{-1}(\mathrm{d}w) + \delta_1(\mathrm{d}w)\}$ given by

$$q\big((x,v),(y,w)\big) = \big\{\rho \mathbb{1}_v(w) + (1-\rho)\mathbb{1}_{-v}(w)\big\} q_w(x,y), \tag{4.27}$$

where $\rho \in (0,1)$. In words, starting from (x,v), we either "keep" $w = v$ with probability ρ or "flip" $w = -v$ the direction otherwise, and then propose a candidate y according to $q_w(x,\cdot)$. In the original implementation of the lifting procedure [TCV11], ρ is set to 1; taking $\rho < 1$ simply prevents the algorithm from getting "stuck" in one direction which could impede convergence of the algorithm.

From (4.7) and (4.11), the acceptance ratio α writes, for $q_w(x,y)\Pi(x) \neq 0$, see Section 4.7.2,

$$\alpha\big((x,v),(y,w)\big) = \mathsf{a}\left(\frac{q_{-w}(y,x)\Pi(y)}{q_w(x,y)\Pi(x)}\right) \;, \tag{4.28}$$

and $\alpha\big((x,v),(y,w)\big) = 1$ otherwise, where a satisfies (4.8). The GMH kernel is given by (4.9) with $a(z) = 0$ and $b(z) = 1 - Q_\alpha\big((x,v), \mathbb{Z}\big)$. Note that if the proposal move is rejected, then the direction is automatically flipped.

In the case $q_{-1} = q_1$, then the acceptance probability α (4.28) does not depend on v, w and the GMH kernel (4.9) can be marginalized w.r.t. v yielding the π_0-reversible MH algorithm of proposal density q_1. Since $\rho \in (0,1)$, the expression for q in (4.150) implies the following result.

Proposition 8. *Assume that for any $y \in \mathbb{R}^d$, $\pi_0(y) > 0$ implies $q_{-1}(x,y) > 0$ and $q_1(x,y) > 0$, for all $x \in \mathbb{R}^d$. Then the conditions of Theorem 6 hold, and the GMH kernel* (4.9) *is ergodic.*

Similarly to Section 4.3.1, the proposal densities $q_v(x,\cdot)$ are often associated to C^1-diffeomorphisms $G_{v,x}\colon p \mapsto G_{v,x}(p)$. From a state X_k, we sample P_{k+1} from φ positive density on $\mathbb{R}^d$ and set $Y_{k+1} = G_{V_k,X_k}(P_{k+1})$. In this case,

$$q_v(x,y) = \varphi\big(G_{v,x}^{-1}(y)\big)\mathrm{J}_{G_{v,x}^{-1}}(y) \; . \tag{4.29}$$

We illustrate the construction above with two examples of mappings G satisfying the conditions we consider.

Example 9 ((MALA-cIT) lifted kernel)**.** *Assume that π_0 is positive and continuously differentiable. For $x \in \mathbb{R}^d$, we define two transforms $G_{1,x}, G_{-1,x}$. For $G_{1,x}$, we set*

$$G_{1,x}\colon p \mapsto x + \gamma \nabla \log \pi(x) + \sqrt{2\gamma} p \; , \tag{4.30}$$

which corresponds to the proposal of the Metropolis Adjusted Langevin Algorithm (MALA). In particular, for any $x \in \mathbb{R}^d$, the transformation $G_{1,x}$ is a C^1-diffeomorphism, with $\mathrm{J}_{G_{1,x}}(p) = (2\gamma)^{d/2}$ and

$$G_{1,x}^{-1}(y) = \{y - x - \gamma \nabla \log \pi(x)\}/\sqrt{2\gamma} \; . \tag{4.31}$$

For $G_{-1,x}$ we consider conditional invertible transforms [Ard+19]

$$G_{-1,x}(p) = \mathrm{G}_{K,x} \circ \cdots \circ \mathrm{G}_{1,x}(p) \; , \tag{4.32}$$

where for $i \in \{1,\ldots,K\}$, $\mathrm{G}_{i,x}$ splits its input into two parts $(p_{i,1}, p_{i,2}) \in \mathbb{R}^{d_{i,1}} \times \mathbb{R}^{d-d_{i,1}}$ and applies affine transformations between them

$$\begin{aligned} p_{i+1,1} &= p_{i,1} \odot \exp\big(R_{i,1}(p_{i,2}, x)\big) + M_{i,1}(p_{i,2}, x) \; , \\ p_{i+1,2} &= p_{i,2} \odot \exp\big(R_{i,2}(p_{i+1,1}, x)\big) + M_{i,2}(p_{i+1,1}, x) \; . \end{aligned} \tag{4.33}$$

Here $R_{i,1}, M_{i,1}$ (resp. $R_{i,2}, M_{i,2}$) are any functions from $\mathbb{R}^{d_{i,1}}$ (resp. $\mathbb{R}^{d-d_{i,1}}$) to $\mathbb{R}^d$. This structure is an extension of the affine coupling block architecture suggested in [DSB17]. Note that for any $i \in \{1,\ldots,K\}$, $\mathrm{G}_{i,x}$ is a C^1-diffeomorphism on $\mathbb{R}^d$ of Jacobian determinant given by $\mathrm{J}_{\mathrm{G}_{i,x}}(p) = \exp\big(R_{i,1}(p_2, x) + R_{i,2}(p_1', x)\big)$. Therefore, $G_{-1,x}$ is a C^1-diffeomorphism with Jacobian determinant which can be explicitly computed. (4.29) *gives a nonreversible MH algorithm with convergence guarantees provided by Proposition 8; see details in Section 4.7.3.*

A specific case corresponds to the choice

$$G_{v,x}(p) = \mathrm{proj}_1 \circ \Psi^v(x,p) \; , \tag{4.34}$$

where Ψ is a C^1-diffeomorphism on $\mathbb{R}^{2d}$. We establish in the following result an alternative expression for α using (4.28) and (4.29), which relies on Ψ^v and J_{Ψ^v} and for which $\mathrm{J}_{G_{v,x}}$ is not required anymore (see Section 4.7.4).

Lemma 10. *Assume that, for any $(x,v) \in \mathbb{Z}$, the mapping $G_{v,x}$ is a C^1-diffeomorphism on $\mathbb{R}^d$. Then, for any $x, y \in \mathbb{R}^d$, $v, w \in \mathsf{V}$, the acceptance ratio α defined in* (4.28) *is given by*

$$\mathsf{a}\left(\frac{\mu\Big(\Psi^w\big(x, G_{w,x}^{-1}(y)\big)\Big)}{\mu\big(x, G_{w,x}^{-1}(y)\big)} \mathrm{J}_{\Psi^w}\big(x, G_{w,x}^{-1}(y)\big) \right) \; , \tag{4.35}$$

where $\mu(x,p) = \pi_0(x)\varphi(p)$.

This result is of practical interest because in many cases, the computation of $\mathrm{J}_{\Psi^v}(x,p)$ is much simpler than that of $\mathrm{J}_{G_{v,x}}(p)$. As an example, if Ψ is the generalized HMC transform $\Psi = \mathrm{F}_m \circ \cdots \circ \mathrm{F}_1$ where F_i is defined in (4.18), $\mathrm{J}_{\Psi^v}(x,p) = 1$ while $\mathrm{J}_{G_{v,x}}(p)$ has no simple closed-form expression.

Deterministic proposals. Using a C^1-diffeomorphism Ψ on $\mathbb{R}^{2d}$, we may also consider deterministic moves like in Section 4.3.1. Consider the extended state space $\mathsf{Z} = \mathbb{R}^{2d} \times \mathsf{V}$, the target distribution $\pi = \pi_0 \otimes \varphi \otimes [\{\delta_{-1} + \delta_1\}/2]$, where φ is a symmetric density w.r.t. Leb_d, and the involution $s(x,p,v) = (x,p,-v)$. Define $\Phi\big(x,p,v\big) = \big(\Psi^v(x,p), v\big)$. Then, it is immediate to see that $s \circ \Phi \circ s = \Phi^{-1}$. We consider the deterministic proposal kernel

$$Q\big((x,p,v), \mathrm{d}(y,q,w)\big) = \delta_{\Psi^v(x,p)}(\mathrm{d}(y,q))\delta_v(\mathrm{d}w) \; . \tag{4.36}$$

In the case, the acceptance ratio (4.14) reads for $x, p \in \mathbb{R}^d$, $v \in \mathsf{V}$ satisfying $\pi_0(x)\varphi(p) > 0$

$$\bar{\alpha}(x,p,v) = \mathsf{a}\left(\mu\big(\Psi^v(x,p)\big)\mathrm{J}_{\Psi^v}(x,p)/\mu(x,p)\right) \; , \tag{4.37}$$

and is equal to 1 if $\mu(x,p) = 0$, where $\mu(x,p) = \pi_0(x)\varphi(p)$; see Section 4.7.5.

Example 11 (L2HMC)**.** *Assume that π_0 is positive and continuously differentiable. Using the framework depicted above, we show how the L2HMC algorithm [LHS17a] (Learning To Hamiltonian Monte Carlo) can be turned into a nonreversible MCMC method by considering the map*

$$\Psi(x,p) = G_K \circ \cdots \circ G_1(x,p) \; , \tag{4.38}$$

where $G_i = H_i \circ F_i \circ H_{i-1/2}$ with, for $\delta > 0$,
- *for $j \in \{i, i-1/2\}$, $H_j(x,p) = (x, H_{j,x}(p))$ with*

$$H_{j,x}(p) = p \odot \exp\left(\delta R_j^H(x)\right) + \delta\left[\nabla \log \pi_0(x) \odot \exp\left(\delta R_j^H(x)\right) + M_j^H(x)\right] \; . \tag{4.39}$$

Note that H_j is a C^1-diffeomorphism on $\mathbb{R}^{2d}$ of Jacobian $\mathrm{J}_{H_j}(x,p) = \exp\Big(\delta R_j^H(x)\Big)$.
- *$F_i(x,p) = \big(F_{i,p}(x), p\big)$, where $F_{i,p}$ splits its input into two parts x_1, x_2 and applies affine transformations*

$$\begin{aligned} x_1' &= x_1 \odot \exp\big(\delta R_{i,1}^F(x_2,p)\big) + \delta M_{i,1}^F(x_2,p) \; , \\ x_2' &= x_2 \odot \exp\big(\delta R_{i,2}^F(x_1',p)\big) + \delta M_{i,2}^F(x_1',p) \; . \end{aligned} \tag{4.40}$$

Clearly, F_i is a C^1-diffeomorphism on $\mathbb{R}^{2d}$ with $\mathrm{J}_{F_i}(x,p) = \exp\big(\delta R_{i,1}^F(x_2,p) + \delta R_{i,2}^F(x_1',p)\big)$.
Then, Ψ defined by (4.38) is a C^1-diffeomorphism whose Jacobian can be recursively computed. Then, the kernel $P(x,p,w), \mathrm{d}(y,q,w))$ given by

$$\bar{\alpha}(x,p,v)\delta_{\Psi^v(x,p)}(\mathrm{d}(y,q))\delta_v(\mathrm{d}w) \quad + \quad (1 \quad - \quad \bar{\alpha}(x,p,v))\delta_{(x,p,-v)}(\mathrm{d}(y,q,w)) \tag{4.41}$$

where $\bar{\alpha}$ is defined in (4.37) is (π, S)-reversible. This kernel should be combined with (possibly partial) refreshment steps as discussed in Section 4.3.1; see Section 4.7.6 for details.

Supplementary Material

4.4 Notations, definitions and general Markov chain theory

In this section, we recall some basic facts and notations in a form that is useful for establishing properties of Markov chains. Let $(\mathbb{Z}, \mathcal{Z})$ be a measurable space where $\mathcal{Z}$ is a countably generated σ-algebra.

Definition 12 (Kernel). *A* kernel *on $\mathbb{Z} \times \mathcal{Z}$ is a map $P\colon \mathbb{Z} \times \mathcal{Z} \to \mathbb{R}_+$ such that*

(i) for any $A \in \mathcal{Z}$, $z \mapsto P(z, A)$ is measurable;

(ii) for any $z \in \mathbb{Z}$, the function $A \mapsto P(z, A)$ is a finite measure on $\mathcal{Z}$.

Definition 13 (Markov and sub-Markovian kernel). *A kernel P is Markovian (or P is a Markov kernel) if $P(z, \mathbb{Z}) = 1$ for all $z \in \mathbb{Z}$. A kernel P is submarkovian (or P is a sub-Markov kernel) if $P(z, \mathbb{Z}) \leqslant 1$ for all $z \in \mathbb{Z}$.*

For $f\colon \mathbb{Z} \to \mathbb{R}$ a measurable function, ν a probability distribution, and P a kernel on $\mathbb{Z} \times \mathcal{Z}$, we let $\nu(f) \coloneqq \int f(z)\nu(\mathrm{d}z)$ and denote for $(z, A) \in \mathbb{Z} \times \mathcal{Z}$,

$$\nu P(A) = \int \nu(\mathrm{d}z) P(z, A) \;, \quad Pf(z) = \int P(z, \mathrm{d}z') f(z') \;.$$

Further, for $(z, A) \in \mathbb{Z} \times \mathcal{Z}$ define recursively for $n \geqslant 2$: $P^n(z, A) = \int P^{n-1}(z, \mathrm{d}z') P(z', A)$.

Definition 14 (Total variation distance). *For μ, ν two probability distributions on $(\mathbb{Z}, \mathcal{Z})$ we define the total variation distance between μ and ν by $\|\mu - \nu\|_{\mathrm{TV}} := \sup_{|f| \leqslant 1} |\mu(f) - \nu(f)|$, where the supremum is taken over the measurable function $f\colon \mathbb{Z} \to \mathbb{R}$.*

Definition 15 (Harmonic function). *Let P be a kernel on $(\mathbb{Z}, \mathcal{Z})$. Then a non-negative measurable function $h : \mathbb{Z} \to \mathbb{R}$ is said to be* harmonic *if $Ph = h$.*

Definition 16 (Irreducibility). *Let ν be a non trivial σ-finite measure on $\big(\mathbb{Z}, \mathcal{Z}\big)$. A kernel P is said to be ν-irreducible if for all $(z, A) \in \mathbb{Z} \times \mathcal{Z}$ such that $\nu(A) > 0$ there exists $n = n(z, A) \in \mathbb{N}$ such that $P^n(z, A) > 0$.*

Definition 17 (Periodicity and Aperiodicity). *P is periodic if there exists $n \in \mathbb{N}$, $n \geqslant 2$, and $A_i \in \mathcal{Z}$ for $i \in 1, \ldots, n$, non-empty and disjoint, such that for $z \in A_i$, $P(z, A_{i+1}) = 1$ with the convention $A_{n+1} = A_1$. Aperiodicity is the negation of periodicity.*

General Markov chain theory provides us with powerful tools to establish validity and convergence of MCMC algorithms, leading to basic convergence theorems such as those found in [Tie94, Theorem 1 and 3] and distilled below. We informally comment on the result below.

Theorem 18 ([Tie94]). *Suppose P is such that $\pi P = P$ and is $\pi-$irreducible. Then π is the unique invariant probability distribution of P and for any $f\colon \mathbb{Z} \to \mathbb{R}$ such that $\pi(|f|) < \infty$*

$$\lim_{n \to \infty} n^{-1} \sum_{i=1}^{n} f(Z_i) = \pi(f) \;, \tag{4.42}$$

almost surely for $\pi-$almost all $z \in \mathbb{Z}$. If in addition P is aperiodic then for $\pi-$almost all $z \in \mathbb{Z}$

$$\lim_{n \to \infty} \|P^n(z, \cdot) - \pi(\cdot)\|_{\mathrm{TV}} = 0 \;. \tag{4.43}$$

The result is fairly intuitive. Invariance of π is a fixed point property ensuring that if $Z_i \sim \pi$ then $Z_{i+1} \sim \pi$. $\pi-$irreducibility simply says that the Markov chain should be able to reach any set of $\pi-$positive probability from any $z \in \mathbb{Z}$ in a finite number of iterations. Periodicity would clearly prevent (S2) since the Markov chain would then periodically avoid visiting sets of positive $\pi-$probability. Averaging in (S1) removes the need for this property. We note that establishing these properties is often overlooked and a necessary prerequisite to any more refined analysis characterising their performance, such as quantitative finite time convergence bounds as found for example in [Dal17; DK19; DM17].

4.5 Standard reversible MH

We summarize in this Section the results presented in [Tie98, Section 2].

Definition 19 (Reversible kernel)**.** *A sub-Markovian kernel P on $(\mathbb{Z}, \mathcal{Z})$, P is π-reversible if and only if*

$$\check{\nu}\big(\mathrm{d}(z,z')\big) = \check{\nu}^F\big(\mathrm{d}(z,z')\big) \;, \tag{4.44}$$

where $\check{\nu}\big(\mathrm{d}(z,z')\big) = \pi(\mathrm{d}z)P(z,\mathrm{d}z')$ and $\check{\nu}^F\big(\mathrm{d}(z,z')\big) = F_{\#}\nu\big(\mathrm{d}(z,z')\big) = \pi(\mathrm{d}z')P(z',\mathrm{d}z)$ is the push-foward measure of ν by $F\colon (z,z') \mapsto (z',z)$.

From a proposal Markov kernel Q, the MH method consists of considering a sub-Markovian kernel $Q_\alpha(z,\mathrm{d}z') = \alpha(z,z')Q(z,\mathrm{d}z')$. If π and Q admit a common dominating σ-finite measure μ on Z, such that $\pi(\mathrm{d}z) = \pi(z)\mu(\mathrm{d}z)$ (we use the same notation for the probability and the density) and $Q(z,\mathrm{d}z') = q(z,z')\mu(\mathrm{d}z')$, Q_α is π-reversible if

$$\alpha(z,z') = \begin{cases} \mathsf{a}\left(\frac{\pi(z')q(z',z)}{\pi(z)q(z,z')}\right) & \pi(z)q(z,z') > 0 \;, \\ 1 & \text{otherwise} \;, \end{cases}$$

where for any $t \in \mathbb{R}_+^*$,

$$t\mathsf{a}(1/t) = \mathsf{a}(t) \;. \tag{4.45}$$

We may take for example $\mathsf{a}(t) = \min(1,t)$ or $\mathsf{a}(t) = t/(1+t)$ which correspond to the classical Metropolis-Hastings and Barker ratio, respectively. To obtain a π-reversible Markov kernel P, it suffices to add a Dirac mass, *i.e.*

$$P(z,\mathrm{d}z') = Q_\alpha(z,\mathrm{d}z') + \big(1 - Q_\alpha(z,\mathsf{Z})\big)\delta_z(\mathrm{d}z') \;. \tag{4.46}$$

This construction can be generalized to the case where π or Q do not admit a density. In particular, let Φ be an invertible mapping on $\mathbb{Z}$ satisfying $\Phi^{-1} = \Phi$ (*i.e.* Φ is an involution) and consider $Q(z,\mathrm{d}z') = \delta_{\Phi(z)}(\mathrm{d}z')$ (when the current state is z, then the proposal is $\Phi(z)$). Define the measure $\nu = \pi + \Phi_{\#}\pi$ and denote by $h(z) = \mathrm{d}\pi/\mathrm{d}\nu(z)$ (h is the density of π w.r.t. ν). Then, $h\big(\Phi(z)\big)$ is a density of $\Phi_{\#}\pi$ w.r.t. ν. Denote $A = \{z \in \mathbb{Z} : h(z) \times h \circ \Phi(z) > 0\}$. Detailed balance holds if and only if for π-almost all $z \in A$ (see [Tie98]):

$$\alpha\big(z,\Phi(z)\big)h(z)/h\circ\Phi(z) = \alpha\big(\Phi(z),z\big) \;.$$

If $\mathbb{Z} = \mathbb{R}^d$ and ν is the Lebesgue measure, we obtain $\alpha\big(z,\Phi(z)\big) = \bar{\alpha}(z)$, where

$$\bar{\alpha}(z) = \mathsf{a}\left(\frac{\pi\circ\Phi(z)}{\pi(z)}\mathrm{J}_\Phi(z)\right) \;.$$

4.6 Proofs of Section 4.2

4.6.1 Proof of (4.4)

Let $f\colon \mathbb{Z}^2 \to \mathbb{R}_+$ be a measurable function. The condition $\check{\mu}_P = \check{\mu}_P^{\mathrm{s}}$ implies

$$I = \iint \check{\mu}_P\big(\mathrm{d}(z,z')\big)f(z,z') = \iint \check{\mu}_P\big(\mathrm{d}(z,z')\big)f\big(s(z'),s(z)\big) = \iint \pi(\mathrm{d}z)P(z,\mathrm{d}z')f\big(s(z'),s(z)\big) \;.$$

Using the change of variable $\tilde{z}' = s(z)$ and since s is an involution, we get

$$I = \iint s_{\#}\pi(\mathrm{d}\tilde{z}')P\big(s(\tilde{z}'),\mathrm{d}z'\big)f\big(s(z'),\tilde{z}'\big) \;.$$

Applying now the change of variable $\tilde{z} = s(z')$, we finally obtain

$$I = \iint s_{\#}\pi(\mathrm{d}\tilde{z}')s_{\#}P\big(s(\tilde{z}'),\mathrm{d}\tilde{z}\big)f(\tilde{z},\tilde{z}') \; .$$

Note that, for any $z \in \mathbb{Z}$ and $A \in \mathcal{Z}$,

$$s_{\#}P(z,A) = \int P(z,\mathrm{d}z')\mathbb{1}_A\big(s(z')\big) = PS(z,A) \; ,$$

showing that

$$I = \iint s_{\#}\pi(\mathrm{d}\tilde{z}')PS\big(s(\tilde{z}'),\mathrm{d}\tilde{z}\big)f(\tilde{z},\tilde{z}') = \iint s_{\#}\pi(\mathrm{d}\tilde{z}')SPS(\tilde{z}',\mathrm{d}\tilde{z})f(\tilde{z},\tilde{z}'),$$

where we have used $\int SPg(\tilde{z}',\mathrm{d}\tilde{z}) = Pg\big(s(\tilde{z}')\big)$.

4.6.2 Proof of Proposition 3

We set $\check{\lambda} = \check{\nu} + \check{\nu}^s$. Note that $\check{\nu}$ and $\check{\nu}^s$ are absolutely continuous w.r.t. to $\check{\lambda}$. Denote by $\check{\lambda}^s = (F_s)_{\#}\check{\lambda}$ the pushforward of $\check{\lambda}$ by the transform $F_s(z,z') = \big(s(z'),s(z)\big)$: for any $C \in \mathcal{Z}^{\otimes 2}$

$$\check{\lambda}^s(C) = \int \mathbb{1}_C\big(s(z'),s(z)\big)\check{\lambda}\big(\mathrm{d}(z,z')\big) \; . \tag{4.47}$$

Since $(\check{\nu}^s)^s = \check{\nu}$, $\check{\lambda} = \check{\lambda}^s$. This implies, for any measurable function $f\colon \mathbb{Z}^2 \to \mathbb{R}_+$,

$$\iint f(z,z')\check{\lambda}\big(\mathrm{d}(z,z')\big) = \iint f\big(s(z'),s(z)\big)\check{\lambda}\big(\mathrm{d}(z,z')\big) \; . \tag{4.48}$$

We choose h to be a version of the Radon-Nikodym derivative $\mathrm{d}\check{\nu}/\mathrm{d}\check{\lambda}$ (the function is defined up to $\check{\lambda}$-negligible sets). Then by definition of $\check{\nu}^s$,

$$\iint f(z,z')\check{\nu}^s\big(\mathrm{d}(z,z')\big) = \iint f\big(s(z'),s(z)\big)\check{\nu}\big(\mathrm{d}(z,z')\big) = \iint f\big(s(z'),s(z)\big)h(z,z')\check{\lambda}\big(\mathrm{d}(z,z')\big) \tag{4.49}$$

$$= \iint f\big(s(z'),s(z)\big)h(z,z')\check{\nu}\big(\mathrm{d}(z,z')\big) + \iint f\big(s(z'),s(z)\big)h(z,z')\check{\nu}^s\big(\mathrm{d}(z,z')\big) \tag{4.50}$$

$$= \iint f(z,z')h\big(s(z'),s(z)\big)\check{\nu}^s\big(\mathrm{d}(z,z')\big) + \iint f(z,z')h\big(s(z'),s(z)\big)\check{\nu}\big(\mathrm{d}(z,z')\big) \tag{4.51}$$

$$= \iint f(z,z')h\big(s(z'),s(z)\big)\check{\lambda}\big(\mathrm{d}(z,z')\big) \; , \tag{4.52}$$

showing that

$$h\big(s(z'),s(z)\big) = \frac{\mathrm{d}\check{\nu}^s}{\mathrm{d}\check{\lambda}}(z,z') \; . \tag{4.53}$$

We then define

$$A_{\check{\nu}} = \big\{(z,z') \in \mathbb{Z}^2 \colon h(z,z') \times h\big(s(z'),s(z)\big) > 0\big\} \; . \tag{4.54}$$

In other words, if $(z,z') \notin A_{\check{\nu}}$, then either $h(z,z') = 0$ or $h\big(s(z'),s(z)\big) = 0$. Therefore, $\check{\nu}_{A,\mathrm{c}}$ and $\check{\nu}^s_{A,\mathrm{c}}$ are singular since $B_1 = \{(z,z') \in \mathbb{Z}^2 \colon h(z,z') > 0\}$, $B_2 = \{(z,z') \in \mathbb{Z}^2 \colon h\big(s(z'),s(z)\big) > 0\}$ are disjoint subsets of $A^{\mathrm{c}}_{\check{\nu}}$ and $\check{\nu}(B_2) = 0$, $\check{\nu}(B_1) = 0$. In addition, since for any set $B \in \mathbb{Z}^2$,

$$\mathbb{1}_{A_{\check{\nu}}\cap B}h = 0 \; \check{\lambda} - \text{a.e. if and only if } \mathbb{1}_{A_{\check{\nu}}\cap B}h^s = 0 \; \check{\lambda} - \text{a.e.}$$

the restrictions $\check{\nu}_A$ and $\check{\nu}^s_A$ are equivalent. In addition,

$$\frac{\mathrm{d}\check{\nu}_A}{\mathrm{d}\check{\nu}^s}(z,z') = \frac{h(z,z')}{h\big(s(z'),s(z)\big)} = r(z,z') \; , \quad (z,z') \in A_{\check{\nu}} \; , \tag{4.55}$$

satisfying $r(z,z') = 1/r\big(s(z'),s(z)\big)$.

4.6.3 Proof of Theorem 4

Define the σ-finite measure $\check{\rho}\big(\mathrm{d}(z,z')\big) = \alpha(z,z')\check{\nu}\big(\mathrm{d}(z,z')\big)$ and denote by $\check{\rho}^s = (F_s)_{\#}\check{\rho}$ the pushforward of $\check{\rho}$ by the transform $F_s(z,z') = \big(s(z'), s(z)\big)$: for any $C \in \mathcal{Z}^{\otimes 2}$

$$\check{\rho}^s(C) = \int \mathbb{1}_C\big(s(z'), s(z)\big)\check{\rho}\big(\mathrm{d}(z,z')\big) \; . \tag{4.56}$$

Note by definition of $\check{\nu}^s$ that

$$\check{\rho}^s\big(\mathrm{d}(z,z')\big) = \alpha\big(s(z'), s(z)\big)\check{\nu}^s\big(\mathrm{d}(z,z')\big) \tag{4.57}$$

We show below that under the stated assumptions $\check{\rho} = \check{\rho}^s$.

Define the function $\tilde{\alpha}(z,z') = \alpha\big(s(z'), s(z)\big)$. Since the set $A_{\check{\nu}}$ is s-symmetric, the set $A_{\check{\nu}}^{\mathrm{c}}$ is also s-symmetric and $\check{\nu}(\{(z,z') \in A_{\check{\nu}}^{\mathrm{c}} \, : \, \tilde{\alpha}(z,z') > 0\}) = 0$ using (i). Hence $\check{\rho}(A_{\check{\nu}}^{\mathrm{c}}) = \check{\rho}^s(A_{\check{\nu}}^{\mathrm{c}}) = 0$.

We have by Proposition 3 and (ii),

$$\begin{aligned} \mathbb{1}_{A_{\check{\nu}}}(z,z')\check{\rho}\big(\mathrm{d}(z,z')\big) &= \mathbb{1}_{A_{\check{\nu}}}(z,z')\alpha(z,z')\check{\nu}\big(\mathrm{d}(z,z')\big) = \alpha(z,z')r(z,z')\check{\nu}^s\big(\mathrm{d}(z,z')\big) \\ &= \mathbb{1}_{A_{\check{\nu}}}(z,z')\alpha\big(s(z'),s(z)\big)\nu^s\big(\mathrm{d}(z,z')\big) = \mathbb{1}_{A_{\check{\nu}}}(z,z')\check{\rho}^s\big(\mathrm{d}(z,z')\big) \; . \end{aligned} \tag{4.58}$$

Conversely, assume that $\check{\rho} = \check{\rho}^s$. Since by Proposition 3, $\check{\nu}_{A,\mathrm{c}}$ and $\check{\nu}^s_{A,\mathrm{c}}$ are mutually singular, there exist $B_1, B_2 \subset A_{\check{\nu}}^{\mathrm{c}}$ (see also the proof Proposition 3) such that $\check{\nu}_{A,\mathrm{c}}(B_2) = 0$ and $\check{\nu}^s_{A,\mathrm{c}}(B_1) = 0$. Therefore, we obtain using that $\check{\rho} = \check{\rho}^s$ and (4.57) that

$$\check{\rho}(A_{\check{\nu}}^{\mathrm{c}} \cap B_1) = \check{\rho}^s(A_{\check{\nu}}^{\mathrm{c}} \cap B_1) = 0 \; . \tag{4.59}$$

This result and $\check{\rho}(A_{\check{\nu}}^{\mathrm{c}} \cap B_2)$ imply $\check{\rho}(A_{\check{\nu}}^{\mathrm{c}}) = 0$ and therefore $\check{\nu}(\{(x,z') \in A_{\check{\nu}}^{\mathrm{c}} \, : \, \alpha(z,z') > 0\}) = 0$ showing (i). Finally, under the condition $\check{\rho} = \check{\rho}^s$ and Proposition 3, (4.58) holds and (ii) follows.

4.6.4 Checking the GMH rule (4.7)

We first check (i). By Proposition 3 and (4.7), $A_\nu^{\mathrm{c}} = B_1 \cup B_2$ where $B_1 = \big\{(z,z') \in \mathbb{Z}^2 \, : \, h(z,z') = 0\big\}$ and $B_2 = \big\{(z,z') \in \mathbb{Z}^2 \, : \, h\big(s(z'),s(z)\big) = 0\big\}$, and for any $(z,z') \in B_2 \setminus B_1$, $\alpha(z,z') = 0$. Therefore, to show (i), it suffices to establish that $\check{\nu}(\{\alpha = 0\} \cap B_1) = 0$ which follows from

$$\check{\nu}(B_1) = \int \mathbb{1}_{B_1}(z,z')\check{\nu}\big(\mathrm{d}(z,z')\big) = \int \mathbb{1}_{B_1}(z,z')h(z,z')\check{\lambda}\big(\mathrm{d}(z,z')\big) = 0 \; .$$

We now check (ii). Note that by Proposition 3 and using that F_s is an involution, for $(z,z') \in A_{\check{\nu}}$,

$$\begin{aligned} \alpha(z,z')r(z,z') &= \mathsf{a}\big(1/r(z,z')\big)r(z,z') && (4.60) \\ &= \mathsf{a}\big(r(z,z')\big) = \alpha\big(s(z'),s(z)\big) \; . && (4.61) \end{aligned}$$

4.6.5 Expressions for a and b

We check the conditions on the nonnegative weights a and b so that the sub-Markovian kernel

$$R(z,\mathrm{d}z') = a(z)\delta_z(\mathrm{d}z') + b(z)\delta_{s(z)}(\mathrm{d}z') \tag{4.62}$$

is (π, S)-reversible. For f a nonnegative measurable function, we get

$$SRSf(z') = a\big(s(z')\big)f(z') + b\big(s(z')\big)f\big(s(z')\big) \; .$$

Hence, we obtain, for any nonnegative measurable function g,

$$\iint \pi(\mathrm{d}z')SRS(z',\mathrm{d}z)f(z)g(z') = \int \pi(\mathrm{d}z')a\big(s(z')\big)f(z')g(z') + \int \pi(\mathrm{d}z')b\big(s(z')\big)f\big(s(z')\big)g(z') \tag{4.63}$$

$$= \int \pi(\mathrm{d}z)a\big(s(z')\big)\delta_z(\mathrm{d}z')f(z)g(z') + \int \pi(\mathrm{d}z')b(z')f(z')g\big(s(z')\big) \tag{4.64}$$

$$= \int \pi(\mathrm{d}z)a\big(s(z')\big)\delta_z(\mathrm{d}z')f(z)g(z') + \int \pi(\mathrm{d}z)b(z)f(z)\delta_{s(z)}(\mathrm{d}z')g(z') \ , \tag{4.65}$$

where we have used $s_{\#}\pi = \pi$. The result implies that

$$\pi(\mathrm{d}z')SRS(z',\mathrm{d}z) = \pi(\mathrm{d}z)a\big(s(z)\big)\delta_z(\mathrm{d}z') + \pi(\mathrm{d}z)b(z)\delta_{s(z)}(\mathrm{d}z').$$

Therefore, (4.4) is satisfied (e.g. $\pi(\mathrm{d}z')SRS(z',\mathrm{d}z) = \pi(\mathrm{d}z)R(z,\mathrm{d}z')$) and R is (π,S)-reversible if $a(z) = a\big(s(z)\big)$.

In addition, the total mass of $Q_\alpha(z,\mathrm{d}z')$ is $Q_\alpha(z,\mathbb{Z})$. The missing mass is therefore $1 - Q_\alpha(z,\mathbb{Z})$. Since the total mass of R is $a(z) + b(z)$ we must have $a(z) + b(z) = 1 - Q_\alpha(z,\mathbb{Z})$.

We may for example set $a(z) = 0$ and $b(z) = 1 - Q_\alpha(z,\mathsf{Z})$, which coincides with the classical MH rule when $s = \mathrm{Id}$. We may also take $a(z) = 1 - Q_\alpha(z,\mathbb{Z}) - b(z)$ where b satisfies $0 \leqslant b(z) \leqslant 1 - Q_\alpha(z,\mathbb{Z})$ and $b(z) - b\big(s(z)\big) = Q_\alpha(s(z),\mathbb{Z}) - Q_\alpha(z,\mathbb{Z})$. As suggested in [TCV11], we may set $b(z) = \max\big(0, Q_\alpha\big(s(z),\mathbb{Z}\big) - Q_\alpha(z,\mathbb{Z})\big)$ which is shown to be optimal w.r.t. to the Peskun ordering in [AL19b]. Note however that this choice for b is not always easily computable.

4.6.6 Applications of (4.7): case with densities

Note that $\check{\nu}\big(\mathrm{d}(z,z')\big) = \pi(z)q(z,z')\mu^{\otimes 2}\big(\mathrm{d}(z,z')\big)$, $\check{\nu}^s\big(\mathrm{d}(z,z')\big) = \pi\big(s(z')\big)q\big(s(z'),s(z)\big)\mu^{\otimes 2}\big(\mathrm{d}(z,z')\big)$, since $s_{\#}\mu = \mu$. In addition, $h = \tilde{h}/\{\tilde{h} + \tilde{h}\circ F_s\}$, $\tilde{h}(z,z') = \pi(z)q(z,z')$ and therefore $A_{\check{\nu}}$ in (4.10) is given by

$$A_{\check{\nu}} = \big\{\pi(z)q(z,z') \times \pi\big(s(z')\big)q\big(s(z'),s(z)\big) > 0\big\} \ ,$$

and for $(z,z') \in A_{\check{\nu}}$,

$$r(z,z') = \frac{\pi(z)q(z,z')}{\pi\big(s(z')\big)q\big(s(z'),s(z)\big)} \ .$$

Therefore, we obtain using (4.7) that

$$\alpha(z,z') = \begin{cases} \mathsf{a}\left[\frac{\pi\left(s(z')\right)q\left(s(z'),s(z)\right)}{\pi(z)q(z,z')}\right] & \pi(z)q(z,z')\mathrm{ne}0, \\ 1 & \pi(z)q(z,z') = 0 \ . \end{cases} \tag{4.66}$$

In addition, note that using $s_{\#}\pi = \pi$, $s_{\#}\mu = \mu$ and s is an involution, we obtain that

$$\pi = \pi \circ s \ . \tag{4.67}$$

Therefore, we obtain

$$\alpha(z,z') = \begin{cases} \mathsf{a}\left[\frac{\pi(z')q\left(s(z'),s(z)\right)}{\pi(z)q(z,z')}\right] & \pi(z)q(z,z')\mathrm{ne}0, \\ 1 & \pi(z)q(z,z') = 0 \ . \end{cases} \tag{4.68}$$

4.6.7 Proof of Theorem 6

We preface the proof by the following result. Define $\mathsf{Z}^+ = \{z \in \mathsf{Z} \colon \pi(z) > 0\}$ and set

$$P(z,\mathrm{d}z') = Q_\alpha(z,\mathrm{d}z') + \{1 - Q_\alpha(z,\mathbb{Z})\}\delta_{s(z)}(\mathrm{d}z') \ . \tag{4.69}$$

where $Q_\alpha(z,z') = \alpha(z,z')Q(z,\mathrm{d}z')$ and α is given by (4.7). Note that P corresponds to (4.9) with $a \equiv 0$ and $b(z) = 1 - Q_\alpha(z,\mathbb{Z})$.

Proposition 20. *Consider P defined by (4.69). Assume that P is π-irreducible and $Q(z, \mathsf{Z}^+) = 1$ for any $z \notin \mathsf{Z}^+$. Further, suppose that π is not a Dirac mass. Then, P is Harris recurrent.*

Proof. Since π is invariant for P by Theorem 4, P is recurrent by [Dou+18, Theorem 10.1.6]. Therefore, [Dou+18, Corollary 9.2.16, Proposition 5.2.12] show that for any bounded harmonic function $h\colon \mathsf{Z} \to \mathbb{R}$, *i.e.* satisfying $Ph = h$, $h = \pi(h)$, π a.e.. Then, if $A_h = \{h \neq \pi(h)\}$, $\pi(A_h) = 0$. By [Dou+18, Theorem 10.2.11], P is Harris recurrent if

$$h(z) = \pi(h) \text{ for any } z \in \mathsf{Z} \,. \tag{4.70}$$

First, consider $z \in \mathsf{Z}^+$. Define $B = \{z' \colon \pi(z')q\big(s(z'), s(z)\big) > 0\}$ and $C = \{z' \colon q(z, z') = 0\}$. Let A be a π-negligible set, $\pi(A) = 0$. Using $\mathsf{a}(t) \leqslant t$ by (4.8) for any $t \in \mathbb{R}_+^*$, $\pi(z) \neq 0$ and (4.68), we get

$$\begin{aligned}
\int \mathbb{1}_A(z')\alpha(z, z')q(z, z')\mu(\mathrm{d}z') &= \int \mathbb{1}_{A\cap B\cap C}(z')\alpha(z, z')q(z, z')\mu(\mathrm{d}z') \\
&\quad + \int \mathbb{1}_{A\cap B\cap C^{\mathrm{c}}}(z')\mathsf{a}\left[\frac{\pi(z')q\big(s(z'), s(z)\big)}{\pi(z)q(z, z')}\right] q(z, z')\mu(\mathrm{d}z') \\
&\leqslant \int \mathbb{1}_{A\cap B\cap C^{\mathrm{c}}}(z')\frac{\pi(z')q\big(s(z'), s(z)\big)}{\pi(z)}\mu(\mathrm{d}z') \\
&\leqslant \int \mathbb{1}_{A\cap B\cap C^{\mathrm{c}}}(z')\frac{q\big(s(z'), s(z)\big)}{\pi(z)}\pi(\mathrm{d}z') = 0 \,,
\end{aligned} \tag{4.71}$$

where the last identity follows from $\pi(A) = 0$. Applying this identity with A_h yields to

$$\int \alpha(z, z')q(z, z')h(z')\mathrm{d}\mu(z') = \int \mathbb{1}_{A_h^{\mathrm{c}}}(z')\alpha(z, z')q(z, z')h(z')\mathrm{d}\mu(z') = \pi(h)Q_\alpha(z, \mathsf{Z}) \,. \tag{4.72}$$

Therefore, the condition $Ph(z) = h(z)$ for any $z \in \mathbb{Z}$ and (4.69) imply that

$$h(z) = \pi(h)Q_\alpha(z, \mathsf{Z}) + h \circ s(z)\{1 - Q_\alpha(z, \mathsf{Z})\} \,. \tag{4.73}$$

Applying P to the previous equation, we obtain, denoting $\bar\alpha(z) = Q_\alpha(z, \mathbb{Z})$

$$\begin{aligned}
h(z) = Ph(z) &= \pi(h)\{Q_\alpha\bar\alpha(z) + \{1 - \bar\alpha(z)\}\bar\alpha \circ s(z)\} \\
&\quad + \int Q_\alpha(z, \mathrm{d}z')h \circ s(z')\{1 - \bar\alpha(z')\} + h(z)\{1 - \bar\alpha(z)\}\big\{1 - \bar\alpha\big(s(z)\big)\big\} \,.
\end{aligned} \tag{4.74}$$

Denote $A_{h\circ s} = \{z \in \mathbb{Z} \,:\, h \circ s(z) \mathrm{ne} \pi(h)\}$. Note that, $\pi(A_{h\circ s}) = s_\# \pi(A_h) = \pi(A_h) = 0$. Using (4.71), we get for $z \in \mathbb{Z}^+$, $Q_\alpha(z, A_{h\circ s}) = 0$, which implies

$$\begin{aligned}
\int Q_\alpha(z, \mathrm{d}z')h \circ s(z')\{1 - \bar\alpha(z')\} &= \int \mathbb{1}_{A_{h\circ s}^{\mathrm{c}}}(z')Q_\alpha(z, \mathrm{d}z')h \circ s(z')\{1 - \bar\alpha(z')\} &(4.75)\\
&= \pi(h) \int Q_\alpha(z, \mathrm{d}z')\{1 - \bar\alpha(z')\} \,. &(4.76)
\end{aligned}$$

Plugging this relation into (4.74) we obtain

$$h(z) = \pi(h)[Q_\alpha\bar\alpha(z) + \{1 - \bar\alpha(z)\}\bar\alpha \circ s(z)] + \pi(h)[\bar\alpha(z) - Q_\alpha\bar\alpha(z)] + h(z)\{1 - \bar\alpha(z)\}\{1 - \bar\alpha \circ s(z)\} \,. \tag{4.77}$$

Using straightforward algebra, the previous identity implies

$$\{\pi(h) - h(z)\}\{\bar\alpha(z) + \bar\alpha \circ s(z) - \bar\alpha(z) \times \bar\alpha \circ s(z)\} = 0.$$

Since P is π-irreducible and π is not a Dirac mass, $\bar\alpha(z) \neq 0$, we get that for all $z \in \mathsf{Z}^+$,

$$h(z) = \pi(h) \,. \tag{4.78}$$

Consider now the case $z \notin \mathsf{Z}_+$. Using that $Q(z, \mathsf{Z}_+)$ by assumption and $\alpha(z, z') = 1$ by (4.68) for any $z' \in \mathsf{Z}$, we get

$$h(z) = Ph(z) = \int_{\mathsf{Z}_+} q(z, z')h(z')\mu(z') = \int_{\mathsf{Z}_+} \{q(z, z')h(z')/\pi(z')\}\pi(z')\mu(z') = \pi(h) \; . \tag{4.79}$$

Combining this result with (4.78) completes the proof of (4.70). □

Proposition 21. *Assume the conditions of Theorem 6. Then for any $A \in \mathcal{Z}$ such that $\pi(A) > 0$, we have*

$$P(z, A) > 0 \quad \textit{for any } z \in \mathbb{Z} \; . \tag{4.80}$$

Proof. Consider first the case $z \in \mathsf{Z}_+ = \{z \in \mathbb{Z} : \pi(z) > 0\}$. Then, by (4.68) and the condition if $\pi(z') > 0$, then $q(\tilde{z}, z') \times q\big(s(\tilde{z}), s(z')\big) > 0$ for any $\tilde{z} \in \mathsf{Z}$, we have

$$P(z, A) \geqslant \int \mathbb{1}_{A \cap \mathsf{Z}_+}(z') \mathsf{a}\left[\frac{\pi(z')q\big(s(z'), s(z)\big)}{\pi(z)q(z, z')}\right] q(z, z')\mu(\mathrm{d}z') > 0 \; , \tag{4.81}$$

since $\pi(A) > 0$ implies that $\mu(A \cap \mathsf{Z}_+) > 0$. Second consider the case $z \notin \mathsf{Z}_+$. Then, $\alpha(z, z') = 1$ for any $z' \in \mathsf{Z}$ and we get

$$P(z, A) \geqslant \int \mathbb{1}_{A \cap \mathsf{Z}_+}(z')q(z, z')\mu(\mathrm{d}z') > 0 \; , \tag{4.82}$$

which concludes the proof of (4.80). □

Proof of Theorem 6. π-irreducibility of P follows from Proposition 21. We show that P is π-irreducible and aperiodic. Indeed, this result and [Dou+18, Theorem 7.2.1, Theorem 11.3.1] imply (S2) for all $z \in \mathbb{Z}$. Finally, [Dou+18, Corollary 9.2.16, Proposition 5.2.14] establish (S1) for all $z \in \mathbb{Z}$.

The fact that P is aperiodic is a direct consequence of (4.80) and [Dou+18, Theorem 9.3.10]. □

4.6.8 Proofs of (4.13) and (4.14)

Consider $\check{\nu}\big(\mathrm{d}(z, z')\big) = \pi(\mathrm{d}z)\delta_{\Phi(z)}(\mathrm{d}z')$, where Φ is an invertible mapping on $\mathbb{Z}$ satisfying $\Phi^{-1} = s \circ \Phi \circ s$. For any measurable function $f \colon \mathbb{Z}^2 \to \mathbb{R}_+$, we get

$$\iint \check{\nu}^s\big(\mathrm{d}(z, z')\big) f(z, z') = \iint \check{\nu}\big(\mathrm{d}(z, z')\big) f\big(s(z'), s(z)\big) = \int \pi(\mathrm{d}z) f\big(s \circ \Phi(z), s(z)\big) \tag{4.83}$$

$$= \int s_{\#}\pi(\mathrm{d}z') f(s \circ \Phi \circ s(z'), z') = \iint \pi(\mathrm{d}z')\delta_{\Phi^{-1}(z')}(\mathrm{d}z) f(z, z') \; . \tag{4.84}$$

Define

$$\check{\lambda}\big(\mathrm{d}(z, z')\big) = \check{\nu}\big(\mathrm{d}(z, z')\big) + \check{\nu}^s\big(\mathrm{d}(z, z')\big) = \pi(\mathrm{d}z)\delta_{\Phi(z)}(\mathrm{d}z') + \pi(\mathrm{d}z')\delta_{\Phi^{-1}(z')}(\mathrm{d}z) \; . \tag{4.85}$$

Set $\lambda = \pi + \Phi^{-1}_{\#}\pi$ and define $k(z) = (\mathrm{d}\pi/\mathrm{d}\lambda)(z)$. Note that for any measurable function $f \colon \mathbb{Z}^2 \to \mathbb{R}_+$,

$$\int f(z, z')\check{\lambda}\big(\mathrm{d}(z, z')\big) = \int \pi(\mathrm{d}z) f\big(z, \Phi(z)\big) + \int \pi(\mathrm{d}z') f\big(\Phi^{-1}(z'), \Phi \circ \Phi^{-1}(z')\big) \tag{4.86}$$

$$= \int f\big(z, \Phi(z)\big)\lambda(\mathrm{d}z) \; . \tag{4.87}$$

Then, for any measurable function $f \colon \mathbb{Z}^2 \to \mathbb{R}_+$, we get since $k(z) = \mathrm{d}\pi/\mathrm{d}\lambda(z)$,

$$\iint \check{\nu}\big(\mathrm{d}(z, z')\big) f(z, z') = \int \pi(\mathrm{d}z) f\big(z, \Phi(z)\big) = \int k(z) f\big(z, \Phi(z)\big)\lambda(\mathrm{d}z) \; . \tag{4.88}$$

On the other hand,

$$\iint \check{\nu}\big(\mathrm{d}(z,z')\big) f(z,z') = \int h(z,z') f(z,z') \check{\lambda}\big(\mathrm{d}(z,z')\big) = \int h\big(z,\Phi(z)\big) f\big(z,\Phi(z)\big) \lambda(\mathrm{d}z) \; . \tag{4.89}$$

showing that $h\big(z,\Phi(z)\big) = (\mathrm{d}\check{\nu}/\mathrm{d}\check{\lambda})\big(z,\Phi(z)\big) = k(z)$ λ-a.e. . Setting $z' = s(z)$ and using $\Phi^{-1} = s \circ \Phi \circ s$, we get

$$h\big(s \circ \Phi(z), s(z)\big) = h(s \circ \Phi \circ s(z'), z') = h(\Phi^{-1}(z'), z')$$

Setting now $z'' = \Phi^{-1}(z')$, *i.e.* $z'' = \Phi^{-1} \circ s(z) = s \circ \Phi(z)$, we finally obtain

$$h\big(s \circ \Phi(z), s(z)\big) = h\big(z'', \Phi(z'')\big) = k(z'') = k\big(\Phi^{-1} \circ s(z)\big) = k\big(s \circ \Phi(z)\big) \; .$$

The proof of (4.13) and (4.14) is concluded using Theorem 4.

4.6.9 Proof of (4.17)

We now consider the case $\mathbb{Z} = \mathbb{R}^d$ and $\pi(\mathrm{d}z) = \pi(z)\mathrm{d}z$. We first identify the dominating measure λ defined in (4.13). For any nonnegative measurable function f,

$$\lambda(f) = \int f(z)\pi(z)\mathrm{d}z + \int f \circ \Phi^{-1}(z)\pi(z)\mathrm{d}z \tag{4.90}$$

$$= \int f(z)\pi(z)\mathrm{d}z + \int f(z)\pi \circ \Phi(z)\mathrm{J}_\Phi(z)\mathrm{d}z \; . \tag{4.91}$$

Hence, $\lambda(\mathrm{d}z) = \lambda(z)\mathrm{d}z$ with

$$\lambda(z) = \pi(z) + \pi \circ \Phi(z)\mathrm{J}_\Phi(z) \; . \tag{4.92}$$

Plugging this expression in (4.13), we get that

$$k(z) = \frac{\mathrm{d}\pi}{\mathrm{d}\lambda}(z) = \frac{\pi(z)}{\pi(z) + \pi \circ \Phi(z)\mathrm{J}_\Phi(z)} \; . \tag{4.93}$$

We have for any function nonnegative measurable function f,

$$s_\# \pi(f) = \int \pi(z) f \circ s(z)\mathrm{d}z = \int \pi \circ s(z)\mathrm{J}_s(z) f(z)\mathrm{d}z, \tag{4.94}$$

which implies since $s_\#\pi = \pi$, that $\pi \circ s(z) = \pi(z)/\mathrm{J}_s(z)$ Leb_d-a.e.. Hence, we get that

$$\begin{aligned} k\big(s \circ \Phi(z)\big) &= \frac{\pi\big(s \circ \Phi(z)\big)}{\pi\big(s \circ \Phi(z)\big) + \pi\big(\Phi \circ s \circ \Phi(z)\big)\mathrm{J}_\Phi\big(s \circ \Phi(z)\big)} \\ &= \frac{\pi \circ \Phi(z)}{\pi \circ \Phi(z) + \pi(z)\rho_\Phi(z)}, \end{aligned} \tag{4.95}$$

where we have set

$$\rho_\Phi(z) = \frac{\mathrm{J}_\Phi\big(s \circ \Phi(z)\big)\mathrm{J}_s\big(\Phi(z)\big)}{\mathrm{J}_s(z)} \; . \tag{4.96}$$

Since $\Phi \circ s \circ \Phi(z) = s(z)$, we get that

$$\mathrm{J}_\Phi\big(s \circ \Phi(z)\big)\mathrm{J}_s\big(\Phi(z)\big)\mathrm{J}_\Phi(z) = \mathrm{J}_s(z),$$

which implies that

$$\rho_\Phi(z) = 1/\mathrm{J}_\Phi(z) \; .$$

Plugging this expression into (4.95), we finally get that

$$k\big(s \circ \Phi(z)\big) = \frac{\pi \circ \Phi(z)\mathrm{J}_\Phi(z)}{\pi(z) + \pi \circ \Phi(z)\mathrm{J}_\Phi(z)} \; .$$

Combining this result with (4.14) and (4.93) concludes the proof of (4.17).

4.7 Proofs of Section 4.3

4.7.1 Generalized Hamiltonian Monte Carlo algorithms

Consider the two following assumptions:

NICE1. *For any $i \in \{1, \dots, m\}$, $N_{m+1-i} = M_i$.*

NICE2. *For any $i \in \{1, \dots, m\}$, M_i and N_i are* L*-Lipschitz and $h \leqslant c_0/[\mathrm{L}^{1/2}m]$, where $c_0 \approx 0.3$.*

Lemma 22. *Assume **NICE**1. Then, $s \circ \Phi \circ s = \Phi^{-1}$.*[1]

Proof. Denote for $i \in \{1, \dots, m\}$, $\mathrm{F}_i = \Psi_{N_i} \circ \Upsilon \circ \Psi_{M_i}$, where $\Upsilon(x, p) = (x + hp, p)$, $\Psi_M(x, p) = \big(x, p + hM(x)\big)$ and $\Psi_N(x, p) = \big(x, p + hN(x)\big)$. Each of those transforms verify

$$s \circ \Upsilon \circ s = \Upsilon^{-1} \ , \ s \circ \Psi_M \circ s = \Psi_M^{-1} \ , \ s \circ \Psi_N \circ s = \Psi_N^{-1} \ . \tag{4.97}$$

Then, $s \circ \mathrm{F}_i \circ s = \Psi_{N_i}^{-1} \circ \Upsilon^{-1} \circ \Psi_{M_i}^{-1}$ and thus,

$$s \circ \Phi \circ s = \Psi_{N_m}^{-1} \circ \Upsilon^{-1} \circ \Psi_{M_m}^{-1} \circ \cdots \circ \Psi_{N_1}^{-1} \circ \Upsilon^{-1} \circ \Psi_{M_1}^{-1} \ . \tag{4.98}$$

On the other hand,

$$\Phi^{-1} = \mathrm{F}_1^{-1} \circ \cdots \circ \mathrm{F}_m^{-1} = \Psi_{M_1}^{-1} \circ \Upsilon^{-1} \circ \Psi_{N_1}^{-1} \circ \cdots \circ \Psi_{M_m}^{-1} \circ \Upsilon^{-1} \circ \Psi_{N_m}^{-1} \ . \tag{4.99}$$

Applying **NICE**1 concludes the proof. □

Reversibility vs. persistency

[2] In Subsection 4.3.1 we define the deterministic Markov kernel on $\mathbb{Z}$,

$$P\big((x, p); \mathrm{d}(x', p')\big) = \bar{\alpha}(x, p)\delta_{\Phi(x,p)}\big(\mathrm{d}(x', p')\big) + \big(1 - \bar{\alpha}(x, p)\big)\delta_{(x,-p)}\big(\mathrm{d}(x', p')\big) \ , \tag{4.100}$$

where $\bar{\alpha}(x, p)$ is given by (4.21). Such kernels are most likely not ergodic and the momentum must be refreshed in order to lead to an ergodic Markov chain $\big(Z_i = (X_i, P_i)\big)_{i\in\mathbb{N}}$. We focus on "full refreshment", that is the scenario where the momentum is drawn afresh from its stationary distribution before applying P, in which case it can be checked that $(X_i)_{i\in\mathbb{N}}$ is a Markov chain of (marginal) Markov kernel,

$$K(x, \mathrm{d}y) = K_\alpha(x, \mathrm{d}y) + \{1 - K_\alpha(x, \mathbb{R}^d)\}\delta_x(\mathrm{d}y) \ , \tag{4.101}$$

where $K_\alpha(x, \mathrm{d}y) = \int \bar{\alpha}(x, p)\varphi(p)\delta_{G_x(p)}(\mathrm{d}y)\mathrm{d}p$ with $G_x(p) = \mathrm{proj}_1 \circ \Phi(x, p)$, $\mathrm{proj}_1(x, p) = x$. Sampling from K is described in Algorithm 1. It is also the case that $(X_i)_{i\in\mathbb{N}}$ is time-reversible (see e.g. [Dua+87]), which has the disadvantage of loosing the potentially advantageous persistency features of P. It is possible to recover persistency by considering the mixture of kernels

$$T := \omega P + (1 - \omega)L \tag{4.102}$$

for $\omega \in [0, 1]$, where $L\big((x, p), \mathrm{d}(y, q)\big) = K(x, \mathrm{d}y)\varphi(q)\mathrm{d}q$. The kernel L refreshes independently the position x and the momentum p. Since the target distribution is the product of π_0 and φ (thus the position and the momentum are independent), it is easily checked as well that L leaves π-invariant since

$$\pi L\big(\mathrm{d}(y, q)\big) = \int \pi_0(\mathrm{d}x)K(x, \mathrm{d}y)\varphi(p)\mathrm{d}p\varphi(q)\mathrm{d}q = \pi_0(\mathrm{d}y)\varphi(q)\mathrm{d}q = \pi\big(\mathrm{d}(y, q)\big) \ . \tag{4.103}$$

Note further that L is π-reversible as K is π_0-reversible. In what follows we establish π_0-irreducibility of K, and in particular that Proposition 21 holds, which immediately implies $\pi_0 \otimes (\varphi \times \mathrm{Leb})-$irreducibility of T and allows us to apply Theorem S79 and conclude about convergence. Sampling from T is described in Algorithm 2. In future work we will consider the scenario where p is updated using partial refreshment such as suggested in [Hor91], for example by using an AR(1) process when φ is a normal distribution, which requires an extension of our results; see Algorithm 3.

[1] This condition is missing in the main text due to a late error with our versioning system.

[2] We follow the order of the main text here, but this discussion should be after the proofs of the following subsections.

Proof of (4.24) **and Theorem 7**

We first establish the elementary equation (4.24).

Lemma 23. *Assume that for each $x \in \mathbb{R}^d$, $G_x : p \mapsto \mathrm{proj}_1 \circ \Phi(x,p)$ is a C^1-diffeomorphism. Then, $K_\alpha(x, \mathrm{d}y)$ has a density $K_\alpha(x, \mathrm{d}y) = \alpha(x,y)q(x,y)\mathrm{d}y$ where*

$$\alpha(x,y) = \mathsf{a}\left(\frac{\Pi(y)\varphi\big\{H_x\big(G_x^{-1}(y)\big)\big\}}{\Pi(x)\varphi\big(G_x^{-1}(y)\big)}\right) , \tag{4.104}$$

$$q(x,y) = \varphi(G_x^{-1}(y))\mathrm{J}_{G_x^{-1}}(y) . \tag{4.105}$$

Proof. First by (4.21), for any $(x,p) \in \mathbb{R}^{2d}$, we have by definition

$$K_\alpha f(x) = \int \bar{\alpha}(x,p)\varphi(p) f\big(G_x(p)\big)\mathrm{d}p = \int \mathsf{a}\left(\frac{\pi \circ \Phi(x,p)}{\pi(x,p)}\right)\varphi(p) f\big(G_x(p)\big)\mathrm{d}p . \tag{4.106}$$

Then, using the change of variable $y = G_x(p)$, we obtain

$$K_\alpha f(x) = \int \mathsf{a}\left(\frac{\pi \circ \Phi\big(x, G_x^{-1}(y)\big)}{\pi\big(x, G_x^{-1}(y)\big)}\right)\varphi\big(G_x^{-1}(y)\big)\mathrm{J}_{G_x^{-1}}(y) f(y)\mathrm{d}y , \tag{4.107}$$

which concludes the proof of (4.24) since $\pi = \pi_0 \otimes \varphi$. □

We now prove Theorem 7 which gives conditions on the mappings $\{M_i, N_i\}_{i=1}^m$ that ensure that for all $x \in \mathbb{R}^d$, G_x is a C^1-diffeomorphism.

Theorem 24. *Assume **NICE**2. Then, for any $x \in \mathbb{R}^d$, the function $G_x(p) = \mathrm{proj}_1 \circ \Phi(x,p)$ is a C^1 diffeomorphism. Moreover, the GMH kernel based on NICE transitions is ergodic.*

We preface the proof by some auxiliary results. Recall that one step of the NICE transition is given by $\mathrm{F}_i(x_i, p_i) = (x_{i+1}, p_{i+1})$, where:

$$\begin{cases} p_{i+1/2} &= p_i + hM_i(x_i), \\ x_{i+1} &= x_i + hp_{i+1/2}, \\ p_{i+1} &= p_{i+1/2} + hN_i(x_{i+1}). \end{cases} \tag{4.108}$$

Denote

$$\Lambda^{(j)} = \mathrm{F}_j \circ \cdots \circ \mathrm{F}_1 . \tag{4.109}$$

Lemma 25. *For all $k \in \mathbb{N}^*$, we get*

$$x_k = x_1 + (k-1)hp_1 + h^2 \sum_{i=1}^{k-1}(k-i)M_i(x_i) + h^2 \sum_{i=1}^{k-2}(k-1-i)N_i(x_{i+1}) , \tag{4.110}$$

$$p_k = p_1 + h\sum_{i=1}^{k-1} M_i(x_i) + h\sum_{i=1}^{k-1} N_i(x_{i+1}) . \tag{4.111}$$

Proof. The proof proceeds by induction. The assertion is obviously true for $k = 2$. Let us suppose that

the assertion holds true for some $k \in \mathbb{N}^*$.

$$p_{k+1/2} = p_k + hM_k(x_k) = p_1 + h\sum_{i=1}^{k} M_i(x_i) + h\sum_{i=1}^{k-1} N_i(x_{i+1}), \tag{4.112}$$

$$x_{k+1} = x_k + hp_{k+1/2} \tag{4.113}$$

$$= x_1 + (k-1)hp_1 + h^2\sum_{i=1}^{k-1}(k-i)M_i(x_i) + h^2\sum_{i=1}^{k-2}(k-1-i)N_i(x_{i+1}) \tag{4.114}$$

$$+ h\left(p_1 + h\sum_{i=1}^{k} M_i(x_i) + h\sum_{i=1}^{k-1} N_i(x_{i+1})\right) \tag{4.115}$$

$$= x_1 + khp_1 + h^2\sum_{i=1}^{k}(k+1-i)M_i(x_i) + h^2\sum_{i=1}^{k-1}(k-i)N_i(x_{i+1}), \tag{4.116}$$

$$p_{k+1} = p_{k+1/2} + hN_k(x_{k+1}) = p_1 + h\sum_{i=1}^{k} M_i(x_i) + h\sum_{i=1}^{k} N_i(x_{i+1}). \tag{4.117}$$

This concludes the proof. □

Denote for all $(x_1, p_1) \in \mathbb{R}^{2d}$,

$$G_{x_1}(p_1) = x_1 + mhp_1 + h^2\Theta_m(x_1, p_1), \tag{4.118}$$

$$\Theta_m(x_1, p_1) = \sum_{i=1}^{m}(m+1-i)M_i(x_i) + \sum_{i=1}^{m-1}(m-i)N_i(x_{i+1}). \tag{4.119}$$

Since the mappings $\{M_k\}_{k=1}^m$, $\{N_k\}_{k=1}^m$ are continuously differentiable, the mapping Ξ_k is continuously differentiable.

Lemma 26. *For $(x_1, p_1), (\tilde{x}_1, \tilde{p}_1) \in \mathbb{R}^{2d}$, denote $(x_{k+1}, p_{k+1}), (\tilde{x}_{k+1}, \tilde{p}_{k+1})$ the states obtained after k NICE-based transitions. Under the Lipschitz constraint* L*, we have*

$$\|x_{k+1} - \tilde{x}_{k+1}\| + \mathrm{L}^{-1/2}\|p_{k+1} - \tilde{p}_{k+1}\| \leqslant \left\{1 + h\mathrm{L}^{1/2}\vartheta_1\left(h\mathrm{L}^{1/2}\right)\right\}^k \left\{\|x_1 - \tilde{x}_1\| + \mathrm{L}^{-1/2}\|p_1 - \tilde{p}_1\|\right\}, \tag{4.120}$$

where $\vartheta_1(s) = 2 + s + s^2$.

Proof. We show this result for $k = 1$ and the apply a straightforward induction. For $k = 1$, we have

$$\|x_2 - \tilde{x}_2\| = \left\|x_1 + h^2 M_1(x_1) + hp_1 - \left\{\tilde{x}_1 + h^2 M_1(\tilde{x}_1) + h\tilde{p}_1\right\}\right\| \tag{4.121}$$

$$\leqslant \left(1 + h^2\mathrm{L}\right)\|x_1 - \tilde{x}_1\| + h\|p_1 - \tilde{p}_1\| . \tag{4.122}$$

Moreover, we have

$$\|p_2 - \tilde{p}_2\| = \|p_1 - \tilde{p}_1 - h\{N_1(x_2) + M_1(x_1)\} + h\{N_1(\tilde{x}_2) + M_1(\tilde{x}_1)\}\| \tag{4.123}$$

$$\leqslant \|p_1 - \tilde{p}_1\| + h\mathrm{L}\{\|\tilde{x}_2 - x_2\| + \|\tilde{x}_1 - x_1\|\} \tag{4.124}$$

$$\leqslant \left(1 + h^2\mathrm{L}\right)\|p_1 - \tilde{p}_1\| + h\mathrm{L}\left(2 + h^2\mathrm{L}\right)\|x_1 - \tilde{x}_1\| . \tag{4.125}$$

Summing the two previous expression, we get the desired result for $k = 1$. □

Lemma 27. *For any $h > 0$, we have*

$$\sup_{(x,p,v)\in\mathbb{R}^{3d}} \{\|\Theta_m(x,p) - \Theta_m(x,v)\| / \|p - v\|\} \leqslant (m/h)\left\{\left(1 + h\mathrm{L}^{1/2}\vartheta_1(h\mathrm{L})\right)^m - 1\right\} . \tag{4.126}$$

Proof. By Lemma 26, we have that, for any $(x,p,v) \in \mathbb{R}^{3d}$,

$$\| \operatorname{proj}_1 \circ \Lambda^{(m)}(x,p) - \operatorname{proj}_1 \circ \Lambda^{(m)}(x,v)\| \leqslant \left\{1 + h\mathrm{L}^{1/2}\vartheta_1\left(h\mathrm{L}^{1/2}\right)\right\}^m \mathrm{L}^{-1/2}\|p-v\| \,. \tag{4.127}$$

Denote $\Lambda_1^{(i)} = \operatorname{proj}_1 \circ \Lambda^{(i)}$ and as a convention $\Lambda_1^{(0)} = \operatorname{proj}_1$. We obtain

$$\begin{aligned}
&\|\Theta_m(x,p) - \Theta_m(x,v)\| && (4.128)\\
&\quad \leqslant \mathrm{L}\left(\sum_{i=1}^{m-1} 2(m+1-i)\|\Lambda_1^{(i-1)}(x,p) - \Lambda_1^{(i-1)}(x,v)\| + \|\Lambda_1^{(m-1)}(x,p) - \Lambda_1^{(m-1)}(x,v)\|\right) && (4.129)\\
&\quad \leqslant \mathrm{L}^{1/2}\left(\sum_{i=1}^{m-1} 2(m+1-i)\left\{1 + h\mathrm{L}^{1/2}\vartheta_1\left(h\mathrm{L}^{1/2}\right)\right\}^{i-1} + \left\{1 + h\mathrm{L}^{1/2}\vartheta_1\left(h\mathrm{L}^{1/2}\right)\right\}^{m-1}\right)\|p-v\| && (4.130)\\
&\quad \leqslant 2m\mathrm{L}^{1/2}\left\{\left(1 + h\mathrm{L}^{1/2}\vartheta_1\left(h\mathrm{L}^{1/2}\right)\right)^m - 1\right\} / \left(h\mathrm{L}^{1/2}\vartheta_1\left(h\mathrm{L}^{1/2}\right)\right)\|p-v\| && (4.131)\\
&\quad \leqslant (m/h)\left\{\left(1 + h\mathrm{L}^{1/2}\vartheta_1\left(h\mathrm{L}^{1/2}\right)\right)^m - 1\right\}\|p-v\| \,, && (4.132)
\end{aligned}$$

as $\vartheta_1\left(h\mathrm{L}^{1/2}\right) \geqslant 2$. □

We can now prove Theorem 24.

Proof. Note that for any $h > 0$, $m \in \mathbb{N}^*$, we have

$$\left(1 + h\mathrm{L}^{1/2}\vartheta_1\left(h\mathrm{L}^{1/2}\right)\right)^m - 1 \leqslant \exp\left\{h\mathrm{L}^{1/2}m\vartheta_1\left(h\mathrm{L}^{1/2}m\right)\right\} - 1 \,, \tag{4.133}$$

as ϑ_1 is non decreasing. The function $c \to \mathrm{e}^{c\vartheta_1(c)}$ is continuous and strictly increasing, from 0 to ∞ on $\mathbb{R}$, thus $\mathrm{e}^{c\vartheta_1(c)} = 2$ admits a unique solution, for $c_0 \approx 0.29$. For $c < c_0$, we have $\mathrm{e}^{c\vartheta_1(c)} < 2$. In particular, if $h < h_0(\mathrm{L},k) = c_0/\mathrm{L}^{1/2}m$, then

$$\left\{\left(1 + h\mathrm{L}^{1/2}\vartheta_1\left(h\mathrm{L}^{1/2}k\right)\right)^m - 1\right\} < 1 \,. \tag{4.134}$$

We first prove that, for all $x_1 \in \mathbb{R}^d$,

$$\text{the function } p \mapsto p + h/m\Theta_m(x_1,p) \text{ is one-to-one.} \tag{4.135}$$

By Lemma 27, there exists $0 < \kappa < 1$ such that for all $p, v \in \mathbb{R}^d$,

$$\|H_{y_1}(p) - H_{y_1}(v)\| \leqslant \frac{h}{m}\|\Theta_m(x_1,p) - \Theta_m(x_1,v)\| \leqslant \kappa\|p-v\| \,,$$

where $H_{y_1} : p \mapsto y_1 - h/m\Theta_m(x_1,p)$. Hence, by the Banach fixed point theorem, for any $y_1 \in \mathbb{R}^d$, H_{y_1} has a unique fixed point p_1 and

$$y_1 = p_1 + \frac{h}{m}\Theta_m(x_1,p_1)$$

showing (4.135). Hence

$$p \mapsto G_{x_1}(p) = x_1 + mhp + h^2\Theta_m(x_1,p)$$

is one-to-one. Since in addition Θ_m is continuously differentiable and the Jacobian of G_{x_1} is invertible, the function G_x is a C^1 diffeomorphism. □

Proof of (4.26)

The result (4.26) is directly linked to Theorem 28 which ensures convergence of the Markov kernel based on NICE proposals.

Theorem 28. *Assume* ***NICE****1 and* ***NICE****2. Then, the Markov kernel* K *defined in* (4.101) *is a* Π*-reversible MH kernel with transition density*

$$q(x,y) = \varphi\big(G_x^{-1}(y)\big)\mathrm{J}_{G_x^{-1}}(y)\ , \tag{4.136}$$

and acceptance probability

$$\alpha(x,y) = \mathsf{a}\left(\frac{\Pi(y)q(y,x)}{\Pi(x)q(x,y)}\right)\ . \tag{4.137}$$

In addition, Theorem 1 applies.

Proof. Note that for all $(x,p) \in \mathbb{R}^{2d}$,

$$\Phi^{-1} \circ \Phi(x,p) = \Phi^{-1}\big(G_x(p), H_x(p)\big) = (x,p), \tag{4.138}$$

where we have used $G_x(p) = \mathrm{proj}_1 \circ \Phi(x,p)$ and $H_x(p) = \mathrm{proj}_2 \circ \Phi(x,p)$. Under **NICE**2, for any $x \in \mathbb{R}^d$, $p \mapsto G_x(p)$ is a diffeomorphism. Then, plugging $y = G_x(p)$, $p = G_x^{-1}(y)$ in (4.138), we obtain

$$\Phi^{-1}\big(y, H_x \circ G_x^{-1}(y)\big) = \big(x, G_x^{-1}(y)\big)\ . \tag{4.139}$$

Under **NICE**1, $s \circ \Phi \circ s = \Phi^{-1}$. Then, we get

$$\Phi\big(y, -H_x \circ G_x^{-1}(y)\big) = \big(x, -G_x^{-1}(y)\big)\ . \tag{4.140}$$

Hence $G_y\big(-H_x \circ G_x^{-1}(y)\big) = x$ or equivalently, $-H_x \circ G_x^{-1}(y) = G_y^{-1}(x)$. Since φ is even, this implies

$$\varphi\big(H_x \circ G_x^{-1}(y)\big) = \varphi\big(G_y^{-1}(x)\big)\ . \tag{4.141}$$

Recall that $\mathrm{J}_\Phi(x,p) = 1$, for all $(x,p) \in \mathbb{R}^{2d}$. Using again $-H_x \circ G_x^{-1}(y) = G_y^{-1}(x)$ in (4.141), we get

$$\Phi\big(y, G_y^{-1}(x)\big) = \big(x, -G_x^{-1}(y)\big)\ . \tag{4.142}$$

Using the chain rule for Jacobian matrices, we get

$$\mathrm{J}_{G_y^{-1}}(x) = \mathrm{J}_{G_x^{-1}}(y). \tag{4.143}$$

Combining (4.141) and (4.143) leads to (4.26) by noting that

$$\frac{q(y,x)}{q(x,y)} = \frac{\varphi\big(H_x \circ G_x^{-1}(y)\big)}{\varphi\big(G_x^{-1}(y)\big)}\ . \tag{4.144}$$

Hence, the acceptance ratio α coincides with the standard MH ratio and the marginal Markov kernel K is thus π_0-reversible. We also note that $q(x,y)$ satisfies the conditions of Theorem 6 given the assumptions on φ and G_x. Moreover, K is Π-irreducible, by Theorem 7. Then, Theorem 1 applies. □

Implementation details

Algorithm 1 presents the methodology for sampling according to the kernel K (4.22), which is Π-reversible.

Algorithm 1 NICE with full refreshment at each iteration

Input: Transformation Φ and momentum-flip involution s, acceptance function a, unnormalized target density π, density φ of momentum p, initial point x_0, number of steps N

for $i = 0$ **to** $N - 1$ **do**

Draw $q_i \sim \varphi$;

Compute proposal $(y_{i+1}, q_{i+1}) = \Phi(x_i, q_i)$;

Draw $B_i \sim \mathrm{Ber}(a_i)$ where

$$a_i = \mathsf{a}\left(\frac{\Pi(y_{i+1})\varphi(q_{i+1})}{\Pi(x_i)\varphi(q_i)}\right) ;$$

if $B_i \equiv 1$ **then**

Set $x_{i+1} = y_{i+1}$;

else

Set $x_{i+1} = x_i$;

end if

end for

Return $(x_{0:N})$

In order to recover persistency, as discussed in Section 4.7.1, we consider the mixture of kernels T (4.102); see Algorithm 2.

Algorithm 2 NICE with randomized full refreshment

Input: Transformation Φ and momentum-flip involution s, acceptance function a, unnormalized target π, density φ of momentum p, probability of refreshment ω, initial point x_0 and initial momentum p_0, number of steps N;
for $i = 0$ **to** $N - 1$ **do**
  Draw $R_i \sim \mathrm{Ber}(\omega)$;
  if $R_i \equiv 0$ **then**
    Compute proposal $(y_{i+1}, q_{i+1}) = \Phi(x_i, p_i)$; *### No refreshment, deterministic dynamics*
    Draw $B_i \sim \mathrm{Ber}(a_i)$ where

$$a_i = \mathsf{a}\left(\frac{\Pi(y_{i+1})\varphi(q_{i+1})}{\Pi(x_i)\varphi(q_i)}\right) ;$$

    if $B_i \equiv 1$ **then**
      Set $(x_{i+1}, p_{i+1}) = (y_{i+1}, q_{i+1})$; *### accept the move and keep the momentum*
    else
      Set $(x_{i+1}, p_{i+1}) = s(x_i, p_i)$; *### reject the move and flip the momentum*
    end if
  else
    Sample $q_i \sim \varphi$; *### Full refreshment of the momentum to update the position*
    Compute proposal $(y_{i+1}, q_{i+1}) = \Phi(x_i, q_i)$;
    Draw $B_i \sim \mathrm{Ber}(a_i)$ where

$$a_i = \mathsf{a}\left(\frac{\Pi(y_{i+1})\varphi(q_{i+1})}{\Pi(x_i)\varphi(q_i)}\right) ;$$

    Draw $p_{i+1} \sim \varphi$;
    if $B_i \equiv 1$ **then**
      Set $x_{i+1} = y_{i+1}$;
    else
      Set $x_{i+1} = x_i$;
    end if
  end if
end for
Return $(x_{0:N})$

4.7.2 Proof of (4.28)

By (4.150), we get

$$q\big((x,v),(y,w)\big) = \{\rho \mathbb{1}_v(w) + (1-\rho)\mathbb{1}_{-v}(w)\} q_w(x,y) \;, \tag{4.145}$$
$$q\big(s(y,w),s(x,v)\big) = q\big((y,-w),(x,-v)\big) \tag{4.146}$$
$$= \{\rho \mathbb{1}_{-w}(-v) + (1-\rho)\mathbb{1}_w(-v)\} q_{-w}(y,x) \tag{4.147}$$
$$= \{\rho \mathbb{1}_v(w) + (1-\rho)\mathbb{1}_{-v}(w)\} q_{-w}(y,x) \;, \tag{4.148}$$

which implies that

$$\frac{q\big(s(y,w),s(x,v)\big)}{q\big((x,v),(y,w)\big)} = \frac{q_{-w}(y,x)}{q_w(x,y)} \;. \tag{4.149}$$

The proof follows from (4.11).

Algorithm 3 NICE with persistence

Input: Transformation Φ and momentum-flip involution s, acceptance function a, unnormalized target π, density φ of momentum p, hyperparameter β, initial point x_0 and initial momentum p_0, number of steps N

for $i = 0$ **to** $N-1$ **do**

Draw $u_i \sim \varphi$ and set $q_i = \beta p_i + \sqrt{1-\beta^2}u_i$;

Compute proposal $(y_{i+1}, q_{i+1}) = \Phi(x_i, q_i)$;

Draw $B_i \sim \mathrm{Ber}(a_i)$ where

$$a_i = \mathsf{a}\left(\frac{\Pi(y_{i+1})\varphi(q_{i+1})}{\Pi(x_i)\varphi(q_i)}\right) ;$$

if $B_i \equiv 1$ **then**

Set $(x_{i+1}, p_{i+1}) = (y_{i+1}, q_{i+1})$; *### accept the move and keep the momentum*

else

Set $(x_{i+1}, p_{i+1}) = s(x_i, q_i)$; *###reject the move and flip the momentum*

end if

end for

Return $(x_{0:N})$

4.7.3 Implementation details of Example 9

We define here a probability of refresh ω. At each iteration, we refresh the direction with probability ω, in which case we draw $v \sim \mathcal{U}\{-1, 1\}$. With this definition, we can reinterpret the parameter ρ (4.150)

$$q\big((x,v),(y,w)\big) = \big\{\rho \mathbb{1}_v(w) + (1-\rho)\mathbb{1}_{-v}(w)\big\} q_w(x,y) \,, \tag{4.150}$$

as $\omega = 2\rho$. In particular, we can write the lifted algorithm with randomized direction refresh in Algorithm 4.

4.7.4 Proof of Lemma 10

From (4.28) and (4.29), we get

$$\alpha\big((x,v),(y,w)\big) = \mathsf{a}\left(\frac{q_{-w}(y,x)\Pi(y)}{q_w(x,y)\Pi(x)}\right) \tag{4.151}$$

$$= \mathsf{a}\left(\frac{\Pi(y)\varphi\big(\tilde{G}^{-1}_{-w,y}(x)\big)\mathrm{J}_{\tilde{G}^{-1}_{-w,y}}(x)}{\Pi(x)\varphi\big(\tilde{G}^{-1}_{w,x}(y)\big)\mathrm{J}_{\tilde{G}^{-1}_{w,x}}(y)}\right) = \mathsf{a}\left(\frac{\mu\big(y, \tilde{G}^{-1}_{-w,y}(x)\big)\mathrm{J}_{\tilde{G}^{-1}_{-w,y}}(x)}{\mu\big(x, \tilde{G}^{-1}_{w,x}(y)\big)\mathrm{J}_{\tilde{G}^{-1}_{w,x}}(y)}\right) . \tag{4.152}$$

Set $\tilde{H}_{w,x}(p) = \mathrm{proj}_2 \circ \Psi^w(x,p)$. Note that

$$\Psi^w(x,p) = \big(\tilde{G}_{w,x}(p), \tilde{H}_{w,x}(p)\big) .$$

Hence, we obtain

$$\Psi^{-w}\big(y, \tilde{G}^{-1}_{-w,y}(x)\big) = \big(\tilde{G}_{-w,y} \circ \tilde{G}^{-1}_{-w,y}(x), \tilde{H}_{-w,y} \circ \tilde{G}^{-1}_{-w,y}(x)\big) = \big(x, \tilde{H}_{-w,y} \circ \tilde{G}^{-1}_{-w,y}(x)\big) ,$$

which implies,

$$\big(y, \tilde{G}^{-1}_{-w,y}(x)\big) = \Psi^w\big(x, \tilde{H}_{-w,y} \circ \tilde{G}^{-1}_{-w,y}(x)\big) , \tag{4.153}$$

$$= \big(\tilde{G}_{w,x} \circ \tilde{H}_{-w,y} \circ \tilde{G}^{-1}_{-w,y}(x), \tilde{H}_{w,x} \circ \tilde{H}_{-w,y} \circ \tilde{G}^{-1}_{-w,y}(x)\big) . \tag{4.154}$$

This identity in particular shows that $y = \tilde{G}_{w,x} \circ \tilde{H}_{-w,y} \circ \tilde{G}^{-1}_{-w,y}(x)$ or equivalently $\tilde{G}^{-1}_{w,x}(y) = \tilde{H}_{-w,y} \circ \tilde{G}^{-1}_{-w,y}(x)$, which used in (4.153) establishes

$$\big(y, \tilde{G}^{-1}_{-w,y}(x)\big) = \Psi^w\big(x, \tilde{G}^{-1}_{w,x}(y)\big) . \tag{4.155}$$

Algorithm 4 Lifted Markov sampling

Input: Transformations $G_{1,x}, G_{-1,x}$, acceptance function a, unnormalized target π, density φ of momentum p, initial point x_0 and initial direction v_0, probability of refreshment ω, number of steps N

for $i = 0$ **to** $N-1$ **do**
  Draw $R_i \sim \mathrm{Ber}(\omega)$;
  if $R_i \equiv 1$ **then**
    Refresh direction $w_i \sim \mathcal{U}\{-1, 1\}$;
  else
    Keep direction $w_i = v_i$;
  end if
  Draw $q_i \sim \varphi$;
  Compute proposal $y_{i+1} = G_{w_i,x_i}(q_i)$;
  Draw $B_i \sim \mathrm{Ber}(a_i)$ where

$$a_i = \mathsf{a}\left(\frac{\Pi(y_{i+1})\varphi\big(G^{-1}_{-w_i,y_{i+1}}(x_i)\big)\mathrm{J}_{G^{-1}_{-w_i,y_{i+1}}}(x_i)}{\Pi(x_i)\varphi\big(G^{-1}_{w_i,x_i}(y_{i+1})\big)\mathrm{J}_{G^{-1}_{w_i,x_i}}(y_{i+1})}\right) ;$$

  if $B_i \equiv 1$ **then**
    Set $(x_{i+1}, v_{i+1}) = (y_{i+1}, w_i)$; *### accept the move and keep the direction*
  else
    Set $(x_{i+1}, v_{i+1}) = (x_i, -w_i)$; *###reject the move and flip the direction*
  end if
end for
Return $(x_{0:N})$

Set $A_w(x,y) = \big(y, \tilde{G}^{-1}_{-w,y}(x)\big)$ and $B_w(x,y) = \big(x, \tilde{G}^{-1}_{w,x}(y)\big)$. Note that $\mathrm{J}_{A_w}(x,y) = \mathrm{J}_{\tilde{G}^{-1}_{-w,y}}(x)$, $\mathrm{J}_{B_w}(x,y) = \mathrm{J}_{\tilde{G}^{-1}_{w,x}}(y)$ and by (4.155) and the chain rule

$$\mathrm{J}_{A_w}(x,y) = \mathrm{J}_{\Psi^w}\big(B_w(x,y)\big)\mathrm{J}_{B_w}(x,y) ,$$

which implies

$$\mathrm{J}_{\Psi^w}\big(x, \tilde{G}^{-1}_{w,x}(y)\big) = \frac{\mathrm{J}_{\tilde{G}^{-1}_{-w,y}}(x)}{\mathrm{J}_{\tilde{G}^{-1}_{w,x}}(y)}. \tag{4.156}$$

The proof of Lemma 10 is concluded by plugging (4.155) and (4.156) into (4.152).

4.7.5 Lifted acceptance probability with deterministic proposals

In this case $\Phi(x,p,v) = (\Psi^v(x,p), v)$, $(x,p) \in \mathbb{R}^{2d}$, $s \in \mathsf{V}$. Clearly, $\Phi^{-1}(x,p,v) = (\Psi^{-v}(x,p), v)$ and it is easily checked that $\Phi^{-1} = s \circ \Phi \circ s$. Denote

$$\pi\big(\mathrm{d}(x,p,v)\big) = \pi_0(x)\varphi(p)\mathrm{d}x\mathrm{d}p\rho(\mathrm{d}v) = \mu(x,p)\mathrm{d}x\mathrm{d}p\rho(\mathrm{d}v) . \tag{4.157}$$

To compute the acceptance probability (4.14), we need to evaluate the density $k(z) = \mathrm{d}\mu/\mathrm{d}\lambda(z)$, where

$$\lambda = \pi + \Phi^{-1}_{\#}\pi. \tag{4.158}$$

Let $f\colon \mathbb{R}^d \times \mathbb{R}^d \times \mathsf{V} \to \mathbb{R}_+$ be a measurable function. We get

$$\lambda(f) = \int f(x,p,v)\mu(x,p)\mathrm{d}x\mathrm{d}p\rho(\mathrm{d}v) + \int f(\Psi^{-v}(x,p),v)\mu(x,p)\mathrm{d}x\mathrm{d}p\rho(\mathrm{d}v) \tag{4.159}$$

$$= \int f(x,p,v)\mu(x,p)\mathrm{d}x\mathrm{d}p\rho(\mathrm{d}v) + \int f(x',p',v)\mu\big(\Psi^v(x',p')\big)\mathrm{J}_{\Psi^v}(x',p')\mathrm{d}x'\mathrm{d}p'\rho(\mathrm{d}v) . \tag{4.160}$$

Therefore, we get

$$\frac{\mathrm{d}\lambda}{\mathrm{dLeb}_{2d}\otimes\rho}(x,p,v) = \mu(x,p) + \mu\big(\Psi^v(x,p)\big)\mathrm{J}_{\Psi^v}(x,p)\ . \tag{4.161}$$

This implies that, for all $(x,v)\in\mathbb{R}^d\times\mathsf{V}$,

$$k(x,p,v) = \frac{\mu(x,p)}{\mu(x,p)+\mu\big(\Psi^v(x,p)\big)\mathrm{J}_{\Psi^v}(x,p)}\ . \tag{4.162}$$

Since $s\circ\Phi(x,p,v) = (\Psi^v(x,p),-v)$, we obtain

$$k\big(s\circ\Phi(x,p,v)\big) = \frac{\mu\big(\Psi^v(x,p)\big)}{\mu\big(\Psi^v(x,p)\big)+\mu(x,p)\mathrm{J}_{\Psi^{-v}}\big(\Psi^v(x,p)\big)} \tag{4.163}$$

$$= \frac{\mu\big(\Psi^v(x,p)\big)\mathrm{J}_{\Psi^v}(x,p)}{\mu(x,p)+\mu\big(\Psi^v(x,p)\big)\mathrm{J}_{\Psi^v}(x,p)}\ , \tag{4.164}$$

where we have used $\mathrm{J}_{\Psi^v}(x,p) = 1/\mathrm{J}_{\Psi^{-v}}\big(\Psi^v(x,p)\big)$. Therefore, the acceptance probability is given by

$$\alpha\big((x,v),(y,w)\big) = \begin{cases} \mathsf{a}\left(\frac{\mu\left(\Psi^v(x,p)\right)\mathrm{J}_{\Psi^v}(x,p)}{\mu(x,p)}\right)\ , \text{ if } \mu(x,p)>0, (y,w)=(\Psi^v(x,p),v), \\ 1,\ \text{if } \mu(x,p)=0 \text{ or } (y,w)\mathrm{ne}(\Psi^v(x,p),v)\ . \end{cases} \tag{4.165}$$

4.7.6 L2HMC Algorithms

In this section, we discuss the sampling algorithms associated to the L2HMC kernel, Example 11. Again, we first describe a version of this algorithm in which the momentum is fully refreshed at each iteration. This is a lifted version of the original L2HMC algorithm [LHS17a], because we keep the direction variable at each iteration instead of refreshing the direction at each iteration. Consider the following assumption.

L2HMC1. *For all $v,x\in\{-1,+1\}\times\mathbb{R}^d$,*

$$G_{v,x}\colon p\mapsto \mathrm{proj}_1\circ\Psi^v(x,p) \tag{4.166}$$

is a C^1-diffeomorphism.

As in the NICE case, establishing **L2HMC**1 requires conditions on the mapping Ψ defining the L2MHC transitions, which is subject to a future work. Under **L2HMC**1, Lemma 10 shows that the lifted L2HMC with full momentum refresh satisfies the assumption of Proposition 8. We may therefore apply Theorem 6 to show convergence of the algorithm in the sense of Theorem 1.

As said above, the original L2HMC algorithm (Algorithm 5) refreshes at each iteration both the direction and the momentum, whereas the lifted algorithm keeps the direction and refreshes only the momentum. Similar to the NICE case, define the marginal kernel, acting on the position only:

$$K(x,\mathrm{d}y) = K_\alpha(x,\mathrm{d}y) + \big(1-\bar\alpha(x)\big)\delta_x(\mathrm{d}y)\ ,$$

where $\bar\alpha(x) = K_\alpha(x,\mathbb{R}^d)$ and for a measurable nonnegative function f,

$$K_\alpha f(x) = \iint f\big(\mathrm{proj}_1\circ\Psi^v(x,p)\big)\bar\alpha(x,p,v)Q\big((x,p,v),\mathrm{d}(y,q,w)\big)\varphi(p)\mathrm{d}p\rho(\mathrm{d}v)\ ,$$

where

$$Q\big((x,p,v),\mathrm{d}(y,q,w)\big) = \delta_{\Psi^v(x,p)}\big(\mathrm{d}(y,q)\big)\delta_v(\mathrm{d}w)\ . \tag{4.167}$$

and

$$\bar\alpha(x,p,v) = \mathsf{a}\left(\mu\big(\Psi^v(x,p)\big)/\mu(x,p)\ \mathrm{J}_{\Psi^v}(x,p)\right)\ , \tag{4.168}$$

Algorithm 5 Original L2HMC

Input: Transformation Ψ, acceptance function a, unnormalized target π, density φ of momentum p, initial point x_0, number of steps N

for $i = 0$ **to** $N - 1$ **do**

Refresh momentum $q_i \sim \varphi$ and direction $v_i \sim \mathcal{U}\{-1, 1\}$;

Compute proposal $(y_{i+1}, q_{i+1}) = \Psi^{v_i}(x_i, q_i)$;

Draw $B_i \sim \mathrm{Ber}(a_i)$ where

$$a_i = \mathsf{a}\left(\frac{\pi(y_{i+1})\varphi(q_{i+1})}{\pi(x_i)\varphi(q_i)}\mathrm{J}_{\Psi^{v_i}}(x_i, q_i)\right) ;$$

if $B_i \equiv 1$ **then**

Set $x_{i+1} = y_{i+1}$;

else

Set $x_{i+1} = x_i$;

end if

end for

Return $(x_{0:N})$

Algorithm 6 Lifted L2HMC with full momentum refreshment

Input: Transformation Ψ, acceptance function a, unnormalized target π, density φ of momentum p, probability of direction refreshment ω, initial point x_0 and initial direction v_0, number of steps N

for $i = 0$ **to** $N - 1$ **do**

Draw $R_i \sim \mathrm{Ber}(\omega)$;

if $R_i \equiv 1$ **then**

Refresh direction $w_i \sim \mathcal{U}\{-1, 1\}$;

else

Keep direction $w_i = v_i$;

end if

Refresh momentum $q_i \sim \varphi$;

Compute proposal $(y_{i+1}, q_{i+1}) = \Psi^{w_i}(x_i, q_i)$;

Draw $B_i \sim \mathrm{Ber}(a_i)$ where

$$a_i = \mathsf{a}\left(\frac{\pi(y_{i+1})\varphi(q_{i+1})}{\pi(x_i)\varphi(q_i)}\mathrm{J}_{\Psi^{w_i}}(x_i, q_i)\right) ;$$

if $B_i \equiv 1$ **then**

Set $(x_{i+1}, v_{i+1}) = (y_{i+1}, w_i)$; *### accept the move and keep the direction*

else

Set $(x_{i+1}, v_{i+1}) = (x_i, -w_i)$; *###reject the move and flip the direction*

end if

end for

Return $(x_{0:N})$

Recall that the Markov kernel $Q_\alpha\big(x, p, v; \mathrm{d}(y, q, w)\big) = \bar{\alpha}(x, p, v)Q\big((x, p, v), \mathrm{d}(y, q, w)\big)$ is (π, S)-reversible. Let f and g be two positive measurable functions on $\mathbb{R}^d$. Since Q_α is (π, S)-reversible and ρ is symmetric,

we get

$$\int \Pi(\mathrm{d}x) K_\alpha f(x) g(x) = \int \pi\big(\mathrm{d}(x,p,v)\big) Q_\alpha\big((x,p,v);\mathrm{d}(y,q,w)\big) f(y) g(x) \tag{4.169}$$

$$= \int \pi\big(\mathrm{d}(y,q,w)\big) S Q_\alpha S g(y) f(y) \tag{4.170}$$

$$=^{(3)} \int \pi\big(\mathrm{d}(y,q,w)\big) \bar{\alpha}(y,q,-w) Q\big((y,q,-w);\mathrm{d}(x,p,v)\big) g(x) f(y) \tag{4.171}$$

$$=^{(4)} \int \pi\big(\mathrm{d}(y,q,w)\big) \bar{\alpha}(y,q,w) Q\big((y,q,w);\mathrm{d}(x,p,v)\big) g(x) f(y) \tag{4.172}$$

$$=^{(5)} \int \Pi(\mathrm{d}x) K_\alpha g(y) f(y) \tag{4.173}$$

where we have used in (3) that $S Q_\alpha S g(y,q,w) = \bar{\alpha}(y,q,-w) \int Q\big((y,q,-w);\mathrm{d}(x,p,v)\big) g(x)$, in (4) the symmetry of ρ and finally in (5) the definition of K_α. Hence the L2HMC kernel is Π-reversible. Moreover, note that under **L2HMC**1, we obtain

$$K_\alpha f(x) = \int \left\{ \int \bar{\alpha}(x, G_{v,x}^{-1}(y), v) \varphi\big(G_{v,x}^{-1}(y)\big) \mathrm{J}_{G_{v,x}^{-1}}(y) \rho(\mathrm{d}v) \right\} f(y) \mathrm{d}y \; . \tag{4.174}$$

Denoting by $q_v(x,y)$ the transition density $q_v(x,y) = \varphi\big(G_{v,x}^{-1}(y)\big) \mathrm{J}_{G_{v,x}^{-1}}(y)$ and then setting $q(x,y) = \int q_v(x,y)\rho(\mathrm{d}v)$, we finally get

$$K_\alpha f(x) = \int \alpha(x,y) q(x,y) f(y) \mathrm{d}y \tag{4.175}$$

with

$$\alpha(x,y) = \frac{\int \bar{\alpha}(x, G_{v,x}^{-1}(y), v) \varphi\big(G_{v,x}^{-1}(y)\big) \mathrm{J}_{G_{v,x}^{-1}}(y) \rho(\mathrm{d}v)}{\int \varphi\big(G_{v,x}^{-1}(y)\big) \mathrm{J}_{G_{v,x}^{-1}}(y) \rho(\mathrm{d}v)} \tag{4.176}$$

Write now, following [Tie94],

$$r(x,y) = \frac{\Pi(x) q(x,y)}{\Pi(y) q(y,x)} \; . \tag{4.177}$$

Moreover, by Lemma 10, we can write

$$\bar{\alpha}(x, G_{v,x}^{-1}(y), v) = \mathsf{a}\left(\frac{\Pi(y) q_{-v}(y,x)}{\Pi(x) q_v(x,y)} \right) \; . \tag{4.178}$$

In that case, we have

$$\alpha(x,y)\Pi(x)q(x,y) = \int \Pi(x) q_v(x,y) \mathsf{a}\left(\frac{\Pi(y) q_{-v}(y,x)}{\Pi(x) q_v(x,y)} \right) \rho(\mathrm{d}v) \tag{4.179}$$

$$= \int \Pi(y) q_{-v}(y,x) \mathsf{a}\left(\frac{\Pi(x) q_v(x,y)}{\Pi(y) q_{-v}(y,x)} \right) \rho(\mathrm{d}v) \tag{4.180}$$

$$= \int \Pi(y) q_w(y,x) \mathsf{a}\left(\frac{\Pi(x) q_{-w}(x,y)}{\Pi(y) q_w(y,x)} \right) \rho(\mathrm{d}w) \; , \tag{4.181}$$

where we have use the fact that $t\mathsf{a}(1/t) = \mathsf{a}(t)$ and the change of variable $w = -v$. Then,

$$\alpha(x,y)\Pi(x)q(x,y) = \alpha(y,x)\Pi(y)q(y,x) \; . \tag{4.182}$$

We thus have $\alpha(x,y) r(x,y) = \alpha(y,x)$, which proves by [Tie94], Theorem 2, that the ratio α is exactly the classical MH ratio satisfying the detailed balance condition.

To retrieve persistency, we can use as for the NICE algorithm a mixture of a deterministic L2HMC move and a full independent refreshment of the position, momentum and the direction; see Algorithm 7.

Algorithm 7 Lifted L2HMC with randomized full refreshment

Input: Transformation Ψ, acceptance function a, unnormalized target π, density φ of momentum p, probability of refreshment ω, initial point x_0, initial momentum p_0 and initial direction v_0, number of steps N

for $i = 0$ **to** $N - 1$ **do**

Draw $R_i \sim \mathrm{Ber}(\omega)$;

if $R_i \equiv 0$ **then**

Compute proposal $(y_{i+1}, q_{i+1}) = \Psi^{v_i}(x_i, p_i)$; *###No refreshment, deterministic dynamics*

Draw $B_i \sim \mathrm{Ber}(a_i)$ where

$$a_i = \mathsf{a}\left(\frac{\pi(y_{i+1})\varphi(q_{i+1})}{\pi(x_i)\varphi(p_i)} \mathrm{J}_{\Psi^{v_i}}(x_i, p_i)\right) ;$$

if $B_i \equiv 1$ **then**

Set $(x_{i+1}, p_{i+1}, v_{i+1}) = (y_{i+1}, q_{i+1}, v_i)$; *### accept the move and keep the direction*

else

Set $(x_{i+1}, p_{i+1}, v_{i+1}) = (x_i, p_i, -v_i)$; *###reject the move and flip the direction*

end if

else

Draw $q_i \sim \varphi$, $w_i \sim \mathcal{U}\{-1, 1\}$; *### refresh independently the momentum and the direction*

Compute proposal $(y_{i+1}, q_{i+1}) = \Psi^{w_i}(x_i, q_i)$;

Draw $B_i \sim \mathrm{Ber}(a_i)$ where

$$a_i = \mathsf{a}\left(\frac{\pi(y_{i+1})\varphi(q_{i+1})}{\pi(x_i)\varphi(q_i)} \mathrm{J}_{\Psi^{w_i}}(x_i, q_i)\right) ;$$

Draw $p_{i+1} \sim \varphi$, $v_{i+1} \sim \mathcal{U}\{-1, 1\}$; *### refresh the momentum and the direction*

if $B_i \equiv 1$ **then**

Set $x_{i+1} = y_{i+1}$;

else

Set $x_{i+1} = x_i$;

end if

end if

end for

Return $(x_{0:N})$

Much like the persistent HMC algorithm, we may also design a lifted persistent HMC algorithm in which, at each iteration, we keep the direction and partially refresh the momentum using an autoregressive scheme; see Algorithm 8.

Algorithm 8 Lifted L2HMC with persistence

Input: Transformation Ψ, acceptance function a, unnormalized target π, density φ of momentum p, hyperparameter β, initial point x_0, initial momentum p_0 and initial direction v_0, number of steps N
for $i = 0$ **to** $N - 1$ **do**
  Sample $u_i \sim \varphi$ and refresh momentum $q_i = \beta p_i + \sqrt{1-\beta^2} u_i$; *### partially update the momentum*
  Compute proposal $(y_{i+1}, q_{i+1}) = \Psi^{v_i}(x_i, q_i)$;
  Draw $B_i \sim \mathrm{Ber}(a_i)$ where

$$a_i = \mathsf{a}\left(\frac{\pi(y_{i+1})\varphi(q_{i+1})}{\pi(x_i)\varphi(q_i)} \mathrm{J}_{\Psi^{v_i}}(x_i, q_i)\right) ;$$

  if $B_i \equiv 1$ **then**
    Set $(x_{i+1}, p_{i+1}, v_{i+1}) = (y_{i+1}, q_{i+1}, v_i)$; *### accept the move and keep the direction*
  else
    Set $(x_{i+1}, p_{i+1}, v_{i+1}) = (x_i, p_i, -v_i)$; *### reject the move and flip the direction*
  end if
end for
Return $(x_{0:N})$

4.8 Experiments

In this section, we aim to compare the developed irreversible algorithms with their reversible counterparts. We mainly focus on the proposal mappings based on the Normalizing Flows [Pap+19] but also consider more classical MH algorithms such as MALA.

The proposal mappings based on the Normalizing flows are heavily parametrized and these parameters need to be chosen. In this work, we follow [LHS17a] and choose the parameter values, that optimize the Expected Squared Jump distance with the reciprocal term. Importantly, in all the experiments below we report the results for already learned proposals, i.e. the sampling is done *after* the learning with all the parameters being fixed. Moreover, our goal is not to introduce a criterion for choosing the parameters of the transitions, but merely to introduce a general framework in which irreversible and normalizing flow based MCMC can be understood as part of a general recipe, with convergence guarantees.

We mainly focus on two algorithms: NICE and L2HMC. For both algorithms we consider the original reversible versions (see Algorithms 1 and 5), the versions with randomized full refreshment (see Algorithms 2 and 7) and, finally, their persistent counterparts (see Algorithms 3 and 8). We apply these methods on mixtures of Gaussians (MoG) in dimension 10 with different variances. The modes of the distribution differ only by the first coordinate, respectively -2 and 2 for the first and the second mode. We also consider a Strongly Correlated Gaussian in dimension 2, introduced in [LHS17a]. Additionally, we compare classical MALA transitions with a conditional Invertible Transforms transition introduced in Section 4.3, see also Algorithm 4.

We report several metrics to show the effectiveness of the (π, S)-reversibility in practice, and demonstrate the importance of the convergence conditions we derive. First, we report Effective Sample Size (ESS). We also display a convergence metric to assess the quality of the different samplers considered. We compute the log-likelihood of the samples produced, and additionally train a Kernel Density Estimator (KDE – here a Gaussian kernel with a bandwidth of 0.1) on the samples received. We can thus compute backward Kullback-Leibler divergence obtained between π the target distribution and q the estimated density by the KDE, $\mathrm{KL}(q\|\pi)$. We see in general the advantage of persistency features compared to fully reversible algorithms, at no expense in computational cost.

First, we display in Figure 4.1 a visualization of samples from L2HMC, a cIT-MALA lifted kernel

Figure 4.1: From left to right: L2HMC, Lifted irreversible kernel with cIT and MALA transitions, MALA algorithm. L2HMC performs high jump with less cover of the modes, while lifted kernels with cIT and MALA transitions cover effectively the mixture. A classical MALA algorithm struggles to mix.

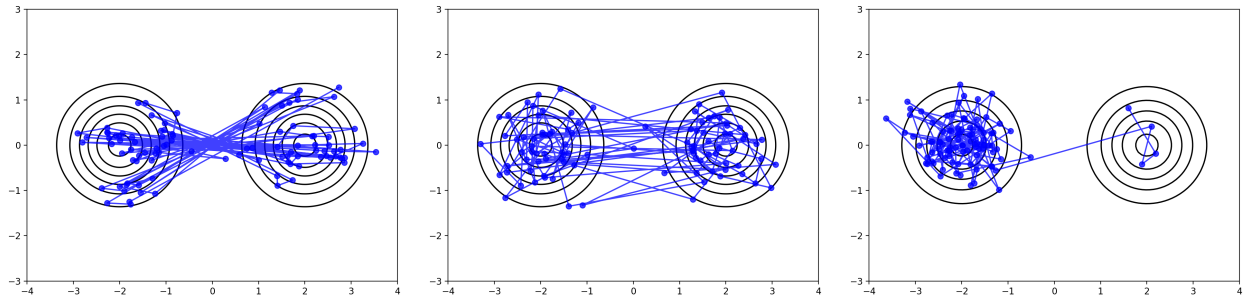


Table 4.1: Mixture of Gaussians with diagonal covariances 0.1 and 0.25 in dimension 10

Proposal	***ESS***	***KL***	***LogLikelihood***
NICE	0.0538819	311.132	-14.7606
NICE with randomized full refreshment	**0.449657**	**195.963**	**-7.19236**
NICE with persistence	0.149203	280.394	-12.0524

and MALA. We express in this figure the benefits of the convergence results of Theorem 1, and the effectiveness of neural enhanced samplers.

Moreover, the results of the experiments for NICE and L2HMC algorithms are summarized in Tables 4.1, 4.2 and 4.3. Without increasing the number of parameters and the expressiveness of the sampler, persistency through partial refreshment or randomized full refreshment comes with an increase in the competitiveness of the sampler, which can be spectacular; see Tables 4.1 and 4.2. Note that NICE-based samplers struggle on the Strongly Correlated Gaussian challenge, which was one of the main improvement of the L2HMC sampler. Finally, we show in Figures 4.2 and 4.3, the autocorrelation plots of the different samplers on the problems considered, as a visual representation of the tables.

Table 4.2: Mixture of Gaussians with diagonal covariances 0.3 and 0.3 in dimension 10

Proposal	***ESS***	***KL***	***LogLikelihood***
NICE	0.061500	345.224242	-23.166510
NICE with randomized full refreshment	**0.139867**	**34.312647**	**-10.130489**
NICE with persistence	0.141844	312.134878	-19.516108

Table 4.3: Strongly Correlated Gaussian in dimension 2

Proposal	***ESS***	***KL***	***LogLikelihood***
NICE	0.003849	20.548244	**-4.03679**
NICE with randomized full refreshment	0.003768	585.054601	-4.80058
NICE with persistence	**0.004165**	**17.614930**	-4.19062
Original L2HMC	0.199865	**0.575168**	**-3.98623**
L2HMC with randomized full refreshment	0.015753	1146.000895	-20.5811
L2HMC with persistence	**0.235942**	101.317806	-7.28438

Figure 4.2: From left to right: Autocorrelation plot for different implementations of NICE, for a Mixture of Gaussian with diagonal variances of respectively 0.1 and 0.25, and 0.3 and 0.3

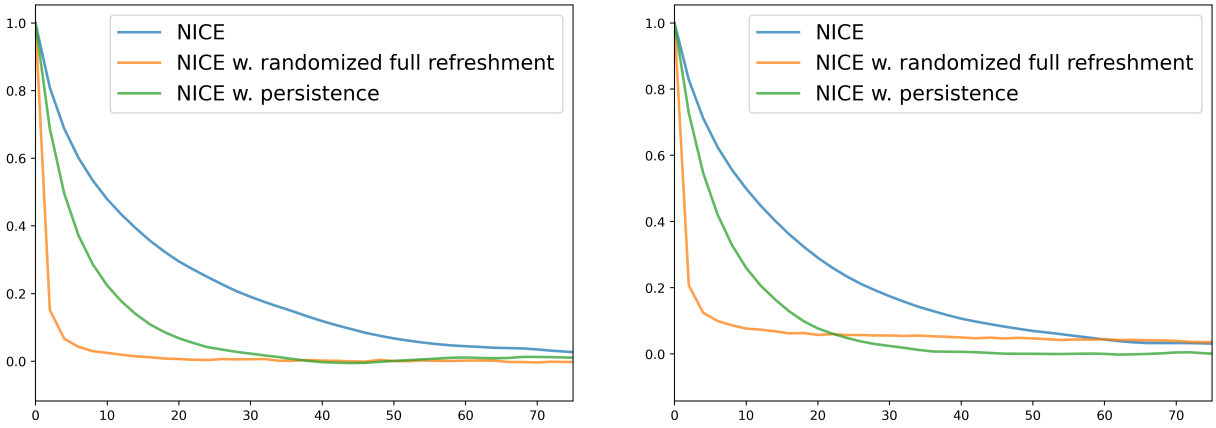
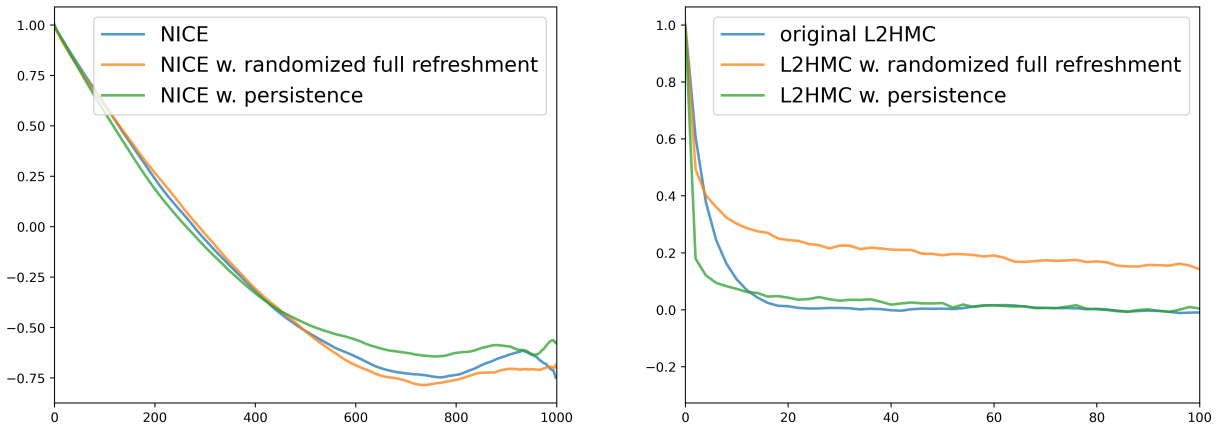


Figure 4.3: From left to right: Autocorrelation plots for different implementations of NICE and L2HMC algorithms on a Strongly Correlated Gaussian.



Chapter 5

NEO: Non Equilibrium Sampling on the Orbit of a Deterministic Transform

Achille Thin[1], Yazid Janati[2], Sylvain Le Corff[2], Charles Ollion[1], Arnaud Doucet[3], Alain Durmus[4], Eric Moulines[1], Christian Robert[5]

Abstract

Sampling from a complex distribution π and approximating its intractable normalizing constant Z are challenging problems. In this paper, a novel family of importance samplers (IS) and Markov chain Monte Carlo (MCMC) samplers is derived. Given an invertible map T, these schemes combine (with weights) elements from the forward and backward Orbits through points sampled from a proposal distribution ρ. The map T does not leave the target π invariant, hence the name NEO, standing for Non-Equilibrium Orbits. NEO-IS provides unbiased estimators of the normalizing constant and self-normalized IS estimators of expectations under π while NEO-MCMC combines multiple NEO-IS estimates of the normalizing constant and an iterated sampling-importance resampling mechanism to sample from π. For T chosen as a discrete-time integrator of a conformal Hamiltonian system, NEO-IS achieves state-of-the art performance on difficult benchmarks and NEO-MCMC is able to explore highly multimodal targets. Additionally, we provide detailed theoretical results for both methods. In particular, we show that NEO-MCMC is uniformly geometrically ergodic and establish explicit mixing time estimates under mild conditions.

5.1 Introduction

Consider a target distribution of the form $\Pi(x) \propto \Lambda(x)\mathrm{L}(x)$ where Λ is a probability density function (pdf) on $\mathbb{R}^d$ and L is a nonnegative function. Typically, in a Bayesian setting, π is a posterior distribution associated with a prior distribution ρ and a likelihood function L. Another situation of interest is generative modeling where π is the distribution implicitly defined by a Generative Adversarial Networks (GAN) discriminator-generator pair where ρ is the distribution of the generator and L is derived from the discriminator [Tur+19a; Che+20b]. In a Variational Auto Encoder (VAE) context [KW14; BGS15], π could be the true posterior distribution, ρ the approximate posterior distribution output by the encoder and L an importance weight between the true posterior and approximate posterior distributions. We are interested in this paper in sampling from π and approximating its intractable normalizing constant

[1]Centre de Mathématiques Appliquées, UMR 7641, Ecole polytechnique, France
[2]Samovar, Télécom SudParis, département CITI, TIPIC, Institut Polytechnique de Paris, Palaiseau
[3]Department of Statistics, University of Oxford
[4]Université Paris-Saclay, ENS Paris-Saclay, CNRS, Centre Borelli, F-91190 Gif-sur-Yvette, France
[5]Ceremade, Université Paris-Dauphine & Department of Statistics, University of Warwick

$\mathrm{Z} = \int \Lambda(x)\mathrm{L}(x)\mathrm{d}x$. These problems arise in many applications in statistics, molecular dynamics or machine learning, and remain challenging.

Many approaches to compute normalizing constants are based on Importance Sampling (IS) - see [Aga+17; AM21] and the references therein - and its variations, among others, Annealed Importance Sampling (AIS) [Nea01a; Wu+16; DF19] and Sequential Monte Carlo (SMC) [DDJ06a]. More recently, Neural IS has also become very popular in machine learning; see e.g. [EM12; Mül+19; Pap+19; Pra19a; Wir+20; WKN20a]. Neural IS is an adaptive IS which relies on an importance function obtained by applying a normalizing flow to a reference distribution. The parameters of this normalizing flow are chosen by minimizing a divergence between the proposal and the target (such as the Kullback–Leibler [Mül+19] or the χ^2-divergence [Aga+17]). Recent work on the subject proposes to add stochastic moves in order to enhance the performance of the normalizing flows [WKN20a].

More recently, the *Non-Equilibrium IS* (NEIS) method has been introduced by [RV19] as an alternative to these approaches. Similar to Neural IS, NEIS consists in transporting samples $\{X^i\}_{i=1}^N$ from a reference distribution using a family of deterministic mappings. For NEIS, this family is chosen to be an homogeneous differential flow $(\phi_t)_{t\in\mathbb{R}}$. In contrast to Neural IS, for any $i \in [N]$, the sample X^i is propagated both forward and backward in time along the orbits associated with $(\phi_t)_{t\in\mathbb{R}}$ until stopping conditions are met. Moreover, the resulting estimator of the normalizing constant is obtained by computing weighted averages of the whole orbit $(\phi_t(X^i))_{t\in[\tau_{+,i},\tau_{-,i}]}$, where $\tau_{+,i}, \tau_{-,i}$ are the resulting stopping times, and not only the endpoints $\phi_{\tau_{+,i}}(X^i), \phi_{\tau_{-,i}}(X^i)$. In [RV19], the authors provide an application of NEIS with $(\phi_t)_{t\in\mathbb{R}}$ associated with a conformal Hamiltonian dynamics, and reports impressive numerical results on difficult normalizing constants estimation problems, in particular for high-dimensional multimodal distributions.

We propose in this work NEO-IS which alleviates the shortcomings of NEIS. Similar to NEIS, samples are drawn from a reference distribution, typically set to Λ, and are propagated under the forward and backward orbits of a *discrete-time* dynamical system associated with an invertible transform T. An estimator of the normalizing constant is obtained by reweighting all the points on the whole orbits using the IS rule. Contrary to NEIS, the NEO-IS estimator of Z is unbiased under assumptions that are mild and easy to verify. It is more flexible than NEIS because it does not rely on the accuracy of the discretization of a continuous-time dynamical system.

We then show how it is possible to leverage the unbiased estimator of Z defined by NEO-IS to obtain NEO-MCMC, a novel massively parallel MCMC algorithm to sample from π. In a nutshell, NEO-MCMC relies on parallel walkers which each estimates the normalizing constant but are allowed to interact through a resampling mechanism.

Our contributions can be summarized as follows.

(i) We present a novel class of IS estimators of the normalizing constant Z referred to as NEO-IS. More broadly, a small modification of this algorithm also allows us to estimate integrals with respect to Π. Both finite sample and asymptotic guarantees are provided for these two methodologies.

(ii) We develop a new massively parallel MCMC method, NEO-MCMC. NEO-MCMC combines NEO-IS unbiased estimator of the normalizing constant with iterated sampling-importance resampling methods. We prove that it is Π-reversible and ergodic under very general conditions. We derive also conditions which imply that NEO-MCMC is uniformly geometrically ergodic (with an explicit expression of the mixing time).

(iii) We illustrate our findings using numerical benchmarks which show that both NEO-IS and NEO-MCMC outperform state-of-the-art (SOTA) methods in difficult settings.

5.2 NEO-IS algorithm

In this section, we derive the NEO-IS algorithm. The two key ingredients for this algorithm are (1) the reference distribution Λ and (2) a transformation T assumed to be a C^1-diffeomorphism with inverse T^{-1}.

Write, for $k \in \mathbb{N}^* = \mathbb{N}\backslash\{0\}$, $\mathrm{T}^k = \mathrm{T} \circ \mathrm{T}^{k-1}$, $\mathrm{T}^0 = \mathrm{Id}_d$ and similarly $\mathrm{T}^{-k} = \mathrm{T}^{-1} \circ \mathrm{T}^{-(k-1)}$. For any $k \in \mathbb{Z}$, denote by $\rho_k : \mathbb{R}^d \to \mathbb{R}_+$ the pushforward of ρ by T^k, defined for $x \in \mathbb{R}^d$ by $\rho_k(x) = \rho(\mathrm{T}^{-k}(x))\mathrm{J}_{\mathrm{T}^{-k}}(x)$, where $\mathrm{J}_\Phi(x) \in \mathbb{R}^+$ is the absolute value of the Jacobian determinant of $\Phi : \mathbb{R}^d \to \mathbb{R}^d$ evaluated at x. In line with multiple importance sampling *à la* Owen and Zhou [OZ00a], we introduce the proposal density

$$\rho_{\mathrm{T}}(x) = \Omega^{-1} \sum\nolimits_{k\in\mathbb{Z}} \varpi_k \rho_k(x) \;, \tag{5.1}$$

where $\{\varpi_k\}_{k\in\mathbb{Z}}$ is a nonnegative sequence and $\Omega = \sum_{k\in\mathbb{Z}} \varpi_k$. Note that we assume in the sequel that the support of the weight sequence defined as $\{k \in \mathbb{Z} : \varpi_k \neq 0\}$ is finite. Thus, the mixture distribution in (5.1) is a **finite mixture**. Given $x \in \mathbb{R}^d$, $\rho_{\mathrm{T}}(x)$ is a function of the forward and backward orbit of T through x.

For any nonnegative function f, the definition of ρ_{T} implies that

$$\int f(y)\rho_{\mathrm{T}}(y)\mathrm{d}y = \Omega^{-1} \int \sum_{k\in\mathbb{Z}} \varpi_k f(\mathrm{T}^k(x))\rho(x)\mathrm{d}x \;.$$

Assuming that $\varpi_0 > 0$, the ratio $\rho(x)/\rho_{\mathrm{T}}(x) \leqslant \varpi_0^{-1}\Omega < \infty$ is bounded. We can therefore apply the IS principle which allows to write the identity

$$\boxed{\int f(x)\rho(x)\mathrm{d}x = \int \left(f(y)\frac{\rho(y)}{\rho_{\mathrm{T}}(y)} \right) \rho_{\mathrm{T}}(y)\mathrm{d}y = \int \sum_{k\in\mathbb{Z}} f(\mathrm{T}^k(x)) w_k(x) \rho(x)\mathrm{d}x \;,} \tag{5.2}$$

where the weights are given by (see Section 5.7.2 for a detailed derivation),

$$w_k(x) = \varpi_k \rho(\mathrm{T}^k(x))/\{\Omega\rho_{\mathrm{T}}(\mathrm{T}^k(x))\} = \varpi_k\rho_{-k}(x) \Big/ \sum\nolimits_{i\in\mathbb{Z}} \varpi_{k+i}\rho_i(x) \quad . \tag{5.3}$$

We assume in the sequel that $\varpi_0 > 0$. In particular, note that under this condition, the weights w_k are also upper bounded uniformly in x: for any $x \in \mathbb{R}^d$, $w_k(x) \leqslant \varpi_k/\varpi_0$. Equations (5.2) and (5.3) suggest to estimate the integral $\int f(x)\rho(x)\mathrm{d}x$ by $I^{\mathrm{NEO}}_{\varpi,N}(f) = N^{-1}\sum_{i=1}^N \sum_{k\in\mathbb{Z}} w_k(X^i) f(\mathrm{T}^k(X^i))$ where $\{X^i\}_{i=1}^N$ are i.i.d. samples from the proposal ρ, which is denoted by $X^{1:N} \overset{\mathrm{iid}}{\sim} \Lambda$.

Algorithm 9 NEO-IS Sampler

1. Sample $X^{1:N} \overset{\mathrm{iid}}{\sim} \rho$ for $i \in [N]$.
2. For $i \in [N]$, compute the path $(\mathrm{T}^j(X^i))_{j\in\mathbb{Z}}$ and weights $(w_j(X^i))_{j\in\mathbb{Z}}$.
3. $I^{\mathrm{NEO}}_{\varpi,N}(f) = N^{-1}\sum_{i=1}^N \sum_{k\in\mathbb{Z}} w_k(X^i) f(\mathrm{T}^k(X^i))$.

This estimator is obtained by a weighted combination of the elements of the independent forward and backward orbits $\{\mathrm{T}^k(X^i)\}_{k\in\mathbb{Z}}$ with $X^{1:N} \overset{\mathrm{iid}}{\sim} \Lambda$. This estimator is referred to as NEO-IS. Choosing $f \equiv \mathrm{L}$ provides the NEO-IS estimator of the normalizing constant of Π:

$$\widehat{\mathrm{Z}}^{\varpi}_{X^i} = \sum\nolimits_{k\in\mathbb{Z}} \mathrm{L}(\mathrm{T}^k(X^i)) w_k(X^i) \;, \quad \widehat{\mathrm{Z}}^{\varpi}_{X^{1:N}} = N^{-1}\sum\nolimits_{i=1}^N \widehat{\mathrm{Z}}^{\varpi}_{X^i} \;. \tag{5.4}$$

We now study the performance of the NEO-IS estimator. The following two quantities play a fundamental role in the analysis:

$$E^{\varpi}_{\mathrm{T}} = \mathbb{E}_{X\sim\rho}\big[\big(\sum\nolimits_{k\in\mathbb{Z}} w_k(X)\mathrm{L}(\mathrm{T}^k(X))/\mathrm{Z}\big)^2\big], \mathrm{L} = \sup\nolimits_{x\in\mathbb{R}^d} \sum\nolimits_{k\in\mathbb{Z}} w_k(x)\mathrm{L}(\mathrm{T}^k(x))/\mathrm{Z} \;. \tag{5.5}$$

Theorem 29. *$\widehat{\mathrm{Z}}^{\varpi}_{X^{1:N}}$ is an unbiased estimator of* Z. *If $E^{\varpi}_{\mathrm{T}} < \infty$, then, $\mathbb{E}[|\widehat{\mathrm{Z}}^{\varpi}_{X^{1:N}}/\mathrm{Z} - 1|^2] = N^{-1}(E^{\varpi}_{\mathrm{T}} - 1)$. If $\mathrm{L} < \infty$, then, for any $\delta \in (0,1)$, with probability $1-\delta$, $\sqrt{N}\left|\widehat{\mathrm{Z}}^{\varpi}_{X^{1:N}}/\mathrm{Z} - 1\right| \leqslant \mathrm{L}\sqrt{\log(2/\delta)/2}$.*

The (elementary) proof is postponed to Section 5.7.3. E_{T}^{ϖ} plays the role of the second-order moment of the importance weights $\mathbb{E}_{X\sim\Lambda}[\mathrm{L}^2(X)]$ which is key to the performance of IS algorithms [Aga+17; AM21]. In addition, since the NEO-IS estimator $\widehat{\mathrm{Z}}^{\varpi}_{X^{1:N}}$ is unbiased, the Cauchy–Schwarz inequality implies that $\mathbb{E}_{X\sim\rho}\big[\big(\sum_{k\in\mathbb{Z}} w_k(X)\mathrm{L}(\mathrm{T}^k(X))\big)^2\big] \geqslant \mathrm{Z}^2$ and hence that $E_{\mathrm{T}}^{\varpi} \geqslant 1$. Note that if $\|\mathrm{L}\|_\infty = \sup_{x\in\mathbb{R}^d} \mathrm{L}(x) < \infty$, then since the weights are uniformly bounded by $\Omega\varpi_0^{-1}$, we have $M_{\mathrm{T}}^{\varpi} \leqslant \|\mathrm{L}\|_\infty \Omega\varpi_0^{-1}/\mathrm{Z}$.

Using the NEO-IS estimate $\widehat{\mathrm{Z}}^{\varpi}_{X^{1:N}}$ of the normalizing constant, we can construct a self-normalized IS estimate of $\int f(x)\Pi(x)\mathrm{d}x$:

$$J^{\mathrm{NEO}}_{\varpi,N}(f) = N^{-1}\sum_{i=1}^{N} \frac{\widehat{\mathrm{Z}}^{\varpi}_{X^i}}{\widehat{\mathrm{Z}}^{\varpi}_{X^{1:N}}} \sum_{k\in\mathbb{Z}} \frac{\mathrm{L}(\mathrm{T}^k(X^i))w_k(X^i)}{\widehat{\mathrm{Z}}^{\varpi}_{X^i}} f(\mathrm{T}^k(X^i)) \,, \tag{5.6}$$

referred to as NEO-SNIS estimator. This expression may seem unnecessarily complicated but highlights the hierarchical structure of the estimator. We combine estimators $(\widehat{\mathrm{Z}}^{\varpi}_{X^i})^{-1}\sum_{k\in\mathbb{Z}}\mathrm{L}(\mathrm{T}^k(X^i))w_k(X^i)f(\mathrm{T}^k(X^i))$ evaluated on the forward and backward orbits through the points $\{X^i\}_{i=1}^N$ using the normalized weights $\{\widehat{\mathrm{Z}}^{\varpi}_{X^i}/\widehat{\mathrm{Z}}^{\varpi}_{X^{1:N}}\}_{i=1}^N$. Although the NEO-IS estimator is unbiased, the NEO-SNIS is in general biased. However, for bounded functions, both the bias and the variance of the NEO-SNIS estimator are $O(N^{-1})$, with constants proportional to E_{T}^{ϖ}. For g a Π-integrable function, we set $\Pi(g) = \int g(x)\Pi(x)\mathrm{d}x$.

Theorem 30. *Assume that $E_{\mathrm{T}}^{\varpi} < \infty$. Then, for any function g satisfying $\sup_{x\in\mathbb{R}^d}|g(x)| \leqslant 1$ on $\mathbb{R}^d$, and $N \in \mathbb{N}$*

$$\mathbb{E}_{X^{1:N}\overset{\mathrm{iid}}{\sim}\rho}\left[|J^{\mathrm{NEO}}_{\varpi,N}(g) - \pi(g)|^2\right] \leqslant 4\cdot N^{-1}E_{\mathrm{T}}^{\varpi} \,, \tag{5.7}$$

$$\left|\mathbb{E}_{X^{1:N}\overset{\mathrm{iid}}{\sim}\rho}\left[J^{\mathrm{NEO}}_{\varpi,N}(g) - \pi(g)\right]\right| \leqslant 2\cdot N^{-1}E_{\mathrm{T}}^{\varpi} \,. \tag{5.8}$$

If $\mathrm{L} < \infty$, then for $\delta \in (0,1]$, with probability at least $1-\delta$,

$$\sqrt{N}|J^{\mathrm{NEO}}_{\varpi,N}(g) - \pi(g)| \leqslant \|g\|_\infty \mathrm{L}\sqrt{32\log(4/\delta)} \,. \tag{5.9}$$

The proof is postponed to Section 5.7.4. These results extend to NEO-SNIS estimators the results known for self-normalized IS estimators; see e.g., [Aga+17; AM21] and the references therein. The upper bounds stated in this result suggest it is good practice to keep $E_{\mathrm{T}}^{\varpi}/N$ small in order to obtain sensible approximations. For two pdfs p and q on $\mathbb{R}^d$, denote by $\mathrm{D}_{\chi^2}(p,q) = \int\{p(x)/q(x)-1\}^2 q(x)\mathrm{d}x$ the χ^2-divergence between p and q.

Lemma 31. *For any nonnegative sequence $(\varpi_k)_{k\in\mathbb{Z}}$, we have $E_{\mathrm{T}}^{\varpi} \leqslant \mathrm{D}_{\chi^2}(\pi\|\rho_{\mathrm{T}}) + 1$.*

The proof is postponed to Section 5.7.5. Lemma 31 suggests that accurate sampling requires N to scale linearly with the χ^2-divergence between the target π and the extended proposal ρ_{T}.

Remark 32. We can extend NEO to non homogeneous flows, replacing the family $\{\mathrm{T}^k : k\in\mathbb{Z}\}$ with a collection of mappings $\{\mathsf{T}_k : k\in\mathbb{Z}\}$. This would allow us to consider further flexible classes of transformations such as normalizing flows; see e.g. [Pap+19]. The χ^2-divergence $\mathrm{D}_{\chi^2}(\pi\|\rho_{\mathrm{T}})$ provides a natural criterion for learning the transformation. We leave this extension to future work.

Conformal Hamiltonian transform The efficiency of NEO relies heavily on the choice of T. Intuitively, a sensible choice of T requires that (i) E_{T}^{ϖ} is small, i.e. ρ_{T} should be close to π by Lemma 31 (see (5.5)), (ii) the inverse T^{-1} and the Jacobian of T are easy to compute. Following [RV19], we use for T a discretization of a conformal Hamiltonian dynamics. Assume that $U(\cdot) = -\log\Pi(\cdot)$ is continuously differentiable. We consider the augmented distribution $\tilde{\Pi}(q,p) \propto \exp\{-U(q)-K(p)\}$ on $\mathbb{R}^{2d}$, where q is the position, p is the momentum, and $K(p) = p^T\mathrm{M}^{-1}p/2$ is the kinetic energy, with M a positive

definite mass matrix. By construction, the marginal distribution of the momentum under $\tilde{\Pi}$ is the target pdf $\Pi(q) = \int \tilde{\Pi}(q,p)\mathrm{d}p$. The conformal Hamiltonian ODE associated with $\tilde{\Pi}$ is defined by

$$\begin{aligned} \mathrm{d}q_t/\mathrm{d}t &= \nabla_p H(q_t,p_t) = \mathrm{M}^{-1}p_t \ , \\ \mathrm{d}p_t/\mathrm{d}t &= -\nabla_q H(q_t,p_t) - \gamma p_t = -\nabla U(q_t) - \gamma p_t \ , \end{aligned} \tag{5.10}$$

where $H(q,p) = U(q) + K(p)$, and $\gamma > 0$ is a damping constant. Any solution $(q_t,p_t)_{t\geqslant 0}$ of (5.10) satisfies setting $H_t = H(q_t,p_t)$, $\mathrm{d}H_t/\mathrm{d}t = -\gamma p_t^T \mathrm{M}^{-1} p_t \leqslant 0$. Hence, all orbits converge to fixed points that satisfy $\nabla U(q) = 0$ and $p = 0$; see e.g. [Fra+20; Mad+18].

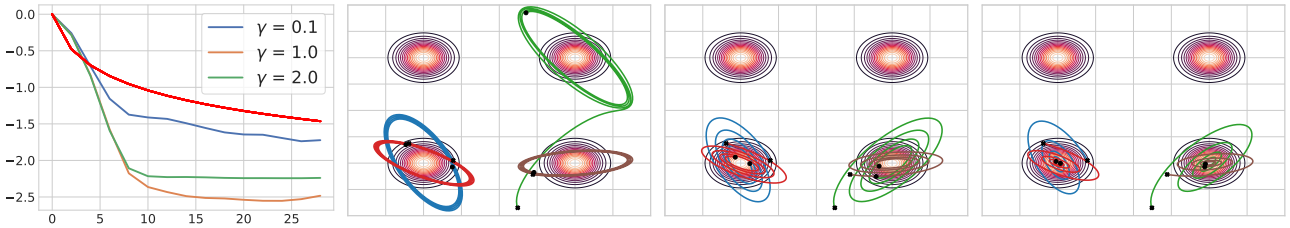


Figure 5.1: Left: $E_{\mathrm{T}_h}^{\mathbb{1}_{[K]}}(K)-1$ vs $E^{\mathrm{IS}}(K)-1$ (red) in $\log_{10}$-scale as a function of the length of trajectories K (the lower the better). Second left to right: Four examples of orbits with the same random seed for different values of γ (from left to right, $\gamma = 0.1, 1, 2$).

In the applications below, we consider the conformal version of the symplectic Euler (SE) method of (5.10), see [Fra+20]. This integrator can be constructed as a splitting of the two conformal and conservative parts of the system (5.10). When composing a dissipative with a symplectic operator, we set for all $(q,p) \in \mathbb{R}^{2d}$, $\mathrm{T}_h(q,p) = (q + h\mathrm{M}^{-1}\{\mathrm{e}^{-h\gamma}p - h\nabla U(q)\}, \mathrm{e}^{-h\gamma}p - h\nabla U(q))$, where $h > 0$ is a discretization stepsize. This transformation can be connected with classical momentum optimization schemes, see [Fra+20, Section 4]. By [Fra+20, Section 3], for any $h > 0$ T_h is a C^1-diffeomorphism on $\mathbb{R}^{2d}$ with Jacobian given by $\mathrm{J}_{\mathrm{T}_h}(q,p) = \mathrm{e}^{-\gamma hd}$. In addition, its inverse is $\mathrm{T}_h^{-1}(q,p) = (q - h\mathrm{M}^{-1}p, \mathrm{e}^{\gamma h}\{p + h\nabla U(q - h\mathrm{M}^{-1}p)\})$. Therefore, the weight (5.3) of the NEO estimator is given by

$$w_k(q,p) = \frac{\varpi_k \tilde{\rho}(\mathrm{T}_h^k(q,p))\mathrm{e}^{-\gamma khd}}{\sum_{j\in\mathbb{Z}} \varpi_{k+j}\tilde{\rho}(\mathrm{T}_h^{-j}(q,p))\mathrm{e}^{\gamma jhd}} \ ,$$

where $\tilde{\rho}(q,p) \propto \rho(q)\mathrm{e}^{-K(p)}$. Figure 5.1 displays for different values of γ on a log-scale the bound $E_{\mathrm{T}_h}^{\mathbb{1}_{[0:K]}} - 1$ appearing in Theorem 29 as a function of K, here we use the sequence of weights $(\varpi_k)_{k\in\mathbb{Z}} = (\mathbb{1}_{[0:K]}(k))_{k\in\mathbb{Z}}$ (i.e. only the $K+1$ first elements of the forward orbits are used and are equally weighted). For comparison, we also present on the same plot the bounds achieved by averaging $K+1$ independent IS estimates, $E^{\mathrm{IS}}(K) - 1 = (K+1)^{-1}\mathbb{E}_{X\sim\Lambda}[\mathrm{L}(X)^2]$. Interestingly, Figure 5.1 shows that there is a trade-off in the choice of γ which controls the exploration of the state space by the Hamiltonian dynamics since the higher γ, the faster the orbits converge towards the modes. This fast convergence prevents a "good" exploration of the space; e.g. $E_{\mathrm{T}_h}^{\mathbb{1}_{[0:K]}}$ is smaller for $\gamma = 1.0$ than for $\gamma = 2.0$ when $K > 7$.

5.3 NEO-MCMC algorithm

We now derive an MCMC method to sample from π based on the NEO-IS estimator. A natural idea consists in adapting the Sampling Importance Resampling procedure (SIR) (see for example [Rub87; SBH03]) to the NEO framework.

Algorithm 10 NEO-MCMC Sampler

At step $n \in \mathbb{N}^*$, given the conditioning orbit point Y_{n-1}.

Step 1: Update the conditioning point

1. Set $X_n^1 = Y_{n-1}$ and for any $i \in \{2, \dots, N\}$, sample $X_n^i \overset{\text{iid}}{\sim} \rho$.
2. Sample the orbit index I_n with probability proportional to $(\widehat{\mathrm{Z}}^{\varpi}_{X_n^i})_{i\in[N]}$, (5.4).
3. Set $Y_n = X_n^{I_n}$.

Step 2: Output a sample

4. Sample index K_n with probability proportional to $\{w_k(Y_n)\mathrm{L}(\mathrm{T}^k(Y_n))/\widehat{\mathrm{Z}}^{\varpi}_{Y_n}\}_{k\in\mathbb{Z}}$
5. Output $U_n = \mathrm{T}^{K_n}(Y_n)$.

The SIR method to sample $J^{\mathrm{NEO}}_{\varpi,N}$ (see (5.6)) consists of 4 steps.

(SIR-1) Draw independently $X^{1:N} \overset{\text{iid}}{\sim} \rho$ and compute the associated forward and backward orbits $\{\mathrm{T}^k(X^i)\}_{k\in\mathbb{Z}}$ of the point.
(SIR-2) Compute the normalizing constants associated with each orbit $\{\widehat{\mathrm{Z}}^{\varpi}_{X^i}\}_{i=1}^N$.
(SIR-3) Sample an orbit index $I^N \in [N]$ with probability $\{\widehat{\mathrm{Z}}^{\varpi}_{X^i}/\sum_{j=1}^N \widehat{\mathrm{Z}}^{\varpi}_{X^j}\}_{i=1}^N$.
(SIR-4) Draw the iteration index K^N on the I^N-th orbit with probability $\{\mathrm{L}(\mathrm{T}^k(X^{I^N}))w_k(X^{I^N})/\widehat{\mathrm{Z}}^{\varpi}_{X^{I^N}}\}_{k\in\mathbb{Z}}$.

The resulting draw is denoted by $U^N = \mathrm{T}^{K^N}(X^{I^N})$. By construction, for any bounded function f, we get that $\mathbb{E}\left[f(U^N)\left|X^{1:N}, I^N\right.\right] = \{\widehat{\mathrm{Z}}^{\varpi}_{X^{I^N}}\}^{-1}\sum_{k\in\mathbb{Z}} w_k(X_{I^N})\mathrm{L}(\mathrm{T}^k(X_{I^N}))$ which implies $\mathbb{E}\left[f(U^N)\left|X^{1:N}\right.\right] = J^{\mathrm{NEO}}_{\varpi,N}(f)$ (see (5.6)). Using Theorem 30, we therefore obtain $|\mathbb{E}[f(U^N)] - \int f(z)\Pi(z)\mathrm{d}z| \leqslant 10^{1/2}\|f\|_\infty E^{\varpi}_{\mathrm{T}} N^{-1}$, showing that the law of the random variable $\mu_N = \mathrm{Law}(U^N)$ converges in total variation to Π as $N \to \infty$,

$$\|\mu_N - \pi\|_{\mathrm{TV}} = \sup_{\|f\|_\infty \leqslant 1} |\mu_N(f) - \pi(f)| \leqslant 10^{1/2} E^{\varpi}_{\mathrm{T}} N^{-1} \,. \tag{5.11}$$

Based on [ADH10], we now derive the NEO-MCMC procedure, which in a nutshell consists in iterating the SIR procedure while keeping a conditioning point (or equivalently, orbit); see Section 5.9. The convergence of NEO-MCMC does not rely on letting $N \to \infty$: the NEO-MCMC works as soon as $N \geqslant 2$, although as we will see below the mixing time decreases as N increases.

This procedure is summarized in Algorithm 10. The NEO-MCMC procedure is an iterated algorithm which produces a sequence $\{(Y_n, U_n)\}_{n\in\mathbb{N}}$ of points in $\mathbb{R}^d$. The n-th iteration of the NEO-MCMC algorithm consists in two main steps: 1) updating the conditioning point $Y_{n-1} \to Y_n$ 2) sampling U_n by selecting a point in the orbit $\{\mathrm{T}^k(Y_n)\}_{k\in\mathbb{Z}}$ of the conditioning point. Compared to SIR, only the generation of the points (step (SIR-1)) is modified: we set $X_n^1 = Y_{n-1}$ (the **conditioning point**), and then draw $X_n^{2:N} \overset{\text{iid}}{\sim} \Lambda$.

The sequence $\{Y_n\}_{n\in\mathbb{N}}$ defined by Algorithm 10 is a Markov chain: $\mathbb{P}\left(Y_n \in \mathsf{A} \,|\, Y_{0:n-1}\right) = \mathbb{P}\left(Y_n \in \mathsf{A} \,|\, Y_{n-1}\right) = P(Y_n, \mathsf{A})$ where

$$P(y, \mathsf{A}) = \int \delta_y(\mathrm{d}x^1) \prod_{j=2}^N \rho(x^j)\mathrm{d}x^j \sum_{i=1}^N \frac{\widehat{\mathrm{Z}}^{\varpi}_{x^i}}{\sum_{j=1}^N \widehat{\mathrm{Z}}^{\varpi}_{x^j}} \mathbb{1}_{\mathsf{A}}(x^i) \,, \quad y \in \mathbb{R}^d \,, \mathsf{A} \in \mathcal{B}(\mathbb{R}^d) \,. \tag{5.12}$$

Note that this Markov kernel describes the way, at stage $n+1$, the conditioning point Y_{n+1} is selected given Y_n, which **depends only on** the estimator of the normalizing constants associated with each orbit, **but not** on the sample U_n selected on the conditioning orbit. In addition, given the conditioning point Y_n at the n-th iteration, the conditional distribution of the output sample U_n is

$\mathbb{P}\left(U_n \in \mathsf{B} \,|\, I_n, X_n^{1:N}\right) = \mathbb{P}\left(U_n \in \mathsf{B} \,|\, Y_n\right) = Q(Y_n, \mathsf{B})$ where

$$Q(y, \mathsf{B}) = \sum_{k\in\mathbb{Z}} \frac{w_k(y)\mathrm{L}(\mathrm{T}^k(y))}{\widehat{\mathrm{Z}}_y^{\varpi}} \mathbb{1}_{\mathsf{B}}(\mathrm{T}^k(y)) \;, \quad y \in \mathbb{R}^d \;, \mathsf{B} \in \mathcal{B}(\mathbb{R}^d) \;. \tag{5.13}$$

With these notations, if the Markov chain is started at $Y_0 = y$, then for any $n \in \mathbb{N}$, the law of the n-th conditioning point is $\mathbb{P}\left(Y_n \in \mathsf{A} \,|\, Y_0 = y\right) = P^n(y, \mathsf{A})$ and the law of the n-th sample is $\mathbb{P}\left(U_n \in \mathsf{B} \,|\, Y_0\right) = P^n Q(y, \mathsf{B})$. Define $\tilde{\pi}$ the pdf given for $y \in \mathbb{R}^d$ by

$$\tilde{\pi}(y) = \frac{\rho(y)}{\mathrm{Z}} \sum_{k\in\mathbb{Z}} w_k(y)\mathrm{L}(\mathrm{T}^k(y)) = \frac{\rho(y)\widehat{\mathrm{Z}}_y^{\varpi}}{\mathrm{Z}} \;. \tag{5.14}$$

The following theorem shows that, for any initial condition $y \in \mathbb{R}^d$, the distribution of the variable Y_n converges in total variation to $\tilde{\pi}$ and that the distribution of U_n converges to Π.

Theorem 33. *The Markov kernel P is reversible w.r.t. the distribution $\tilde{\pi}$, ergodic and Harris positive, i.e., for all $y \in \mathbb{R}^d$, $\lim_{n\to\infty} \|P^n(y,\cdot) - \tilde{\pi}\|_{\mathrm{TV}} = 0$. In addition, $\pi = \tilde{\pi}Q$ and $\lim_{n\to\infty} \|P^n Q(y,\cdot) - \pi\|_{\mathrm{TV}} = 0$. Moreover, for any bounded function g and any $y \in \mathbb{R}^d$, $\lim_{n\to\infty} n^{-1} \sum_{i=0}^{n-1} g(U_i) = \pi(g)$, $\mathbb{P}$-almost surely, where $\{U_i\}_{i\in\mathbb{N}}$ is defined in Algorithm 10 with $Y_0 = y$.*

The proof is postponed to Section 5.7.6.

Remark 34. We may provide another sampling procedure of $\{Y_n\}_{n\in\mathbb{N}}$. Define the pdf on the extended space $[N] \times \mathbb{R}^{dN}$ by $\check{\pi}(i, x^{1:N}) = N^{-1}\tilde{\pi}(x^i) \prod_{j=1, j\neq i}^N \rho(x^j)$. Consider a Gibbs sampler targeting $\check{\pi}$ consisting in (a) sampling $X_n^{1:N\setminus\{I_{n-1}\}} | (I_{n-1}, X_{n-1}) \sim \prod_{j\neq I_{n-1}} \rho(x^j)$, (b) sampling $I_n | X_n^{1:N} \sim \mathrm{Cat}(\{\widehat{\mathrm{Z}}_{X_n^i}^{\varpi} / \sum_{j=1}^N \widehat{\mathrm{Z}}_{X_n^j}^{\varpi}\}_{i=1}^N$ and (c) set $Y_n = X_n^{I_n}$. This algorithm is a Gibbs sampler on $\check{\pi}$ and we easily verify that the distribution of $\{Y_n\}_{n\in\mathbb{N}}$ is the same as Algorithm 10.

The next theorem provides non asymptotic quantitative bounds on the convergence in total variation. The main interest of NEO-MCMC algorithm is motivated empirically from observed behaviour: the mixing time of the corresponding Markov chain improves as N increases. This behaviour is quantified theoretically in the next theorem. Moreover, this improvement is obtained with little extra computational overhead, since sampling N points from the proposal distribution Λ, computing the forward and backward orbits of the points and evaluating the normalizing constants $\{\widehat{\mathrm{Z}}_{X_n^i}^{\varpi}\}_{i=1}^N$ can be performed in parallel.

Theorem 35. *Assume that $\mathrm{L} < \infty$, see (5.5). Set $\epsilon_N = (N-1)/(2\mathrm{L} + N - 2)$ and $\kappa_N = 1 - \epsilon_N$. Then, for any $y \in \mathbb{R}^d$ and $k \in \mathbb{N}$, $\|P^k(y,\cdot) - \tilde{\pi}\|_{\mathrm{TV}} \leqslant \kappa_N^k$ and $\|P^k Q(y,\cdot) - \pi\|_{\mathrm{TV}} \leqslant \kappa_N^k$.*

Instead of sampling the new points $X_n^{2:N}$ independently from Λ (Step 1 in Algorithm 10), it is possible to draw the proposals $X_n^{1:N}$ conditional to the current point Y_{n-1}; see [So06; CL07a; SN+18; Rui+21] for related works. Following [Rui+21], we use a reversible Markov kernel w.r.t. the proposal Λ, i.e., such that $\Lambda(x)m(x,x') = \Lambda(x')m(x',x)$, assuming for simplicity that this kernel has density $m(x,x')$. If $\Lambda = \mathrm{N}(0, \sigma^2 \mathrm{Id}_d)$, an appropriate choice is an autoregressive kernel $m(x,x') = \mathrm{N}(x'; \alpha x, \sigma^2(1-\alpha^2)\mathrm{Id}_d)$. More generally, we can use a Metropolis–Hastings kernel with invariant distribution Λ. In this case, $r_1(x^1, x^{1:N\setminus\{1\}}) = \prod_{j=2}^N m(x^{j-1}, x^j)$ and for each $i \in [2:N]$,

$$r_i(x^i, x^{1:N\setminus\{i\}}) = \prod_{j=1}^{i-1} m(x^{j+1}, x^j) \prod_{j=i+1}^{N} m(x^{j-1}, x^j) \;. \tag{5.15}$$

Since m is reversible w.r.t. Λ, for all $i, j \in [N]$, $\Lambda(x^i) r_i(x^i, x^{1:N\setminus\{i\}}) = \Lambda(x^j) r_j(x^j, x^{1:N\setminus\{j\}})$ where $r_i(x^i; x^{1:N\setminus\{i\}})$ defines the the conditional distribution of $X^{1:N\setminus\{i\}}$ given $X^i = x^i$. The only modification in Algorithm 10 is Step 1, which is replaced by: *Draw $U_n \in [N]$ uniformly, set $X_n^{U_n} = Y_{n-1}$ and sample $X_n^{1:N\setminus\{U_n\}} \sim r_{U_n}(X_n^{U_n}, \cdot)$.* The validity of this procedure is established in Section 5.7.6.

5.4 Continuous-time version of NEO and NEIS

The NEO framework can be thought of as an extension of NEIS introduced in [RV19]. NEIS focuses on normalizing constant estimation and should be therefore compared with NEO-IS. In [RV19], the authors do not consider possible extensions of these ideas to sampling problems. We consider here how NEO could be adapted to continuous-time dynamical system. Proofs of the statements and detailed technical conditions are postponed to Section 5.8.

Consider the Ordinary Differential Equation (ODE) $\dot{x}_t = b(x_t)$, where $b\colon \mathbb{R}^d \to \mathbb{R}^d$ is a smooth vector field. Denote by $(\phi_t)_{t\in\mathbb{R}}$ the flow of this ODE (assumed to be well-behaved). Under appropriate regularity condition $\mathrm{J}_{\phi_t}(x) = \exp(\int_0^t \nabla \cdot b(\phi_s(x))\mathrm{d}s)$; see Lemma 40. Let $\varpi : \mathbb{R} \to \mathbb{R}_+$ be a nonnegative smooth function with finite support, with $\Omega^c = \int_{-\infty}^{\infty} \varpi(t)\mathrm{d}t$. The continuous-time counterpart of the proposal distribution (5.1) is $\rho_{\mathrm{T}}^{\mathrm{c}}(x) = (\Omega^c)^{-1} \int_{-\infty}^{\infty} \varpi(t)\rho(\phi_{-t}(x))\mathrm{J}_{\phi_{-t}}(x)\mathrm{d}t$, which is a continuous mixture of the pushforward of the proposal Λ by the flow of $(\phi_s)_{s\in\mathbb{R}}$. Assuming for simplicity that $\Lambda(x) > 0$ for all $x \in \mathbb{R}^d$, then $\rho_{\mathrm{T}}^{\mathrm{c}}(x) > 0$ for all $x \in \mathbb{R}^d$, and using again the IS formula, for any nonnegative function f,

$$\int f(x)\rho(x)\mathrm{d}x = \int f(x)\frac{\rho(x)}{\rho_{\mathrm{T}}^{\mathrm{c}}(x)}\rho_{\mathrm{T}}^{\mathrm{c}}(x)\mathrm{d}x = \int \left[\int_{-\infty}^{\infty} w_t^{\mathrm{c}}(x) f(\phi_t(x))\mathrm{d}t\right] \rho(x)\mathrm{d}x, \tag{5.16}$$

$$w_t^{\mathrm{c}}(x) = \varpi(t)\rho(\phi_t(x))\mathrm{J}_{\phi_t}(x) \Big/ \int_{-\infty}^{\infty} \varpi(s+t)\rho(\phi_s(x))\mathrm{J}_{\phi_s}(x)\mathrm{d}s \; . \tag{5.17}$$

These relations are the continuous-time counterparts of (5.2). Eqs. (5.16)-(5.17) define a version of NEIS [RV19], with a finite support weight function ϖ; see Sections 5.8.2 and 5.8.3 for weight functions with infinite support. This identity is of theoretical interest but must be discretized to obtain a computationally tractable estimator. For $h > 0$, denote by T_h an integrator with stepsize $h > 0$ of the ODE $\dot{x} = b(x)$. We may construct NEO-IS and NEO-SNIS estimators based on the transform $\mathrm{T} \leftarrow \mathrm{T}_h$ and weights $\varpi_k \leftarrow \varpi(kh)$. We might show that for any bounded function f and for any $x \in \mathbb{R}^d$, $\lim_{h\downarrow 0} \sum_{k\in\mathbb{Z}} w_k(x) f(\mathrm{T}_h^k(x)) = \int_{-\infty}^{\infty} w_t^{\mathrm{c}}(x) f(\phi_t(x))\mathrm{d}t$, where we omitted here the dependency in h of w_k. Therefore, taking $h \downarrow 0^+$, the NEO-IS converges to the continuous-version (5.16)-(5.17). There is however an important difference between NEO and the NEIS method in [RV19] which stems from the way (5.16)-(5.17) are discretized. Compared to NEIS, NEO-IS using $\mathrm{T} \leftarrow \mathrm{T}_h$ and weights $\varpi_k \leftarrow \varpi(kh)$ is unbiased for any stepsize $h > 0$. NEIS uses an approach inspired by the nested-sampling approach, which amounts to discretizing the integral in (5.16) also in the state-variable x; see [Ski06; CR10]. This discretization is biased which prevents the use of this approach to develop MCMC sampling algorithm; see Section 5.8.

5.5 Experiments and Applications

Normalizing constant estimation The performance of NEO-IS is assessed on different normalizing constant estimation benchmarks; see [JS20]. We focus on two challenging examples. Additional experiments and discussion on hyperparameter choice are given in the supplementary material, see Section 5.10.1.

(1) Mixture of Gaussian (`MG25`): $\Pi(x) = P^{-1}\sum_{i=1}^{P} \mathrm{N}(x; \mu_{i,j}, D_d)$, where $d \in \{10, 20, 45\}$, $D_d = \mathrm{diag}(0.01, 0.01, 0.1, \ldots, 0.1)$ and $\mu_{i,j} = [i, j, 0, \ldots, 0]^T$ with $i, j \in \{-2, \ldots, 2\}$.

(2) Funnel distribution (`Fun`) $\Pi(x) = \mathrm{N}(x_1; 0, a^2)\prod_{i=1}^{d} \mathrm{N}(x_i; 0, \mathrm{e}^{2bx_1})$ with $d \in \{10, 20, 45\}$, $a = 1$, and $b = 0.5$. In both case, the proposal is $\rho = \mathrm{N}(0, \sigma_\rho^2 \mathrm{Id}_d)$ with $\sigma_\rho^2 = 5$.

The NEO-IS estimator is compared with (i) the IS estimator using the proposal ρ, (ii) the Adaptive Importance Sampling (AIS) estimator of [TK10], (iii) Stochastic Normalizing Flows (SNF)[1] and (iv) the Neural Importance Sampling (NIS)[2]. For NEO-IS, we use $\varpi_k = \mathbb{1}_{[K]}(k)$ with $K = 10$ (ten steps on the

[1]Implementation available at `https://github.com/noegroup/stochastic_normalizing_flows`.
[2]Implementation available at `https://github.com/ndeutschmann/zunis`.

forward orbit), and conformal Hamiltonian dynamics $\gamma = 1$, $M = 5 \cdot \mathrm{Id}_d$ for dimensions $d = \{10, 20\}$, and $\gamma = 2.5$ for $d = 45$ (where γ is the damping factor, M the mass matrix, h is the stepsize of the integrator). The parameters of AIS are set to obtain a complexity comparable to NEO-IS; see Section 5.10.1. For NIS, we use the default parameters and for SNF we used the same architectures as in [WKN20a]. In `Fun`, we set $\gamma = 0.2$, $K = 10$, $M = 5 \cdot \mathrm{Id}_d$, and $h = 0.3$. The IS estimator was based on $5 \cdot 10^5$ samples, and NIS, NEO-IS and AIS were computed with $5 \cdot 10^4$ samples. Figure 5.2 shows that NEO-IS consistently outperforms the competing methods.

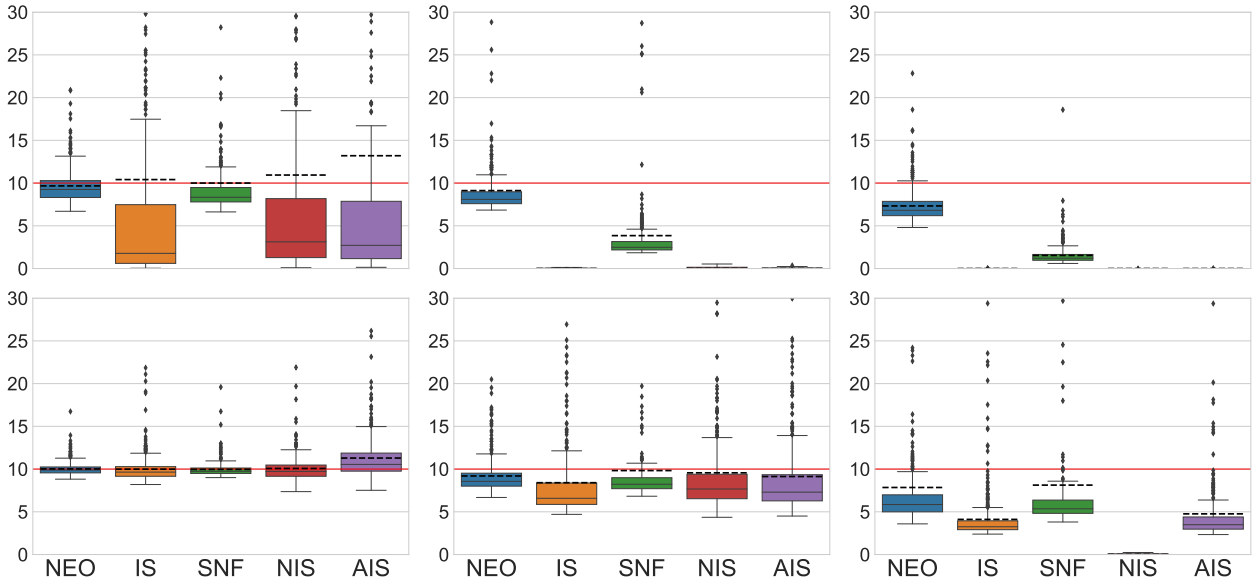


Figure 5.2: Boxplots of 500 independent estimations of the normalizing constant in dimension $d = \{10, 20, 45\}$ (from left to right) for `MG25` (top) and `Fun` (bottom). The true value is given by the red line. The figure displays the median (solid lines), the interquartile range, and the mean (dashed lines) over the 500 runs.

Sampling NEO-MCMC is assessed for the distributions `(MG25)` ($d = 40$) and `Fun` ($d = 20$). NEO-MCMC sampler is compared with (i) the No-U-Turn Sampler - Pyro library [Bin+19] - and (ii) i-SIR algorithm [Rui+21]. The proposal distribution is $\Lambda = \mathrm{N}(0, \sigma_\rho^2 \mathrm{Id}_d)$ with $\sigma_\rho^2 = 5$. Dependent proposals are used (see (5.15)) with $m(x, x') = \mathrm{N}(x'; \alpha x, \sigma_\rho^2(1 - \alpha^2)\mathrm{Id}_d)$ with $\alpha = 0.99$. For NUTS, the default parameters are used. For i-SIR, we use the same number of proposals $N = 10$, proposal distribution and dependent proposal as for NEO-MCMC. To perform a fair comparison, we use the same clock time for all three algorithms. The number of iterations for correlated i-SIR, NEO-MCMC, and NUTS are $n = 4 \cdot 10^6$, $n = 4 \cdot 10^5$, and $n = 5 \cdot 10^5$, respectively. Figure 5.3 displays the empirical two-dimensional histograms of the two first coordinates of samples from the ground truth, i-SIR, NUTS and NEO-MCMC sampler. It is worthwhile to note that NEO-MCMC algorithm performs much better for `MG25` which is a very challenging distribution, even for SOTA algorithm such as NUTS, which struggles to cross energy barriers between modes. For `Fun`, NEO-MCMC performs favourably w.r.t. NUTS, which is well adapted for this type of distributions.

Block Gibbs Inpainting with Deep Generative models and NEO-MCMC We apply NEO-MCMC to the task of sampling the posterior of a deep latent variable model. To be consistent with the rest of the paper, we use non-standard notation here with x being the latent variable and z the observation. More precisely, we assume that $x \sim \mathrm{N}(0, \mathrm{Id}_d)$ and a conditional distribution $p(z \mid x)$ which generates an image $z = (z^1, \ldots, z^D) \in \mathbb{R}^D$. Given a family of parametric *decoders* $\{x \mapsto p_\theta(z \mid x), \theta \in \Theta\}$, and a training set $\mathcal{D} = \{z_i\}_{i=1}^M$, training involves finding the MLE $\theta^* = \arg\max_{\theta \in \Theta} p_\theta(\mathcal{D})$. As $p_\theta(z) = \int p_\theta(z \mid x) p(x) \mathrm{d}x$, the likelihood is intractable and to alleviate this problem, [KW14]

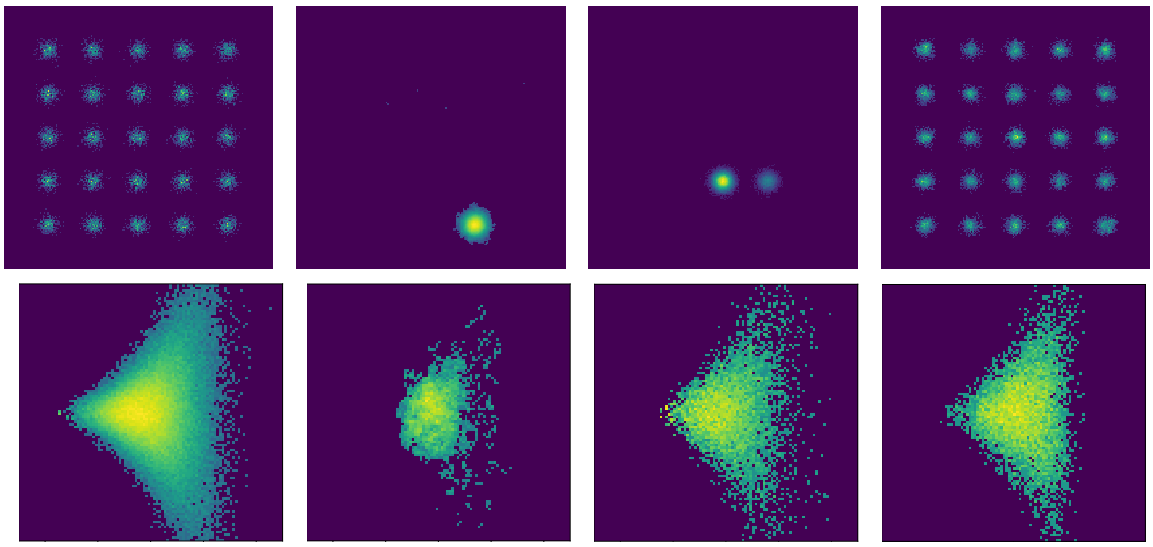


Figure 5.3: Empirical 2-D histogram of the samples of different algorithms targeting `MG25` (top) and `Fun` (bottom). Left to right: samples from the target distribution, correlated i-SIR, NUTS, NEO-MCMC.

proposed to train jointly an approximate posterior $q_\phi(x|z)$ that maximizes a tractable lower-bound on the log-likelihood: $\mathcal{L}(z, \theta, \phi) = \mathbb{E}_{X\sim q_\phi(\cdot|z)}[\log p_\theta(z, X)/q_\phi(X|z)] \leqslant p_\theta(z)$, where $q_\phi(x \mid z)$ is a tractable conditional distribution with parameters $\phi \in \Phi$. It is assumed in the sequel that conditional to the latent variable x, the coordinates are independent, *i.e.* $p_\theta(z \mid x) = \prod_{i=1}^D p_\theta(z^i|x)$.

Note that it is possible to train VAE with the NEO algorithm, using the unbiased estimate of the normalizing constant to construct an ELBO. This approach is described in the supplement Section 5.11. We do not focus on this approach here and assume that the VAE has been trained and we are only interested in the sampling problem. In our experiment, we use a VAE trained on CelebA dataset [3] [Liu+18]. We consider the Block Gibbs inpainting task introduced in [LHS18], Section 5.2.2. Given an image z, denote by $[z^t, z^b]$ the top and the bottom half pixels. Assume only $z^t_\star$ is observed, then we are interested in in-painting the bottom of an image by the posterior distribution of z^b given $z^t_\star$. This is achieved using Block Gibbs sampling. A two-stage Gibbs sampler amounts to (a) sampling $p_{\theta^*}(x|z^t, z^b)$ and (b) sampling $p_{\theta^*}(z^b|x, z^t) = p_{\theta^*}(z^b|x)$ (since z^b and z^t are independent conditional on x). Given $z_k = (z^t_\star, z^b_k)$, we sample at each step $x_k \sim p_{\theta^*}(x \mid z_k)$ and then $z^b_{k+1} \sim p_{\theta^*}(z^b \mid x_k)$. We then set $z_{k+1} = (z^t_*, z^b_{k+1})$. Stage (b) is elementary but stage (a) is challenging. We use an MCMC-within-Gibbs scheme using different samplers. We use the following decomposition of $p_{\theta^*}(x \mid z) \propto \Lambda(x)\mathrm{L}(x)$ for $\Lambda(x) \propto q^\beta_{\phi^*}(x \mid z)$ and $\mathrm{L}(x) = p_{\theta^*}(x, z)/q^\beta_{\phi^*}(x \mid z)$ with $\beta \in (0, 1)$. It is possible to sample from $\Lambda(x)$ as $q_{\phi^*}(x \mid z)$ is Gaussian. In our experiments with CelebA and the chosen trained VAE, we have $x \in \mathbb{R}^{10}$ (recall that x is our latent variable here), $z \in \mathbb{R}^{12288}$, and use $\beta = 0.1$. We then compare i-SIR, HMC and NEO-MCMC sampler in stage (a), with the same computational complexity ($N = 10$, $K = 12$, $\gamma = 0.2$ for NEO-MCMC, $N = 120$ for i-SIR, and HMC is run with $K = 20$ leap-frog steps). Again, NEO-MCMC and i-SIR use dependent proposals, with m a Random Walk Metropolis kernel with stepsize 0.1. For each algorithm, 10 steps are performed. Figure 5.8 displays the evolution of the resulting Markov chains. The samples clearly illustrate that NEO-MCMC mixes better than i-SIR and HMC. More details and examples are presented in the supplementary.

[3] Publicly available online, see `https://github.com/YannDubs/disentangling-vae/tree/master/results/betaH_celeba`

Figure 5.4: Two examples for the Gibbs inpainting task for CelebA dataset. From top to bottom (twice) : i-SIR, HMC and NEO-MCMC: From left to right, original image, blurred image to reconstruct, and output every 5 iterations of the Markov chain. Last line: a forward orbit used in NEO-MCMC for the second example.

5.6 Conclusion

In this paper, we have proposed a new family of algorithms, NEO, for computing normalizing constants and sampling from complex distributions. This methodology comes with asymptotic and non-asymptotic convergence guarantees. For normalizing constant estimation, NEO-IS compares favorably to state-of-the-art algorithms on difficult benchmarks. NEO-MCMC is also able to sample some complex distributions: it is particularly well-adapted to sampling multimodal distributions, thanks to its proposal mechanism which avoids being trapped in local modes. There are numerous potential extensions to this work. For example, it would be interesting to consider deterministic transformations other than conformal Hamiltonian dynamics integrators. These transformations could be trained, as for Neural IS, using a variation lower bound. It would also be interesting to further investigate the influence of the mixture weights $\{\varpi_k\}_{k\in\mathbb{Z}}$ on the efficiency of NEO.

Broader impact: Sampling from complex target distributions and computing their normalizing constants has numerous applications. Our work proposes novel methods to address such problems and has thus potential applications in many areas. This work does not present any foreseeable societal consequence.

Acknowledgements

Arnaud Doucet is partly supported by the EPSRC grant EP/R034710/1. He also acknowledges support of the UK Defence Science and Technology Laboratory (DSTL) and EPSRC under grant EP/R013616/1. This is part of the collaboration between US DOD, UK MOD and UK EPSRC under the Multidisciplinary University Research Initiative. Alain Durmus and Eric Moulines acknowledge support of the Lagrange Mathematical and Computing Research Center.

Supplementary material

5.7 Proofs

5.7.1 Additional notation

By abuse of notation, we denote by Λ and $\tilde{\pi}$ the probability measures with density w.r.t. the Lebesgue measure Λ and $\tilde{\pi}$ respectively.

5.7.2 Proof of (5.3)

The second expression of w_k follows from $\mathrm{J}_{\mathrm{T}^{-j}}(\mathrm{T}^k(x)) = \mathrm{J}_{\mathrm{T}^{k-j}}(x)/\mathrm{J}_{\mathrm{T}^k}(x)$ which implies

$$\begin{aligned} w_k(x) &= \varpi_k \rho(\mathrm{T}^k(x)) / \sum\nolimits_{j\in\mathbb{Z}} \varpi_j \rho(\mathrm{T}^{k-j}(x)) \mathrm{J}_{\mathrm{T}^{-j}}(\mathrm{T}^k(x)) \,, \\ &= \varpi_k \rho(\mathrm{T}^k(x)) \mathrm{J}_{\mathrm{T}^k}(x) / \sum\nolimits_{j\in\mathbb{Z}} \varpi_j \rho(\mathrm{T}^{k-j}(x)) \mathrm{J}_{\mathrm{T}^{k-j}}(x) = \varpi_k \rho_{-k}(x) \Big/ \sum\nolimits_{i\in\mathbb{Z}} \varpi_{k+i} \rho_i(x) \quad . \end{aligned}$$

5.7.3 Proof of Theorem 29

The unbiasedness of $\widehat{\mathrm{Z}}^{\varpi}_{X^{1:N}}$ follows directly from (5.2). Moreover, as $\widehat{\mathrm{Z}}^{\varpi}_{X^{1:N}}$ is unbiased and $E^{\varpi}_{\mathrm{T}} < \infty$, we can write

$$\mathrm{Var}_\rho[\widehat{\mathrm{Z}}^{\varpi}_X/\mathrm{Z}] = \mathbb{E}_\rho[(\widehat{\mathrm{Z}}^{\varpi}_X/\mathrm{Z})^2] - 1 = E^{\varpi}_{\mathrm{T}} - 1 \,. \tag{5.18}$$

As $X^{1:N} \overset{\text{iid}}{\sim} \rho$, $\mathrm{Var}_\rho[\widehat{\mathrm{Z}}^{\varpi}_{X^{1:N}}/\mathrm{Z}] = N^{-1}\,\mathrm{Var}_\rho[\widehat{\mathrm{Z}}^{\varpi}_X/\mathrm{Z}]$. Finally, if $\mathrm{L} < \infty$, then Hoeffding's inequality applies and we can write for any $\epsilon > 0$,

$$\mathbb{P}(|\widehat{\mathrm{Z}}^{\varpi}_{X^{1:N}}/\mathrm{Z} - 1| > \epsilon) \leqslant 2\exp(-2N\epsilon^2/(\mathrm{L})^2) \,. \tag{5.19}$$

Writing $\delta = 2\exp(-2N\epsilon^2/(\mathrm{L})^2)$, we identify $\log(2/\delta) = 2N\epsilon^2/(\mathrm{L})^2$ and $\epsilon = \mathrm{L}\sqrt{\log(2/\delta)/(2N)}$. Plugging this expression of ϵ in (5.19) concludes the proof.

5.7.4 Proof of Theorem 30

We first present two auxiliary lemmas necessary to establish Theorem 30.

Lemma 36. *Let A, B be two integrable random variables satisfying $|A/B| \leqslant M$ almost surely and denote $a = \mathbb{E}[A]$, $b = \mathbb{E}[B]$. Then,*

$$|\mathbb{E}[A/B] - a/b| \leqslant \frac{\sqrt{\mathrm{Var}(A/B)\,\mathrm{Var}(B)}}{b} \,, \tag{5.20}$$

$$\mathrm{Var}(A/B) \leqslant \mathbb{E}\left[|A/B - a/b|^2\right] \leqslant \frac{2}{B^2}\left(\mathbb{E}\left[|A_N - A|^2\right] + M^2\mathbb{E}\left[|B_N - B|^2\right]\right) \,. \tag{5.21}$$

Proof. Write first, using the Cauchy-Schwarz inequality,

$$\begin{aligned} \left|\mathbb{E}\left[\frac{A}{B}\right] - \frac{a}{b}\right| &= \left|\mathbb{E}\left[\frac{A}{B}\right] - \frac{\mathbb{E}\left[A\right]}{b}\right| = \left|\mathbb{E}\left[A\left(\frac{1}{B} - \frac{1}{b}\right)\right]\right| \,, \\ &= \left|\mathbb{E}\left[\frac{A}{B}\left(\frac{b-B}{b}\right)\right]\right| = \left|\mathbb{E}\left[\left(\frac{A}{B} - \mathbb{E}\left[\frac{A}{B}\right]\right)\left(\frac{B-b}{b}\right)\right]\right| \,, \\ &\leqslant \frac{\sqrt{\mathrm{Var}(A/B)}\sqrt{\mathrm{Var}(B)}}{b} \,. \end{aligned}$$

Moreover, using $|A/B| \leqslant M$ yields

$$\begin{aligned} \left|\frac{A}{B} - \frac{a}{b}\right| = \left|\frac{1}{b}(A-a) + A\left(\frac{1}{B} - \frac{1}{b}\right)\right| &\leqslant \frac{1}{b}|A-a| + \frac{|A|}{Bb}|B-b| \,, \\ &\leqslant \frac{1}{b}|A-a| + \frac{M}{b}|B-b| \,. \end{aligned}$$

Therefore,

$$|A/B - a/b|^2 \leqslant \frac{2}{b^2}\left(|A-a|^2 + M^2|B-b|^2\right) ,$$

Using that $\mathbb{E}\left[|A/B - a/b|^2\right] = \mathrm{Var}(A/B) + |\mathbb{E}[A/B] - a/b|^2$ concludes the proof. □

We get the following lemma from [Dou+11], Lemma 4.

Lemma 37. *Assume that A and B are random variables and that there exist positive constants b, M, C, K such that*

(i) $|A/B| \leqslant M$, $\mathbb{P}$*-a.s.* ,

(ii) for all $\epsilon > 0$ *and all* $N \geqslant 1$, $\mathbb{P}\left(|B-b| > \epsilon\right) \leqslant K\exp(-R\epsilon^2)$,

(iii) for all $\epsilon > 0$ *and all* $N \geqslant 1$, $\mathbb{P}\left(|A| > \epsilon\right) \leqslant K\exp\left(-R\epsilon^2/M^2\right)$,

then,

$$\mathbb{P}(|A/B| \geqslant \epsilon) \leqslant 2K\exp(-Rb^2\epsilon^2/4M^2) .$$

Proof. By the triangle inequality,

$$\begin{aligned} |A/B| &= \left|\frac{A}{B}(b-B)b^{-1} + b^{-1}A\right| , \\ &\leqslant b^{-1}\left|A/B\right|\left|b-B\right| + b^{-1}\left|A\right| \leqslant Mb^{-1}\left|b-B\right| + b^{-1}\left|A\right| . \end{aligned}$$

Therefore,

$$\{|A/B| \geqslant \epsilon\} \subseteq \left\{|B-b| \geqslant \frac{\epsilon b}{2M}\right\} \cup \left\{|A| \geqslant \frac{\epsilon b}{2}\right\} .$$

Then, conditions (ii) and (iii) imply that

$$\begin{aligned} \mathbb{P}\left(|A/B| \geqslant \epsilon\right) &\leqslant \mathbb{P}\left(|B-b| \geqslant \frac{\epsilon b}{2M}\right) + \mathbb{P}\left(|A| \geqslant \frac{\epsilon b}{2}\right) , \\ &\leqslant 2K\exp(-Rb^2\epsilon^2/(4M^2)) . \end{aligned}$$

□

Proof of Theorem 30. Let $g : \mathbb{R}^d \to \mathbb{R}$ such that $\sup_{x\in\mathbb{R}^d}|g|(x) \leqslant 1$ and denote $\pi(g) = \int g\mathrm{d}\pi$. We use Lemma 36 with $A = A_N$ and $B = \widehat{\mathrm{Z}}^{\varpi}_{X^{1:N}}$ where

$$A_N = \frac{1}{N}\sum_{i=1}^{N}\sum_{k\in\mathbb{Z}} w_k(X^i)\mathrm{L}(\mathrm{T}^k(X^i))g(\mathrm{T}^k(X^i)) , \ \widehat{\mathrm{Z}}^{\varpi}_{X^{1:N}} = \frac{1}{N}\sum_{i=1}^{N}\sum_{k\in\mathbb{Z}} w_k(X^i)\mathrm{L}(\mathrm{T}^k(X^i)) . \tag{5.22}$$

By construction, since $\sup_{x\in\mathbb{R}^d}|g|(x) \leqslant 1$, almost surely $A_N/\widehat{\mathrm{Z}}^{\varpi}_{X^{1:N}} \leqslant 1$ and $\mathrm{Var}(\widehat{\mathrm{Z}}^{\varpi}_{X^{1:N}}) = N^{-1}\mathrm{Var}(\widehat{\mathrm{Z}}^{\varpi}_{X^1})$. Then, using (5.2) with $a = \mathbb{E}[A_N] = \mathrm{Z}\pi(g)$ and $b = \mathbb{E}[\widehat{\mathrm{Z}}^{\varpi}_{X^{1:N}}] = \mathrm{Z}$, Lemma 36 implies

$$\left|J^{\mathrm{NEO}}_{\varpi,N}(g) - \pi(g)\right| = \left|\mathbb{E}[A_N/\widehat{\mathrm{Z}}^{\varpi}_{X^{1:N}}] - a/b\right| \leqslant N^{-1/2}\sqrt{\mathrm{Var}(A_N/\widehat{\mathrm{Z}}^{\varpi}_{X^{1:N}})\mathrm{Var}(\widehat{\mathrm{Z}}^{\varpi}_{X^1})} . \tag{5.23}$$

On the other hand,

$$\mathbb{E}\left[|A_N - a|^2\right] = N^{-1}\mathbb{E}_{X\sim\rho}\left[\left\{\sum_{k\in\mathbb{Z}} w_k(X)\mathrm{L}(\mathrm{T}^k(X))g(\mathrm{T}^k(X)) - \mathrm{Z}\pi(g)\right\}^2\right] \leqslant N^{-1}\mathrm{Z}^2E^{\varpi}_{\mathrm{T}} .$$

These inequalities yield using $\mathrm{Var}(\widehat{\mathrm{Z}}^{\varpi}_{X^1}) \leqslant E^{\varpi}_{\mathrm{T}}$ and Lemma 36 again:

$$\mathbb{E}\left[|J^{\mathrm{NEO}}_{\varpi,N}(g) - \pi(g)|^2\right] \leqslant \frac{2}{N}(E^{\varpi}_{\mathrm{T}} + \mathrm{Var}(\widehat{\mathrm{Z}}^{\varpi}_{X^1})) \leqslant \frac{4}{N}E^{\varpi}_{\mathrm{T}} ,$$
$$|\mathbb{E}\left[J^{\mathrm{NEO}}_{\varpi,N}(g) - \pi(g)\right]| \leqslant \frac{\sqrt{2(E^{\varpi}_{\mathrm{T}} + \mathrm{Var}(\widehat{\mathrm{Z}}^{\varpi}_{X^1}))\mathrm{Var}(\widehat{\mathrm{Z}}^{\varpi}_{X^1})}}{N} \leqslant \frac{2E^{\varpi}_{\mathrm{T}}}{N} ,$$

which concludes the proof.

Define

$$\tilde{A}_N = N^{-1}\sum_{i=1}^{N}\sum_{k\in\mathbb{Z}} w_k(X^i)\mathrm{L}(\mathrm{T}^k(X^i))\left(g(\mathrm{T}^k(X^i)) - \pi(g)\right) .$$

With this notation, the proof of (5.9) relies on the application of Lemma 37 to $A = \tilde{A}_N$ and $B = \widehat{\mathrm{Z}}^{\varpi}_{X^{1:N}}$, since

$$J^{\mathrm{NEO}}_{\varpi,N}(g) - \pi(g) = A_N/\widehat{\mathrm{Z}}^{\varpi}_{X^{1:N}} .$$

As $\sup_{x\in\mathbb{R}^d}|g|(x) \leqslant 1$, we get that $\tilde{A}_N/\widehat{\mathrm{Z}}^{\varpi}_{X^{1:N}} \leqslant 2$. By (5.2), $\mathbb{E}[\widehat{\mathrm{Z}}^{\varpi}_{X^{1:N}}] = \mathrm{Z}$ and $\widehat{\mathrm{Z}}^{\varpi}_{X^{1:N}} = N^{-1}\sum_{i=1}^{N} W_i$ with $W_i = \sum_{k\in\mathbb{Z}} w_k(X^i)\mathrm{L}(\mathrm{T}^k(X^i)) \leqslant \mathrm{L}$. Then, by Hoeffding's inequality, for all $\varepsilon > 0$,

$$\mathbb{P}(|B_N - \mathrm{Z}| > \varepsilon) \leqslant 2\exp(-2N(\varepsilon/\mathrm{L})^2) .$$

Similarly, A_N is centered and $A_N = N^{-1}\sum_{i=1}^{N} U_i$ with

$$U_i = \sum_{k\in\mathbb{Z}} w_k(X^i)\mathrm{L}(\mathrm{T}^k(X^i))\{g(\mathrm{T}^k(X^i)) - \pi(g)\}$$

and $|U_i| \leqslant 2\mathrm{L}$ almost surely. By Hoeffding's inequality, for all $\varepsilon > 0$,

$$\mathbb{P}(|A_N| > \epsilon) \leqslant 2\exp(-N\varepsilon^2/(8(\mathrm{L})^2)) .$$

The assumptions of Lemma 37 are met so that

$$\mathbb{P}(|J^{\mathrm{NEO}}_{\varpi,N}(g) - \pi(g)| > \varepsilon) \leqslant 4\exp(-\varepsilon^2 N\mathrm{Z}^2/[32(\mathrm{L})^2]) ,$$

which concludes the proof. □

5.7.5 Proof of Lemma 31

As $w_k(x) = \varpi_k\rho(\mathrm{T}^k(x))/\{\Omega\rho_{\mathrm{T}}(\mathrm{T}^k(x))\}$, by Jensen's inequality,

$$E^{\varpi}_{\mathrm{T}} = \int\left(\sum_{k\in\mathbb{Z}} w_k(x)\mathrm{L}(\mathrm{T}^k(x))/\mathrm{Z}\right)^2\rho(x)\mathrm{d}x = \int\left(\sum_{k\in\mathbb{Z}}\frac{\varpi_k}{\Omega}\frac{\pi(\mathrm{T}^k(x))}{\rho_{\mathrm{T}}(\mathrm{T}^k(x))}\right)^2\rho(x)\mathrm{d}x ,$$
$$\leqslant \int\sum_{k\in\mathbb{Z}}\frac{\varpi_k}{\Omega}\left(\frac{\pi(\mathrm{T}^k(x))}{\rho_{\mathrm{T}}(\mathrm{T}^k(x))}\right)^2\rho(x)\mathrm{d}x ,$$
$$\leqslant \Omega^{-1}\sum_{k\in\mathbb{Z}}\varpi_k\int\left(\frac{\pi(\mathrm{T}^k(x))}{\rho_{\mathrm{T}}(\mathrm{T}^k(x))}\right)^2\rho(x)\mathrm{d}x .$$

Using the change of variables $y = \mathrm{T}^k(x)$ yields, by (5.1),

$$E^{\varpi}_{\mathrm{T}} \leqslant \Omega^{-1}\sum_{k\in\mathbb{Z}}\varpi_k\int\left(\frac{\pi(y)}{\rho_{\mathrm{T}}(y)}\right)^2\rho(\mathrm{T}^{-k}(y))\mathrm{J}_{\mathrm{T}^{-k}}(y)\mathrm{d}y \leqslant \int\left(\frac{\pi(y)}{\rho_{\mathrm{T}}(y)}\right)^2\rho_{\mathrm{T}}(y)\mathrm{d}y .$$

5.7.6 Proofs of NEO MCMC sampler

Proof of Theorem 33. Note first that by symmetry, we have

$$P(y,\mathsf{A}) = N^{-1}\int \sum_{i=1}^{N}\delta_y(\mathrm{d}x^i)\prod_{j=1,j\neq i}^{N}\rho(x^j)\mathrm{d}x^j\sum_{k=1}^{N}\frac{\widehat{\mathrm{Z}}^{\varpi}_{x^k}}{\sum_{j=1}^{N}\widehat{\mathrm{Z}}^{\varpi}_{x^j}}\mathbb{1}_{\mathsf{A}}(x^k)\;. \tag{5.24}$$

We begin with the proof of reversibility of P w.r.t. $\tilde{\pi}$. Let f,g be nonnegative measurable functions. By definition of P,

$$\begin{aligned}
\int \tilde{\pi}(\mathrm{d}y)P(y,\mathrm{d}y')f(y)g(y') &= \frac{1}{N\mathrm{Z}}\int\sum_{i=1}^{N}\Lambda(\mathrm{d}y)\widehat{\mathrm{Z}}^{\varpi}_{y}f(y)\delta_y(\mathrm{d}x^i)\prod_{l=1,l\neq i}^{N}\Lambda(\mathrm{d}x^l)\sum_{k=1}^{N}\frac{\widehat{\mathrm{Z}}^{\varpi}_{x^k}}{\sum_{j=1}^{N}\widehat{\mathrm{Z}}^{\varpi}_{x^j}}g(x^k)\;,\\
&= \frac{1}{N\mathrm{Z}}\int\sum_{i=1}^{N}\widehat{\mathrm{Z}}^{\varpi}_{x^i}f(x^i)\prod_{l=1}^{N}\Lambda(\mathrm{d}x^l)\sum_{k=1}^{N}\frac{\widehat{\mathrm{Z}}^{\varpi}_{x^k}}{\sum_{j=1}^{N}\widehat{\mathrm{Z}}^{\varpi}_{x^j}}g(x^k)\;,\\
&= \frac{1}{N\mathrm{Z}}\int\prod_{l=1}^{N}\Lambda(\mathrm{d}x^l)\frac{\sum_{i=1}^{N}\widehat{\mathrm{Z}}^{\varpi}_{x^i}f(x^i)\sum_{k=1}^{N}\widehat{\mathrm{Z}}^{\varpi}_{x^k}g(x^k)}{\sum_{j=1}^{N}\widehat{\mathrm{Z}}^{\varpi}_{x^j}}\;,\\
&= \int\tilde{\pi}(\mathrm{d}y)P(y,\mathrm{d}y')f(y')g(y)\;,
\end{aligned}$$

which shows that P is $\tilde{\pi}$-reversible. We now establish that P is $\tilde{\pi}$-irreducible. We have for $y\in\mathbb{R}^d$, $\mathsf{A}\in\mathcal{B}(\mathbb{R}^d)$,

$$\begin{aligned}
P(y,\mathsf{A}) &= \int\delta_y(\mathrm{d}x^1)\sum_{i=1}^{N}\frac{\widehat{\mathrm{Z}}^{\varpi}_{x^i}}{N\widehat{\mathrm{Z}}^{\varpi}_{x^{1:N}}}\mathbb{1}_{\mathsf{A}}(x^i)\prod_{j=2}^{N}\Lambda(\mathrm{d}x^j)\\
&= \int\frac{\widehat{\mathrm{Z}}^{\varpi}_{y}}{\widehat{\mathrm{Z}}^{\varpi}_{y}+\sum_{j=2}^{N}\widehat{\mathrm{Z}}^{\varpi}_{x^j}}\mathbb{1}_{\mathsf{A}}(x)\prod_{j=2}^{N}\Lambda(\mathrm{d}x^j)+\int\sum_{i=2}^{N}\frac{\widehat{\mathrm{Z}}^{\varpi}_{x^i}}{\widehat{\mathrm{Z}}^{\varpi}_{y}+\sum_{j=2}^{N}\widehat{\mathrm{Z}}^{\varpi}_{x^j}}\mathbb{1}_{\mathsf{A}}(x^i)\prod_{j=2}^{N}\Lambda(\mathrm{d}x^j)\\
&\geqslant \sum_{i=2}^{N}\int\frac{\widehat{\mathrm{Z}}^{\varpi}_{x^i}}{\widehat{\mathrm{Z}}^{\varpi}_{y}+\widehat{\mathrm{Z}}^{\varpi}_{x^i}+\sum_{j=2,j\neq i}^{N}\widehat{\mathrm{Z}}^{\varpi}_{x^j}}\mathbb{1}_{\mathsf{A}}(x^i)\prod_{j=2}^{N}\Lambda(\mathrm{d}x^j)\\
&\geqslant \sum_{i=2}^{N}\int\tilde{\pi}(\mathrm{d}x^i)\mathbb{1}_{\mathsf{A}}(x^i)\int\frac{\mathrm{Z}}{\widehat{\mathrm{Z}}^{\varpi}_{y}+\widehat{\mathrm{Z}}^{\varpi}_{x^i}+\sum_{j=2,j\neq i}^{N}\widehat{\mathrm{Z}}^{\varpi}_{x^j}}\prod_{j=2,j\neq i}^{N}\Lambda(\mathrm{d}x^j)\;.
\end{aligned}$$

Since the function $f\colon z\mapsto(z+a)^{-1}$ is convex on $\mathbb{R}_+$ for $a>0$, we get for $i\in\{2,\ldots,N\}$,

$$\begin{aligned}
\int\frac{\mathrm{Z}}{\widehat{\mathrm{Z}}^{\varpi}_{y}+\widehat{\mathrm{Z}}^{\varpi}_{x^i}+\sum_{j=2,j\neq i}^{N}\widehat{\mathrm{Z}}^{\varpi}_{x^j}}\prod_{j=2,j\neq i}^{N}\Lambda(\mathrm{d}x^j) &\geqslant \frac{\mathrm{Z}}{\widehat{\mathrm{Z}}^{\varpi}_{y}+\widehat{\mathrm{Z}}^{\varpi}_{x^i}+\int\sum_{j=2,j\neq i}^{N}\widehat{\mathrm{Z}}^{\varpi}_{x^j}\prod_{j=2,j\neq i}^{N}\Lambda(\mathrm{d}x^j)}\\
&\geqslant \frac{\mathrm{Z}}{\widehat{\mathrm{Z}}^{\varpi}_{y}+\widehat{\mathrm{Z}}^{\varpi}_{x^i}+\mathrm{Z}(N-2)}\;.
\end{aligned} \tag{5.25}$$

Therefore, for $\mathsf{A}\in\mathcal{B}(\mathbb{R}^d)$ satisfying $\tilde{\pi}(\mathsf{A})>0$, we get $P(y,\mathsf{A})>0$ for any $y\in\mathbb{R}^d$ since $\widehat{\mathrm{Z}}^{\varpi}_{x}<\infty$ for any $x\in\mathbb{R}^d$. By definition, P is $\tilde{\pi}$-irreducible.

We show that P is Harris recurrent using [Tie94], Corollary 2. To this end, since P is $\tilde{\pi}$-irreducible, it is sufficient to show that P is a Metropolis type kernel. Define $\alpha(x^1,x^2)=(N-1)\int\prod_{j=3}^{N}\Lambda(\mathrm{d}x^j)\widehat{\mathrm{Z}}^{\varpi}_{x^2}/\sum_{j=1}^{N}\widehat{\mathrm{Z}}^{\varpi}_{x^j}$ for $x^1,x^2\in\mathbb{R}^d$ and $\rho_{2:N}(\mathrm{d}x^{2:N})=\{\prod_{j=2}^{N}\rho_{2:N}(x^j)\}\mathrm{d}x^{2:N}$. Then, by

(5.12), we get with this notation, for $y \in \mathbb{R}^d$, $\mathsf{A} \in \mathcal{B}(\mathbb{R}^d)$,

$$
\begin{aligned}
&P(y, \mathsf{A}) \\
&= \int \delta_y(\mathrm{d}x^1)\rho_{2:N}(\mathrm{d}x^{2:N}) \sum_{i=2}^N \frac{\widehat{\mathrm{Z}}^{\varpi}_{x^i}}{N\widehat{\mathrm{Z}}^{\varpi}_{x^{1:N}}} \mathbb{1}_{\mathsf{A}}(x^i) + \int \delta_y(\mathrm{d}x^1)\rho_{2:N}(\mathrm{d}x^{2:N}) \frac{\widehat{\mathrm{Z}}^{\varpi}_{x^1}}{N\widehat{\mathrm{Z}}^{\varpi}_{x^{1:N}}} \mathbb{1}_{\mathsf{A}}(x^1) \\
&= \sum_{i=2}^N \int \delta_y(\mathrm{d}x^1)\rho_{2:N}(\mathrm{d}x^{2:N}) \frac{\widehat{\mathrm{Z}}^{\varpi}_{x^i}}{N\widehat{\mathrm{Z}}^{\varpi}_{x^{1:N}}} \mathbb{1}_{\mathsf{A}}(x^i) + \int \delta_y(\mathrm{d}x^1)\rho_{2:N}(\mathrm{d}x^{2:N}) \frac{\widehat{\mathrm{Z}}^{\varpi}_{x^1}}{N\widehat{\mathrm{Z}}^{\varpi}_{x^{1:N}}} \mathbb{1}_{\mathsf{A}}(x^1) \\
&= \sum_{i=2}^N \int \delta_y(\mathrm{d}x^1)\Lambda(\mathrm{d}x^i) \int \prod_{j=2, j\neq i}^N \Lambda(x^j)\mathrm{d}x^j \frac{\widehat{\mathrm{Z}}^{\varpi}_{x^i}\mathbb{1}_{\mathsf{A}}(x^i)}{N\widehat{\mathrm{Z}}^{\varpi}_{x^{1:N}}} + \int \delta_y(\mathrm{d}x^1)\rho_{2:N}(\mathrm{d}x^{2:N}) \frac{\widehat{\mathrm{Z}}^{\varpi}_{x^1}\mathbb{1}_{\mathsf{A}}(x^1)}{N\widehat{\mathrm{Z}}^{\varpi}_{x^{1:N}}} \\
&= \sum_{i=2}^N \int \frac{\alpha(y, x^i)}{(N-1)} \mathbb{1}_{\mathsf{A}}(x^i)\Lambda(\mathrm{d}x^i) + \int \delta_y(\mathrm{d}x^1)\rho_{2:N}(\mathrm{d}x^{2:N}) \left\{1 - \sum_{i=2}^N \frac{\widehat{\mathrm{Z}}^{\varpi}_{x^i}}{N\widehat{\mathrm{Z}}^{\varpi}_{x^{1:N}}}\right\} \mathbb{1}_{\mathsf{A}}(x^1) \\
&= \int_{\mathsf{A}} \alpha(y, y')\Lambda(y')\mathrm{d}y' + \left(1 - \int \alpha(y, y')\Lambda(y')\mathrm{d}y'\right) \delta_y(\mathsf{A}) \, . \qquad (5.26)
\end{aligned}
$$

With the terminology of [Tie94], Corollary 2, P is Metropolis type kernel and therefore is Harris recurrent.

Note that Algorithm 10 defines a Markov chain $\{Y_i, U_i\}_{i\in\mathbb{N}}$ taking for U_0 an arbitrary initial point with Markov kernel denoted by $\tilde{P}$. By abuse of notation, we denote by $\{Y_i, U_i\}_{i\in\mathbb{N}}$ the canonical process on the canonical space $(\mathbb{R}^d \times \mathbb{R}^d)^{\mathbb{N}}$ endowed with the corresponding σ-field and denote by $\mathbb{P}_{y,u}$ the distribution associated with the Markov chain with kernel $\tilde{P}$ and initial distribution $\delta_y \otimes \delta_u$. Denote for any $y \in \mathbb{R}^d$ by $\mathbb{P}_y$ the marginal distribution of $\mathbb{P}_{y,u}$ with respect to $\{Y_i\}_{i\in\mathbb{N}}$, *i.e.* $\mathbb{P}_y(\mathsf{A}) = \mathbb{P}_{(y,u)}(\{Y_i\}_{i\in\mathbb{N}} \in \mathsf{A})$ for $u \in \mathbb{R}^d$, noting that by definition, $\mathbb{P}_{(y,u)}(\mathsf{A} \times (\mathbb{R}^d)^{\mathbb{N}})$ does not depend on u. In addition, under $\mathbb{P}_y$, $\{Y_i\}_{i\in\mathbb{N}}$ is a Markov chain associated with P. Therefore, since P is $\tilde{\pi}$-irreducible and Harris recurrent, we get by [Dou+18], Theorem 11.3.1, and [Tie94], Theorem 2, 3. for any $y \in \mathbb{R}^d$, $\lim_{k\to\infty} \|\delta_y P^k - \tilde{\pi}\|_{\mathrm{TV}} = 0$ and for any bounded and measurable function g,

$$
n^{-1} \sum_{k=1}^n g(Y_k) = \tilde{\pi}(g) \, , \qquad \mathbb{P}_y\text{-almost surely} \, . \qquad (5.27)
$$

We now turn to proving the properties regarding Q. For any $\mathsf{B} \in \mathcal{B}(\mathbb{R}^d)$, using (5.2), we obtain

$$
\int \tilde{\pi}(y)Q(y, \mathsf{B})\mathrm{d}y = \mathrm{Z}^{-1} \int \Lambda(y) \sum_{k\in\mathbb{Z}} w_k(y)\mathrm{L}(\mathrm{T}^k(y))\mathbb{1}_{\mathsf{B}}(\mathrm{T}^k(y))\mathrm{d}y = \pi(\mathsf{B}) \, .
$$

Using for all $y \in \mathbb{R}^d$, $\lim_{n\to\infty} \|P^n(y, \cdot) - \tilde{\pi}\|_{\mathrm{TV}} = 0$, we get $\lim_{n\to\infty} \|P^n Q(y, \cdot) - \pi\|_{\mathrm{TV}} = 0$. It remains to show the stated Law of Large Numbers. Let $y, u \in \mathbb{R}^d$ and g be a bounded measurable function. Define for any $i \in \mathbb{N}^*$, $\tilde{U}_i = g(U_i) - Qg(Y_i)$. By definition, for any $i \in \mathbb{N}^*$, $\left|\tilde{U}_i\right| \leqslant 2\sup_{x\in\mathbb{R}^d} |g(x)|$ and $\mathbb{E}_{(y,u)}[\tilde{U}_i|\mathcal{F}_{i-1}] = 0$, where $\{\mathcal{F}_k\}_{k\in\mathbb{N}}$ is the canonical filtration. Therefore, $\{\tilde{U}_i\}_{i\in\mathbb{N}^*}$ are $\{\mathcal{F}_k\}_{k\in\mathbb{N}}$-martingale increments and $\{S_k = \sum_{i=1}^k \tilde{U}_i\}_{k\in\mathbb{N}}$ is a $\{\mathcal{F}_k\}_{k\in\mathbb{N}}$-martingale. Using [HH80], Theorem 2.18, we get

$$
\lim_{n\to\infty} \{S_n/n\} = 0 \, , \quad \mathbb{P}_{(y,u)}\text{-almost surely} \, . \qquad (5.28)
$$

The proof is completed using that $\lim_{n\to\infty}\{n^{-1}\sum_{i=1}^n Qg(Y_i)\} = \tilde{\pi}(Qg) = \pi(g)$, $\mathbb{P}_y$-almost surely by (5.27) and therefore by definition, $\mathbb{P}_{(y,u)}$-almost surely. □

Proof of Theorem 35. We have for $(x, \mathsf{A}) \in \mathbb{R}^d \times \mathcal{B}(\mathbb{R}^d)$,

$$
P(y, \mathsf{A}) \geqslant \sum_{i=2}^N \int \tilde{\pi}(\mathrm{d}x^i)\mathbb{1}_{\mathsf{A}}(x^i) \int \frac{\mathrm{Z}}{\widehat{\mathrm{Z}}^{\varpi}_{y} + \widehat{\mathrm{Z}}^{\varpi}_{x^i} + \sum_{j=2, j\neq i}^N \widehat{\mathrm{Z}}^{\varpi}_{x^j}} \prod_{j=2, j\neq i}^N \Lambda(\mathrm{d}x^j) \, .
$$

Moreover, as for any $x \in \mathbb{R}^d$, $\widehat{\mathrm{Z}}_x^\varpi / \mathrm{Z} \leqslant \mathrm{L}$,

$$\int \frac{\mathrm{Z}}{\widehat{\mathrm{Z}}_y^\varpi + \widehat{\mathrm{Z}}_{x^i}^\varpi + \sum_{j=2, j\neq i}^N \widehat{\mathrm{Z}}_{x^j}^\varpi} \prod_{j=2, j\neq i}^N \Lambda(\mathrm{d}x^j) \geqslant \frac{\mathrm{Z}}{\widehat{\mathrm{Z}}_y^\varpi + \widehat{\mathrm{Z}}_{x^i}^\varpi + \mathrm{Z}(N-2)} \geqslant \frac{1}{2\mathrm{L} + N - 2} \, .$$

We finally obtain the inequality

$$P(x, \mathsf{A}) \geqslant \tilde{\pi}(\mathsf{A}) \times \frac{N-1}{2\mathrm{L} + N - 2} = \epsilon_N \tilde{\pi}(\mathsf{A}) \, . \tag{5.29}$$

The proof for P is concluded from [Dou+18, Theorem 18.2.4].

As $\|P^k(y, \cdot) - \tilde{\pi}\|_{\mathrm{TV}} \leqslant \kappa_N^k$, for any bounded function f, $\|f\|_\infty \leqslant 1$, we have $|P^k f(y) - \tilde{\pi}(f)| \leqslant \kappa_N^k$, by definition of the Total Variation Distance. Then, writing $f = Qg$ for any bounded function g, $\|g\|_\infty \leqslant 1$, we have $\|f\|_\infty \leqslant 1$ and

$$|P^k f(y) - \tilde{\pi}(f)| = |P^k Q g(y) - \tilde{\pi} Q(g)| = |P^k Q g(y) - \pi(g)| \leqslant \kappa_N^k \, . \tag{5.30}$$

□

Write now P the Markov kernel extending to correlated proposals: for $y \in \mathbb{R}^d$ and $\mathsf{A} \in \mathcal{B}(\mathbb{R}^d)$,

$$P(y, \mathsf{A}) = N^{-1} \int \sum_{i=1}^N \delta_y(\mathrm{d}x^i) r_i(x^i, \mathrm{d}x^{1:n\setminus\{i\}}) \sum_{k=1}^N \frac{\widehat{\mathrm{Z}}_{x^k}^\varpi}{N \widehat{\mathrm{Z}}_{x^{1:N}}^\varpi} \mathbb{1}_{\mathsf{A}}(x^k) \, , \tag{5.31}$$

where the Markov kernels R_i are defined by $R_i(x^i, \mathrm{d}x^{1:N\setminus\{i\}}) = r_i(x^i, x^{1:N\setminus\{i\}}) \mathrm{d}x^{1:N\setminus\{i\}}$ and r_i by (5.15).

Theorem 38. *P is $\tilde{\pi}$-invariant.*

Proof. Define the Nd-dimensional probability measure $\bar{\Lambda}_N(\mathrm{d}x^{1:N}) = \Lambda(\mathrm{d}x^1) R_1(x^1, \mathrm{d}x^{2:n})$. Let $\mathsf{A} \in \mathcal{B}(\mathbb{R}^d)$. Then, we have

$$\begin{aligned}
\tilde{\pi} P(\mathsf{A}) &= N^{-1} \int \tilde{\pi}(\mathrm{d}y) \int \sum_{i=1}^N \delta_y(\mathrm{d}x^i) R_i(x^i, \mathrm{d}x^{1:n\setminus\{i\}}) \sum_{k=1}^N \frac{\widehat{\mathrm{Z}}_{x^k}^\varpi}{N \widehat{\mathrm{Z}}_{x^{1:N}}^\varpi} \mathbb{1}_{\mathsf{A}}(x^k) \\
&= (N\mathrm{Z})^{-1} \int \sum_{i=1}^N \Lambda(\mathrm{d}x^i) \widehat{\mathrm{Z}}_{x^i}^\varpi R_i(x^i, \mathrm{d}x^{1:n\setminus\{i\}}) \sum_{k=1}^N \frac{\widehat{\mathrm{Z}}_{x^k}^\varpi}{N \widehat{\mathrm{Z}}_{x^{1:N}}^\varpi} \mathbb{1}_{\mathsf{A}}(x^k) \\
&= (N\mathrm{Z})^{-1} \int \bar{\Lambda}_N(\mathrm{d}x^{1:N}) \sum_{i=1}^N \widehat{\mathrm{Z}}_{x^i}^\varpi \sum_{k=1}^N \frac{\widehat{\mathrm{Z}}_{x^k}^\varpi}{N \widehat{\mathrm{Z}}_{x^{1:N}}^\varpi} \mathbb{1}_{\mathsf{A}}(x^k) \\
&= (N\mathrm{Z})^{-1} \int \sum_{k=1}^N \widehat{\mathrm{Z}}_{x^k}^\varpi \bar{\Lambda}_N(\mathrm{d}x^{1:N}) \mathbb{1}_{\mathsf{A}}(x^k) \\
&= (N\mathrm{Z})^{-1} \int \sum_{k=1}^N \widehat{\mathrm{Z}}_{x^k}^\varpi \Lambda(\mathrm{d}x^k) \mathbb{1}_{\mathsf{A}}(x^k) = \tilde{\pi}(\mathsf{A}) \, .
\end{aligned}$$

□

5.8 Continuous-time limit of NEO and NEIS

5.8.1 Proof for the continuous-time limit

Consider $\bar{h} > 0$ and a family $\{\mathrm{T}_h \, : \, h \in (0, \bar{h}]\}$ of C^1-diffeomorphisms. For $N \in \mathbb{N}^*$ and a bounded and continuous $f : \mathbb{R}^d \to \mathbb{R}$, write

$$I_{\varpi, N, h}^{\mathrm{NEO}}(f) = N^{-1} \sum_{i=1}^N \sum_{k \in \mathbb{Z}} w_{k,h}(X^i) f(\mathrm{T}_h^k(X^i)) \, , \tag{5.32}$$

where $\{X_i\}_{i=1}^N \overset{\text{iid}}{\sim} \rho$ and for some weight function $\varpi^{\mathrm{c}} : \mathbb{R} \to \mathbb{R}_+$ with bounded support (see **H**3), $k \in \mathbb{Z}$ and $h > 0$, setting $\varpi_{k,h} = \varpi^{\mathrm{c}}(kh)$,

$$w_{k,h}(x) = \varpi_{k,h}\rho_{-k}(x) \Big/ \sum\nolimits_{i\in\mathbb{Z}} \varpi_{k+i,h}\rho_i(x) \ . \tag{5.33}$$

We show in this section the convergence of the sequence of NEO-IS estimators $\{I^{\mathrm{NEO}}_{\varpi,N,h}(f) : h \in \big(0, \bar{h}\big]\}$ as $h \downarrow 0$ to its continuous counterpart, the version (5.16) of NEIS [RV19], with weight function ϖ, in the case where for any $h \in \big(0, \bar{h}\big]$, T_h corresponds to one step of a discretization scheme with stepsize h of the ODE

$$\dot{x}_t = b(x_t) \ , \tag{5.34}$$

where $b : \mathbb{R}^d \to \mathbb{R}^d$ is a drift function. We are particularly interested in the case where (5.34) corresponds to the conformal Hamilonian dynamics (5.10) and $\{\mathrm{T}_h : h \in \big(0, \bar{h}\big]\}$ to its conformal symplectic Euler discretization: for all $(q, p) \in \mathbb{R}^{2d}$,

$$\mathrm{T}_h(q, p) = (q + h\mathrm{M}^{-1}\{\mathrm{e}^{-h\gamma}p - h\nabla U(q)\}, \mathrm{e}^{-h\gamma}p - h\nabla U(q)) \ . \tag{5.35}$$

We make the following conditions on b, ρ, ϖ^{c} and $\{\mathrm{T}_h : h \in \big(0, \bar{h}\big]\}$.

H1. *The function b is continuously differentiable and L_b-Lipschitz.*

Under **H**1, consider $(\phi_t)_{t\geqslant 0}$ the differential flow associated with (5.34), *i.e.* $\phi_t(x) = x_t$ where $(x_t)_{t\in\mathbb{R}}$ is the solution of (5.34) starting from x. Note that **H**1 implies that $(t, x) \mapsto \phi_t(x)$ is continuously differentiable on $\mathbb{R} \times \mathbb{R}^d$, see [Har82, Theorem 4.1 Chapter V].

H1 is satisfied in the case of the conformal Hamiltonian dynamics if the potential U is continuously differentiable and with Lipschitz gradient, that is there exists $L_U \in \mathbb{R}_+^*$ such that for any $x_1, x_2 \in \mathbb{R}^d$, $\|\nabla U(x_1) - \nabla U(x_2)\| \leqslant L_U\|x_1 - x_2\|$.

H2. *For any $h \in \big(0, \bar{h}\big]$, $\mathrm{T}_h : \mathbb{R}^d \to \mathbb{R}^d$ is a C^1-diffeomorphism. In addition, it holds:*

(i) there exist $C \geqslant 0$ and $\delta \in (0, 1]$ such that for any $x \in \mathbb{R}^d$,

$$\|\,\mathrm{T}_h(x) - (x + hb(x))\| \leqslant Ch^{1+\delta}(1 + \|x\|) \ ;$$

(ii) for any $x \in \mathbb{R}^d$ and $T \in \mathbb{R}_+^*$,

$$\lim_{h\downarrow 0} \max_{k\in[-\lfloor T/h\rfloor:\lfloor T/h\rfloor]} \|\mathrm{J}_{\phi_{kh}}(x) - \mathrm{J}_{\mathrm{T}_h^k}(x)\| = 0 \ .$$

Note that **H**2 is automatically satisfied for the conformal symplectic Euler discretization (5.35) of the conformal Hamiltonian dynamics. Indeed, in that case $\mathrm{D}\, b(\phi_t(x)) = \gamma d$, and therefore $\mathrm{J}_{\phi_t}(x) = e^{\gamma dt}$ for $t \in \mathbb{R}$, and for any $h > 0, k \in \mathbb{Z}$, $\mathrm{J}_{\mathrm{T}_h^k}(x) = e^{\gamma dhk}$; see [Fra+20].

Define

$$\mathrm{support}(\varpi^{\mathrm{c}}) = \{t \in \mathbb{R} \ : \ \varpi^{\mathrm{c}}(t) \neq 0\} \ . \tag{5.36}$$

H3. *(i) Λ is continuous and positive on $\mathbb{R}^d$*

(ii) ϖ^{c} is piecewise continuous on $\mathbb{R}$, its support $\mathrm{support}(\varpi^{\mathrm{c}})$ *is bounded and* $\sup_{(s,t)\in\mathsf{A}_\varpi} \varpi^{\mathrm{c}}(t)/\varpi^{\mathrm{c}}(t+s) = m < \infty$ *where*

$$\mathsf{A}_\varpi = \{(s, t) \in \mathbb{R}^2; \ t \in \mathrm{support}(\varpi^{\mathrm{c}}), \ (s + t) \in \mathrm{support}(\varpi^{\mathrm{c}})\} \ .$$

(iii) Moreover, for any $x \in \mathbb{R}^d$, we have $\rho^c_{\mathrm{T}}(x) = \int \varpi^{\mathrm{c}}(t)\Lambda(\phi_t(x))\mathrm{J}_{\phi_t}(x)\mathrm{d}t > 0$.

Note that **H**3 implies that $\sup_{t\in\mathbb{R}} |\varpi^{\mathrm{c}}(t)| < +\infty$. **H**3 is automatically satisfied for example in the case $\varpi^{\mathrm{c}} = \mathbb{1}_{[-T_1,T_2]}$ for $T_1, T_2 \geqslant 0$.

Theorem 39. *Assume **H**1, **H**2, **H**3. For any $x \in \mathbb{R}^d$ and $f : \mathbb{R}^d \to \mathbb{R}$ continuous and bounded,*

$$\lim_{h\downarrow 0} \left| \sum_{k\in\mathbb{Z}} w_{k,h}(x) f(\mathrm{T}_h^k(x)) - \int_{-\infty}^{\infty} w_t^{\mathrm{c}}(x) f(\phi_t(x)) \mathrm{d}t \right| = 0 \,,$$

where $\{w_{k,h}\}_{k\in\mathbb{Z}}$ and w_t^{c} are defined in (5.33) *and* (5.17) *respectively, i.e. for $x \in \mathbb{R}^d$ and $t \in \mathbb{R}$,*

$$w_t^{\mathrm{c}}(x) = \varpi^{\mathrm{c}}(t)\rho(\phi_t(x))\mathrm{J}_{\phi_t}(x) \Big/ \int_{-\infty}^{\infty} \varpi^{\mathrm{c}}(s+t)\rho(\phi_s(x))\mathrm{J}_{\phi_s}(x)\mathrm{d}s \ . \tag{5.37}$$

Proof. Let f be a bounded continuous function, $x \in \mathbb{R}^d$. Setting

$$\begin{aligned} g_{k,h}(x) &= \rho(\mathrm{T}_h^k(x))\varpi^{\mathrm{c}}(kh)\mathrm{J}_{\mathrm{T}_h^k}(x) f(\mathrm{T}_h^k(x)) \\ h\Delta_{k,h}(x) &= h\sum_{i\in\mathbb{Z}} \Lambda(\mathrm{T}_h^i(x))\varpi^{\mathrm{c}}((k+i)h)\mathrm{J}_{\mathrm{T}_h^i(x)} \,, \end{aligned}$$

we have that

$$\sum_{k\geqslant 0} \frac{h g_{k,h}(x)}{h\Delta_{k,h}(x)} = \int_0^{T_\varpi} \frac{1}{h\Delta_{\lfloor t/h\rfloor,h}(x)} g_{\lfloor t/h\rfloor,h}(x)\mathrm{d}t + \int_{T_\varpi}^{h\lfloor T_\varpi/h\rfloor+h} \frac{1}{h\Delta_{\lfloor t/h\rfloor,h}(x)} g_{\lfloor t/h\rfloor,h}(x)\mathrm{d}t \,,$$

as $g_{k,h}(x) = 0$ when $k > \lfloor T_\varpi/h\rfloor$. Therefore, we can consider the following decomposition,

$$\left| \sum_{k\geqslant 0} \frac{\rho(\mathrm{T}_h^k(x))\varpi^{\mathrm{c}}(kh)\mathrm{J}_{\mathrm{T}_h^k}(x) f(\mathrm{T}_h^k(x))}{\sum_{i\in\mathbb{Z}} \Lambda(\mathrm{T}_h^i(x))\varpi^{\mathrm{c}}((k+i)h)\mathrm{J}_{\mathrm{T}_h^i(x)}} - \int_0^{T_\varpi} \frac{\varpi^{\mathrm{c}}(t)\Lambda(\phi_t(x))\mathrm{J}_{\phi_t}(x) f(\phi_t(x))\mathrm{d}t}{\int \varpi^{\mathrm{c}}(t+s)\Lambda(\phi_s(x))\mathrm{J}_{\phi_s}(x)\mathrm{d}s} \right| \leqslant A + B$$

with

$$\begin{aligned} A = \left| \int_0^{T_\varpi} \frac{1}{h\Delta_{\lfloor t/h\rfloor,h}(x)} \left\{ g_{\lfloor t/h\rfloor,h}(x) - \varpi^{\mathrm{c}}(t)\Lambda(\phi_t(x))\mathrm{J}_{\phi_t}(x) f(\phi_t(x)) \right\} \mathrm{d}t \right| \\ + \left| \int_{T_\varpi}^{h\lfloor T_\varpi/h\rfloor+h} \frac{1}{h\Delta_{\lfloor t/h\rfloor,h}(x)} g_{\lfloor t/h\rfloor,h}(x)\mathrm{d}t \right| \,, \end{aligned}$$

and

$$B = \int_0^{T_\varpi} \left| \frac{\varpi^{\mathrm{c}}(t)\Lambda(\phi_t(x))\mathrm{J}_{\phi_t}(x) f(\phi_t(x))\mathrm{d}t}{h\Delta_{\lfloor t/h\rfloor,h}(x)} - \frac{\varpi^{\mathrm{c}}(t)\Lambda(\phi_t(x))\mathrm{J}_{\phi_t}(x) f(\phi_t(x))}{\int \varpi^{\mathrm{c}}(t+s)\Lambda(\phi_s(x))\mathrm{J}_{\phi_s}(x)\mathrm{d}s} \right| \mathrm{d}t \,,$$

We bound those terms separately. First of all, under **H**3-(ii), for any k such that $kh \in [0, T_\varpi]$, we have $h\Delta_{k,h}(x) \geqslant hm^{-1}\Delta_{0,h}(x)$. Second, as $\lim_{h\downarrow 0} h\Delta_{0,h}(x) = \int_0^{T_\varpi} \Lambda(\phi_s(x))\mathrm{J}_{\phi_s}(x)\varpi^{\mathrm{c}}(s)\mathrm{d}s > 0$, there exists some $\tilde{h} > 0$ and $c > 0$ such that for all $k \in \mathbb{Z}$, $h < \tilde{h}$ implies

$$\int_0^{T_\varpi} \varpi^{\mathrm{c}}(t)\Lambda(\phi_t(x))\mathrm{J}_{\phi_t}(x)\mathrm{d}t > c \,, \quad h\Delta_{k,h}(x) \geqslant hm^{-1}\Delta_{0,h}(x) > c \,. \tag{5.38}$$

Then, for $h < \tilde{h}$,

$$\begin{aligned} A \leqslant c^{-1} \int_0^{T_\varpi} |g_{\lfloor t/h\rfloor,h}(x) - \varpi^{\mathrm{c}}(t)\Lambda(\phi_t(x))\mathrm{J}_{\phi_t}(x) f(\phi_t(x))|\mathrm{d}t \\ + c^{-1} \int_{T_\varpi}^{h\lfloor T_\varpi/h\rfloor+h} \left| g_{\lfloor t/h\rfloor,h}(x) \right| \mathrm{d}t \,. \end{aligned}$$

By **H**1 and **H**3, the function $t \to \varpi^{\mathrm{c}}(t)\Lambda(\phi_t(x))\mathrm{J}_{\phi_t}(x) f(\phi_t(x))$ is continuous on the compact $[0, 2T_\varpi]$ and thus is bounded. Therefore, for any $h \in \big(0, \bar{h}\big)$,

$$\sup_{t\in[0,2T_\varpi]} |\varpi^{\mathrm{c}}(t)\Lambda(\phi_t(x))\mathrm{J}_{\phi_t}(x) f(\phi_t(x))| \leqslant \sup_{t\in\mathbb{R}} |\varpi^{\mathrm{c}}| \sup_{x\in\mathbb{R}^d} |f(x)| \sup_{t\in[0,2T_\varpi]} |\Lambda(\phi_t(x))\mathrm{J}_{\phi_t}(x)| < \infty \,. \tag{5.39}$$

Under **H**2, (5.39) and Lemma 43 imply that

$$\sup_{t\in[0,h\lfloor T_\varpi/h\rfloor+h)} g_{\lfloor t/h\rfloor,h}(x) \leqslant \sup_{t\in\mathbb{R}}|\varpi^{\mathrm{c}}(t)|\sup_{x\in\mathbb{R}^d}|f(x)|\sup_{t\in[0,h\lfloor T_\varpi/h\rfloor+h)}\rho(\mathrm{T}_h^{\lfloor t/h\rfloor}(x))\mathrm{J}_{\mathrm{T}_h^{\lfloor t/h\rfloor}}(x)<\infty\,,$$

Then, $\lim_{h\downarrow 0}\int_{T_\varpi}^{h\lfloor T_\varpi/h\rfloor+h}\left|g_{\lfloor t/h\rfloor,h}(x)\right|\mathrm{d}t=0$. Finally, Lemma 44 implies that $\lim_{h\downarrow 0}A=0$.

Moreover, setting for $t\in[0,T_\varpi]$,

$$\begin{aligned}&\Delta_{t,h}^B(x) \qquad (5.40)\\ &=\int|\Lambda(\phi_{h\lfloor s/h\rfloor}(x))\varpi^{\mathrm{c}}(h(\lfloor s/h\rfloor+\lfloor t/h\rfloor))\mathrm{J}_{\phi_{h\lfloor s/h\rfloor}(x)}-\varpi^{\mathrm{c}}(s+t)\Lambda(\phi_s(x))\mathrm{J}_{\phi_s}(x))|\mathbb{1}_{\mathsf{A}_\varpi}(s,t)\mathrm{d}s\\ &\quad+\int_{T_\varpi-h\lfloor t/h\rfloor}^{h(\lfloor T_\varpi/h\rfloor-\lfloor t/h\rfloor+1)}|\Lambda(\phi_{h\lfloor s/h\rfloor}(x))\varpi^{\mathrm{c}}(h(\lfloor s/h\rfloor+\lfloor t/h\rfloor))\mathrm{J}_{\phi_{h\lfloor s/h\rfloor}(x)}|\mathbb{1}_{\mathsf{A}_\varpi}(s,t)\mathrm{d}s\,,\end{aligned}$$

we have for $h<\tilde{h}$, by (5.38) and **H**3-(ii),

$$\begin{aligned}B&=\int_0^{T_\varpi}\left|\frac{\varpi^{\mathrm{c}}(t)\Lambda(\phi_t(x))\mathrm{J}_{\phi_t}(x)f(\phi_t(x))}{h\Delta_{\lfloor t/h\rfloor,h}(x)}-\frac{\varpi^{\mathrm{c}}(t)\Lambda(\phi_t(x))\mathrm{J}_{\phi_t}(x)f(\phi_t(x))}{\int\varpi^{\mathrm{c}}(s+t)\Lambda(\phi_s(x))\mathrm{J}_{\phi_s}(x)\mathrm{d}s}\right|\mathrm{d}t\\ &\leqslant\int_0^{T_\varpi}\frac{\varpi^{\mathrm{c}}(t)\Lambda(\phi_t(x))\mathrm{J}_{\phi_t}(x)f(\phi_t(x))}{h\Delta_{\lfloor t/h\rfloor,h}(x)\int\varpi^{\mathrm{c}}(s+t)\Lambda(\phi_s(x))\mathrm{J}_{\phi_s}(x)\mathrm{d}s}\Delta_{t,h}^B(x)\mathrm{d}t\\ &\leqslant mc^{-2}\int_0^{T_\varpi}\varpi^{\mathrm{c}}(t)\Lambda(\phi_t(x))\mathrm{J}_{\phi_t}(x)f(\phi_t(x))\Delta_{t,h}^B(x)\mathrm{d}t\\ &\leqslant mc^{-2}\sup_{t\in\mathbb{R}}|\varpi^{\mathrm{c}}(t)|\sup_{x\in\mathbb{R}^d}|f(x)|\sup_{t\in[0,T_\varpi]}|\Lambda(\phi_s(x))\mathrm{J}_{\phi_s}(x)|\int_0^{T_\varpi}\Delta_{t,h}^B(x)\mathrm{d}t\,. \qquad (5.41)\end{aligned}$$

By **H**1 and **H**3, the function $s\to\Lambda(\phi_s(x))\mathrm{J}_{\phi_s}(x)$ is continuous on the interval $[-T_\varpi,T_\varpi]$ and thus is bounded. Therefore, for any $h\in\left(0,\bar{h}\right)$,

$$\begin{aligned}&\sup_{(s,t)\in\mathsf{A}_\varpi}|\varpi^{\mathrm{c}}(h(\lfloor t/h\rfloor+\lfloor s/h\rfloor))\Lambda(\phi_{h\lfloor s/h\rfloor}(x))\mathrm{J}_{\phi_{h\lfloor s/h\rfloor}}(x)|\\ &\quad\leqslant\sup_{(s,t)\in\mathsf{A}_\varpi}|\varpi^{\mathrm{c}}(s+t)\Lambda(\phi_s(x))\mathrm{J}_{\phi_s}(x)|<T_\varpi\sup_{s\in\mathbb{R}}|\varpi^{\mathrm{c}}(s)|\sup_{s\in[-T_\varpi,T_\varpi]}|\Lambda(\phi_s(x))\mathrm{J}_{\phi_s}(x)|<\infty\,. \qquad (5.42)\end{aligned}$$

This implies that

$$\lim_{h\downarrow0}\int_{T_\varpi-h\lfloor t/h\rfloor}^{h(\lfloor T_\varpi/h\rfloor-\lfloor t/h\rfloor+1)}|\Lambda(\phi_{h\lfloor s/h\rfloor}(x))\varpi^{\mathrm{c}}(h(\lfloor s/h\rfloor+\lfloor t/h\rfloor))\mathrm{J}_{\phi_{h\lfloor s/h\rfloor}(x)}|\mathrm{d}s=0\,.$$

Moreover, for any $t\in[0,T_\varpi]$, the function

$$s\mapsto|\varpi^{\mathrm{c}}(h(\lfloor t/h\rfloor+\lfloor s/h\rfloor))\Lambda(\phi_{h\lfloor s/h\rfloor}(x))\mathrm{J}_{\phi_{h\lfloor s/h\rfloor}}(x)-\varpi^{\mathrm{c}}(t+s)\Lambda(\phi_s(x))\mathrm{J}_{\phi_s}(x)|\mathbb{1}_{\mathsf{A}_\varpi}(s,t)$$

converges pointwise to 0 for almost all $s\in\mathbb{R}$ when $h\downarrow0$ using **H**1, **H**3 and the continuity of $s\mapsto\phi_s(x)$. The Lebesgue dominated convergence theorem applies and by (5.40), for all $t\in[0,T_\varpi]$,

$$\lim_{h\downarrow0}\Delta_{t,h}^B(x)=0\,.$$

Moreover, using $h\Delta_{k,h}(x)=h\sum_{i\in\mathbb{Z}}\Lambda(\mathrm{T}_h^i(x))\varpi^{\mathrm{c}}((k+i)h)\mathrm{J}_{\mathrm{T}_h^i(x)}$ and (5.42),

$$\sup_{t\in[0,T_\varpi]}\sup_{h\in\left(0,\bar{h}\right)}\Delta_{t,h}^B(x)<\infty\,.$$

The Lebesgue dominated convergence theorem and (5.41) show that $\lim_{h\downarrow0}B=0$ which concludes the proof. □

Supporting Lemmas

For $f \in \mathrm{C}^1(\mathbb{R}^d, \mathbb{R}^d)$, define $\mathfrak{J}_f(x)$ the Jacobian matrix of f evaluated at x and the divergence operator by $\mathrm{D}\, f(x) = \mathrm{Tr}[\mathfrak{J}_f(x)]$.

Lemma 40. *Let b be a C^1 vector field in $\mathbb{R}^d$ and $(\phi_t)_{t\in\mathbb{R}}$ be the flow of the ODE (5.34). For any $t \in \mathbb{R}$, the Jacobian of ϕ_t is given by*

$$\mathrm{J}_{\phi_t}(x) = \exp(\textstyle\int_0^t \mathrm{D}\, b(\phi_s(x))\mathrm{d}s) \; .$$

Proof. First, for $t \in \mathbb{R}$ and $x \in \mathbb{R}$, write $A(t,x) = \mathfrak{J}_{\phi_t}(x)$ the Jacobian matrix of ϕ_t evaluated at x. By Jacobi's formula, $\dot{\det} A(t,x) = \mathrm{Tr}[\mathrm{adj}(A(t,x)) \cdot \dot{A}(t,x)]$, where $\mathrm{Tr}[M]$ denotes the trace of a matrix M and $\mathrm{adj}(M)$ its adjugate, i.e. the transpose of the cofactor matrix of M such that $\mathrm{adj}(M)M = \det(M)\,\mathrm{Id}$. Since for all t and x, $\dot{A}(t,x) = \mathfrak{J}_{b\circ\phi_t}(x) = \mathfrak{J}_b(\phi_t(x)) \cdot A(t,x)$, then

$$\dot{\mathrm{J}}_{\phi_t}(x) = \mathrm{Tr}[\mathrm{adj}(A(t,x)) \cdot \mathfrak{J}_b(\phi_t(x)) \cdot A(t,x)] = \mathrm{Tr}[\mathfrak{J}_b(\phi_t(x))]\mathrm{J}_{\phi_t}(x) \; . \tag{5.43}$$

Integrating this ODE yields $\mathrm{J}_{\phi_t}(x) = \exp(\int_0^t \mathrm{D}\, b(\phi_s(x))\mathrm{d}s)$. □

Lemma 41. *Assume **H**1. Then, there exists $C > 0$ such that for any $x \in \mathbb{R}^d, t \in \mathbb{R}$, $k \in \mathbb{Z}$, $h > 0$,*

$$\begin{aligned} \|\phi_t(x)\| &\leqslant Ce^{C|t|}(\|x\| + 1) \; , \\ \|\,\mathrm{T}_h^k(x)\| &\leqslant Ce^{C|kh|}(\|x\| + 1) \; . \end{aligned}$$

This lemma follows from Gronwall's inequality and **H**1.

Lemma 42. *Assume **H**1 and **H**2-(i). There exists $C > 0$ such that for any $x \in \mathbb{R}^d, h \in \big(0, \bar{h}\big)$,*

$$\|\,\mathrm{T}_h(x) - \phi_h(x)\| \leqslant C\{1 + \|x\|\}\|h^{1+\delta} \; . \tag{5.44}$$

Proof. Under **H**1 and **H**2-(i), we have

$$\|\,\mathrm{T}_h(x) - \phi_h(x)\| \leqslant \|x + hb(x) - \phi_h(x)\| + C_F h^{1+\delta}(1 + \|x\|) \; ,$$

and as $\phi_h(x) = x + \int_0^h b(\phi_s(x))\mathrm{d}s$,

$$\begin{aligned} \|x + hb(x) - \phi_h(x)\| = \|hb(x) - \textstyle\int_0^h b(\phi_s(x))\| &\leqslant hL_b \sup_{s\in[0,h]} \|\phi_s(x) - x\| \\ &\leqslant L_b h^2\{L_b \sup_{s\in[0,h]} \phi_s(x) + \|b(0)\|\} \; . \end{aligned} \tag{5.45}$$

The proof is completed using Lemma 41. □

Lemma 43. *Assume **H**1 and **H**2-(i). There exists $C > 0$ such that for any $x \in \mathbb{R}^d, k \in \mathbb{N}, h \in \big(0, \bar{h}\big)$, $kh \leqslant T_\varpi$,*

$$\|\,\mathrm{T}_h^k(x) - \phi_{kh}(x)\| \leqslant Ce^{khC}(1 + \|x\|)h^\delta \; . \tag{5.46}$$

Proof. Using Lemma 42, **H**1 and **H**2-(i), there exist $C_1, C_2, C_3 > 0$ such that for any $x \in \mathbb{R}^d, k \in \mathbb{N}, h \in \big(0, \bar{h}\big)$, $kh \leqslant T_\varpi$,

$$\begin{aligned} &\|\,\mathrm{T}_h^{k+1}(x) - \phi_{(k+1)h}(x)\| \leqslant \|\,\mathrm{T}_h^{k+1}(x) - \mathrm{T}_h \circ\phi_{kh}(x)\| + \|\,\mathrm{T}_h \circ\phi_{kh}(x) - \phi_{(k+1)h}(x)\| \\ &\leqslant (1 + hL_b)\|\,\mathrm{T}_h^k(x) - \phi_{kh}(x)\| \\ &\qquad + h^{1+\delta}C_1\{2 + \|\,\mathrm{T}_h^k(x)\| + \|\phi_{kh}(x)\|\} + \|\,\mathrm{T}_h \circ\phi_{kh}(x) - \phi_{(k+1)h}(x)\| \\ &\leqslant (1 + hL_b)\|\,\mathrm{T}_h^k(x) - \phi_{kh}(x)\| + h^{1+\delta}2C_1C_2e^{C_2T_\varpi}\{1 + \|x\|\} + C_3\{1 + \|\phi_{kh}(x)\|\}h^{1+\delta} \\ &\leqslant (1 + hL_b)\|\,\mathrm{T}_h^k(x) - \phi_{kh}(x)\| \\ &\qquad + h^{1+\delta}2C_1C_2e^{C_2T_\varpi}\{1 + \|x\|\} + C_3\{1 + C_2(1 + \|x\|)\}h^{1+\delta}e^{C_2T_\varpi} \\ &\leqslant (1 + hL_b)\|\,\mathrm{T}_h^k(x) - \phi_{kh}(x)\| + A_T\{1 + \|x\|\}h^{1+\delta} \; , \end{aligned}$$

with $A_T = (2C_1C_2 + C_3(1+C_2))e^{C_2T_\varpi}$. A straightforward induction yields

$$\| \mathrm{T}_h^k(x) - \phi_{kh}(x)\| \leqslant \frac{(1+hL_b)^k}{L_b} A_T(1+\|x\|)h^\delta \ .$$

□

Lemma 44. *Assume **H**1, **H**2, **H**3 . For any $x \in \mathbb{R}^d$, and $f : \mathbb{R}^d \to \mathbb{R}^d$ bounded and continuous,*

$$\lim_{h \downarrow 0} \int_0^{T_\varpi} \left| \varpi^{\mathrm{c}}(h\lfloor t/h \rfloor)\Lambda(\mathrm{T}_h^{\lfloor t/h \rfloor}(x))\mathrm{J}_{\mathrm{T}_h^{\lfloor t/h \rfloor}}(x) f(\mathrm{T}_h^{\lfloor t/h \rfloor}(x)) - \varpi^{\mathrm{c}}(t)\Lambda(\phi_t(x))\mathrm{J}_{\phi_t}(x) f(\phi_t(x)) \right| \mathrm{d}t = 0 \ .$$

Proof. Let $x \in \mathbb{R}^d$. Consider the following decomposition, for any $h < \bar{h}$,

$$\begin{aligned}
&\int_0^{T_\varpi} \left| \varpi^{\mathrm{c}}(h\lfloor t/h \rfloor)\Lambda(\mathrm{T}_h^{\lfloor t/h \rfloor}(x))\mathrm{J}_{\mathrm{T}_h^{\lfloor t/h \rfloor}}(x) f(\mathrm{T}_h^{\lfloor t/h \rfloor}(x)) - \varpi^{\mathrm{c}}(t)\Lambda(\phi_t(x))\mathrm{J}_{\phi_t}(x) f(\phi_t(x)) \right| \mathrm{d}t \\
&\leqslant \tfrac{h}{T_\varpi} \textstyle\sum_{k \in \mathbb{Z}} \varpi^{\mathrm{c}}(kh) |\Lambda(\mathrm{T}_h^k(x))\mathrm{J}_{\mathrm{T}_h^k}(x) f(\mathrm{T}_h^k(x)) - \Lambda(\phi_{kh}(x))\mathrm{J}_{\phi_{kh}}(x) f(\phi_{kh}(x))| \\
&+ \textstyle\int_0^{T_\varpi} |\varpi^{\mathrm{c}}(t)\Lambda(\phi_t(x))\mathrm{J}_{\phi_t}(x) f(\phi_t(x)) - \varpi^{\mathrm{c}}(h\lfloor t/h \rfloor)\Lambda(\phi_{h\lfloor t/h \rfloor}(x))\mathrm{J}_{\phi_{h\lfloor t/h \rfloor}}(x) f(\phi_{h\lfloor t/h \rfloor}(x))| \mathrm{d}t \ .
\end{aligned}$$

The first term converges to 0 by Lemma 43 and **H**2-(ii) as $\varpi^{\mathrm{c}}(kh) = 0$ for $kh > T_\varpi$. By **H**1 and **H**3, the function $t \to \varpi^{\mathrm{c}}(t)\Lambda(\phi_t(x))\mathrm{J}_{\phi_t}(x) f(\phi_t(x))$ is continuous on the compact $[0, T_\varpi]$ and thus is bounded. Therefore, for any $h \in \big(0, \bar{h}\big)$,

$$\begin{aligned}
&\sup_{t \in [0, T_\varpi]} |\varpi^{\mathrm{c}}(h\lfloor t/h \rfloor)\Lambda(\phi_{h\lfloor t/h \rfloor}(x))\mathrm{J}_{\phi_{h\lfloor t/h \rfloor}}(x) f(\phi_{h\lfloor t/h \rfloor}(x))| \\
&\qquad \leqslant \sup_{t \in \mathbb{R}} |\varpi^{\mathrm{c}}| \sup_{x \in \mathbb{R}^d} |f(x)| \sup_{t \in [0, T_\varpi]} |\Lambda(\phi_t(x))\mathrm{J}_{\phi_t}(x)| < \infty \ . \quad (5.47)
\end{aligned}$$

Moreover, $t \mapsto \varpi^{\mathrm{c}}(h\lfloor t/h \rfloor)\Lambda(\phi_{h\lfloor t/h \rfloor}(x))\mathrm{J}_{\phi_{h\lfloor t/h \rfloor}}(x) f(\phi_{h\lfloor t/h \rfloor}(x))$ converges pointwise when $h \downarrow 0$ to $t \to \varpi^{\mathrm{c}}(t)\Lambda(\phi_t(x))\mathrm{J}_{\phi_t}(x) f(\phi_t(x))$ by continuity, using **H**1 and **H**3. The Lebesgue dominated convergence theorem applies and the second term goes to 0 as $h \downarrow 0$. □

5.8.2 NEIS algorithm after [RV19]

Non Equilibrium Importance Sampling (NEIS) has been introduced in the pioneering work of [RV19]. NEIS relies on the flow of the ODE $\dot{x}_t = b(x_t)$ and the introduction of a set $\mathsf{o} \subset \mathbb{R}^d$. As in Section 5.8, we assume **H**1 holds and denote by $(\phi_t)_{t \in \mathbb{R}}$ the flow of this ODE.

Define for $x \in \mathsf{o}$, the exit times $\tau^+(x) \geqslant 0$ (resp. $\tau^-(x) \leqslant 0$) satisfying

$$\tau^+(x) = \inf\{t \geqslant 0 \, : \, \phi_t(x) \notin \mathsf{o}\} \ , \ \tau^-(x) = \inf\{t \leqslant 0 \, : \, \phi_t(x) \notin \mathsf{o}\} \ . \quad (5.48)$$

The validity of NEIS relies on the following assumption.

H4. *The average time of an orbit in o is finite, i.e.*

$$\mathrm{Z}_\tau = \int_{\mathsf{o}} (\tau^+(x) - \tau^-(x))\Lambda(x)\mathrm{d}x < \infty \ . \quad (5.49)$$

Under **H**4, we can define the proposal distribution

$$\rho_{\mathrm{T}}(x) = \mathrm{Z}_\tau^{-1} \int_{\mathsf{o}} \mathbb{1}_{[\tau^-(x), \tau^+(x)]}(t)\rho(\phi_t(x))\mathrm{J}_{\phi_t}(x)\mathrm{d}t \ . \quad (5.50)$$

Under **H**4, [RV19], Equation (8), derive the following estimator of $\rho(f)$, closely related to (5.16), in the case $\varpi \equiv 1$, on the restricted set $\mathsf{o} \subset \mathbb{R}^d$:

$$I_N^{\mathrm{NEIS}}(f) = \frac{1}{N} \sum_{i=1}^N \int_{\tau^-(X^i)}^{\tau^+(X^i)} w_t(X^i) f(\phi_t(X^i))\mathrm{d}t \quad (5.51)$$

$$w_t(x) = \frac{\rho(\phi_t(x))\mathrm{J}_{\phi_t(x)}}{\int_{\tau^-(x)}^{\tau^+(x)} \rho(\phi_t(x))\mathrm{J}_{\phi_t(x)}\mathrm{d}t} \ . \quad (5.52)$$

Note that in practice, in order for **H** 4 to be verified, one typically requires that o be bounded, as discussed in [RV19].

Following [RV19], consider a d-dimensional system with position $q \in \mathbb{R}^d$, momentum $p \in \mathbb{R}^d$ and Hamiltonian $H(p,q) = (1/2)\|p\|^2 + U(q)$ where $U(q)$ is a potential assumed to be bounded from below. Denote by $V(E)$ the volume of the phase-space below some threshold energy E,

$$V(E) = \int \mathbb{1}_{\{H(p,q)\leqslant E\}} \mathrm{d}p\mathrm{d}q \; . \tag{5.53}$$

To calculate (5.53), we set $x = (p,q)$, define $\mathsf{o} = \{x; H(x) \leqslant E_{\max}\}$ for some $E_{\max} < \infty$, and use the dissipative Langevin dynamics with $b(x) = (p, -\nabla U(q) - \gamma p)$, *i.e.*

$$\dot{q} = p \; , \quad \dot{p} = -\nabla U(q) - \gamma p \; ,$$

for some friction coefficient $\gamma > 0$. With this choice, $\mathrm{J}_{\phi_t}(x) = \mathrm{e}^{-d\gamma t}$. Taking Λ to be the uniform distribution on the (bounded) set o, write the estimator for $E \leqslant E_{\max}$, $V(E)/V(E_{\max}) = \int \mathbb{1}_{\{H(p,q)\leqslant E\}} \Lambda(p,q)\mathrm{d}p\mathrm{d}q$, where $\Lambda(p,q) = \mathbb{1}_{\mathsf{o}}(p,q)/V(E_{\max})$, we get

$$\begin{aligned} V(E)/V(E_{\max}) &= \frac{1}{N}\sum_{i=1}^{N} \frac{\int_{\tau^-(X^i)}^{\tau^+(X^i)} \mathrm{J}_{\phi_t(X^i)} \mathbb{1}_{\{H(\phi_t(X^i))\leqslant E\}} \mathrm{d}t}{\int_{\tau^-(X^i)}^{\tau^+(X^i)} \mathrm{J}_{\phi_t(X^i)} \mathrm{d}t} \\ &= \frac{1}{N}\sum_{i=1}^{N} \frac{\int_{\tau^E(X^i)}^{\tau^+(X^i)} \mathrm{J}_{\phi_t(X^i)} \mathrm{d}t}{\int_{\tau^-(X^i)}^{\tau^+(X^i)} \mathrm{J}_{\phi_t(X^i)} \mathrm{d}t} = \frac{1}{N}\sum_{i=1}^{N} \mathrm{e}^{-d\gamma(\tau^E(X^i) - \tau^-(X^i))} \; , \end{aligned} \tag{5.54}$$

where $\tau^E(x)$ denotes the (possibly infinite) time for a trajectory initiated at $x = (p,q)$ to reach the energy $E \leqslant E_{\max}$.

Finally, to estimate the normalizing constant, [RV19] discretize the energy levels $\{E_0, \ldots, E_P\}$ and write their estimator as

$$\widehat{\mathrm{Z}}^{\mathrm{NEIS}}_{X^{1:N}} = \frac{1}{N}\sum_{i=1}^{N}\sum_{\ell=1}^{P} \mathrm{e}^{-d\gamma(\tau^E_\ell(X^i) - \tau^-(X^i))}(E_\ell - E_{\ell-1}) \; , \tag{5.55}$$

using an approximation of the identity

$$\mathrm{Z} = \int_{\mathsf{o}}\int_0^\infty \mathbb{1}_{\{\mathrm{L}(x)>L\}} \rho(x)\mathrm{d}L\mathrm{d}x = \int_0^\infty \mathbb{P}_{X\sim\rho}(\mathrm{L}(X) > L)\mathrm{d}L \; ,$$

which is at the core of nested sampling [CR10].

5.8.3 NEO with exit times

Consider $\mathsf{o} \subset \mathbb{R}^d$ and let T be a C^1-diffeomorphism on $\mathbb{R}^d$. We introduce here an estimator based on the forward and backward orbits in o associated with T. Define the exit times $\tau^+ : \mathbb{R}^d \to \mathbb{N}$ and $\tau^- : \mathbb{R}^d \to \mathbb{N}_-$, given, for all $x \in \mathbb{R}^d$, by

$$\tau^+(x) = \inf\{k \geqslant 1 \; : \; \mathrm{T}^k(x) \notin \mathsf{o}\} \; , \tag{5.56}$$

$$\tau^-(x) = \sup\{k \leqslant -1 \; : \; \mathrm{T}^k(x) \notin \mathsf{o}\} \; , \tag{5.57}$$

with the convention $\inf\emptyset = +\infty$ and $\sup\emptyset = -\infty$, and set

$$\mathrm{I} = \{(x,k) \in \mathsf{o}\times\mathbb{Z} \; : \; k \in [\tau^-(x)+1 : \tau^+(x)-1]\} \; . \tag{5.58}$$

For any $k \in \mathbb{Z}$, define $\rho_k : \mathbb{R}^d \to \mathbb{R}_+$ by

$$\rho_k(x) = \rho(\mathrm{T}^{-k}(x))\mathrm{J}_{\mathrm{T}^{-k}}(x)\mathbb{1}_{\mathrm{I}}(x,-k) \; . \tag{5.59}$$

The density ρ_k is the push-forward of $\mathbb{1}_{\mathrm{I}}(x,k)\rho(x)$ by T^k, *i.e.* for any $k \in \mathbb{Z}$ and any bounded function $g : \mathbb{R}^d \to \mathbb{R}$,

$$\int_{\mathsf{o}} g(y)\rho_k(y)\mathrm{d}y = \int_{\mathsf{o}} g(\mathrm{T}^k(x))\mathbb{1}_{\mathrm{I}}(x,k)\rho(x)\mathrm{d}x \; . \tag{5.60}$$

Consider the following assumption:

H5. *The nonnegative sequence $(\varpi_k)_{k\in\mathbb{Z}}$ satisfies $\varpi_0 > 0$ and*

$$\mathrm{Z}_{\mathrm{T}}^{\varpi} = \int_{\mathsf{o}} \sum_{k\in\mathbb{Z}} \varpi_k \rho_k(x)\mathrm{d}x = \int_{\mathsf{o}} \sum_{k\in\mathbb{Z}} \varpi_k \rho(\mathrm{T}^k(x))\mathrm{J}_{\mathrm{T}^k}(x)\mathbb{1}_{\mathrm{I}}(x,k)\mathrm{d}x < \infty \; . \tag{5.61}$$

Consider the pdf

$$\rho_{\mathrm{T}}(x) = \frac{1}{\mathrm{Z}_{\mathrm{T}}^{\varpi}} \sum_{k\in\mathbb{Z}} \varpi_k \rho_k(x) \; , \tag{5.62}$$

where $\mathrm{Z}_{\mathrm{T}}^{\varpi}$ is the normalizing constant. This is a *non-equilibrium* distribution, since ρ_{T} is not invariant by T in general. Using ρ_{T} as an importance distribution to obtain an unbiased estimator of $\int f(x)\Lambda(x)\mathrm{d}x$ is feasible since as $\varpi_0 > 0$, $\sup_{x\in\mathsf{o}} \Lambda(x)/\rho_{\mathrm{T}}(x) \leqslant \mathrm{Z}_{\mathrm{T}}/\varpi_0 < \infty$, hence

$$\int_{\mathsf{o}} f(x)\rho(x)\mathrm{d}x = \int_{\mathsf{o}} \left(f(x)\frac{\rho(x)}{\rho_{\mathrm{T}}(x)} \right) \rho_{\mathrm{T}}(x)\mathrm{d}x \; .$$

From (5.60), the right hand side can be computed using the following key result.

Theorem 45. *For any $f : \mathbb{R}^d \to \mathbb{R}$ measurable bounded function, we have*

$$\int_{\mathsf{o}} f(x)\rho(x)\mathrm{d}x = \int_{\mathsf{o}} \sum\nolimits_{k\in\mathbb{Z}} f(\mathrm{T}^k(x))w_k(x)\rho(x)\mathrm{d}x \; , \tag{5.63}$$

where, for any $x \in \mathbb{R}^d$ and $k \in \mathbb{Z}$,

$$w_k(x) = \varpi_k \rho_{-k}(x) \Big/ \sum\nolimits_{j\in\mathbb{Z}} \varpi_{j+k}\rho_j(x) \quad . \tag{5.64}$$

Proof. Let $f : \mathbb{R}^d \to \mathbb{R}$ be a measurable bounded function. By (5.60), writing $g \leftarrow f\rho/\rho_{\mathrm{T}}$,

$$\begin{aligned}\int_{\mathsf{o}} f(x)\rho(x)\mathrm{d}x = \int_{\mathsf{o}} \left(f(x)\frac{\rho(x)}{\rho_{\mathrm{T}}(x)} \right) \rho_{\mathrm{T}}(x)\mathrm{d}x \\ = \int_{\mathsf{o}} \sum_{k\in\mathbb{Z}} \left(f(\mathrm{T}^k(x)) \frac{\varpi_k \rho(\mathrm{T}^k(x))\mathbb{1}_{\mathrm{I}}(x,k)}{\mathrm{Z}_{\mathrm{T}}^{\varpi}\rho_{\mathrm{T}}(\mathrm{T}^k(x))} \right) \rho(x)\mathrm{d}x \; .\end{aligned}$$

We now need to prove:

$$\frac{\varpi_k \rho(\mathrm{T}^k(x))\mathbb{1}_{\mathrm{I}}(x,k)}{\mathrm{Z}_{\mathrm{T}}^{\varpi}\rho_{\mathrm{T}}(\mathrm{T}^k(x))} = \frac{\varpi_k \rho(\mathrm{T}^k(x))\mathbb{1}_{\mathrm{I}}(x,k)}{\mathbb{1}_{\mathrm{I}}(x,k)\sum_{i\in\mathbb{Z}} \varpi_i \rho_i(\mathrm{T}^k(x))} = \frac{\varpi_k \rho_{-k}(x)}{\sum_{j\in\mathbb{Z}} \varpi_{j+k}\rho_j(x)} = w_k(x) \quad ,$$

with the convention $0/0 = 0$. We thus need to show that for any $x \in \mathsf{o}$, $k \in \mathbb{Z}$,

$$\mathbb{1}_{\mathrm{I}}(x,k) \sum_{i\in\mathbb{Z}} \varpi_i \rho_i(\mathrm{T}^k(x)) = \frac{\mathbb{1}_{\mathrm{I}}(x,k)}{\mathrm{J}_{\mathrm{T}^k}(x)} \sum_{j\in\mathbb{Z}} \varpi_{j+k}\rho_j(x) \; .$$

Using the identity $\mathrm{J}_{\mathrm{T}^{-i+k}}(x) = \mathrm{J}_{\mathrm{T}^{-i}}(\mathrm{T}^k(x))\mathrm{J}_{\mathrm{T}^k}(x)$, we obtain

$$\begin{aligned}\mathbb{1}_{\mathrm{I}}(x,k) \sum_{i\in\mathbb{Z}} \varpi_i \rho_i(\mathrm{T}^k(x)) &= \sum_{i\in\mathbb{Z}} \mathbb{1}_{\mathrm{I}}(x,k)\varpi_i \rho(\mathrm{T}^{-i}(\mathrm{T}^k(x)))\mathrm{J}_{\mathrm{T}^{-i}}(\mathrm{T}^k(x))\mathbb{1}_{\mathrm{I}}(\mathrm{T}^k(x),-i) \\ &= \frac{1}{\mathrm{J}_{\mathrm{T}^k}(x)} \sum_{i\in\mathbb{Z}} \mathbb{1}_{\mathrm{I}}(x,k)\varpi_i \rho(\mathrm{T}^{-i+k}(x))\mathrm{J}_{\mathrm{T}^{-i+k}}(x)\mathbb{1}_{\mathrm{I}}(\mathrm{T}^k(x),-i) \\ &= \frac{1}{\mathrm{J}_{\mathrm{T}^k}(x)} \sum_{j\in\mathbb{Z}} \varpi_{j+k}\rho(\mathrm{T}^{-j}(x))\mathrm{J}_{\mathrm{T}^{-j}}(x)\mathbb{1}_{\mathrm{I}}(\mathrm{T}^k(x),-j-k)\mathbb{1}_{\mathrm{I}}(x,k)\end{aligned}$$

Note that if $(x,k) \in \mathrm{I}$, we have $(x,-j) \in \mathrm{I}$ if and only if $(\mathrm{T}^k(x), -j-k) \in \mathrm{I}$ by definition of I (5.58). The proof is concluded by noting that:

$$\mathbb{1}_{\mathrm{I}}(\mathrm{T}^k(x)), -j-k)\mathbb{1}_{\mathrm{I}}(x,k) = \mathbb{1}_{\mathrm{I}}(x,-j)\mathbb{1}_{\mathrm{I}}(x,k) \; .$$

□

5.9 Iterated SIR

Let us recall the principle of the Sampling Importance Resampling method (SIR; [Rub87; SG92]) whose goal is to approximately sample from the target distribution Π using samples drawn from a proposal distribution Λ.

In SIR, a N-i.i.d. sample $X^{1:N}$ is first generated from the proposal distribution Λ. A sample X^* is approximately drawn from the target Π by choosing randomly a value in $X^{1:N}$ with probabilities proportional to the importance weights $\{\mathrm{L}(X^i)\}_{i=1}^N$, where $\mathrm{L}(x) = \Pi(x)/\Lambda(x)$. Note that the importance weights are required to be known only up to a constant factor.

For SIR, as $N \to \infty$, the sample X^* is *asymptotically* distributed according to Π; see [SG92].

A subsequent algorithm is the *iterated SIR* (i-SIR) [ADH10]. Here, N is not necessarily large ($N \geqslant 2$), the whole process of sampling a set of proposals, computing the importance weights, and picking a candidate, is iterated. At the n-th step of i-SIR, the active set of N proposals $X_n^{1:N}$ and the index $I_n \in [N]$ of the conditioning proposal are kept. First i-SIR updates the active set by setting $X_{n+1}^{I_n} = X_n^{I_n}$ (keep the conditioning proposal) and then draw independently $X_{n+1}^{1:N\setminus\{I_n\}}$ from Λ. Then it selects the next proposal index $I_{n+1} \in [N]$ by sampling with probability proportional to $\{\tilde{w}(X_{n+1}^i)\}_{i=1}^N$. As shown in [ADH10], this algorithm defines a partially collapsed Gibbs sampler (PCG) of the augmented distribution

$$\bar{\boldsymbol{\pi}}(x^{1:N}, i) = \frac{1}{N}\Pi(x^i)\prod_{j\neq i}\Lambda(x^j) = \frac{1}{N}\tilde{w}(x^i)\prod_{j=1}^N \Lambda(x^j) \; .$$

The PCG sampler can be shown to be ergodic provided that Λ and Π are continuous and Λ is positive on the support of Π. If in addition the importance weights are bounded, the Gibbs sampler can be shown to be uniformly geometrically ergodic [LDM15; ALV+18]. It follows that the distribution of the conditioning proposal $X_n^* = X_n^{I_n}$ converges to π as the iteration index n goes to infinity. Indeed, for any integrable function f on $\mathbb{R}^d$, with $(X_{1:N}, I) \sim \bar{\boldsymbol{\pi}}$,

$$\mathbb{E}[f(X^I)] \;=\; \int \sum_{i=1}^N f(x^i)\bar{\boldsymbol{\pi}}(x^{1:N}, i)\mathrm{d}x^{1:N} \;=\; N^{-1}\sum_{i=1}^N \int f(x^i)\Pi(x^i)\mathrm{d}x_i \;=\; \int f(x)\Pi(x)\mathrm{d}x \quad .$$

When the state space dimension d increases, designing a proposal distribution Λ guaranteeing proper mixing properties becomes more and more difficult. A way to circumvent this problem is to use dependent proposals, allowing in particular *local moves* around the conditioning orbit. To implement this idea, for each $i \in [N]$, we define a proposal transition, $r_i(x^i; x^{1:N\setminus\{i\}})$ which defines the the conditional distribution of $X^{1:N\setminus\{i\}}$ given $X^i = x^i$. The key property validating i-SIR with dependent proposals is that all one-dimensional marginal distributions are equal to Λ, which requires that for each $i, j \in [N]$,

$$\Lambda(x^i)r_i(x^i; x^{1:N\setminus\{i\}}) = \Lambda(x^j)r_j(x^j; x^{1:N\setminus\{j\}}) \tag{5.65}$$

The (unconditional) joint distribution of the particles is therefore defined as

$$\Lambda_N\big(x^{1:N}\big) = \Lambda(x^1)r_1(x^1; x^{1:N\setminus\{1\}}) \; . \tag{5.66}$$

The resulting modification of the i-SIR algorithm is straightforward: $X^{1:N\setminus\{I_n\}}$ is sampled jointly from the conditional distribution $r_{I_n}(X_n^{I_n}, \cdot)$ rather than independently from Λ.

5.10 Additional Experiments

5.10.1 Normalizing constant estimation

We consider here the problem of the estimation of the normalizing constant of Cauchy mixtures. The Cauchy distribution with scale σ has a pdf defined by $\text{Cauchy}(x;\mu,\sigma) = [\pi\sigma(1+\{(x-\mu)/\sigma\}^2]^{-1}$. The target distribution is a product of mixtures of two Cauchy distributions,

$$\Pi(x) = \prod_{i=1}^{n} \frac{1}{2}\left[\text{Cauchy}\left(x_i;\mu,\sigma\right) + \text{Cauchy}\left(x_i;-\mu,\sigma\right)\right], \quad \mu = 5, \sigma = 1\ .$$

NEO-IS is compared with IS estimator using the same proposal Λ. We also compare NEO-IS to Neural IS [Mül+19] with a Cauchy as base distribution.

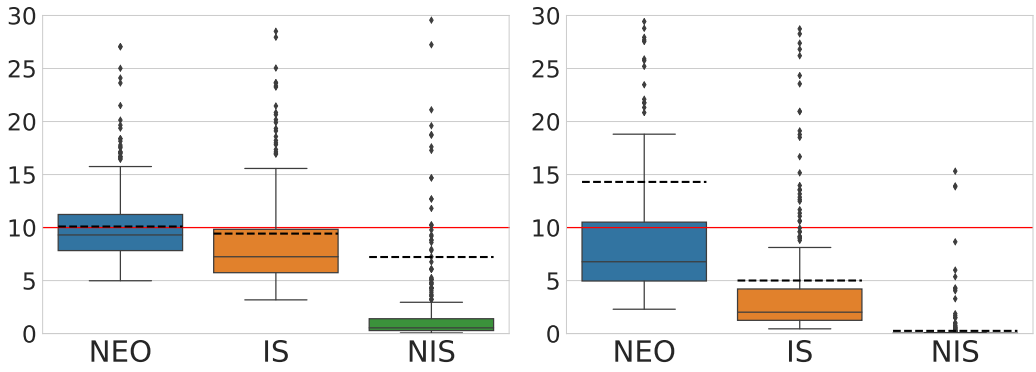


Figure 5.5: Boxplots of 500 independent estimations of the normalizing constant of the Cauchy mixture in dimension $d = 10, 15$ (top, bottom). The true value is given by the red line. The figure displays the median (solid lines), the interquartile range, and the mean (dashed lines) over the 500 runs

Finally, we compare NEO-IS with NEIS[4]. We consider here `MG25` in dimension 5 and 10, where all the covariances of the Gaussian distributions are diagonal and equal to $0.005\,\text{Id}$. NEIS and NEO-IS are run for the same computational time. We add an IS scheme as a baseline for comparison. All algorithms (NEO-IS, NEIS, IS) are run for 7.20s and 11.30s wall clock time respectively for $d = 5$ and $d = 10$. For NEO-IS, we use a conformal transform with $h = 0.1$, $K = 10$ and $\gamma = 1$. For NEIS, we choose $\gamma = 1$ and consider a stepsize $h = 10^{-4}$ corresponding to an optimal trade-off between the discretization bias inherent to NEISand its computational budget. We can observe that NEO-IS always outperforms NEIS, which suffers from a non-negligeable bias if the stepsize h is not chosen small enough.

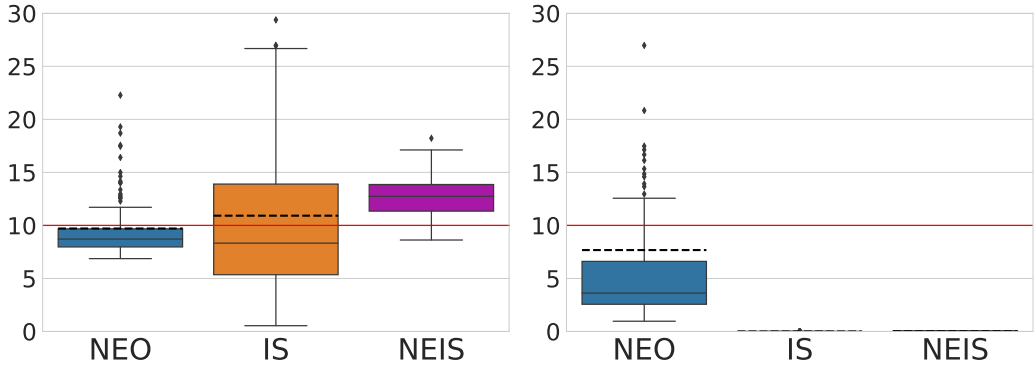


Figure 5.6: NEO v. NEIS. 25 GM with $\sigma^2 = 0.005$, $d = 5$. 500 runs each.

[4]The code from [RV19] we run is available at `https://gitlab.com/rotskoff/trajectory_estimators/-/tree/master`.

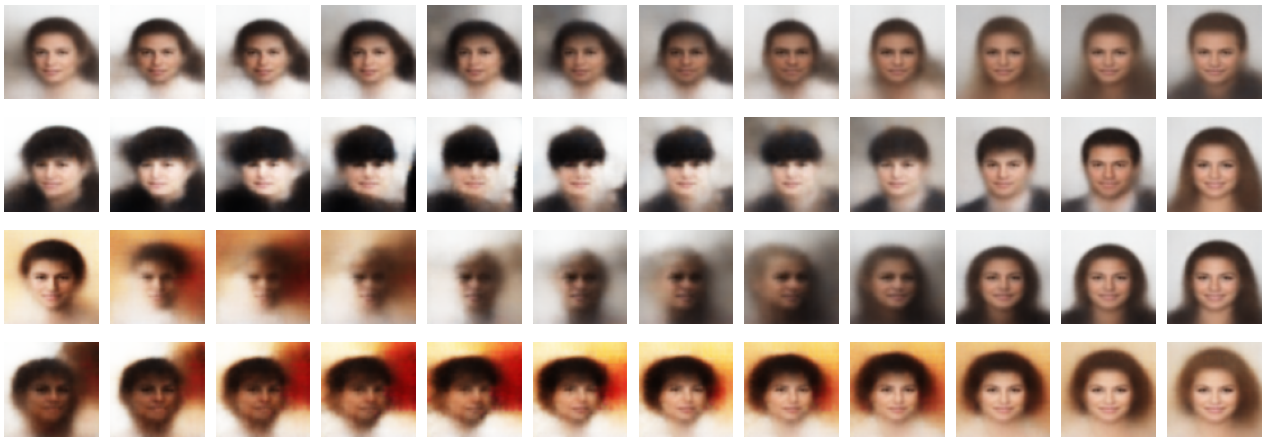


Figure 5.7: Forward orbits of NEO-MCMC.

5.10.2 Gibbs inpainting

We display here additional results for the Gibbs inpainting experiment presented in Section 5.5. We emphasize that the starting images are chosen at random in the test set.

5.11 NEO and VAEs

Denote by $p_\theta(x, z)$ the joint distribution of the observation $z \in \mathbb{R}^p$ and the latent variable $x \in \mathbb{R}^d$. The marginal likelihood is given, for $z \in \mathbb{R}^p$ by $p_\theta(z) = \int p_\theta(x, z)\mathrm{d}x$. Given a training set $\mathcal{D} = \{z_i\}_{i=1}^M$, the objective is to estimate θ by maximizing the likelihood, *i.e.* maximizing $\log p_\theta(\mathcal{D}) = \sum_{i=1}^M \log p_\theta(z_i)$. We show two experiments in the following, first the evaluation of independently trained VAEs, and then the derivation and learning of a VAE based on NEO, and NEO-VAE.

5.11.1 Log-likelihood estimation

We present here first the evaluation of the log-likelihood of a trained VAE on the dynamically binarized MNIST dataset. The models we compare share the same architecture: the inference network q_ϕ is given by a convolutional network with 2 convolutional layers and one linear layer, which outputs the parameters $\mu_\phi(x), \sigma_\phi(x) \in \mathbb{R}^d$ of a factorized Gaussian distribution, while the generative model $p_\theta(\cdot|z)$ is given by another symmetrical convolutional network g_θ. This outputs the parameters for the factorized Bernoulli distribution (for MNIST dataset), that is

$$p_\theta(z|x) = \prod_{i=1}^N \mathrm{Ber}\Big(z^{(i)}|\big(g_\theta(x)\big)^{(i)}\Big) \,.$$

We here follow the experimental setting of [Wu+16]. Given a test set $\mathcal{T} = \{z_i\}_{i=1}^{M_\mathcal{T}}$, we estimate $\sum_{i=1}^{M_\mathcal{T}} \log p_{\theta^*}(z_i)$. We also estimate similarly the log-likelihood of an Importance Weighted Auto Encoder (IWAE) [BGS15]. Following [Wu+16], we compare IS, AIS, and NEO-IS. As previously, AIS, IS, and NEO-IS are given a similar computational budget, choosing here $K = 12$, $N = 5 \cdot 10^3$. For NEO, we choose $\gamma = 1.$ and $h = 0.2$. Similarly, the stepsize of HMC transitions in AIS is $h = 0.1$ in order to achieve an acceptance ratio of around 0.6 in the HMC transitions. We report in Table 5.1 the log-likelihood computed on the test set for VAE, IWAE with latent dimension in $\{16, 32\}$. For the same computational budget, NEO-IS yields consistently better values for the estimation of the log-likelihood of the VAE.

Figure 5.8: Additional examples for the Gibbs inpainting task for CelebA dataset. From top to bottom: i-SIR, HMC and NEO-MCMC: From left to right, original image, blurred image to reconstruct, and output every 5 iterations of the Markov chain.

Model	VAE, $d = 32$	VAE, $d = 16$	IWAE, $d = 32$	IWAE, $d = 16$
IS	-90.17	-90.44	-88.76	-90.13
AIS	-89.67	-89.97	-88.30	-89.61
NEO-IS	-88.81	-89.17	-87.46	-88.99

Table 5.1: Evaluation of the log-likelihood (normalizing constant) of different Variational Auto Encoders.

5.11.2 Definition of a NEO-VAE

Variational inference (VI) provides us with a tool to simultaneously approximate the intractable posterior $p_\theta(x|z)$ and maximize the marginal likelihood $p_\theta(\mathcal{D})$ in the parameter θ. This is achieved by introducing a parametric family $\{q_\phi(x|z), \phi \in \Phi\}$ to approximate the posterior $p_\theta(x|z)$ and maximizing the Evidence Lower Bound (ELBO) (see [KW19]) $\mathcal{L}_{\mathrm{ELBO}}(\mathcal{D}, \theta, \phi) = \sum_{i=1}^M \mathcal{L}_{\mathrm{ELBO}}(z_i, \theta, \phi)$ where

$$\begin{aligned}\mathcal{L}_{\mathrm{ELBO}}(z, \theta, \phi) &= \int \log\left(\frac{p_\theta(x, z)}{q_\phi(x \mid z)}\right) q_\phi(x \mid z)\mathrm{d}x \\ &= \log p_\theta(z) - \mathrm{KL}(q_\phi(\cdot \mid z) \| p_\theta(\cdot \mid z)) \,,\end{aligned} \tag{5.67}$$

and KL is the Kullback–Leibler divergence. In the sequel, we set $\Lambda(x) = q_\phi(x \mid z)$ and $\mathrm{L}(x) = p_\theta(x, z)/q_\phi(x \mid z)$. In such a case, $\Pi(x) = \Lambda(x)\mathrm{L}(x)/\mathrm{Z} = p_\theta(x \mid z)$ and $\mathrm{Z} = p_\theta(z)$ (in these notations, the dependence in the observation z is implicit).

We follow the the auxiliary variational inference framework (AVI) provided by [AB04]. We consider a joint distribution $\bar{p}_\theta(x, u, z)$ which is such that $p_\theta(z) = \int p_\theta(x, u, z)\mathrm{d}x\mathrm{d}u$ where $u \in \mathsf{U}$ is an auxiliary variable (the auxiliary variable can both have discrete and continuous components; when u has discrete components the integrals should be replaced by a sum). Then as the usual VI approach, we consider a parametric family $\{\bar{q}_\phi(x, u|z), \phi \in \Phi\}$. Introducing auxiliary variables loses the tractability of (5.67) but they allow for their own ELBO as suggested in [AB04; Law+19] by minimizing

$$\mathrm{KL}(\bar{q}_\phi(\cdot \mid z) \| \bar{p}_\theta(\cdot \mid z)) = \int \bar{q}_\phi(x, u|z) \log\left(\frac{\bar{p}_\theta(x, u, z)}{\bar{q}_\phi(x, u|z)}\right) \mathrm{d}x\mathrm{d}u \,. \tag{5.68}$$

The auxiliary variable u is naturally associated with the extended target $\bar{p}$ defined similar to Remark 34,

$$\bar{p}_N([x, x^{1:N\setminus\{i\}}], i) = \check{\pi}(x^{1:N}, i) = \frac{\widehat{\mathrm{Z}}_x^\varpi}{N\mathrm{Z}}\Lambda_N(x^{1:N}) \tag{5.69}$$

with $(x, u) = ([x, x^{1:N\setminus\{i\}}], i)$, a shorthand notation for a N-tuple $x^{1:N}$ with $x^i = x$, and, with r_i defined in (5.15),

$$\Lambda_N(x^{1:N}) = \Lambda(x^1)r_1(x^1, x^{2:N}) = \Lambda(x^j)r_j(x^j, x^{1:N\setminus\{j\}}) \,, \quad j \in \{1, \ldots, N\} \,, \tag{5.70}$$

generally for Markov transitions $\{r_j\}_{j\in[N]}$. We might write simply in the following

$$\Lambda_N(x^{1:N}) = \prod_{i=1}^N \rho(x^i) \,.$$

An extended proposal playing the role of $\bar{q}_\phi(x, u|z)$ is derived from the NEO-MCMC sampler, i.e.

$$\bar{q}_N([x, x^{1:N\setminus\{i\}}], i) = \frac{\widehat{\mathrm{Z}}_x^\varpi}{N\widehat{\mathrm{Z}}_{x^{1:N}}^\varpi}\Lambda_N(x^{1:N}) \,. \tag{5.71}$$

where $\widehat{\mathrm{Z}}_{x^{1:N}}^\varpi$ is the NEO estimator (5.4) of the normalizing constant. Note that, by construction,

$$\sum_{i=1}^N \bar{q}_N(x^{1:N}, i) = \Lambda_N(x^{1:N}) \tag{5.72}$$

showing that this joint proposal can be sampled by drawing the proposals $x^{1:N} \sim \rho_N$, then sampling the path index $i \in [N]$ with probability proportional to $(\widehat{\mathrm{Z}}_{x^i}^\varpi)_{i=1}^N$ (with $\widehat{\mathrm{Z}}_x^\varpi$ defined in (5.4)). The ratio of (5.69) over (5.71) is

$$\bar{p}_N(x^{1:N}, i)/\bar{q}_N(x^{1:N}, i) = \widehat{\mathrm{Z}}_{x^{1:N}}^\varpi/\mathrm{Z} \,. \tag{5.73}$$

Table 5.2: Negative Log Likelihood estimates for VAE models for different latent space dimensions.

	$d = 4$		$d = 8$		$d = 16$		$d = 50$	
model	IS	NEO	IS	NEO	IS	NEO	IS	NEO
VAE	115.01	113.49	97.96	97.64	90.52	90.42	88.22	88.36
IWAE, $N = 5$	113.33	111.83	97.19	96.61	89.34	89.05	87.49	87.27
IWAE, $N = 30$	111.92	110.36	96.81	95.94	88.99	88.64	86.97	86.93
NEO VAE, $K = 3$	109.14	107.47	94.50	94.26	89.03	88.92	88.14	88.16
NEO VAE, $K = 10$	110.02	107.90	94.63	94.22	89.71	88.68	88.25	86.95

Thus, we write the augmented ELBO (5.68)

$$\mathcal{L}_{\text{NEO}} = \int \Lambda_N(x^{1:N}) \log \widehat{\mathrm{Z}}^{\varpi}_{x^{1:N}} \mathrm{d}x^{1:N} = \log \mathrm{Z} - \mathrm{KL}(\bar{q}_N | \bar{p}_N) \,, \tag{5.74}$$

where we have used (5.72) and that the ratio $\bar{p}_N(x^{1:N}, i)/\bar{q}_N(x^{1:N}, i)$ does not depend on the path index i. When $\varpi_k = \delta_{k,0}$, where $\delta_{i,j}$ is the Kronecker symbol, and $\Lambda_N(x^{1:N}) = \prod_{j=1}^N \Lambda(x^j)$, we exactly retrieve the Importance Weighted AutoEncoder (IWAE); see e.g. [BGS15] and in particular the interpretation in [CMD17].

Choosing the conformal Hamiltonian introduced in Section 5.2 allows for a family of invertible flows that depends on the parameter θ which itself is directly linked to the target distribution. Table 5.2 displays the estimated NLL of all models provided by IS and the NEO method. It is interesting to note here again that NEO improves the training of the VAE when the dimension of the latent space is small to moderate.

Chapter 6

Ex^{2}MCMC: Sampling through Exploration Exploitation

Evgeny Lagutin[1], Daniil Selikhanovych[1], Achille Thin[2], Sergey Samsonov[3], Alexey Naumov[3], Denis Belomestny[3], Maxim Panov[3], Eric Moulines[2]

Abstract

We develop an Explore-Exploit Markov chain Monte Carlo algorithm (Ex^{2}MCMC) that combines multiple global proposals and local moves. The proposed method is massively parallelizeable and extremely computationally efficient. We prove V-uniform geometric ergodicity of Ex^{2}MCMC under realistic conditions, and compute explicit bounds on the mixing rate showing the improvement brought by the multiple global moves. We show that Ex^{2}MCMC allows fine-tuning of exploitation (local moves) and exploration (global moves) via a novel approach to proposing dependent global moves. Finally, we develop an adaptive scheme, FlEx^{2}MCMC, that learns the distribution of global moves using normalizing flows. We illustrate the efficiency of Ex^{2}MCMC and its adaptive versions on many classical sampling benchmarks. We also show that these algorithms improve the quality of sampling GANs as energy-based models.

6.1 Introduction

Suppose one is interested in sampling from a probability distribution Π that is known up to a scaling factor. A Markov chain Monte Carlo algorithm (MCMC) consists of simulating a realization of a time-homogeneous Markov chain $\{Y_k,\ k \in \mathbb{N}\}$ with the Markov kernel K, with the property that the distribution of Y_n becomes arbitrarily close to Π as $n \to \infty$, irrespective of the distribution of Y_0. A property that the kernel K must satisfy is that it leaves the distribution Π invariant, i.e., Π should be a fixed point of the Markov kernel. Instead, one can consider the stronger *detailed balance* condition or reversibility, a property that is easier to handle due to its local character. In particular, it leads to the famous Metropolis-Hastings (MH) kernel, the cornerstone of MCMC simulations, and a number of its successful variants.

To improve the available samplers, a number of authors have tried to optimize the usual MH algorithm by generating a pool of proposals at each iteration, e.g. Multiple-Try Metropolis algorithm (MTM; [LLW00; CL07b]). The use of multiple proposals at each iteration, which can be efficiently implemented in parallel computing architectures, allows to increase the local search region without decreasing the acceptance ratio, which leads to an improvement in the mixing rate. This property

[1]CDISE, Skolkovo Institute of Science and Technology, Moscow, Russian Federation
[2]Centre de Mathématiques Appliquées, UMR 7641, Ecole polytechnique, France
[3]CS Departement, HSE University, Russian Federation

has been theoretically supported by results on the high-dimensional scaling limit (see [BDM12]). At the same time, MCMC algorithms based on multiple independent proposals suffer from the fact that their acceptance rate decreases dramatically in large dimension. In this work, we follow the idea of generating multiple proposals for MH-based MCMC but give a new prospective on it and provide a computationally attractive alternative to MTM.

Contributions The main contributions of the paper are as follows:

- We propose an Explore-Exploit MCMC algorithm (Ex^{2}MCMC), that retains most of the desirable properties of MTM, in particular, high degree of parallelization and improved mixing rate, while reducing the computational cost. We prove V-uniform geometric convergence of Ex^{2}MCMC and evaluate its mixing rate;
- We propose an original method to construct dependent proposals for Ex^{2}MCMC , which allows to fine-tune the exploration-exploitation trade-off. Moreover, we propose an adaptive algorithm to learn the proposal distribution (FlEx^{2}MCMC);
- We provide numerical evaluation of Ex^{2}MCMC and FlEx^{2}MCMC on various sampling problems, including sampling from GANs as energy-based model. The results clearly show the benefits of the proposed approaches compared to standard MCMC methods.

6.2 Ex^{2}MCMC

6.2.1 From Importance Sampling to Sampling Importance Resampling

Importance Sampling (IS) is widely used for estimating integrals of the function f w.r.t. of a target distribution Π on a state-space $(\mathbb{X}, \mathcal{X})$, known up to a normalizing factor Z_Π, $\Pi(\mathrm{d}x) = \tilde{\Pi}(\mathrm{d}x)/\,\mathrm{Z}_\Pi$; see, e.g. [RC13b]. IS consists of weighting samples from a proposal Λ. Assume that $\tilde{\Pi}(\mathrm{d}x) = \tilde{w}(x)\Lambda(\mathrm{d}x)$, and that the importance weight function $\tilde{w}$ is positive, i.e. $\tilde{w}(x) > 0$ for all $x \in \mathbb{X}$. We often assume that $\tilde{\Pi}$ (hence, Π) and Λ have positive densities w.r.t. a common dominating measure, denoted by $\tilde{\pi}$, π, and λ, respectively. If this is the case, $\tilde{w}(x) = \tilde{\pi}(x)/\lambda(x)$. The *self-normalized importance sampling* (SNIS) estimator of $\pi(f)$ is then given by $\widehat{\Pi}_N(f) = \sum_{i=1}^N \omega_N^i f(X^i)$, where $X^{1:N} \sim \Lambda$ and $\omega_N^i = \tilde{w}(X^i)/\sum_{j=1}^N \tilde{w}(X^j)$, $i \in \{1, \dots, N\}$ are the self-normalized importance weights.

Provided that $\Lambda(\tilde{w}^2) = \int \tilde{w}(x)^2 \Lambda(\mathrm{d}x) < \infty$, the bias and the mean square error MSE of the SNIS estimator are inversely proportional to N.

Although importance sampling is primarily intended to approximate integrals of the form $\Pi(f)$, it can also be used to (approximately) sample from Π. The latter can be achieved by the Sampling Importance Resampling (SIR; [Rub87]). SIR is a two-stage procedure. In the first stage, an i.i.d. sample $X^{1:N} = X^1, \dots, X^N$ is sampled from Λ and the importance weights $\omega^{1:N} = \omega_N^1, \dots, \omega_N^N$ are computed. In a second step, a sample of size M, $Y^{1:M}$ is obtained by sampling with replacement with the weights $\omega^{1:N}$, denoted $\mathrm{Cat}(\omega^{1:N})$. In other words, given a N-iid sample $X^{1:N}$ from Λ, SIR draws samples from the empirical distribution $\hat{\Pi}(\mathrm{d}x) = \sum_{i=1}^N \omega_N^i \delta_{X^i}(\mathrm{d}x)$ where $\delta_y(\mathrm{d}x)$ denotes the Dirac mass at y. As $N \to \infty$, a sample $Y^1, \dots, Y^M \sim \hat{\Pi}$ will be distributed according to Π; see [SG92; SBH03]. The main issue of SIR method is that it is only asymptotically valid.

6.2.2 From SIR to iterated Sampling Importance Resampling (i-SIR)

A closely related algorithm is the *iterated SIR* (i-SIR), a term coined in [ADH10] and later investigated more deeply in [ALV+18]. For i-SIR, the sample size N is not necessarily large, but the process of sampling from the proposal, computing the normalized importance weights and picking a candidate is iterated. The j-th i-SIR iteration is defined as follows. Given the current state $Y_j \in \mathbb{X}$, (i) Set $X_{j+1}^1 = Y_j$ and draw $X_{j+1}^{2:N}$ independently from the proposal distribution Λ. (ii) Compute the normalized importance weights $\omega_{N,j+1}^i = \tilde{w}(X_{j+1}^i)/\sum_{\ell=1}^N \tilde{w}(X_{j+1}^\ell)$, $i \in \{1, \dots, N\}$. (iii) Draw Y_{j+1} from the proposal set $X_{j+1}^{1:N}$, choosing X_{j+1}^i with probability $\omega_{N,j+1}^i$. The Markov kernel associated

with i-SIR is given, for $(x, \mathsf{A}) \in \mathbb{X} \times \mathcal{X}$, by

$$\mathsf{P}_N(x, \mathsf{A}) = \int \delta_x(\mathrm{d}x^1) \sum_{i=1}^{N} \frac{\tilde{w}(x^i)}{\sum_{j=1}^{N} \tilde{w}(x^j)} \mathbb{1}_{\mathsf{A}}(x^i) \prod_{j=2}^{N} \Lambda(\mathrm{d}x^j). \tag{6.1}$$

Lemma 46. *P_N admits Π as its invariant distribution.*

This result is proven in [ALV+18] (see also Section 6.7.1). To go further, we now establish the V-geometric convergence for i-SIR samples to Π. To state the results, we introduce, for a function $V(x)\colon \mathbb{X} \mapsto [1, \infty)$, the *$V$-norm* of two probability measures ξ and ξ' on $(\mathbb{X}, \mathcal{X})$, $\|\xi - \xi'\|_V \stackrel{:=}{\sup}_{|f(x)| \leqslant V(x)} |\xi(f) - \xi'(f)|$. If $V \equiv 1$, $\|\cdot\|_1$ corresponds to the total variation distance (rather denoted $\|\cdot\|_{\mathrm{TV}}$).

Definition 47 (Geometric Ergodicity)**.** *A Markov kernel Q with invariant probability measure Π is V-geometrically ergodic if there exist constants $\rho \in (0, 1)$ and $M < \infty$ such that, for all $k \in \mathbb{N}$,*

$$\|\mathrm{Q}^k(x, \cdot) - \Pi\|_V \leqslant M\, V(x) \rho^k \quad \textit{for all } x \in \mathbb{X}\,. \tag{6.2}$$

In particular, geometric ergodicity results ensure that the distribution of the n-th step of a Markov chain converges geometrically fast to the invariant probability in V-norm, for all starting points $x \in \mathbb{X}$. Here the dependence on the initial state x appears on the right-hand side only in $V(x)$. Uniform geometric ergodicity of i-SIR is established in [ALV+18, Theorem 1] under the assumption that the normalized importance weight function w (that is, $\Pi(\mathrm{d}x) = w(x)\Lambda(\mathrm{d}x)$) is uniformly bounded.

This result is extended below for arbitrary V-norm satisfying $\Pi(V) = \int V(x)\Pi(\mathrm{d}x) < \infty$ under the following condition.

H6. *For any $x \in \mathbb{X}$, $w(x) \leqslant \mathrm{L}$, with $\mathrm{L} < \infty$.*

The result is up to our best knowledge new.

Theorem 48. *Assume **H**6. Set $\epsilon_N = \frac{N-1}{2\mathrm{L}+N-2}$ and $\kappa_N = 1 - \epsilon_N$. Then,*

(i) For any $x \in \mathbb{X}$ and $k \in \mathbb{N}$, $\|\mathsf{P}_N^k(x, \cdot) - \Pi\|_{\mathrm{TV}} \leqslant \kappa_N^k$

Let $V\colon \mathbb{X} \to [1, \infty)$ be any measurable function such that $\Pi(V) < \infty$ and $\Lambda(V) < \infty$. Then,

(ii) For all $x \in \mathbb{X}$ and $k \in \mathbb{N}$, $\|\mathsf{P}_N^k(x, \cdot) - \Pi\|_V \leqslant c_N\{\Pi(V) + V(x)\}\tilde{\kappa}_N^k$, where the constants c_N, $\tilde{\kappa}_N$ are given in (6.53).

The proof is postponed to Section 6.7.2. We show that $\tilde{\kappa}_N = O(N^{-1/3})$ which can be sharpen but essentially means that the geometric convergence rate decreases to 0 with the reciprocal of the number of proposals N. **H**6 is restrictive, and even if it is satisfied, in most cases the upper bound grows exponentially with dimension (typically, if $\Pi_d = \prod_{i=1}^{d} \Pi$, $\Lambda_d = \prod_{i=1}^{d} \Lambda$, then $\mathrm{L}_d = \mathrm{L}^d$). For this reason, IS is rarely used in high-dimensional space unless the proposal distribution is learned [Aga+17], which is the key point of Adaptive IS methods; see Section 6.3. A natural idea is to couple the i-SIR method with some local MCMC steps to define a sampler that remains V-uniformly geometrically ergodic even if **H**6 is not satisfied. After each step of i-SIR, it suffices to apply a local MCMC kernel R (called *rejuvenation kernel*) that has Π as invariant distribution. We call this algorithm the Ex^{2}MCMC algorithm because it combines steps of exploration by i-SIR and steps of exploitation by the local MCMC moves. The resulting Ex^{2}MCMC algorithm is given by Algorithm 5.

We denote by K_N the associated Markov kernel (see Supplementary Material). Consider the following assumption

H7. *(i) R has Π as its unique invariant distribution; (ii) There exists a function $V\colon \mathbb{X} \to [1, \infty)$, such that for all $d \geqslant d_{\mathsf{R}} > 1$ there exist $\lambda_{\mathsf{R},d} \in [0, 1)$, $\mathsf{b}_{\mathsf{R},d} < \infty$, such that $\mathsf{R}V \leqslant \lambda_{\mathsf{R},d} V + \mathsf{b}_{\mathsf{R},d} \mathbb{1}_{\mathsf{V}_d}$, where $\mathsf{V}_d = \{x\colon V(x) \leqslant d\}$; (iii) For all $d \geqslant d_{\mathsf{R}}$, $\sup_{x \in \mathsf{V}_d} w(x) < w_{\infty,d} < \infty$.*

Procedure Ex^{2}MCMC $(Y_j, \Lambda, \mathsf{R})$**:**
  Input : Previous sample Y_j;
    proposal distribution Λ;
    rejuvenation kernel R;
  Output : New sample Y_{j+1};
  Set $X_{j+1}^1 = Y_j$, draw $X_{j+1}^{2:N} \sim \Lambda$;
  for $i \in [N]$ **do**
    compute the normalized weights $\omega_{i,j+1} = \tilde{w}(X_{j+1}^i) / \sum_{k=1}^N \tilde{w}(X_{j+1}^k)$;
  end
  Set $I_{j+1} = \mathrm{Cat}(\omega_{1,j+1}, \ldots, \omega_{N,j+1})$;
  Draw $Y_{j+1} \sim \mathsf{R}(X_{j+1}^{I_{j+1}}, \cdot)$.
end

Algorithm 1: Single stage of Ex^{2}MCMC algorithm with independent proposals

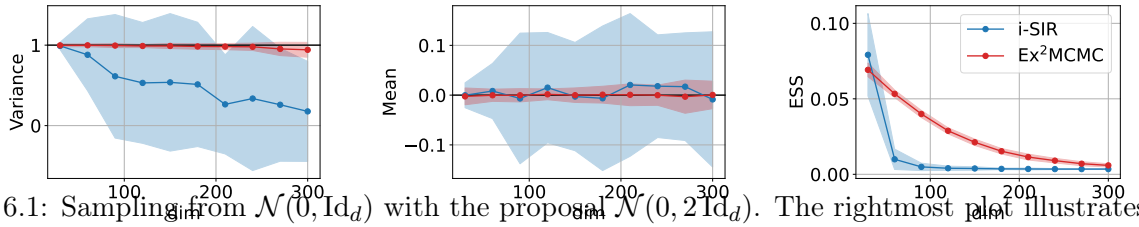


Figure 6.1: Sampling from $\mathcal{N}(0, \mathrm{Id}_d)$ with the proposal $\mathcal{N}(0, 2\,\mathrm{Id}_d)$. The rightmost plot illustrates the number of rejections rapidly growing for vanilla i-SIR algorithm (see Section 6.11.1 for the definition of ESS). The correlated proposals in Ex^{2}MCMC help to achieve efficient sampling even in high dimensions. We display confidence intervals for i-SIR and Ex^{2}MCMC obtained from 20 independent runs as blue and red regions, respectively.

(ii) states that the rejuvenation kernel satisfies a Foster-Lyapunov drift condition for V: it is satisfied by very many MCMC kernels, typically under super-exponential tail conditions for the target distribution; see [RR04] and [Dou+18, Chapter 2] and the references therein. (iii) states that the importance weights are upper bounded on level sets. This is a mild condition: if $\mathbb{X} = \mathbb{R}^d$, and V is norm-like, then the level sets V_d are compact and $\tilde{w}$ is bounded as soon as π and λ are positive and continuous. We emphasize that we do not need to identify the small sets of the rejuvenation kernel. We can now present the main theoretical result of this paper.

Theorem 49. *Assume **H**7. Then, for all $x \in \mathbb{X}$ and $k \in \mathbb{N}$,*

$$\|\mathsf{K}_N^k(x, \cdot) - \Pi\|_V \leqslant c_{N,\mathsf{R}} \{\Pi(V) + V(x)\} \lambda_{\mathrm{r}}[N, \mathsf{R}]^k \,, \tag{6.3}$$

where the constant $c_{N,\mathsf{R}}$, $\lambda_{\mathrm{r}}[N, \mathsf{R}]$ are given in (6.74).

The proof is postponed to Section 6.7.3. The main steps are (i) establishing that the level sets V_d are small for the Markov kernel K_N (with a constant which depends only on N and $w_{\infty,d}$; see Lemma 52); (ii) checking that K_N also satisfies a Foster-Lyapunov with function V with constant depending only on $\lambda_{\mathsf{R},d} \in [0, 1)$, $\mathsf{b}_{\mathsf{R},d}$, and N; see Lemma 54. Note that $\lambda_{\mathrm{r}}[N, \mathsf{R}]$ typically decreases when the number of proposals N grows. In many situations, the mixing rate $\lambda_{\mathrm{r}}[N, \mathsf{R}]$ is significantly better than the corresponding mixing rate of R, provided that N is large enough. This is illustrated in Section 6.8 with the Metropolis Adjusted Langevin Algorithm (MALA) kernel (see e.g. [Bes94a; RT96]).

6.2.3 Dependent proposals for i-SIR and Ex^{2}MCMC algorithms

We now extend Ex^{2}MCMC by relaxing the assumptions that the proposals at each stage are independent. The possibility of using dependent particles (and in particular *local moves* around the conditioning point) allows Ex^{2}MCMC algorithm to fine-tune Exploration v.s. Exploitation. Denote by $\bar{\Lambda}_N(\mathrm{d}x^{1:N})$ the joint distributions of the proposed particles. The key property to satisfy is that the marginal

distribution of the proposed particles is Λ, that is for each $i \in [N] = \{1, \dots, N\}$ there exists a Markov kernel denoted Q_i satisfying

$$\Lambda(\mathrm{d}x^i)\mathsf{Q}_i\big(x^i, \mathrm{d}x^{1:N\setminus\{i\}}\big) = \bar{\Lambda}_N\big(\mathrm{d}x^{1:N}\big) \; . \tag{6.4}$$

The Markov kernel $\mathsf{Q}_i(X^i, \cdot)$ defines the conditional distribution of the particles $X^{1:N\setminus\{i\}}$ given the "conditioning" particle X^i. In the simple i-SIR case, $\mathsf{Q}_i(x^i, \mathrm{d}x^{1:N\setminus\{i\}}) = \prod_{j\mathrm{nei}} \Lambda(\mathrm{d}x^j)$ so that $\bar{\Lambda}_N\big(\mathrm{d}x^{1:N}\big) = \prod_{i=1}^N \Lambda(\mathrm{d}x^i)$. Using these conditional distributions, we now adapt Algorithm 5 by changing `line 1` by

1. Draw U_{j+1} from the uniform distribution in $[N]$ and set $X_{j+1}^{U_{j+1}} = Y_j$;
2. Draw $X_{j+1}^{1:N\setminus\{U_{j+1}\}} \sim \mathsf{Q}_{U_{j+1}}(X_{j+1}^{U_{j+1}}, \cdot)$.

Note that *contrary to the independent case* we *randomize* the index of the conditioning variable, because we have not assumed that the joint proposal $\bar{\Lambda}_N$ is exchangeable. We present below some methods of constructing proposals verifying (6.4). We denote by C_N the corresponding Markov kernel. The following result establishes the validity of the proposed algorithm, which holds virtually without any assumption.

Theorem 50. *For any $N \geqslant 2$, the Markov kernel C_N admits Π as its unique invariant distribution.*

We now describe a general method for constructing joint proposals $\bar{\Lambda}_N$ with the proper marginal Λ. The main motivation for this construction is to seize the opportunity of massive parallelization of proposal sampling, which is a key to efficient implementation. The basic idea is to introduce a hierarchical latent variable model that allows flexible control of the dependency between proposals while preserving the desired symmetry property. Denote by $(\mathbb{E}, \mathcal{E})$ the latent space, Ξ the latent distribution and R, a Markov kernel on $\mathbb{E} \times \mathcal{X}$, the conditional distribution of the proposal given the latent variable. We assume that

$$\Xi\mathsf{R}(\mathrm{d}x) = \int_{\mathbb{E}} \Xi(\mathrm{d}\xi)\mathsf{R}(\xi, \mathrm{d}x) = \Lambda(\mathrm{d}x). \tag{6.5}$$

We associate to each proposal X^i a latent variable ξ^i, $i \in [N]$; $\mathsf{R}(\xi^i, \cdot)$ therefore defines the conditional distribution of X^i given the latent variable ξ^i. Denote by $\bar{\Xi}_N$ the joint distribution of $\xi^{1:N}$. It is assumed that the marginal distribution of any ξ^i is Ξ, and we denote by S_i the Markov kernel satisfying for any $i \in [N]$,

$$\bar{\Xi}_N\big(\mathrm{d}\xi^{1:N}\big) = \Xi(\mathrm{d}\xi^i)\; \mathsf{S}_i\big(\xi^i, \mathrm{d}\xi^{1:N\setminus\{i\}}\big) \; . \tag{6.6}$$

Finally, we set

$$\bar{\Lambda}_N\big(\mathrm{d}x^{1:N}\big) = \int_{\mathbb{E}^N} \bar{\Xi}_N\big(\mathrm{d}\xi^{1:N}\big) \prod_{i=1}^N \mathsf{R}(\xi^i, \mathrm{d}x^i) \; . \tag{6.7}$$

This is the distribution of N random variables $X^{1:N}$ taking values in $\mathbb{X}$. The variables $\{X^i\}_{i=1}^N$ are conditionally independent with respect to the latent variables $\xi^{1:N}$ and are marginally distributed according to Λ thanks to (6.5). This helps to show that the distribution defined by (6.7) satisfies (6.4) with the properly selected kernel $\mathsf{Q}_i\big(x^i, \mathrm{d}x^{1:N\setminus\{i\}}\big)$. Define by $\breve{\mathsf{R}}$ the *reverse* Markov kernel on $\mathbb{X} \times \mathcal{E}$ satisfying

$$\Xi(\mathrm{d}\xi)\mathsf{R}(\xi, \mathrm{d}x) = \Lambda(\mathrm{d}x)\breve{\mathsf{R}}(x, \mathrm{d}\xi) \; . \tag{6.8}$$

The Markov kernel $\breve{\mathsf{R}}$ can be computed using the Bayes rule. This is transparent when the Markov kernel R has density, *i.e.* there exists a function r and a measure μ on $\mathbb{X}$ such that $\mathsf{R}(\xi, \mathrm{d}x) = \mathsf{r}(x \mid \xi)\mu(\mathrm{d}x)$. In this case r is the conditional p.d.f. of the proposal X given the latent variable ξ. In such case, $\Lambda(\mathrm{d}x) = \lambda(x)\mu(\mathrm{d}x)$, with $\lambda(x) = \int_{\mathbb{E}} \Xi(\mathrm{d}\xi)\mathsf{r}(x \mid \xi)$ which is the marginal p.d.f. of the proposal X. The reversal condition in this case writes: $\Xi(\mathrm{d}\xi)\mathsf{R}(\xi, \mathrm{d}x) = \mathsf{r}(x \mid \xi)\Xi(\mathrm{d}\xi)\mu(\mathrm{d}x) = \mathsf{r}(x \mid \xi)/\lambda(x)\Xi(\mathrm{d}\xi)\Lambda(\mathrm{d}x)$. This shows that the reverse kernel has a density w.r.t. Ξ given by $\breve{\mathsf{r}}(\xi \mid x) = \mathsf{r}(x \mid \xi)/\lambda(x)$. Using (6.8), it is easily seen that for all $i \in [N]$

$$\begin{aligned} &\mathsf{Q}_i\big(x^i, \mathrm{d}x^{1:N\setminus\{i\}}\big) = \\ &= \int_{\mathbb{E}^N} \breve{\mathsf{R}}(x^i, \mathrm{d}\xi^i)\mathsf{S}_i\big(\xi^i, \mathrm{d}\xi^{1:N\setminus\{i\}}\big) \prod_{j\mathrm{nei}} \mathsf{R}(\xi^j, \mathrm{d}x^j) \; . \end{aligned} \tag{6.9}$$

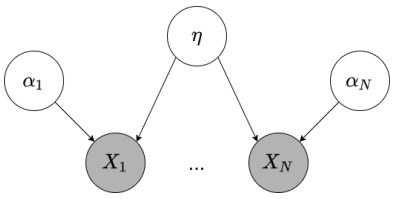


Figure 6.2: Graphical model for the proposals $X^{1:N}$.

Input : Sample Y_j from previous iteration
Output : Set of proposals for the current iteration $X_{j+1}^{1:N}$

1 Draw $U_{j+1} \sim \mathrm{Unif}([N])$ and set $X_{j+1}^{U_{j+1}} = Y_j$
2 Draw $\alpha_{j+1}^{U_{j+1}} \sim \nu$ and $\eta_{j+1} \sim \mathrm{N}(\alpha_{j+1}^{U_{j+1}} X_{j+1}^{U_{j+1}}, \sigma^2[\alpha_{j+1}^{U_{j+1}}] \mathrm{Id}_d)$
3 For $i \in [N] \setminus \{U_{j+1}\}$, draw $W_{j+1}^i \sim \mathrm{N}(0, \mathrm{Id}_d)$, $\alpha_{j+1}^i \sim \nu$, and set $X_{j+1}^i = \alpha_{j+1}^i \eta_{j+1} + \sigma[\alpha_{j+1}^i] W_{j+1}^i$.

Algorithm 2: Ex^{2}MCMC for Gaussian proposal

We can then simply update Algorithm 5 by replacing `line 1` by the following three steps:

1. Draw U_{j+1} from the uniform distribution in $[N]$ and set $X_{j+1}^{U_{j+1}} = Y_j$, $\xi_{j+1}^{U_{j+1}} \sim \check{\mathsf{R}}(X_{j+1}^{U_{j+1}}, \cdot)$,
2. Draw latents $\xi_{j+1}^{1:N\setminus\{U_{j+1}\}} \sim \mathsf{S}_{U_{j+1}}(\xi_{j+1}^{U_{j+1}}, \cdot)$,
3. Draw proposals $X_{j+1}^i \sim \mathsf{R}(\xi_{j+1}^i, \cdot)$, $i \in [N] \setminus \{U_{j+1}\}$.

Below we provide an example of Ex^{2}MCMC algorithm with dependent proposal distributions.

6.2.4 Dependent Gaussian proposals

Let us denote by $\mathrm{g}(x; \mu, \Gamma)$ the Gaussian p.d.f. with mean $\mu \in \mathbb{R}^d$ and covariance $\Gamma \in \mathbb{R}^{d\times d}$. Assume that the proposal distribution is Gaussian, *i.e.* $\lambda(x) = \mathrm{g}(x; 0, \sigma_\Lambda^2 \mathrm{Id}_d)$. As we will see below, this example is important because it is often used when sampling generative models. We give a step-by-step construction that closely follows the presentation given above. We first define the latent space $\mathbb{E} = \mathbb{R}^d \times \mathbb{R}$, the latent variable $\xi = (\eta, \alpha)$. We assume that the latent distribution is a product $\Xi = \Lambda \otimes \nu$, with $\nu(\mathrm{d}\alpha) = \epsilon\delta_a(\mathrm{d}\alpha) + (1-\epsilon)\delta_0(\mathrm{d}\alpha)$, with $\epsilon \in [0,1]$ and $a \in [0,1]$. Expressed with random variables, $X = \alpha\eta + \sigma[\alpha]W$, $\sigma[\alpha] = \sigma_\Lambda(1-\alpha^2)^{1/2}$ where (α, W) are two independent random variables, α takes two values a and 0 with probability and $\mathbb{P}(\alpha = a) = \epsilon$, $W \sim \mathrm{N}(0, \sigma_\Lambda^2 \mathrm{Id}_d)$. Under these assumptions, the conditional p.d.f of the proposal sample x given the latent variable (η, α) is given by $\mathsf{r}\big(x \mid (\eta, \alpha)\big) = \mathrm{g}(x; \alpha\eta, \sigma^2[\alpha] \mathrm{Id}_d)$. The reverse kernel $\check{\mathsf{R}}$ defined by (6.8) can be written as $\check{\mathsf{R}}\big(x, \mathrm{d}(\alpha, \eta)\big) = \epsilon \mathrm{g}(\eta; ax, \sigma^2[a] \mathrm{Id}_d)\delta_a(\mathrm{d}\alpha)\mathrm{d}\eta + (1-\epsilon)\mathrm{g}(\eta; 0, \sigma_\Lambda^2 \mathrm{Id}_d)\delta_0(\mathrm{d}\alpha)$. We now specify the joint law of the latent variables to be, for $i \in [N]$,

$$\mathsf{S}_i\big(\xi^i, \mathrm{d}\xi^{1:N\setminus\{i\}}\big) = \prod_{j \mathrm{nei}} \delta_{\eta^i}(\mathrm{d}\eta^j)\nu(\mathrm{d}\alpha^j) \,, \tag{6.10}$$

that is, the latent variables η^i, $i \in [N]$ are all equal, whereas that the random variables $\alpha^{1:N\setminus\{i\}}$ are conditionally independent with the same distributions ν. The graphical model for the resulting latent variables approach for dependent proposals generation is given in Figure 6.2. Now we can specify the modification of Algorithm 5 in this case by substituting its `line 1` by Algorithm 6.

The value ϵ controls the exploration-exploitation ratio of Ex^{2}MCMC. When $\epsilon = 0$ we recover the independent i-SIR, and for $\epsilon = 1$ i-SIR is bound only to local proposals. In our experiments, we find in practice that a value $\epsilon \in (0,1)$ is relevant. To illustrate the need for introducing dependencies between propositions, consider the toy problem of sampling a high-dimensional Gaussian distribution: $\pi(x) = \mathrm{g}(x; 0, \mathrm{Id}_d)$ and $\lambda(x) = \mathrm{g}(x; 0, 2\,\mathrm{Id}_d)$. This is admittedly an artificial problem, but it provides a diagnosis of why i-SIR fails in high dimensions and why Ex^{2}MCMC (here with $\epsilon = 1$ and without rejuvenation steps at all) works well (see Figure 6.1). Details of the experiment can be found in Supplementary Material, Section 6.11.4.

Input : weights θ_j, batch $Y_j[1:K]$
Output : new weights θ_{j+1}, batch $Y_{j+1}[1:K]$
1 **for** $k \in [K]$ **do**
2 | $Y_{j+1}[k] = \text{Ex}^2\text{MCMC}\ (Y_j[k], T_{\theta_j}\#\Lambda, \mathsf{R})$
3 **end**
4 Draw $\bar{Z}[1:K] \sim \Lambda$.
5 Update $\theta_{j+1} = \theta_j - \gamma\widehat{\nabla\mathcal{L}}(Y_{j+1}, \bar{Z}, \theta_j)$.

Algorithm 3: Single stage of FlEx^{2}MCMC. Steps of Ex^{2}MCMC are done in parallel with common values of proposal parameters θ_j. Step 4 updates the parameters using the gradient estimate obtained from all the chains.

6.2.5 Related Work

i-SIR has been proposed by [ADH10] and further developed in [ALV+18]; see also [Lee+10; Lee11]. [ALV+18] highlights the links of i-SIR with particle Gibbs methods, the main difference being that the proposal distribution is defined on the "path space" used in sequential Monte Carlo methods; see also [Dou+15]. Using this analogy, the rejuvenation kernel plays a role similar to the backward sampling kernel in the particle Gibbs with Backward Sampling (PGBS; [LS13]). The idea of making the moves dependent of the conditioning particle has been suggested in the PGBS context by [SN18].

Ex^{2}MCMC algorithm can also be seen as a collapsed version of the Gibbs sampler proposed in [Tje04]. The algorithm also has similarities with the Multiple Tries Metropolis (MTM) algorithm, but the Ex^{2}MCMC is both computationally simpler and displays more favorable mixing properties. In the MTM algorithm, N i.i.d.trial proposals $\{X^i_{j+1}\}_{i=1}^N$ are drawn from a kernel $\mathsf{T}(y,\cdot)$ in each iteration: this is similar to Ex^{2}MCMC sampling step, except that we do not require $\mathsf{Q}_j\big(x^j, \mathrm{d}x^{1:N\setminus\{j\}}\big) = \prod_{j=1}^N \mathsf{T}(x^j, \mathrm{d}x^j)$. In a second step, a sample Y^*_{j+1} is selected with probability proportional to the weights (the exact expression of the importance weights differs from ours, but this does not change the complexity of the algorithm). In a third step (see [LLW00], section 2), $N-1$ i.i.d. proposals are drawn from the kernel $\mathsf{T}(Y^*_{j+1},\cdot)$ and the move is assumed to be $Y_{j+1} = Y^*_{j+1}$ with a *generalized M-H* ratio, see [LLW00], eq. 3. This step is bypassed in Ex^{2}MCMC, which reduces the computational complexity by a factor of 2.

6.3 Adaptive Ex^{2}MCMC algorithm

As mentioned earlier, the success of IS methods lies in an appropriate choice of proposal distribution. A classical approach is to define families of proposal distributions $\{\lambda_\theta\}$ parameterized by some parameters θ chosen to match the target distribution $\tilde{\pi}$. Such families can be obtained using a sequence of invertible transformations called normalizing flow [Pap+19]. Let $T\colon \mathbb{X} \to \mathbb{X}$ be a C^1 diffeomorphism. We denote the push-forward of measure Λ under T, that is, the distribution of $Y = T(X)$ with $X \sim \lambda$, by $T\#\Lambda$. The corresponding push-forward density is given by $\lambda_T(y) = \lambda\big(T^{-1}(y)\big)\,\mathrm{J}_{T^{-1}}(y)$, where J_T denotes the Jacobian determinant of T; see [RM15b; KPB20; Pap+19] and references therein. The parameterized family of diffeomorphisms $\{T_\theta\}$ defines a family of distributions $\{\lambda_{T_\theta}\}$, denoted for conciseness $\{\lambda_\theta\}$. The parameter θ is then chosen to minimize a discrepancy between the target π and λ_θ. This discrepancy can be e.g. a forward or backward divergence KL or another f-divergence; see e.g. [Pap+19]. The forward KL objective and its gradient are given by

$$\mathcal{L}^f(\theta) = \int \log \frac{\pi(x)}{\lambda_\theta(x)}\pi(x)\mathrm{d}x\ , \tag{6.11}$$

$$\nabla\mathcal{L}^f(\theta) = -\int \nabla \log \lambda_\theta(x)\pi(x)\mathrm{d}x\ . \tag{6.12}$$

The backward KL divergence and its gradient are given by

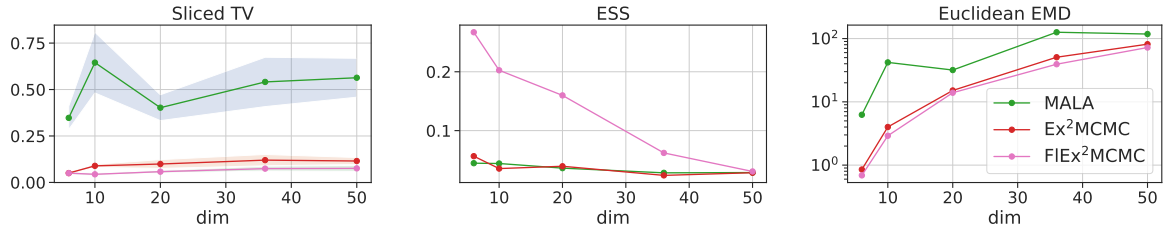


Figure 6.3: Sampling results for the asymmetric banana-shaped distribution. The Sliced TV, ESS and EMD metrics are reported as functions of the dimension of the space.

$$\mathcal{L}^b(\theta) = \int \log \frac{\lambda(x)}{\pi\big(T_\theta(x)\big)\,\mathrm{J}_{T_\theta}(x)} \lambda(x)\mathrm{d}x\,, \tag{6.13}$$

$$\nabla\mathcal{L}^b(\theta) = -\int \nabla \log\big(\pi(T_\theta(x))\,\mathrm{J}_{T_\theta}(x)\big)\lambda(x)\mathrm{d}x\,. \tag{6.14}$$

Note that $\nabla\mathcal{L}^b(\theta)$ does not depend on the normalizing constant of π. Thus we can compute unbiased estimates of $\mathcal{L}^f(\theta)$ and $\mathcal{L}^b(\theta)$, given a sample $Y[k] \sim \pi$ and $Z[k] \sim \lambda$ for $k \in [K]$, by

$$\widehat{\nabla\mathcal{L}^f}(Y[1:K],\theta) = -\frac{1}{K}\sum_{k=1}^{K} \nabla \log \lambda_\theta(Y[k])\,, \tag{6.15}$$

$$\widehat{\nabla\mathcal{L}^b}(Z[1:K],\theta) = -\frac{1}{K}\sum_{k=1}^{K} \nabla \log\big(\tilde{\pi}(T_\theta(Z[k])\,\mathrm{J}_{T_\theta}(Z[k])\big). \tag{6.16}$$

We adapt the proposal distribution using a weighted combination of the forward and backward KL: $\widehat{\mathcal{L}}(Y,Z,\theta) = \alpha_j\widehat{\mathcal{L}^f}(Y,\theta) + \beta_j\widehat{\mathcal{L}^b}(Z,\theta)$; see [GRV21]. In our experiments we use the following scheme for weights: $\alpha_j = \min\left(1, \frac{3j}{n}\right)$, $\beta_j = (1-\alpha_j)$, where n is the number of optimization steps.

We introduce in the following a novel adaptive MCMC scheme, FlEx^{2}MCMC, which combines normalizing flows and Ex^{2}MCMC; see [AT08; LLC11] for a background on adaptive MCMC. The importance weights for FlEx^{2}MCMC become $\tilde{w}_\theta(x) = \tilde{\pi}(x)/\lambda_\theta(x)$. The dependence between proposals can be introduced by first correlating samples with kernels $\{\mathsf{Q}_i\}_{i=1}^N$ which satisfy (6.4), and passing them through the flow T_θ afterwards in order to enhance the initial proposal distribution λ. The j-th step of the algorithm is given in Algorithm 7. We essentially perform K independent Ex^{2}MCMC steps with the same values of flow parameters θ and then update parameters based on the gradient estimate obtained from all the chains.

6.4 Experiments

In this section we illustrate the efficiency of Ex^{2}MCMC and FlEx^{2}MCMC compared to standard MCMC methods. In all our experiments we use the MALA rejuvenation kernel, but it is also possible to use other kernels, e.g. HMC [Nea11] or NUTS [HG+14b]. For FlEx^{2}MCMC proposals we use the RealNVP [DSB17] normalizing flows trained with the Adam optimizer [KB14]. Additional details on the experimental setup, including hyperparameters can be found in Supplementary material, Section 6.11 and Table 6.6. We assess sampling performance using the empirical Sliced Total Variation distance (STV, [Kol+19]), Effective Sample Size (ESS, [Kis65]) and the empirical Euclidean Earth Mover's distance (EMD, [Mon81]). These metrics are defined in Supplementary Material, Section 6.11.1.

6.4.1 Sampling experiments

Distributions with complex geometry We study the ability of Ex^{2}MCMC to sample from the banana-shaped distributions and funnel distributions in high dimensions. Analytical expressions for

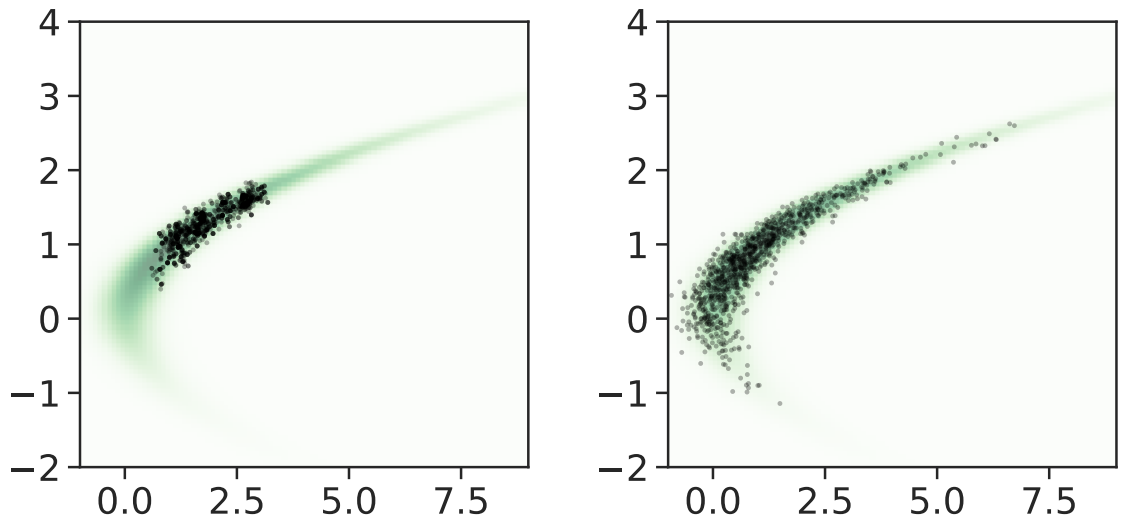


Figure 6.4: Asymmetric banana-shaped distribution in dimension 50: projections on first two coordinates.Left: MALA, right: Ex^{2}MCMC.

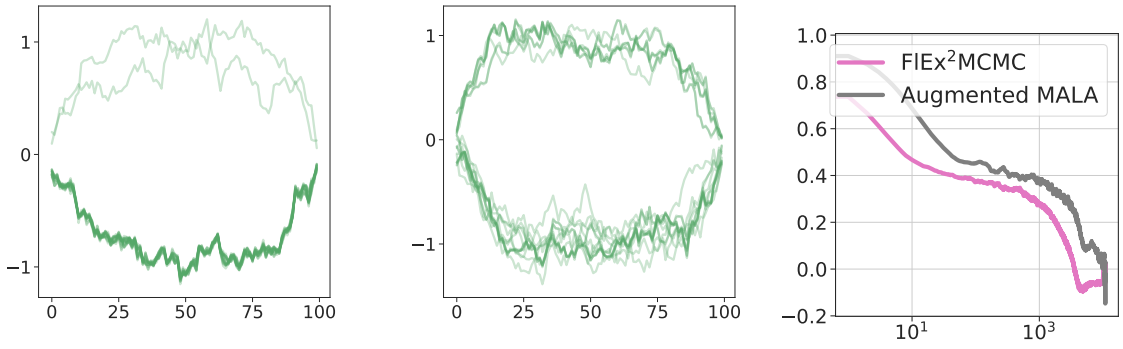


Figure 6.5: Allen-Cahn equation. From left to right,trajectories sampled by Augmented MALA, by FlEx^{2}MCMC and autocorrelation plot.

target distributions and experimental details can be found in Section 6.11.6. We use the HMC-based NUTS sampler [HG+14b] to generate reference samples and compute STV, ESS and EMD for samples obtained from single run of MALA, Ex^{2}MCMC and FlEx^{2}MCMC. The numerical results for asymmetric banana distribution are shown on Figure 6.3 and for funnel distribution on Figure 6.11. We clearly observe the benefits of Ex^{2}MCMC over MALA in approximation quality (see also Figure 6.4), while FlEx^{2}MCMC allows to dramatically improve sampling efficiency.

Allen-Cahn equation We use FlEx^{2}MCMC to sample from the invariant distribution of the Allen-Cahn stochastic differential equation; see [AC75]. For the details about distribution and task setting see [GRV21] as we fully rely on it. In their paper they propose using normalizing flows to augment MALA algorithm that we further refer to as Augmented MALA. Figures 6.4.1 and 6.4.1 show examples of learned maps (100 examples per map) for MALA and FlEx^{2}MCMC respectively. We observe that FlEx^{2}MCMC is good in terms of exploring the sample space while MALA essentially stucks in few modes. Figure 6.4.1 shows the autocorrelations of the examples obtained during the burn-in period and training. We demonstrate that FlEx^{2}MCMC allows for better mixing compared to method from [GRV21]. More details are provided in Supplementary Material, Section 6.11.9.

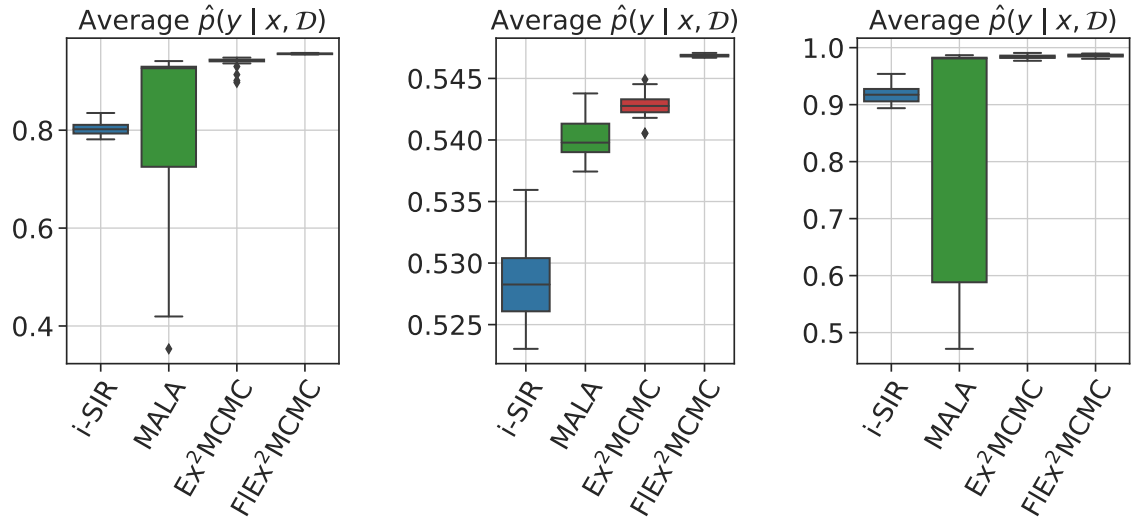


Figure 6.6: Bayesian logistic regression: average $\hat{p}(y|x, \mathcal{D})$ for (left to right) Covertype, EEG and Digits datasets.

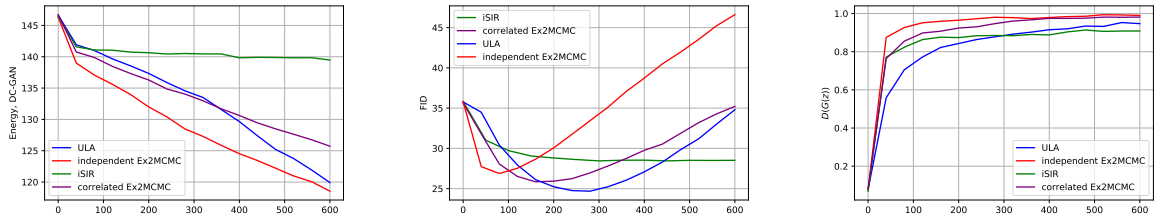


Figure 6.7: CIFAR-10 dataset with DC-GAN architecture. From left to right, average energy values, FID and discriminator scores for 600 sampling iterations.

Bayesian Logistic Regression Consider the training set $\mathcal{D} = \{(x_j, y_j)\}_{j=1}^M$ consisting of pairs (x_j, y_j), where $x_j = (x_j^{(0)}, \dots, x_j^{(d-1)}) \in \mathbb{R}^d$ and labels $y_j \in \{-1, 1\}$. Without loss of generality, we assume that $x_j^{(0)} = 1$. The likelihood for a pair (x, y) is given by $p(y \mid x, \theta) = \text{logit}(y \langle x, \theta \rangle)$, $\theta \in \mathbb{R}^d$. Given a prior distribution $p_0(\theta)$, we sample the posterior distribution $p(\theta \mid \mathcal{D})$ and compute the posterior predictive distribution $p(y \mid x, \mathcal{D})$. We approximate $p(y \mid x, \mathcal{D})$ as $p(y \mid x, \mathcal{D}) \simeq \frac{1}{n} \sum_{i=1}^n p(y \mid x, \mathcal{D}, \theta_i)$ for $\theta_i \sim p(\cdot \mid \mathcal{D})$, $i = 1, \dots, n$. We display boxplots of the posterior predictive distribution averaged over the dataset based on 30 independent runs of the samplers, see Figure 6.6. The results show that Ex²MCMC achieves much higher values compared to i-SIR and MALA while FIEx²MCMC allows to further improve the average values and also simultaneously decrease the variance. The datasets and the implementation details are given in Supplementary Material, Section 6.11.7.

6.4.2 Sampling from GAN as Energy-based model (EBM)

Generative adversarial networks (GANs) represent a class of generative models defined by a pair of a generator network G and a discriminator network D. The generator G takes a latent variable z from a prior density $p_0(z)$, $z \in \mathbb{R}^d$, and produces an observation $G(z) \in \mathbb{R}^{\mathsf{D}}$ in the observation space. The discriminator takes a sample in the observation space and aims to distinguish between real examples and fake ones, produced by the generator. Recently, it has been advocated that considering GAN as an energy-based model increases the quality of the generated samples [Tur+19b; Che+20a]. Following [Che+20a], we consider the EBM model induced by the GAN on the latent space, contrary

to [Tur+19b] which works in the observation space. Recall that an EBM is defined by a Boltzmann-Gibbs distribution $p(z) = \mathrm{e}^{-E(z)}/Z$, $z \in \mathbb{R}^d$, where $E(z)$ is the energy function and Z is the normalizing constant. We set $E(z) = -\log p_0(z) - \mathrm{logit}\big(D(G(z))\big)$, where $\mathrm{logit}(y)$, $y \in (0,1)$ is the inverse of the sigmoid function and $p_0(z) = g(z; 0; \mathrm{Id}_d)$. The distribution $p(z)$ would perfectly reproduce the target one even for the imperfect generator, provided that the discriminator is optimal. In most EBMs, samples are generated from $p(z)$ by an MCMC algorithm, either using the Unadjusted Langevin Dynamics (ULA) or standard MCMC algorithms like MALA or HMC; see [Xie+18; Nij+20; SK21] and the references therein. We advocate using Ex^{2}MCMC algorithm instead.

GANs on synthetic data. Following the setting used in [Che+20a, Section 5.1], we apply Ex^{2}MCMC to a WGAN model [ACB17] trained on synthetic datasets. Implementation details and additional experiments are provided in Supplement Material, Section 6.11.12. Our first example is a mixture of $M = 243$ Gaussians in $\mathbb{R}^5$, that is, the target distribution is $M^{-1}\sum_{i=1}^{M} \mathrm{N}(\mu_i, \sigma^2 \mathrm{Id}_5)$ with $\sigma^2 = 1$ and centers $\{\mu_i\}_{i=1}^M$ equally spaced on a uniform grid at $\{-2; 0; 2\}^5$. To assess the sampling performance, we first assign each point x to its closest mode $\mu_{i(x)} = \arg\min_{j\in[M]} \|x - \mu_j\|^2$. The point x is tagged as an *outlier* if $\|x - \mu_{i(x)}\|^2 \geqslant t$, where t is the 95% quantile of the χ^2-distribution with 5 degrees of freedom. We compute *mode-std* as the sample variance of points in the neighborhood of each mode and number of captured modes as number of modes which were assigned with at least one point from the sample. We also compute the empirical Earth Mover's distance (EMD) between the target and the empirical distribution. When applicable, we provide results both for single-start and multi-start regimes (see Section 6.11.12 for more details). Results are summarized in Tables 6.1 and 6.2. This example illustrates that Langevin-based methods explore the support of distribution only in the multi-start regime. At the same time, each particular chain tends to stuck in one of the modes of the latent distribution $p(z)$. At the same time, Ex^{2}MCMC allows to achieve high sampling quality even for the single chain.

GANs for CIFAR-10. In this experiment we investigate performance of Ex^{2}MCMC algorithm for sampling from EBM for GAN on the CIFAR10 dataset. As a GAN model we consider two popular architectures, DC-GAN [RMC16] and SN-GAN [Miy+18].

We compare ULA, i-SIR , and Ex^{2}MCMC (both with correlated and independent proposals) methods. To evaluate sampling quality, we report the values of the energy function $E(z)$, averaged over 1000 independent runs of each sampler. We present the results on Figure 6.7 together with the dynamics of the Frechet Inception Distance (FID, [Heu+17]), computed over the first 600 sampler iterations. Additional implementation details are provided in Section 6.11.13. Note that all methods except for i-SIR improve in terms of average energy function values, and Ex^{2}MCMC algorithms allows for the best exploration.

At the same time, Section 6.6.2 indicates that more accurate sampling from $p(z)$ does not yield the desired FID improvement. Indeed, after 600 MCMC steps the best FID score corresponds to i-SIR procedure, which keeps energy function almost constant. Hence, essentially the issue is not with sampling from $p(z)$, but with calibrating the EBM for $p(z)$ itself. An interesting future research direction is to check, if the discriminator training procedure or calibration could help to improve the model quality. The reported result is hard to compare with the ones in [Che+20a], since the authors of it do not show the dynamics of FID score. We provide additional experiments (including the ones with SN-GAN) and visualizations in Section 6.11.13.

6.5 Conclusions

We propose a new MCMC algorithm, Ex^{2}MCMC, which allows to improve over the competitors due to the efficiency both at exploration and exploitation steps. We analyze its theoretical properties and suggest an adaptive version of the algorithm, FlEx^{2}MCMC, based on normalizing flows. It allows to overcome the issues of MTM and i-SIR algorithms, which are caused by low acceptance rate when

Table 6.1: GAN sampling from 243 Gaussians

	mode std		# captured modes		EMD	
Model	Mult	Single	Mult	Single	Mult	Single
Vanilla GAN	0.040		34.8		3.86	
ULA [Che+20a]	0.039	0.022	98	1	4.20	30.96
MALA	0.040	0.030	85.8	3.5	3.80	24.03
Ex^{2}MCMC	0.039	0.030	89.4	62.9	3.60	3.78

Table 6.2: Results for Swiss Roll dataset

Model	EMD		STV	
	Mult	Single	Mult	Single
WGAN-GP	0.011		0.053	
ULA [Che+20a]	0.010	2.961	0.048	0.91
MALA	0.010	0.055	0.051	0.056
Ex^{2}MCMC	0.011	0.043	0.063	0.046

sampling from high dimensional distributions. Further studies of FlEx^{2}MCMC, in particular its mixing rate, is an interesting direction for the future work.

6.6 Sampling GANs as energy-based model on CIFAR-10

We consider two popular GAN architectures, DC-GAN [RMC16] and SN-GAN [Miy+18]. Below we provide the details on experimental setup and evaluation for both of the models.

6.6.1 DC-GAN

For DC-GAN experiments, we took the pretrained GAN model after 200 epochs from the open repository `https://github.com/csinva/gan-vae-pretrained-pytorch`, to compute FID we took code from the repository `https://github.com/abdulfatir/gan-metrics-pytorch`.

For DC-GAN, the latent dimension equals $d = 100$. Following [Che+20a], we consider sampling from the latent space distribution

$$p(z) = \mathrm{e}^{-E(z)}/Z\ , \quad z \in \mathbb{R}^d\ , \quad E(z) = -\log p_0(z) - \mathrm{logit}\big(D(G(z))\big)\ , \tag{6.17}$$

where $\mathrm{logit}(y) = \log\left(y/(1-y)\right)$, $y \in (0,1)$ is the inverse of the sigmoid function, and $p_0(z) = g(z; 0; \mathrm{Id}_d)$. For the step size $\eta > 0$ we define $(k+1)$-th iteration of the Unadjusted Langevin Algorithm as

$$Z_{k+1} = Z_k - \eta \nabla E(Z_k) + \sqrt{2\eta}\varepsilon_{k+1}\ , \varepsilon_{k+1} \sim \mathcal{N}(0, \mathrm{Id}_d)\ . \tag{6.18}$$

Note that the ergodic distribution of the corresponding continuous-time diffusion is given by $p(z) = \mathrm{e}^{-E(z)}/Z$. We specify our choice of η in Table 6.6.

Evaluation protocol We run $n = 600$ iterations of the ULA, i-SIR , and Ex^{2}MCMC algorithm in its correlated and vanilla versions. For the vanilla Ex^{2}MCMC algorithm (Algorithm 5), we use the Markov kernel (6.80), corresponding to 1 MALA step, as the rejuvenation kernel. Step size γ, reported for Ex^{2}MCMC algorithm, corresponds to its rejuvenation MALA kernel. For the correlated Ex^{2}MCMC algorithm (see Algorithm 6), we bypass the rejuvenation step, since local exploration is guaranteed by selecting large α. Additional details are provided at Table 6.3.

We run $N = 50000$ independent chains for each of the mentioned MCMC algorithms. Then, for $j-$th iteration, we calculate average value of the energy function $E(z)$ averaged over $M = 1000$ chains. Every 40 MCMC iterations we calculate FID based on 50000 images generated on the current MCMC step and 50000 images from the training data. We report FID results after 600 MCMC iterations at Table 6.5.

6.6.2 SN-GAN

For SN-GAN, we took an implementation available at `https://github.com/pfnet-research/sngan_projection` to reproduce results with the unconditional version of SN-GAN pretrained on CIFAR-10, to compute FID we took statistics from the repository `https://github.com/pfnet-research/chainer-gan-lib/blob/master/common/cifar-10-fid.npz`, used in the SN-GAN repository. We additionally calibrated SN-GAN discriminator as suggested at [Aza+19], by replacing its top linear layer with fully-connected one consisting of 3 consecutive linear layers and finetuning them with binary cross entropy loss for $5k$ iterations.

Method	**sampl. steps**	γ	# particles, N	ϵ	α	**num MALA steps**
ULA	600	0.01	–	–	–	–
i-SIR	600	–	5	–	–	–
Vanilla Ex^{2}MCMC	600	0.02	5	–	–	1
Correlated Ex^{2}MCMC	600	–	5	0.9	0.99	0

Table 6.3: CIFAR-10 hyperparameters for DC-GAN architecture.

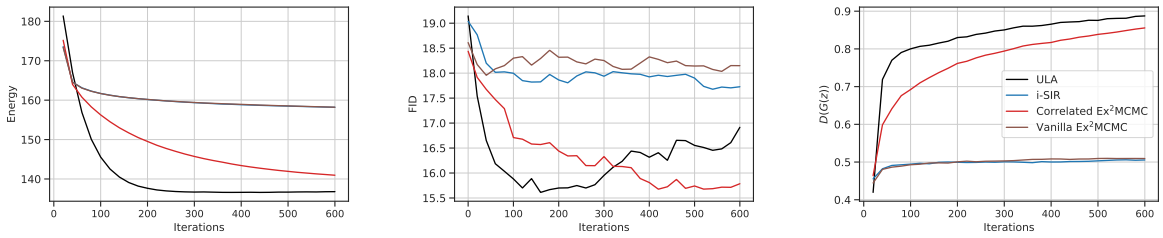


Figure 6.8: CIFAR-10 dataset with SN-GAN architecture: From left to right, average energy values, FID and discriminator scores for first 600 sampling iterations.

For SN-GAN, generator is a mapping $G(z) : \mathbb{R}^d \mapsto \mathbb{R}^{\mathsf{D}}$ with latent dimension $d = 128$ and ambient space dimension $\mathsf{D} = 784$. Following [Che+20a], we performed an additional tempering of the distribution, that is, we sampled from the energy-based model

$$p(z) = \mathrm{e}^{-E(z)/T}/Z\,, \quad z \in \mathbb{R}^d\,, \quad E(z) = -\log p_0(z) - \mathrm{logit}\big(D(G(z))\big)\,. \tag{6.19}$$

In our experiments we observe, that simply setting $T = 1$ does not yield significant sampling quality improvement, compared to vanilla GAN sampling. At the same time, setting $T = 1/4$ allows to efficiently reduce the FID values, as reported in Figure 6.8.

Note that i-SIR and Ex2-MCMC with independent proposals fail to improve sampling quality compared to the vanilla GAN sampling, as it can be observed from Section 6.6.2-(a), and average discriminator scores $D(G(z))$, see Section 6.6.2-(c). Indeed, average discriminator score does not increase. At the same time, both ULA and Ex^{2}MCMC with correlated particles shows considerable energy score improvement and better explore the regions of $p(z)$ with large values $D(G(z))$. Ex^{2}MCMC shows more stable behaviour in terms of FID metrics, reported in Section 6.6.2-(b).

The evaluation protocol for SN-GAN follows the one for DC-GAN but with some adjustments, see the value of hyperparameters in Table 6.4. We run $N = 5000$ independent chains for each of the mentioned MCMC algorithms. Then, for $j-$th iteration, we calculate average value of the energy function $E(z)$ averaged over N chains. Every 20 MCMC iterations we calculate FID based on 5000 images generated on the current MCMC step and 5000 images from the training data. We report FID results after 600 MCMC iterations at Table 6.5.

6.7 Proofs of main theoretical results of Section 6.2

Notations For $k \in \mathbb{N}$, $m, m' \in \mathbb{N}^*$ and Ω, Ω' two open sets of $\mathbb{R}^m, \mathbb{R}^{m'}$ respectively, denote by $\mathrm{C}^k(\Omega, \Omega')$, the set of k-times continuously differentiable functions. For $f \in \mathrm{C}^2(\mathbb{R}^d, \mathbb{R})$, denote by ∇f the gradient of f and by Δf the Laplacian of f. For $k \in \mathbb{N}$ and $f \in \mathrm{C}^k(\mathbb{R}^d, \mathbb{R})$, denote by $\mathrm{D}^i f$ the i-th order differential of f for $i \in \{0, \dots, k\}$. For $x \in \mathbb{R}^d$ and $i \in \{1, \dots, k\}$, define $\left\|\mathrm{D}^0 f(x)\right\| = |f(x)|$, $\left\|\mathrm{D}^i f(x)\right\| = \sup_{u_1,\dots,u_i \in \mathbb{S}(\mathbb{R}^d)} \mathrm{D}^i f(x)[u_1, \dots, u_i]$. For $k, p \in \mathbb{N}$ and $f \in \mathrm{C}^k(\mathbb{R}^d, \mathbb{R})$, define

$$\|f\|_{k,p} = \sup_{x \in \mathbb{R}^d,\, i \in \{0,\dots,k\}} \left\|\mathrm{D}^i f(x)\right\| / (1 + \|x\|^p)\,. \tag{6.20}$$

Method	**sampl. steps**	γ	# particles, N	ϵ	α	**num MALA steps**
ULA	600	0.0025	–	–	–	–
i-SIR	600	–	4	–	–	–
Vanilla Ex^{2}MCMC	600	0.005	4	–	–	1
Correlated Ex^{2}MCMC	600	–	4	0.5	0.995	0

Table 6.4: CIFAR-10 hyperparameters for SN-GAN architecture.

Table 6.5: FID for CIFAR-10 GAN-based models

Model	SN-GAN	DC-GAN
Vanilla GAN	18.9	35.8
ULA [Che+20a]	17.0	34.8
i-SIR	17.7	28.5
Correlated Ex^{2}MCMC	15.8	35.2

Define $\mathrm{C}^k_{\mathrm{poly}}(\mathbb{R}^d, \mathbb{R}) = \left\{ f \in \mathrm{C}^k(\mathbb{R}^d, \mathbb{R}) : \inf_{p \in \mathbb{N}} \|f\|_{k,p} < +\infty \right\}$ and for any $f \in \mathrm{C}^k_{\mathrm{poly}}(\mathbb{R}^d, \mathbb{R})$, we consider the semi-norm

$$\|f\|_k = \|f\|_{k,p} \text{ where } p = \min\{q \in \mathbb{N} : \|f\|_{k,q} < +\infty\} . \tag{6.21}$$

Finally, define $\mathrm{C}^\infty_{\mathrm{poly}}(\mathbb{R}^d, \mathbb{R}) = \cap_{k \in \mathbb{N}} \mathrm{C}^k_{\mathrm{poly}}(\mathbb{R}^d, \mathbb{R})$.

In the sequel, we denote by $w(x)$ the normalized weight function, that is,

$$\Pi(\mathrm{d}x) = w(x)\Lambda(\mathrm{d}x) . \tag{6.22}$$

6.7.1 Proof of Lemma 46

By symmetrisation, note that

$$\mathsf{P}_N(x, \mathsf{A}) = \int \delta_x(\mathrm{d}x_1) \sum_{i=1}^N \frac{w(x^i)}{\sum_{j=1}^N w(x^j)} \mathbb{1}_{\mathsf{A}}(x^i) \prod_{j=2}^N \Lambda(\mathrm{d}x^j) \tag{6.23}$$

$$= \frac{1}{N} \int \sum_{\ell=1}^N \delta_x(\mathrm{d}x_\ell) \prod_{j \mathrm{ne} \ell} \Lambda(\mathrm{d}x^j) \sum_{i=1}^N \frac{w(x^i)}{\sum_{\ell=1}^N w(x^\ell)} \mathbb{1}_{\mathsf{A}}(x^i) . \tag{6.24}$$

Then,

$$\int \Pi(\mathrm{d}x) \mathsf{P}_N(x, \mathsf{A}) = N^{-1} \int \Pi(\mathrm{d}x) \sum_{\ell=1}^N \delta_x(\mathrm{d}x_\ell) \prod_{j \mathrm{ne} \ell} \Lambda(\mathrm{d}x^j) \sum_{i=1}^N \frac{w(x^i)}{\sum_{\ell=1}^N w(x^\ell)} \mathbb{1}_{\mathsf{A}}(x^i) \tag{6.25}$$

$$= N^{-1} \int \Big(\sum_{\ell=1}^N w(x_\ell)\Big) \prod_{j=1}^N \Lambda(\mathrm{d}x^j) \sum_{i=1}^N \frac{w(x^i)}{\sum_{\ell=1}^N w(x^\ell)} \mathbb{1}_{\mathsf{A}}(x^i) \tag{6.26}$$

$$= N^{-1} \int \prod_{j=1}^N \Lambda(\mathrm{d}x^j) \sum_{i=1}^N w(x^i) \mathbb{1}_{\mathsf{A}}(x^i) = \Pi(\mathsf{A}) \tag{6.27}$$

6.7.2 Proof of Theorem 48

We preface the proof by a technical lemma.

Lemma 51. *Let $Y^{1:M}$ be i.i.d. random variables, such that $\mathbb{E}[Y_1] = 1$, and $\mathbb{P}(Y_1 \in [0, \mathrm{L}]) = 1$. Then for $S = \sum_{i=1}^M Y_i$ and $a, b > 0$*

$$\mathbb{E}\left[(a + bS)^{-1}\right] \leqslant (a + bM/2)^{-1} + (1/a)\exp(-M/(2\mathrm{L}^2)). \tag{6.28}$$

Proof. Let $K \geqslant 0$. Then we get

$$\frac{1}{a + bS} = \frac{1}{a + bS} \mathbb{1}\{S < K\} + \frac{1}{a + bS} \mathbb{1}\{S \geqslant K\} \tag{6.29}$$

$$\leqslant \frac{1}{a + bK} \mathbb{1}\{S \geqslant K\} + \frac{1}{a + bS} \mathbb{1}\{S < K\} \leqslant \frac{1}{a + bK} + \frac{1}{a} \mathbb{1}\{S < K\} \tag{6.30}$$

and in particular, $\mathbb{E}[(a+bS)^{-1}] \leqslant (a+bK)^{-1} + a^{-1}\mathbb{P}(S<K)$. By Hoeffding's inequality,

$$\mathbb{P}(S<K) = \mathbb{P}(S-M < -(M-K)) \leqslant \exp(-2(M-K)^2/(M\mathrm{L}^2)). \tag{6.31}$$

In particular, for $K = M/2$, we have $\mathbb{P}(S<K) \leqslant \exp(-M/(2\mathrm{L}^2))$. □

Proof of Theorem 48. (i) Under **H**6, we have, for $(x,\mathsf{A}) \in \mathbb{X}\times\mathcal{X}$,

$$\mathsf{P}_N(x,\mathsf{A}) = \int \delta_x(\mathrm{d}x^1) \sum_{i=1}^N \frac{w(x^i)}{\sum_{j=1}^N w(x^j)} \mathbb{1}_{\mathsf{A}}(x^i) \prod_{j=2}^N \Lambda(\mathrm{d}x^j) \tag{6.32}$$

$$= \int \frac{w(x)}{w(x)+\sum_{j=2}^N w(x^j)} \mathbb{1}_{\mathsf{A}}(x) \prod_{j=2}^N \Lambda(\mathrm{d}x^j) + \int \sum_{i=2}^N \frac{w(x^i)}{w(x)+\sum_{j=2}^N w(x^j)} \mathbb{1}_{\mathsf{A}}(x^i) \prod_{j=2}^N \Lambda(\mathrm{d}x^j) \tag{6.33}$$

$$\geqslant \sum_{i=2}^N \int \frac{w(x^i)}{w(x)+w(x^i)+\sum_{j=2,j\neq i}^N w(x^j)} \mathbb{1}_{\mathsf{A}}(x^i) \prod_{j=2}^N \Lambda(\mathrm{d}x^j) \tag{6.34}$$

$$\geqslant \sum_{i=2}^N \int \Pi(\mathrm{d}x^i)\mathbb{1}_{\mathsf{A}}(x^i) \int \frac{1}{w(x)+w(x^i)+\sum_{j=2,j\neq i}^N w(x^j)} \prod_{j=2,j\neq i}^N \Lambda(\mathrm{d}x^j)\,. \tag{6.35}$$

Finally, since the function $f\colon z \mapsto (z+a)^{-1}$ is convex on $\mathbb{R}_+$ and $a>0$, we get for $i \in \{2,\dots,N\}$,

$$\int \frac{1}{w(x)+w(x^i)+\sum_{j=2,j\neq i}^N w(x^j)} \prod_{j=2,j\neq i}^N \Lambda(\mathrm{d}x^j) \tag{6.36}$$

$$\geqslant \frac{1}{\int w(x)+w(x^i)+\sum_{j=2,j\neq i}^N w(x^j) \prod_{j=2,j\neq i}^N \Lambda(\mathrm{d}x^j)} \tag{6.37}$$

$$\geqslant \frac{1}{w(x)+w(x^i)+N-2} \geqslant \frac{1}{2\mathrm{L}+N-2}\,. \tag{6.38}$$

We finally obtain the inequality

$$\mathsf{P}_N(x,\mathsf{A}) \geqslant \Pi(\mathsf{A}) \times \frac{N-1}{2\mathrm{L}+N-2} = \epsilon_N \Pi(\mathsf{A})\,. \tag{6.39}$$

This means that the whole space $\mathbb{X}$ is $(1,\epsilon_N\Pi)$-small (see [Dou+18, Definition 9.3.5]). Since $\mathsf{P}_N(x,\cdot)$ and Π are probability measures, (6.39) implies

$$\|\mathsf{P}_N(x,\cdot) - \Pi\|_{\mathrm{TV}} = \sup_{\mathsf{A}\in\mathcal{X}} |\mathsf{P}_N(x,\mathsf{A}) - \Pi(\mathsf{A})| \leqslant 1-\epsilon_N = \kappa_N\,. \tag{6.40}$$

Now the statement follows from [Dou+18, Theorem 18.2.4] applied with $m=1$.

(ii) Let $V : \mathbb{X} \to [1,\infty)$ be a measurable function such that $\Pi(V)<\infty$ and $\Lambda(V)<\infty$. We aim to check first the drift condition

$$\mathsf{P}_N V(x) \leqslant \kappa_N V(x) + \mathrm{b}_N \tag{6.41}$$

with the constants κ_N and b_N defined in Theorem 48 and (6.51), respectively. Setting $x^1 = x$, we obtain

$$\mathsf{P}_N V(x) = \int \sum_{i=1}^N \frac{w(x^i)}{\sum_{j=1}^N w(x^j)} V(x^i) \prod_{j=2}^N \Lambda(\mathrm{d}x^j) \tag{6.42}$$

$$= V(x) \int \frac{w(x)}{\sum_{j=1}^N w(x^j)} \prod_{j=2}^N \Lambda(\mathrm{d}x^j) + \int \sum_{i=2}^N \frac{w(x^i)}{\sum_{j=1}^N w(x^j)} V(x^i) \prod_{j=2}^N \Lambda(\mathrm{d}x^j). \tag{6.43}$$

We bound these two terms separately.

$$V(x)\int \frac{w(x)}{\sum_{j=1}^N w(x^j)}\prod_{j=2}^N \Lambda(\mathrm{d}x^j) = V(x)\int \left(1-\frac{\sum_{j=2}^N w(x^j)}{\sum_{j=1}^N w(x^j)}\right)\prod_{j=2}^N \Lambda(\mathrm{d}x^j) \tag{6.44}$$

$$= V(x)\left(1-\sum_{k=2}^N \int \frac{w(x^k)}{w(x)+w(x^k)+\sum_{j=2,j\neq k}^N w(x^j)}\prod_{j=2}^N \Lambda(\mathrm{d}x^j)\right) \tag{6.45}$$

$$= V(x)\left(1-\sum_{k=2}^N \int \frac{\pi(\mathrm{d}x^k)}{w(x)+w(x^k)+\sum_{j=2,j\neq k}^N w(x^j)}\prod_{j=2,j\neq k}^N \Lambda(\mathrm{d}x^j)\right) . \tag{6.46}$$

From (6.36), we get that

$$V(x)\int \frac{w(x)}{\sum_{j=1}^N w(x^j)}\prod_{j=2}^N \Lambda(\mathrm{d}x^j) \leqslant V(x)\left(1-\frac{N-1}{2\mathrm{L}+N-2}\right) = \kappa_N V(x). \tag{6.47}$$

Moreover, we have

$$\int \sum_{i=2}^N \frac{w(x^i)V(x^i)}{\sum_{j=1}^N w(x^j)}\prod_{j=2}^N \Lambda(\mathrm{d}x^j) = (N-1)\int \frac{w(x^2)V(x^2)\Lambda(\mathrm{d}x^2)}{w(x)+w(x^2)+\sum_{j=3}^N w(x^j)}\prod_{j=3}^N \Lambda(\mathrm{d}x^j) . \tag{6.48}$$

Since the function $z \mapsto z/(z+a)$ is concave on $\mathbb{R}_+$ for $a>0$, we have

$$\begin{aligned}
&\int \frac{w(x^2)}{w(x)+w(x^2)+\sum_{j=3}^N w(x^j)} V(x^2)\Lambda(\mathrm{d}x^2) \qquad (6.49)\\
&= \Lambda(V)\int \frac{w(x^2)}{w(x^2)+w(x)+\sum_{j=3}^N w(x^j)}\frac{V(x^2)\Lambda(\mathrm{d}x^2)}{\Lambda(V)}\\
&\leqslant \Lambda(V)\frac{\int w(x^2)V(x^2)\Lambda(\mathrm{d}x^2)/\Lambda(V)}{\int w(x^2)V(x^2)\Lambda(\mathrm{d}x^2)/\Lambda(V)+w(x)+\sum_{j=3}^N w(x^j)} \leqslant \frac{\Pi(V)}{\Pi(V)/\Lambda(V)+w(x)+\sum_{j=3}^N w(x^j)}.
\end{aligned}$$

Using Lemma 51, with $Y_i = w(x^i)$,

$$\int \frac{\Pi(V)}{\Pi(V)/\Lambda(V)+\sum_{j=3}^N w(x^j)}\prod_{j=3}^N \Lambda(\mathrm{d}x^j) \leqslant \frac{\Pi(V)}{\Pi(V)/\Lambda(V)+(N-2)/2} + \Lambda(V)\exp(-(N-2)/2\mathrm{L}^2) . \tag{6.50}$$

Writing in this case

$$b_N = \frac{\Pi(V)(N-1)}{\Pi(V)/\Lambda(V)+(N-2)/2} + \Lambda(V)(N-1)\exp(-(N-2)/2\mathrm{L}^2) \tag{6.51}$$

concludes the proof.
Moreover, (i) implies that for any $d_N > 1$ the level sets $\{x \in \mathbb{X} : V(x) \leqslant d_N\}$ are $(1, \epsilon_N\Pi)$-small. Let us choose $d_N = 1 \vee 4\mathrm{b}_N/(1-\kappa_N) - 1 \vee 2/\kappa_N$. Then $\kappa_N + 2\mathrm{b}_N/(1+d_N) < 1$, and [Dou+18, Theorem 19.4.1] implies

$$\|\mathsf{P}_N^n(x,\cdot) - \Pi\|_V \leqslant c_N\{V(x)+\Pi(V)\}\tilde{\kappa}_N^n , \tag{6.52}$$

where the constants $\tilde{\kappa}_N$ and c_N are given by

$$\log\tilde{\kappa}_N = \frac{\log\kappa_N \log\bar{\lambda}_N}{\left(\log\kappa_N + \log\bar{\lambda}_N - \log\bar{b}_N\right)} , \quad c_N = (\bar{\lambda}_N+1)(1+\bar{b}_N/[\kappa_N(1-\bar{\lambda}_N)]) \tag{6.53}$$

$$\bar{\lambda}_N = \kappa_N + 2\mathrm{b}_N/(1+d_N) , \ \bar{b}_N = \kappa_N \mathrm{b}_N + d_N , d_N = 1 \vee 4\mathrm{b}_N/(1-\kappa_N) - 1 \vee 2/\kappa_N . \tag{6.54}$$

In the expression above we used the fact that $1 - \epsilon_N = \kappa_N$. The choice of d_N in (6.53) implies $2\mathrm{b}_N/(1 + d_N) \leqslant \mathrm{b}_N\kappa_N$. Then the elementary calculations imply $\kappa_N \leqslant \bar{\lambda}_N \leqslant (\mathrm{b}_N + 1)\kappa_N$ and

$$\tilde{\kappa}_N \leqslant \kappa_N^{\theta_N} , \tag{6.55}$$

where

$$\theta_N = \frac{\log(1/\bar{\lambda}_N)}{\big(\log(1/\kappa_N) + \log(1/\bar{\lambda}_N) + \log \bar{b}_N\big)} \geqslant \frac{\log(1/\kappa_N) - \log(\mathrm{b}_N + 1)}{2\log(1/\kappa_N) + \log \bar{b}_N} = \frac{1}{3} + \alpha_N . \tag{6.56}$$

In the expression above $\alpha_N \to 0$ when $N \to \infty$, since $\mathrm{b}_N \leqslant 4\Pi(V)$ for $N \geqslant 3$. □

6.7.3 Proof of Theorem 49

We preface the proof with some preparatory lemmas.

Lemma 52. *Let* $\mathsf{K} \subset \mathbb{X}$*, such that* $\sup_{x\in\mathsf{K}} w(x) < w_{\infty,\mathsf{K}} < \infty$ *and* $\Pi(\mathsf{K}) > 0$*. Then, for all* $(x, \mathsf{A}) \in \mathsf{K} \times \mathcal{X}$*,*

$$\mathsf{P}_N(x, \mathsf{A}) \geqslant \epsilon_{N,K}\Pi_K(\mathsf{A}) , \tag{6.57}$$

with $\epsilon_{N,\mathsf{K}} = (N - 1)\Pi(\mathsf{K})/[2w_{\infty,\mathsf{K}} + N - 2]$ *and* $\Pi_{\mathsf{K}}(\mathsf{A}) = \Pi(\mathsf{A} \cap \mathsf{K})/\Pi(\mathsf{K})$*.*

Note that if the weight function w is continuous, then for any compact K, $\sup_{x\in\mathsf{K}} w(x) < w_{\infty,\mathsf{K}} < \infty$.

Proof. Let $(x, \mathsf{A}) \in \mathbb{X} \times \mathcal{X}$. Then

$$\mathsf{P}_N(x, \mathsf{A}) = \int \frac{w(x)}{w(x) + \sum_{j=2}^N w(x^j)} \mathbb{1}_{\mathsf{A}}(x) \prod_{j=2}^N \Lambda(\mathrm{d}x^j) + \int \sum_{i=2}^N \frac{w(x^i)}{w(x) + \sum_{j=2}^N w(x^j)} \mathbb{1}_{\mathsf{A}}(x^i) \prod_{j=2}^N \Lambda(\mathrm{d}x^j) \tag{6.58}$$

$$\geqslant \sum_{i=2}^N \int \frac{w(x^i)}{w(x) + w(x^i) + \sum_{j=2, j\neq i}^N w(x^j)} \mathbb{1}_{\mathsf{A}}(x^i) \prod_{j=2}^N \Lambda(\mathrm{d}x^j) \tag{6.59}$$

$$\geqslant \sum_{i=2}^N \int \Pi(\mathrm{d}x^i)\mathbb{1}_{\mathsf{A}}(x^i) \int \frac{1}{w(x) + w(x^i) + \sum_{j=2, j\neq i}^N w(x^j)} \prod_{j=2, j\neq i}^N \Lambda(\mathrm{d}x^j) \tag{6.60}$$

$$\geqslant (N - 1) \int \Pi(\mathrm{d}x^1)\mathbb{1}_{\mathsf{A}}(x^1) \frac{1}{w(x) + w(x^1) + N - 2} , \tag{6.61}$$

where the last inequality follows from Jensen's inequality and the convexity of the function $z \mapsto (z+a)^{-1}$ on $\mathbb{R}_+$. Now,

$$P_N(x, \mathsf{A}) \geqslant (N - 1) \int \Pi(\mathrm{d}x^1)\mathbb{1}_{\mathsf{A}\cap\mathsf{K}}(x^1) \frac{1}{w(x) + w(x^1) + N - 2} \tag{6.62}$$

$$\geqslant \frac{N - 1}{2w_{\infty,\mathsf{K}} + N - 2} \int \Pi(\mathrm{d}y)\mathbb{1}_{\mathsf{A}\cap\mathsf{K}}(y) = \frac{(N - 1)\Pi(\mathsf{K})}{2w_{\infty,\mathsf{K}} + N - 2}\Pi_{\mathsf{K}}(\mathsf{A}) . \tag{6.63}$$

□

Lemma 53. *Let* P *be a Markov kernel on* $(\mathbb{X}, \mathcal{X})$*,* γ *be a probability measure on* $(\mathbb{X}, \mathcal{X})$*, and* $\epsilon > 0$*. Let also* $\mathsf{C} \in \mathcal{X}$ *be an* $(1, \epsilon\gamma)$*-small set for* P*. Then for arbitrary Markov kernel* Q *on* $(\mathbb{X}, \mathcal{X})$*, the set* C *is an* $(1, \epsilon\gamma_{\mathrm{Q}})$*-small set for* PQ*, where* $\gamma_{\mathrm{Q}}(\mathsf{A}) = \int \gamma(\mathrm{d}y)\mathrm{Q}(y, \mathsf{A})$ *for* $\mathsf{A} \in \mathcal{X}$*.*

Proof. Let $(x, \mathsf{A}) \in \mathsf{C} \times \mathcal{X}$. Then it holds

$$\mathrm{PQ}(x, \mathsf{A}) = \int \mathrm{P}(x, \mathrm{d}y)\mathrm{Q}(y, \mathsf{A}) \geqslant \epsilon \int_{\mathsf{C}} \gamma(\mathrm{d}y)\mathrm{P}(y, \mathsf{A}) = \epsilon\gamma(\mathsf{A}) . \tag{6.64}$$

□

Lemma 54. *Let* P *and* Q *be two irreducible Markov kernels with* Π *as their unique invariant distribution. Let* $V : \mathbb{X} \to [1, \infty)$ *be a measurable function. Assume that there exists* $\lambda_Q \in [0, 1)$ *and* $\mathsf{b}_\mathrm{P}, \mathsf{b}_\mathrm{Q} \in \mathbb{R}_+$ *such that* $\mathrm{P}V(x) \leqslant V(x) + \mathsf{b}_\mathrm{P}$ *and* $\mathrm{Q}V(x) \leqslant \lambda_\mathrm{Q} V(x) + \mathsf{b}_\mathrm{Q}$*. Let* $d_0 > 1$*. Assume in addition, that for all* $d \geqslant d_0$*, there exist* $\epsilon_d > 0$ *and a probability measure* γ_d *such that for all* $(x, \mathsf{A}) \in \mathsf{V}_d \times \mathcal{X}$*,* $\mathrm{P}(x, \mathsf{A}) \geqslant \epsilon_d \gamma_d(\mathsf{A})$*, where* $\mathsf{V}_d = \{x \in \mathbb{X} : V(x) \leqslant d\}$*. Define* $\mathrm{K} = \mathrm{PQ}$ *and* $\lambda_\mathrm{K} = \lambda_\mathrm{Q}$*,* $\mathsf{b}_\mathrm{K} = \mathsf{b}_\mathrm{P} + \mathsf{b}_\mathrm{Q}$*. Then,*

$$\mathrm{K}V(x) \leqslant \lambda_\mathrm{K} V + \mathsf{b}_\mathrm{K} \text{ and, for all } x \in \mathsf{V}_d,\ \mathrm{K}(x, \mathsf{A}) \geqslant \epsilon_d \gamma_{\mathrm{Q},d}(\mathsf{A}),$$

where $\gamma_{\mathrm{Q},d}(\mathsf{A}) = \int \gamma_d(\mathrm{d}y)\mathrm{Q}(y, \mathsf{A})$*.*

Let $d \geqslant d_0$ *be such that* $\lambda_\mathrm{K} + 2\mathsf{b}_\mathrm{K}/(1 + d) < 1$*. Then, for any* $x \in \mathbb{X}$ *and* $k \in \mathbb{N}$*,*

$$\|\mathrm{K}^k(x, \cdot) - \Pi\|_V \leqslant c_\mathrm{K}\{V(x) + \Pi(V)\}\rho_\mathrm{K}^k$$

with

$$\rho_\mathrm{K} = \frac{\log(1 - \epsilon_d) \log \bar{\lambda}_\mathrm{K}}{\log(1 - \epsilon_d) + \log \bar{\lambda}_\mathrm{K} - \log \bar{b}_\mathrm{K}}\ , \tag{6.65}$$

with $\bar{\lambda}_\mathrm{K} = \lambda_\mathrm{K} + 2\mathsf{b}_\mathrm{K}/(1 + d)$*,* $\bar{b}_\mathrm{K} = \lambda_\mathrm{K}\mathsf{b}_\mathrm{K} + d$*, and* $c_\mathrm{K} = (\lambda_\mathrm{K} + 1)(1 + \bar{b}_\mathrm{K}/[(1 - \epsilon_{\mathsf{V}_d})(1 - \bar{\lambda}_\mathrm{K})])$*.*

Proof. By Lemma 53, we have directly that for any $(x, \mathsf{A}) \in \mathsf{V}_d \times \mathcal{X}$, $\mathrm{K}(x, \mathsf{A}) \geqslant \epsilon_d \gamma_{\mathrm{Q},d}(\mathsf{A})$. Moreover, for any $x \in \mathbb{X}$, $\mathrm{K}V(x) = \mathrm{PQ}V(x) \leqslant \lambda_\mathrm{Q}\mathrm{P}V(x) + \mathsf{b}_\mathrm{Q} \leqslant \lambda_\mathrm{Q} V(x) + \mathsf{b}_\mathrm{Q} + \mathsf{b}_\mathrm{P}$. The proof is completed with [Dou+18, Theorem 19.4.1]. □

Proof of Theorem 55. Note first, that the Markov kernel of Ex^{2}MCMC algorithm can be represented as a composition $\mathsf{K}_N = \mathsf{P}_N\mathsf{R}$ with R being the rejuvenation kernel. Applying the same symmetrisation argument as (6.23), we write K_N for $(x, \mathsf{A}) \in \mathbb{X} \times \mathcal{X}$ as

$$\mathsf{K}_N(x, \mathsf{A}) = \frac{1}{N} \int \sum_{\ell=1}^N \delta_x(\mathrm{d}x_\ell) \prod_{i \neq \ell} \Lambda(\mathrm{d}x^i) \sum_{i=1}^N \frac{w(x^i)}{\sum_{\ell=1}^N w(x^\ell)} \mathsf{R}(x^i, \mathsf{A})\ . \tag{6.66}$$

Applying (6.42)-(6.48)-(6.49), we get

$$\mathsf{P}_N V(x) = V(x) \int \frac{w(x)}{w(x) + \sum_{j=2}^N w(x^j)} \prod_{j=2}^N \Lambda(\mathrm{d}x^j) + \int \sum_{i=2}^N \frac{w(x^i)}{\sum_{j=1}^N w(x^j)} V(x^i) \prod_{j=2}^N \Lambda(\mathrm{d}x^j) \tag{6.67}$$

$$\leqslant V(x) + (N - 1)U_N \quad \text{with} \quad U_N = \int \frac{\Pi(V)}{\Pi(V)/\Lambda(V) + w(x) + \sum_{j=3}^N w(x^j)} \prod_{j=3}^N \Lambda(\mathrm{d}x^j) \tag{6.68}$$

On the other hand, using (6.29) with $K = (N - 2)/2$ together with Markov inequality,

$$U_N \leqslant \frac{\Pi(V)}{\Pi(V)/\Lambda(V) + (N - 2)/2} + \Lambda(V) \int \mathbb{1}_{\{\sum_{j=3}^N \{w(x^j) - 1\} \leqslant -(N-2)/2\}} \prod_{j=3}^N \Lambda(\mathrm{d}x^j)\ , \tag{6.69}$$

$$\leqslant \frac{\Pi(V)}{\Pi(V)/\Lambda(V) + (N - 2)/2} + \frac{4\Lambda(V)\,\mathrm{Var}_\Lambda[w]}{N - 2}\ , \tag{6.70}$$

where $\mathrm{Var}_\Lambda[w] = \int \{w(x) - 1\}^2 \Lambda(\mathrm{d}x)$ is the variance of the normalized weight functions under the proposal distribution. Combining the above results, for any $x \in \mathbb{X}$,

$$\mathsf{P}_N V(x) \leqslant V(x) + \mathsf{b}_{\mathsf{P}_N}\ , \text{ where } \mathsf{b}_{\mathsf{P}_N} = \frac{\Pi(V)(N - 1)}{\Pi(V)/\Lambda(V) + (N - 2)/2} + \frac{4(N - 1)\Lambda(V)\,\mathrm{Var}_\Lambda[w]}{N - 2}\ . \tag{6.71}$$

Assumption **H** 7-(i) implies $\mathsf{R}V(x) \leqslant \lambda_\mathsf{R} V(x) + \mathsf{b}_\mathsf{R}$. Assumption **H** 7-(iii) together with Lemma 52 implies that the level sets V_d are $(1, \epsilon_{d,N}\gamma_d)$-small for the Markov kernel P_N. Here the probability measure γ_d and $\epsilon_{d,N}$ are given, for any $\mathsf{A} \in \mathcal{X}$, by

$$\begin{aligned} \gamma_d(\mathsf{A}) &= \int \Pi_{\mathsf{V}_d}(\mathrm{d}y)\mathsf{R}(y, \mathsf{A})\ , \text{ where } \Pi_{\mathsf{V}_d}(B) = \Pi(B \cap \mathsf{V}_d)/\Pi(\mathsf{V}_d), B \in \mathcal{X}\ , \\ \epsilon_{d,N} &= (N - 1)\Pi(\mathsf{V}_d)/[2w_{\infty,d} + N - 2]\ . \end{aligned} \tag{6.72}$$

Hence all conditions of Lemma 54 are satisfied. Choose $d_N = 1 \vee 4(\mathsf{b}_\mathsf{R} + \mathsf{b}_{\mathsf{P}_N})/(1-\lambda_\mathsf{R}) - 1$. Then $\lambda_\mathsf{R} + 2(\mathsf{b}_\mathsf{R} + \mathsf{b}_{\mathsf{P}_N})/(1+d_N) < 1$, and Lemma 54 implies for any $x \in \mathbb{X}$ and $k \in \mathbb{N}$,

$$\|\mathsf{K}_N^k(x,\cdot) - \Pi\|_V \leqslant c_{N,\mathsf{R}}\{V(x) + \Pi(V)\}\lambda_\mathrm{r}[N,\mathsf{R}]^k \ ,$$

where the constants $c_{N,\mathsf{R}}$ and $\lambda_\mathrm{r}[N,\mathsf{R}]$ are given by

$$\log\lambda_\mathrm{r}[N,\mathsf{R}] = \frac{\log(1-\epsilon_{d,N})\log\bar\lambda_\mathsf{R}}{\log(1-\epsilon_{d,N}) + \log\bar\lambda_\mathsf{R} - \log\bar b_\mathsf{R}} \ , \quad c_{N,\mathsf{R}} = (\lambda_\mathsf{R}+1)(1+\bar b_\mathsf{R}/[(1-\epsilon_{d,N})(1-\bar\lambda_\mathsf{R})]) \tag{6.73}$$

$$\bar\lambda_\mathsf{R} = \lambda_\mathsf{R} + 2\mathsf{b}_\mathsf{R}/(1+d) \ , \ \bar b_\mathsf{R} = \lambda_\mathsf{R}\mathsf{b}_\mathsf{R} + d_N \ , d_N = 1 \vee 4(\mathsf{b}_\mathsf{R} + \mathsf{b}_{\mathsf{P}_N})/(1-\lambda_\mathsf{R}) - 1 \ , \tag{6.74}$$

and $\epsilon_{d,N}$ defined in (6.72). It is easy to see from this expression that, for d_N being fixed, $\lambda_\mathrm{r}[N,\mathsf{R}]$ decreases with $N \to \infty$. □

6.7.4 Proof of Theorem 50

The algorithm Ex^{2}MCMC defines a Markov chain $\{Y_j,\ j \in \mathbb{N}\}$ with Markov kernel

$$\mathsf{C}_N(x,\mathsf{A}) = \frac{1}{N}\int \sum_{j=1}^N \delta_x(\mathrm{d}x^j)\mathsf{Q}_j\big(x^j, \mathrm{d}x^{1:N\setminus\{j\}}\big) \times \sum_{i=1}^N \frac{w(x^i)}{\sum_{\ell=1}^N w(x^\ell)}\mathsf{R}(x^i,\mathsf{A}) \ . \tag{6.75}$$

Let f be a nonnegative measurable function. Using that $\Pi(\mathrm{d}y)\delta_y(\mathrm{d}x^j) = \Pi(\mathrm{d}x^j)\delta_{x^j}(\mathrm{d}y)$, $\Pi(\mathrm{d}x^j) = \Lambda(\mathrm{d}x^j)w(x^j)$, and $\Lambda(\mathrm{d}x^j)\mathsf{Q}_j\big(x^j,\mathrm{d}x^{1:N\setminus\{j\}}\big) = \bar\Lambda_N\big(\mathrm{d}x^{1:N}\big)$, we get

$$\int \Pi(\mathrm{d}y)\mathsf{C}_N(y,\mathrm{d}y')f(y') = N^{-1}\int \sum_{j=1}^N w(x^j)\bar\Lambda_N\big(\mathrm{d}x^{1:N}\big)\sum_{i=1}^N \frac{w(x^i)}{\sum_{\ell=1}^N w(x^\ell)}\mathsf{R}f(x^i) \tag{6.76}$$

$$= N^{-1}\int \sum_{i=1}^N w(x^i)\mathsf{R}f(x^i)\bar\Lambda_N\big(\mathrm{d}x^{1:N}\big) \ . \tag{6.77}$$

Using that

$$w(x^i)\bar\Lambda_N\big(\mathrm{d}x^{1:N}\big) = w(x^i)\Lambda(\mathrm{d}x^i)\mathsf{Q}_j\big(x^j,\mathrm{d}x^{1:N\setminus\{j\}}\big) = \Pi(\mathrm{d}x^i)\mathsf{Q}_i\big(x^i,\mathrm{d}x^{1:N\setminus\{i\}}\big) \ , \tag{6.78}$$

we obtain

$$\begin{aligned} N^{-1}\int \sum_{i=1}^N w(x^i)\mathsf{R}f(x^i)\bar\Lambda_N\big(\mathrm{d}x^{1:N}\big) &= N^{-1}\sum_{i=1}^N \Pi(\mathrm{d}x^i)\mathsf{R}f(x^i)\mathsf{Q}_i\big(x^i,\mathrm{d}x^{1:N\setminus\{i\}}\big) \\ &= \sum_{i=1}^N \int \Pi(\mathrm{d}x^i)f(x^i)\int \mathsf{Q}_i\big(x^i,\mathrm{d}x^{1:N\setminus\{i\}}\big) = \int \Pi(\mathrm{d}y)f(y) \ , \end{aligned} \tag{6.79}$$

where we have used $\int \mathsf{Q}_i\big(x^i,\mathrm{d}x^{1:N\setminus\{i\}}\big) = 1$.

6.8 Metropolis-Adjusted Langevin rejunevation kernel

Assume that $\mathbb{X} = \mathbb{R}^d$ and the target distribution Π is absolutely continuous with $\Pi(\mathrm{d}x) = \pi(x)\mathrm{d}x$, where $\pi(x) = \exp\{-U(x)\}$ with continuously differentiable function $U(x)$. Then the MALA kernel is given, for $\eta > 0$, $x, z \in \mathbb{R}^d$, and $\mathsf{A} \in \mathcal{B}(\mathbb{R}^d)$, by

$$\begin{aligned} \mathsf{R}_\eta^{\mathrm{MALA}}(x,\mathsf{A}) = &\int_{\mathbb{R}^d} \mathbb{1}_\mathsf{A}(x - \eta\nabla U(x) + \sqrt{2\eta}z)\min(1, \mathrm{e}^{-\tau_\eta^{\mathrm{MALA}}(x,z)})\boldsymbol{\varphi}(z)\mathrm{d}z \\ &\delta_x(\mathsf{A})\int_{\mathbb{R}^d}\{1 - \min(1,\mathrm{e}^{-\tau_\eta^{\mathrm{MALA}}(x,z)})\}\boldsymbol{\varphi}(z)\mathrm{d}z \ , \end{aligned} \tag{6.80}$$

$$\tau_\eta^{\mathrm{MALA}}(x,z) = U(x - \eta\nabla U(x) + \sqrt{2\eta}z) - U(x) \tag{6.81}$$

$$+ (1/2)\{\Big\|z - (\eta/2)^{1/2}\Big\{\nabla U(x) + \nabla U(x - \eta\nabla U(x) + \sqrt{2\eta}z)\Big\}\Big\|^2 - \|z\|^2\} \ . \tag{6.82}$$

Note that the MALA kernel leaves the target π invariant. For notation simplicity, in this section we simply write R_η instead of $\mathsf{R}_\eta^{\mathrm{MALA}}$. Consider the following assumptions on the target distribution:

H8. *$U(x) \in \mathrm{C}^\infty_{\mathrm{poly}}\left(\mathbb{R}^d, \mathbb{R}\right)$, and ∇U is Lipschitz, i.e. there exists $L \geqslant 0$ such that $\|\nabla U(x) - \nabla U(y)\| \leqslant L\|x - y\|$ for all $x, y \in \mathbb{R}^d$.*

H9. *There exist $\mathrm{K}_1 \geqslant 0$ and $m > 0$ such that for any $x \notin \mathrm{B}(0, \mathrm{K}_1)$, and $y \in \mathbb{R}^d$, $\left\langle \mathrm{D}^2 U(x)y, y\right\rangle \geqslant m \|y\|^2$. Moreover, there exists $M \geqslant 0$ such that for any $x \in \mathbb{R}^d$, $\left\|\mathrm{D}^3 U(x)\right\| \leqslant M$.*

Theorem 55. *Assume **H**8, **H**9 and that the proposal distribution has a continuous and positive p.d.f. $\Lambda(x)$. Then **H**7 is satisfied with $V(x) = \exp(\bar\eta \|x\|^2)$, $\bar\eta$ given in (6.95).*

Proof. Proposition 63 implies that **H**7-(ii) holds with Assume H8 and H9. Then there exist $\bar\gamma > 0$, $\varpi > 0$, and $\mathrm{K}_2, \bar b \geqslant 0$ such that for any $\eta \in (0, \bar\gamma]$ and $x \in \mathbb{R}^d$,

$$\mathrm{R}_\eta V_{\bar\eta}(x) \leqslant \lambda_{\mathrm{R}_\eta} V_{\bar\eta}(x) + \mathsf{b}_{\mathrm{R}_\eta} \ , \ \text{where } \lambda_{\mathrm{R}_\eta} = 1 - \varpi\eta, \ \mathsf{b}_{\mathrm{R}_\eta} = \bar b \eta \ , \tag{6.83}$$

and the parameters ϖ and $\bar b$ are given in (6.135). Moreover, since Λ and Π are continuous and $\Lambda(x) > 0$, the weight function $w(x)$ is bounded on the compact sets and **H**7-(iii) holds. Hence, all conditions of Lemma 54 are satisfied. □

Consider now the sampling problem described in Section 6.11.4. We state the following result.

Proposition 56. *Let $\pi(x) = \mathrm{g}(x; 0, \mathrm{Id}_{\mathsf{D}})$ and $\lambda(x) = \mathrm{g}(x; 0, 2\,\mathrm{Id}_{\mathsf{D}})$. Let $V_{\bar\eta}(x) = \exp(\bar\eta \|x\|^2)$, where $\bar\eta$ is given in (6.95). Then, for $N \geqslant \mathrm{e}^{2\bar\eta D}$, the Markov kernel of* Ex^{2}MCMC *algorithm with MALA rejuvenation kernel satisfies*

$$\|\mathsf{K}_N^k(x, \cdot) - \Pi\|_V \leqslant c_{\mathrm{K}}\{V_{\bar\eta}(x) + \Pi(V_{\bar\eta})\}\lambda_{\mathrm{r}}[N, \mathsf{R}]^k \ , \ \textit{where } \log \lambda_{\mathrm{r}}[N, \mathsf{R}] \leqslant \frac{\log(1 - \varpi\eta)}{1 + 8\bar\eta} + \alpha_N \ , \tag{6.84}$$

and $\alpha_N \to 0$ for $N \to \infty$, and c_{K} is given in (6.74).

Proof. From the proof of Theorem 55, it follows that

$$\|\mathsf{K}_N^k(x, \cdot) - \Pi\|_V \leqslant c_{N,\mathsf{R}}\{V(x) + \Pi(V)\}\lambda_{\mathrm{r}}[N, \mathsf{R}]^k \ ,$$

where the constants $c_{N,\mathsf{R}}$ and $\lambda_{\mathrm{r}}[N, \mathsf{R}]$ are given in (6.74). We first estimate the quantity $\epsilon_{d_N,N} = (N-1)\Pi(\mathsf{V}_{d_N})/[2w_{\infty,d_N} + N - 2]$. Note that

$$w_{\infty,d_N} \leqslant \sup_{x \in \mathbb{R}^{\mathsf{D}}} \frac{\pi(x)}{\lambda(x)} = 2^{\mathsf{D}/2} \ ,$$

which does not depend on the choice of d_N. Hence, for any choice d_N, $\epsilon_{d_N,N}/\Pi(\mathsf{V}_{d_N}) \to 1$ as $N \to \infty$. Now we can lower bound $\Pi(\mathsf{V}_{d_N})$ as follows. Note that $\mathsf{V}_{d_N} = \{x \in \mathbb{X} : V_{\bar\eta}(x) \leqslant d_N\} = \left\{x \in \mathbb{X} : \|x\|^2 \leqslant \log d_N/\bar\eta\right\}$. Then, for $\log d_N/\bar\eta \geqslant 2\mathsf{D}$, we write

$$\Pi(\mathsf{V}_{d_N}) = \mathbb{P}\left(\|Z\|^2 \leqslant \left(\frac{\log d_N}{\bar\eta} - \mathsf{D}\right) + \mathsf{D}\right) \geqslant 1 - \exp\left\{-\frac{\log d_N}{8\bar\eta} + \frac{\mathsf{D}}{8}\right\} \ , \tag{6.85}$$

where $Z = (Z_1, \ldots, Z_{\mathsf{D}}) \sim \Pi$. Here we use the fact that $\|Z\|^2$ is a chi-squared random variable with D degrees of freedom and [Wai19, Proposition 2.2]. Hence, $\log(1 - \epsilon_{d,N}) \leqslant -\left(\frac{\log d_N}{8\bar\eta} - \frac{\mathsf{D}}{8}\right)$. Then using the expression (6.74) and choosing

$$d_N = 1 \vee 4(\mathsf{b}_{\mathsf{R}} + \mathsf{b}_{\mathsf{P}_N})/(1 - \lambda_{\mathsf{R}}) - 1 \vee N \vee \mathrm{e}^{2\bar\eta D} \ , \tag{6.86}$$

we have

$$\log \lambda_{\mathrm{r}}[N, \mathsf{R}] = \frac{\log(1 - \epsilon_{d,N}) \log \bar\lambda_{\mathsf{R}}}{\log(1 - \epsilon_{d_N,N}) + \log \bar\lambda_{\mathsf{R}} - \log \bar b_{\mathsf{R}}} = \frac{\log \bar\lambda_{\mathsf{R}}}{1 + \log \bar\lambda_{\mathsf{R}}/\log(1 - \epsilon_{d_N,N}) - \log \bar b_{\mathsf{R}}/\log(1 - \epsilon_{d_N,N})} \ . \tag{6.87}$$

Note that $\log \bar\lambda_{\mathsf{R}}/\log(1 - \epsilon_{d_N,N}) \to 0, N \to \infty$. Now the statement follows from $\log \bar b_{\mathsf{R}}/\log(1 - \epsilon_{d_N,N}) = -8\bar\eta + \beta_N$, $\beta_N \to 0$ for $N \to \infty$. □

6.9 Technical lemmas for Metropolis-Adjusted Langevin kernel

The goal of this section is to establish the Foster-Lyapunov drift condition for the Markov kernel $\mathsf{R}_\eta^{\mathrm{MALA}}$ defined in (6.80). As an auxiliary result we need to establish the drift condition for the Markov kernel $\mathrm{Q}_\eta^{\mathrm{ULA}}$, defined as

$$\mathrm{Q}_\eta^{\mathrm{ULA}}(x, \mathsf{A}) = \int_{\mathbb{R}^d} \mathbb{1}_{\mathsf{A}}\left(x - \eta\nabla U(x) + \sqrt{2\eta}z\right)\boldsymbol{\varphi}(z)\mathrm{d}z \;, \tag{6.88}$$

where $\boldsymbol{\varphi}$ is the d-dimensional standard Gaussian density $\boldsymbol{\varphi}(z) = (2\pi)^{-d/2}\mathrm{e}^{-\|z\|^2}$. Define for any $\eta > 0$, $V_\eta(x) : \mathbb{R}^d \to [1, +\infty)$ as

$$V_\eta(x) = \exp(\eta \|x\|^2) \;. \tag{6.89}$$

For notation simplicity, we write in this section Q_η instead of $\mathrm{Q}_\eta^{\mathrm{ULA}}$, and R_η instead of $\mathsf{R}_\eta^{\mathrm{MALA}}$. We begin with the technical lemma.

Lemma 57. *Assume H8 and H9. Then there exists* $\mathrm{K}_2 \geqslant 0$ *such that for any* $x \notin \mathrm{B}(0, \mathrm{K}_2)$, $\langle \nabla U(x), x\rangle \geqslant (m/2)\|x\|^2$ *and in particular* $\|\nabla U(x)\| \geqslant (m/2)\|x\|$.

Proof. Using H8 and H9, we have for any $x \in \mathbb{R}^d$, $\|x\| \geqslant \mathrm{K}_1$,

$$\begin{aligned}\langle \nabla U(x), x\rangle &= \int_0^{\mathrm{K}_1/\|x\|} \mathrm{D}^2 U(tx)[x^{\otimes 2}]\mathrm{d}t + \int_{\mathrm{K}_1/\|x\|}^1 \mathrm{D}^2 U(tx)[x^{\otimes 2}]\mathrm{d}t &(6.90)\\ &\geqslant m\|x\|^2\left\{1 - \mathrm{K}_1(1 + L/m)/\|x\|\right\} \;, &(6.91)\end{aligned}$$

which proves the first statement. The second statement easily follows from the Cauchy-Schwartz inequality. □

Lemma 58. *Assume H8 and H9. Then, for any* $t \in [0,1]$, $\eta \in (0, 1/(4L)]$ *and* $x, z \in \mathbb{R}^d$, $\|z\| \leqslant \|x\|/(4\sqrt{2\eta})$, *it holds*

$$\left\|x + t\{-\eta\nabla U(x) + \sqrt{2\eta}z\}\right\| \geqslant \|x\|/2 \;. \tag{6.92}$$

Proof. Let $t \in [0,1]$, $\eta \in (0, 1/(4L)]$ and $x, z \in \mathbb{R}^d$, $\|z\| \leqslant \|x\|/(4\sqrt{2\eta})$. Using the triangle inequality and H8, we have since $t \in [0,1]$

$$\left\|x + t\{-\eta\nabla U(x) + \sqrt{2\eta}z\}\right\| \geqslant (1 - \eta L)\|x\| - \sqrt{2\eta}\|z\| \;. \tag{6.93}$$

The conclusion then follows from $\eta \leqslant 1/(4L)$ and $\|z\| \leqslant \|x\|/(4\sqrt{2\eta})$. □

Now we establish the drift condition for $\mathrm{Q}_\eta^{\mathrm{ULA}}$.

Lemma 59. *Assume H8 and H9, and let* $\bar\gamma \in \left(0, m/(4L^2)\right]$. *Then, for any* $\eta \in (0, \bar\gamma]$,

$$\mathrm{Q}_\eta V_{\bar\eta}(x) \leqslant \exp\left(-\bar\eta m\gamma\|x\|^2/4\right)V_{\bar\eta}(x) + b_{\bar\eta}\gamma\mathbb{1}_{\mathrm{B}(0,\mathrm{K}_3)}(x) \;, \tag{6.94}$$

where $V_{\bar\eta}$ *is defined by* (6.89), $\bar\eta = \min(m/16, (8\bar\gamma)^{-1})$, $\mathrm{K}_3 = \max(\mathrm{K}_2, 4\sqrt{d/m})$, *and*

$$\begin{aligned}b_{\bar\eta} = &\left[\bar\eta\left\{m/4 + (1 + 16\bar\eta\bar\gamma)(4\bar\eta + 2L + \bar\gamma L^2)\right\}\mathrm{K}_3^2 + 4\bar\eta d\right]\\ &\times \exp\left[\bar\gamma\bar\eta\left\{m/4 + (1 + 16\bar\eta\bar\gamma)(4\bar\eta + 2L + \bar\gamma L^2)\right\}\mathrm{K}_3^2 + (d/2)\log(2)\right] \;.\end{aligned} \tag{6.95}$$

Proof. Let $\eta \in (0, \bar\gamma]$. For any $x \in \mathbb{R}^d$, we have

$$\begin{aligned}\bar\eta\left\|x - \eta\nabla U(x) + \sqrt{2\eta}z\right\|^2 - \|z\|^2/2 = -\frac{1 - 4\bar\eta\eta}{2}\left\|z - \frac{2(2\eta)^{1/2}\bar\eta}{1 - 4\bar\eta\eta}\{x - \eta\nabla U(x)\}\right\|^2\\ + \frac{\bar\eta}{1 - 4\bar\eta\eta}\|x - \eta\nabla U(x)\|^2 \;. \quad (6.96)\end{aligned}$$

Since $1 - 4\bar{\eta}\eta > 0$, we get that

$$\begin{aligned} \mathrm{Q}_\eta V_{\bar{\eta}}(x) &= (2\pi)^{-d/2} \int_{\mathbb{R}^d} \exp\left(\bar{\eta} \left\| x - \eta \nabla U(x) + \sqrt{2\eta} z \right\|^2 - \|z\|^2 / 2\right) \mathrm{d}z \\ &= (1 - 4\bar{\eta}\eta)^{-d/2} \exp\left(\bar{\eta}(1 - 4\bar{\eta}\eta)^{-1} \|x - \eta \nabla U(x)\|^2\right) . \end{aligned} \tag{6.97}$$

We now distinguish the case when $\|x\| \geqslant \mathrm{K}_3$ and $\|x\| < \mathrm{K}_3$. By H9 and Lemma 57, for any $x \in \mathbb{R}^d$, $\|x\| \geqslant \mathrm{K}_3 \geqslant \mathrm{K}_2$, using that $\bar{\eta} \leqslant m/16$ and $\eta \leqslant \bar{\gamma} \leqslant m/(4L^2)$, we have

$$(1 - 4\bar{\eta}\eta)^{-1} \|x - \eta \nabla U(x)\|^2 - \|x\|^2 \leqslant \eta \|x\|^2 (1 - 4\bar{\eta}\eta)^{-1} \left(4\bar{\eta} - m + \eta L^2\right) \leqslant -\eta (m/2) \|x\|^2 (1 - 4\bar{\eta}\eta)^{-1} . \tag{6.98}$$

Therefore, (6.97) becomes

$$\mathrm{Q}_\eta V_{\bar{\eta}}(x) \leqslant \exp\left(-\eta\bar{\eta}(m/2)(1 - 4\bar{\eta}\eta)^{-1} \|x\|^2 - (d/2) \log(1 - 4\bar{\eta}\eta)\right) V_{\bar{\eta}}(x) \tag{6.99}$$

$$\leqslant \exp\left(\eta\bar{\eta}\{-(m/2) \|x\|^2 + 4d\}\right) V_{\bar{\eta}}(x) , \tag{6.100}$$

where we have used for the last inequality that $-\log(1 - t) \leqslant 2t$ for $t \in [0, 1/2]$ and $4\bar{\eta}\eta \leqslant 1/2$. The proof of the statement then follows since $\|x\| \geqslant \mathrm{K}_3 \geqslant 4\sqrt{d/m}$.

In the case $\|x\| < \mathrm{K}_3$, by (6.97), H8 and since $(1 - t)^{-1} \leqslant 1 + 4t$ for $t \in [0, 1/2]$, we obtain

$$(1 - 4\bar{\eta}\eta)^{-1} \|x - \eta \nabla U(x)\|^2 - \|x\|^2 \leqslant \eta (1 - 4\bar{\eta}\eta)^{-1} \{4\bar{\eta} + 2L + \eta L^2\} \|x\|^2 \tag{6.101}$$

$$\leqslant \eta (1 + 16\bar{\eta}\eta) \{4\bar{\eta} + 2L + \eta L^2\} \|x\|^2 , \tag{6.102}$$

which implies that

$$\begin{aligned} \mathrm{Q}_\eta V_{\bar{\eta}}(x) / V_{\bar{\eta}}(x) \leqslant \mathrm{e}^{-\bar{\eta} m \eta \|x\|^2 / 4} + \\ \exp\left[\eta\bar{\eta} \left\{m/4 + (1 + 16\bar{\eta}\eta)(4\bar{\eta} + 2L + \eta L^2)\right\} \|x\|^2 - (d/2) \log(1 - 4\bar{\eta}\eta)\right] - 1 . \end{aligned} \tag{6.103}$$

The proof is then completed using that for any $t \geqslant 0$, $\mathrm{e}^t - 1 \leqslant t\mathrm{e}^t$, for any $s \in [0, 1/2]$, $-\log(1 - s) \leqslant 2s$ and $4\bar{\eta}\eta \leqslant 1/2$. □

We now provide a decomposition in η of $\tau_\eta^{\mathrm{MALA}}$ defined in (6.81). For any $x, z \in \mathbb{R}^d$, by [DMS17, Lemma 24][1], we have that

$$\tau_\eta^{\mathrm{MALA}}(x, z) = \sum_{k=2}^{6} \eta^{k/2} A_{k,\eta}(x, z) \tag{6.104}$$

where, setting $x_t = x + t\{-\eta \nabla U(x) + \sqrt{2\eta} z\}$,

$$A_{2,\eta}(x, z) = 2 \int_0^1 \mathrm{D}^2 U(x_t)[z^{\otimes 2}](1/2 - t)\mathrm{d}t \tag{6.105}$$

$$A_{3,\eta}(x, z) = 2^{3/2} \int_0^1 \mathrm{D}^2 U(x_t)[z \otimes \nabla U(x)](t - 1/4)\mathrm{d}t , \tag{6.106}$$

$$A_{4,\eta}(x, z) = -\int_0^1 \mathrm{D}^2 U(x_t)[\nabla U(x)^{\otimes 2}]t\mathrm{d}t + (1/2) \left\| \int_0^1 \mathrm{D}^2 U(x_t)[z]\mathrm{d}t \right\|^2 \tag{6.107}$$

$$A_{5,\eta}(x, z) = -(1/2)^{1/2} \left\langle \int_0^1 \mathrm{D}^2 U(x_t)[\nabla U(x)]\mathrm{d}t, \int_0^1 \mathrm{D}^2 U(x_t)[z]\mathrm{d}t \right\rangle \tag{6.108}$$

$$A_{6,\eta}(x, z) = (1/4) \left\| \int_0^1 \mathrm{D}^2 U(x_t)[\nabla U(x)]\mathrm{d}t \right\|^2 . \tag{6.109}$$

[1]Note that with the notation of [DMS17], MALA corresponds to HMC with only one leapfrog step and step size equals to $(2\eta)^{1/2}$

Lemma 60. *Assume H8 and H9. Then, for any $\bar{\gamma} > 0$, there exists $C_{1,\bar{\gamma}} < \infty$ such that for any $x, z \in \mathbb{R}^d$, $\gamma \in (0, \bar{\gamma}]$, it holds*

$$\left|\tau_\eta^{\mathrm{MALA}}(x,z)\right| \leqslant C_{1,\bar{\gamma}} \gamma^{3/2} \{1 + \|z\|^4 + \|x\|^2\} \,. \tag{6.110}$$

Proof. Since $\int_0^1 \mathrm{D}^2 U(x)[z^{\otimes 2}](1/2 - t)\mathrm{d}t = 0$, we get setting $x_t = x + t\{-\gamma \nabla U(x) + \sqrt{2\gamma} z\}$,

$$\begin{aligned} &A_{2,\gamma}(x,z) \\ &\quad = \sqrt{\gamma} \iint_0^1 \mathrm{D}^3 U(s x_t + (1-s)x) \left[z^{\otimes 2} \otimes \{-\gamma^{1/2} \nabla U(x) + \sqrt{2} z\}\right] (1/2 - t) t \mathrm{d}s \mathrm{d}t \,. \end{aligned} \tag{6.111}$$

The proof follows from $\sup_{x \in \mathbb{R}^d} \left\|\mathrm{D}^2 U(x)\right\| \leqslant L$ and $\sup_{x \in \mathbb{R}^d} \left\|\mathrm{D}^3 U(x)\right\| \leqslant M$. □

Lemma 61. *Assume H8 and H9. Then, for any $\bar{\gamma} \in \left(0, m^3/(4L^4)\right]$ there exists $C_{2,\bar{\gamma}} < \infty$ such that for any $\eta \in (0, \bar{\gamma}]$, $x, z \in \mathbb{R}^d$ satisfying $\|x\| \geqslant \max(2\,\mathrm{K}_1, \mathrm{K}_2)$ and $\|z\| \leqslant \|x\| / (4\sqrt{2\eta})$, where K_2 is defined in Lemma 57, it holds*

$$\tau_\eta^{\mathrm{MALA}}(x,z) \leqslant C_{2,\bar{\gamma}} \eta \|z\|^2 \left\{1 + \|z\|^2\right\} \,. \tag{6.112}$$

Proof. Let $\eta \in (0, \bar{\gamma}]$, $x, z \in \mathbb{R}^d$ satisfying $\|x\| \geqslant \max(2\,\mathrm{K}_1, \mathrm{K}_2)$ and $\|z\| \leqslant \|x\| / (4\sqrt{2\eta})$. Using (6.104), we get setting

$$A_{4,0,\eta}(x,z) = \int_0^1 \mathrm{D}^2 U(x_t)[\nabla U(x)^{\otimes 2}] t \mathrm{d}t \,,$$

$$\begin{aligned} \tau_\eta^{\mathrm{MALA}}(x,z) &\leqslant 2\eta A_{2,\eta}(x,z) - \eta^2 A_{4,0,\eta}(x,z) \\ &\quad + (2\eta)^{3/2} L^2 \|z\| \|x\| + (\eta^2/2) L^2 \|z\|^2 + (\eta^5/2)^{1/2} L^3 \|z\| \|x\| + (\eta^3/4) L^4 \|x\|^2 \,, \end{aligned} \tag{6.113}$$

By H8, Lemma 57 and Lemma 58, we get for any $x \in \mathbb{R}^d$, $\|x\| \geqslant \max(2\,\mathrm{K}_1, \mathrm{K}_2)$,

$$A_{4,0,\gamma}(x,z) \geqslant (m/2)^3 \|x\|^2 \,. \tag{6.114}$$

Combining this result with (6.111), (6.114) in (6.113), we obtain using $\gamma \leqslant \bar{\gamma} \leqslant m^3/(4L^4)$

$$\tau_\eta^{\mathrm{MALA}}(x,z) \leqslant 2\gamma M \left\{\sqrt{2\gamma} \|z\|^3 + \gamma L \|z\|^2 \|x\|\right\} - \gamma^2 (m^3/2^4) \|x\|^2 \tag{6.115}$$

$$+ (2\gamma)^{3/2} L^2 \|z\| \|x\| + (\gamma^2/2) L^2 \|z\|^2 + (\gamma^5/2)^{1/2} L^3 \|z\| \|x\| \,, \tag{6.116}$$

Since for any $a, b \in \mathbb{R}^+$ and $\varepsilon > 0$, $ab \leqslant (\epsilon/2) a^2 + 1/(2\epsilon) b^2$, we obtain

$$\tau_\eta^{\mathrm{MALA}}(x,z) \leqslant \eta \|z\|^2 \left\{2^{1/2} L^2 \varepsilon^{-1} + (\eta/2) L^2 + 2^{-3/2} \eta^{3/2} L^3 \varepsilon^{-1} \right. \tag{6.117}$$

$$\left. + (2^3 \eta)^{1/2} M \|z\| + \eta M L \varepsilon^{-1} \|z\|^2 \right\} \tag{6.118}$$

$$+ \|x\|^2 \eta^2 \left[\varepsilon \left\{L M + 2^{1/2} L^2 + 2^{-3/2} \bar{\gamma}^{1/2} L^3\right\} - m^3/2^4\right] \,. \tag{6.119}$$

Choosing $\varepsilon = (m^3/2^4) \left\{L M + 2^{1/2} L^2 + 2^{-3/2} \bar{\gamma}^{1/2} L^3\right\}^{-1}$ concludes the proof. □

Lemma 62. *Let $\bar{\gamma} > 0$ and $\gamma \in (0, \bar{\gamma}]$. Then, for any $x \in \mathbb{R}^d$, $\|x\| \geqslant 20\sqrt{2\bar{\gamma} d}$,*

$$\int_{\mathbb{R}^d \setminus \mathrm{B}(0, \|x\|/(4\sqrt{2\gamma}))} \boldsymbol{\varphi}(z) \mathrm{d}z \leqslant \exp(-\|x\|^2 / (128\gamma)) \,. \tag{6.120}$$

Proof. Let $x > 0$. By [LM00, Lemma 1],

$$\mathbb{P}(\|Z\|^2 \geqslant 2\{\sqrt{d} + \sqrt{x}\}^2) \leqslant \mathbb{P}(\|Z\|^2 \geqslant d + 2\sqrt{dx} + 2x) \leqslant \mathrm{e}^{-x} \,, \tag{6.121}$$

where Z is a d-dimensional standard Gaussian vector. Setting $t = 2\{\sqrt{d} + \sqrt{x}\}^2$, we obtain

$$\mathbb{P}(\|Z\|^2 \geqslant t) \leqslant \exp\left(-\left\{d + t/2 - \sqrt{2td}\right\}\right) \,, \tag{6.122}$$

and for $\sqrt{t} \geqslant 5\sqrt{d}$, we get $\mathbb{P}(\|Z\| \geqslant \sqrt{t}) \leqslant \mathrm{e}^{-t/4}$ which gives the result. □

Proposition 63. *Assume H8 and H9. Then there exist $\bar{\gamma} > 0$, $\varpi > 0$, and $\mathrm{K}_2, \bar{b} \geqslant 0$ such that for any $\eta \in (0, \bar{\gamma}]$ and $x \in \mathbb{R}^d$,*

$$\mathrm{R}_\eta V_{\bar{\eta}}(x) \leqslant (1 - \varpi\eta) V_{\bar{\eta}}(x) + \bar{b}\eta \mathbb{1}_{\mathrm{B}(0,\mathrm{K}_2)}(x) \;, \tag{6.123}$$

where $V_{\bar{\eta}}$ is defined by (6.89)*, R_η is the MALA kernel given in* (6.80) *and $\bar{\eta}$ is given by* (6.95)*.*

Proof. Let $\bar{\gamma}_1 = m/(4L^2)$. By Lemma 59, for any $\gamma \in (0, \bar{\gamma}_1]$ and $x \in \mathbb{R}^d$,

$$\mathrm{R}_\eta V_{\bar{\eta}}(x) \leqslant \mathrm{Q}_\eta V_{\bar{\eta}}(x) + V_{\bar{\eta}}(x) \int_{\mathbb{R}^d} \{1 - \min(1, \mathrm{e}^{-\tau_\eta^{\mathrm{MALA}}(x,z)}\} \boldsymbol{\varphi}(z) \mathrm{d}z \tag{6.124}$$

$$\leqslant \mathrm{e}^{-\bar{\eta} m \gamma \|x\|^2/4} V_{\bar{\eta}}(x) + b_{\bar{\eta}} \gamma \mathbb{1}_{\mathrm{B}(0,\mathrm{K}_3)}(x) + V_{\bar{\eta}}(x) \int_{\mathbb{R}^d} \{1 - \min(1, \mathrm{e}^{-\tau_\eta^{\mathrm{MALA}}(x,z)}\} \boldsymbol{\varphi}(z) \mathrm{d}z \;, \tag{6.125}$$

where K_3 and $b_{\bar{\eta}}$ are given in (6.95). Let

$$\bar{\gamma}_2 = \min\left(1, \bar{\gamma}_1, m^3/(4L^4)\right) \;, \quad M_1 = \max\left(1, 2\,\mathrm{K}_1, \mathrm{K}_2, \mathrm{K}_3, 20\sqrt{2\bar{\gamma}_2 d}\right) \;. \tag{6.126}$$

Then, by Lemma 61 and Lemma 62, there exist $C_1 \geqslant 0$ such that for any $x \in \mathbb{R}^d$, $\|x\| \geqslant M_1$ and $\gamma \in (0, \bar{\gamma}_2]$,

$$\mathrm{R}_\eta V_{\bar{\eta}}(x) \leqslant \mathrm{e}^{-\bar{\eta} m \gamma \|x\|^2/4} V_{\bar{\eta}}(x) + V_{\bar{\eta}}(x) \left\{ C_1 \gamma + \exp(-\|x\|^2/(128\gamma)) \right\} \tag{6.127}$$

$$\leqslant \mathrm{e}^{-\bar{\eta} m \gamma \|x\|^2/4} V_{\bar{\eta}}(x) + V_{\bar{\eta}}(x) \left\{ C_1 \gamma + \exp(-1/(128\gamma)) \right\} \;. \tag{6.128}$$

Using that there exists $C_2 \geqslant 0$ such that $\sup_{t \in (0,1)} \{t^{-1} \exp(-1/(128t))\} \leqslant C_2$ we get for any $x \in \mathbb{R}^d$, $\|x\| \geqslant M_1$, $\gamma \in (0, \bar{\gamma}_2]$,

$$\mathrm{R}_\eta V_{\bar{\eta}}(x) \leqslant \mathrm{e}^{-\bar{\eta} m \gamma \|x\|^2/4} V_{\bar{\eta}}(x) + V_{\bar{\eta}}(x) \gamma \left\{ C_1 + C_2 \right\} \;. \tag{6.129}$$

Let

$$M_2 = \max\left(M_1, 4(C_1 + C_2)^{1/2} (\bar{\eta} m)^{-1/2}\right) \;, \quad \bar{\gamma}_3 = \min\left(\bar{\gamma}_2, 4\left\{m \bar{\eta} M_2^2\right\}^{-1}\right) \;. \tag{6.130}$$

Then, since for any $t \in [0,1]$, $\mathrm{e}^{-t} \leqslant 1 - t/2$, we get for any $x \in \mathbb{R}^d$, $\|x\| \geqslant M_2$, $\gamma \in (0, \bar{\gamma}_3]$,

$$\begin{aligned} \mathrm{R}_\eta V_{\bar{\eta}}(x) &\leqslant \mathrm{e}^{-\bar{\eta} m \eta M_2^2/4} V_{\bar{\eta}}(x) + V_{\bar{\eta}}(x) \eta \left\{ C_1 + C_2 \right\} \\ &\leqslant \left[1 - \eta \left\{ \bar{\eta} m M_2^2/8 - C_1 - C_2 \right\}\right] V_{\bar{\eta}}(x) \\ &\leqslant \left\{1 - \eta \bar{\eta} m M_2^2/16\right\} V_{\bar{\eta}}(x) \;. \end{aligned} \tag{6.131}$$

In addition, by Equation (6.110), using that for any $t \in \mathbb{R}$, $1 - \min(1, \mathrm{e}^{-t}) \leqslant |t|$, there exists $C_3 \geqslant 0$ such that for any $x \in \mathbb{R}^d$, $\|x\| \leqslant M_2$ and $\eta \in (0, \bar{\gamma}_3]$,

$$\mathrm{R}_\eta V_{\bar{\eta}}(x) \leqslant V_{\bar{\eta}}(x) + b_{\bar{\eta}} \eta \mathbb{1}_{\mathrm{B}(0,\mathrm{K}_3)}(x) + C_3 \eta^{3/2} \int_{\mathbb{R}^d} \{1 + \|x\|^2 + \|z\|^4\} \boldsymbol{\varphi}(z) \mathrm{d}z \tag{6.132}$$

$$\leqslant (1 - \eta \bar{\eta} m M_2^2/16) V_{\bar{\eta}}(x) + \eta \bar{\eta} m M_2^2 \mathrm{e}^{\bar{\eta} M_2^2}/16 + \eta b_{\bar{\eta}} \tag{6.133}$$

$$+ C_3 \eta \bar{\gamma}_3^{1/2} \left\{1 + M_2^2 + C_4\right\} \;, \tag{6.134}$$

where $C_4 = \int_{\mathbb{R}^d} \|z\|^4 \boldsymbol{\varphi}(z) \mathrm{d}z$. Hence, (6.123) holds with

$$\varpi = \bar{\eta} m M_2^2/16, \qquad \bar{b} = \bar{\eta} m M_2^2 \mathrm{e}^{\bar{\eta} M_2^2}/16 + b_{\bar{\eta}} + C_3 \bar{\gamma}_3^{1/2} \left\{1 + M_2^2 + C_4\right\} \;. \tag{6.135}$$

□

6.10 Algorithms

In this section, we have compiled the detailed description of all the algorithms we use in the text:

- Algorithm 4 for i-SIR;
- Algorithm 5 for Ex^{2}MCMC with independent proposals;
- Algorithm 6 for Ex^{2}MCMC with dependent proposals.

The unifying information for the hyperparameter selection in all the considered experiments is given in Table 6.6.

Input : Sample Y_j from previous iteration
Output : New sample Y_{j+1}
Set $X_{j+1}^1 = Y_j$ and draw $X_{j+1}^{2:N} \sim \Lambda$.
for $i \in [N]$ **do**
  compute the normalized weights $\omega_{i,j+1} = \tilde{w}(X_{j+1}^i) / \sum_{k=1}^N \tilde{w}(X_{j+1}^k)$.
end
Set $I_{j+1} = \mathrm{Cat}(\omega_{1,j+1}, \ldots, \omega_{N,j+1})$.
Draw $Y_{j+1} = X_{j+1}^{I_{j+1}}$.

Algorithm 4: Single stage of i-SIR algorithm with independent proposals

Input : Sample Y_j from previous iteration
Output : New sample Y_{j+1}
Set $X_{j+1}^1 = Y_j$ and draw $X_{j+1}^{2:N} \sim \Lambda$.
for $i \in [N]$ **do**
  compute the normalized weights $\omega_{i,j+1} = \tilde{w}(X_{j+1}^i) / \sum_{k=1}^N \tilde{w}(X_{j+1}^k)$.
end
Set $I_{j+1} = \mathrm{Cat}(\omega_{1,j+1}, \ldots, \omega_{N,j+1})$.
Draw $Y_{j+1} \sim \mathsf{R}(X_{j+1}^{I_{j+1}}, \cdot)$.

Algorithm 5: Single stage of Ex^{2}MCMC algorithm with independent proposals

Input : Sample Y_j from previous iteration
Output : Set of proposals for the current iteration $X_{j+1}^{1:N}$
Draw $U_{j+1} \sim \mathrm{Unif}([N])$ and set $X_{j+1}^{U_{j+1}} = Y_j$
Set $Z_{j+1}^{U_{j+1}} = T_{\theta_j}^{-1}(X_{j+1}^{U_{j+1}})$
Draw $\alpha_{j+1}^{U_{j+1}} \sim \nu$
Draw $\xi_{j+1} \sim \mathrm{N}\big(\alpha_{j+1}^{U_{j+1}} Z_{j+1}^{U_{j+1}}, \sigma_\Lambda^2 \big(1 - (\alpha_{j+1}^{U_{j+1}})^2\big)\,\mathrm{Id}\big)$
For $i \in [N] \setminus \{U_{j+1}\}$, draw $W_{j+1}^i \sim \mathrm{N}(0, \mathrm{Id}_d)$ and $\alpha_{j+1}^i \sim \nu$ and set
$Z_{j+1}^i = \alpha_{j+1}^i \xi_{j+1} + \sqrt{1 - \{\alpha_{j+1}^i\}^2} W_{j+1}^i$.
Set $X_{j+1}^{1:N\setminus\{U_{j+1}\}} = T_{\theta_j}\Big(Z_{j+1}^{1:N\setminus\{U_{j+1}\}}\Big)$

Algorithm 6: Proposal generation procedure for FlEx^{2}MCMC algorithm with dependent proposal

6.11 Numerical experiments

We provide here additional information to the simulation problems of the main document and present results on new simulation problems.

Input : weights θ_j, batch $Y_j[1:K]$
Output : new weights θ_{j+1}, batch $Y_{j+1}[1:K]$

1 **for** $k \in [K]$ **do**
2 Set $X_{j+1}^1[k] = Y_j[k]$.
3 Draw $Z_{j+1}^{1:N\setminus\{1\}}[k] \sim \Lambda$.
4 Set $X_{j+1}^{1:N\setminus\{1\}}[k] = T_{\theta_j}\big(Z_{j+1}^{1:N\setminus\{1\}}[k]\big)$.
5 **for** $i \in [N]$ **do**
6 compute the unnormalized weights $\bar{\omega}_{i,j+1}[k] = \tilde{w}_{\theta_j}(X_{j+1}^i[k])$.
7 **end**
8 Compute $\Omega_N[j+1,k] = \sum_{i=1}^N \bar{\omega}_{i,j+1}[k]$ and the normalized weights $\omega_{i,j+1}[k] = \bar{\omega}_{i,j+1}[k]/\Omega_N[j+1,k]$.
9 Set $I_{j+1}[k] = \mathrm{Cat}(\omega_{1,j+1}[k], \ldots, \omega_{N,j+1}[k])$.
10 Draw $Y_{j+1}[k] \sim \mathsf{R}(X_{j+1}^{I_{j+1}}[k], \cdot)$.
11 **end**
12 Draw $\bar{Z}[1:K] \sim \Lambda$.
13 Update $\theta_{j+1} = \theta_j - \gamma\widehat{\nabla\mathcal{L}}(Y_{j+1}, \bar{Z}, \theta_j)$.

Algorithm 7: Single stage of FlEx^{2}MCMCwith independent proposals. Steps 1-7 are done in parallel for independent chains indexed by k but with common values of proposal parameters θ_j. Step 9 updates the parameters using the gradient estimate obtained from all the chains.

6.11.1 Metrics

ESTD To compute ESTD, we perform 10 random one-dimensional projections and then perform Kernel Density Estimation there for reference and produced samples, and take TV-distance between two distributions over $1D$ grids of 1000 points. We consider the value averaged over the projections to show the divergence between the MCMC distribution and the reference distribution.

EEMD We compute the EEMD as the transport cost between sample and reference points in L_2 using the algorithm proposed in [Bon+11].

ESS The ESS is computed as follows: for sample $\{Y_t\}_{t=1}^M, Y_t \in \mathbb{R}^d$ of size M, we compute ESS component-wise. To be specific, for $i = 1, \ldots, d$, we compute

$$\mathrm{ESS}_i = \frac{M}{1 + \sum_{k=1}^M \rho_k^{(i)}} ,$$

where $\rho_k^{(i)} = \frac{\mathrm{Cov}(Y_{t,i}, Y_{t+k,i})}{\mathrm{Var}(Y_{t,i})}$ is the autocorrelation at lag k for $i-$th component. We replace ρ_k by its sample counterpart. Then we compute

$$\mathrm{ESS} = d^{-1} \sum_{i=1}^d \mathrm{ESS}_i \, .$$

6.11.2 Normalizing flow RealNVP

We use RealNVP architecture ([DSB17]) for our experiments with adaptive MCMC. The key item of RealNVP is a coupling layer, defined as transformation $f : \mathbb{R}^D \to \mathbb{R}^D$:

$$y_{1:d} = x_{1:d} \tag{6.136}$$
$$y_{d+1:D} = x_{d_1:D} \odot \exp(s(x_{1:d}) + t(x_{1:d})) \tag{6.137}$$

Experiment	Method	γ	N	ϵ	α	**num MALA steps**	**num flows**	**batch size**	**num train steps**	**lr**
Funnel, Banana	MALA	.01								
	Ex^{2}MCMC	.01	10	.5	.9	50				
	FlEx^{2}MCMC	.01	10	.5	.9	50	6	200	200	.005
Bayesian Logistic Regression	i-SIR		5							
	MALA	.01								
	Ex^{2}MCMC	.01	5	.5	.9	1				
	FlEx^{2}MCMC	.01	5	.5	.9	1	4	100	200	.01
Gaussian	i-SIR		10							
	Ex^{2}MCMC		10	1.	.95	0				
Mixture of two Gaussian distributions	i-SIR		10							
	MALA	.01								
	Ex^{2}MCMC	.01	5	.5	.9	1				
	FlEx^{2}MCMC	.01	5	.5	.9	1	2	100	200	0.01
Mixture of 25 Gaussian distributions	MALA	.001								
	Ex^{2}MCMC	.001	10	1.	.9	1				
Mixture of 243 Gaussian distributions	MALA	.001								
	Ex^{2}MCMC	.001	10	.9	.99	1				
Swissroll	MALA	.001								
	Ex^{2}MCMC	.001	10	.5	.9	1				
Allen-Cahn equation	([GRV21])	.001				10	2	100	$10k$	.001
	Ex^{2}MCMC	.001	10	.5	.9	10	2	100	$10k$	.001
Ill-conditioned Gaussian distribution	MALA	.01								
	Ex^{2}MCMC	.01	10			10				
	FlEx^{2}MCMC	.01	10	.5	.9	10	2	100	$10k$	.001

Table 6.6: Hyperparameters used in experiments.

where s and t are some functions from $\mathbb{R}^D$ to $\mathbb{R}^D$. It is clear then that the Jacobian of such transformation is triangular matrix with nonzero diagonal terms. We use fully connected neural networks to parameterize the functions s and t.

In all experiments with normalizing flows, we use optimizer Adam ([KB14]) with $\beta_1 = 0.9$, $\beta_2 = 0.999$ and weight decay 0.01 to avoid overfitting.

6.11.3 Adaptive strategy for tuning the stepsize in the MALA algorithm

In all our experiments, except for the Allen-Cahn equation, we use adaptive strategy for the stepsize in MALA, Ex^{2}MCMC and FlEx^{2}MCMC algorithms to keep acceptance rate close to $\alpha = 0.5$: we measure the average acceptance rate μ_{acc} during sampling, and increase the stepsize if $\mu_{acc} > \alpha$ or decrease it if $\mu_{acc} < \alpha$ with some tolerance threshold. The scheme is described in Algorithm 8. In practice, we set tolerance $\delta = 0.03$, factor $s = 1.05$.

Input : average acc. rate μ_{acc}, target acc. rate α, current stepsize γ, tolerance δ, factor s
Output : new stepsize γ'

if $\mu_{acc} - \alpha > \delta$ **then**
  $\gamma' = \gamma s$
else if $\alpha - \mu_{acc} > \delta$ **then**
  $\gamma' = \gamma / s$
else
  $\gamma' = \gamma$

Algorithm 8: Adaptive strategy for stepsize.

Note that in all experiments, wherever it is not said otherwise, we set noise scale coefficient in MALA kernel to $\sqrt{2\gamma}$, where γ is step size.

6.11.4 High-dimensional Gaussian distribution sampling

We consider the problem of sampling from a high-dimensional standard Normal distribution $\mathcal{N}(0, \mathrm{Id}_d)$ and the proposal distribution $\mathcal{N}(0, 2\,\mathrm{Id}_d)$ for different problem dimension $d \in [30; 300]$. The goal of the experiment is to show the efficiency of correlated proposals for i-SIR . We apply Ex^{2}MCMC algorithm with $\epsilon = 1$ and without rejuvenation kernel, as it is described in Algorithm 6. We compute empirical estimates of mean and variance, and report confidence intervals for them based on 20 independent runs of each algorithm. We report empirical estimates of mean and variance, and the ESS. We perform 10^3 burn-in steps for each of the algorithms, and then compute the metrics over the next 5×10^3 samples. Other experimental details are provided in Table 6.6.

6.11.5 Mixture of 25 Gaussian distributions in $2d$

The details on experiments with mixture of 25 Gaussian distributions were moved to Section 6.11.12, see also Table 6.6.

6.11.6 Distributions with complex geometry

In this section, we study the sampling quality from high-dimensional distributions, whose density levels have high curvature (Banana shaped and Funnel distributions, details below). With such distributions, standard MCMC algorithms like MALA or i-SIR, fail to explore fully the density support. In the examples below, we sample reference points with HMC-type algorithm NUTS ([HG+14b]). For each of examples we produce 1000 reference points from the target distribution and consider them as reference points to compute ESTV and EMD for samples from MALA, Ex^{2}MCMC and FlEx^{2}MCMC. For every target distribution, we set proposal to be the standard normal distribution $\mathcal{N}(0, \mathrm{Id}_d)$. As we do 50

rejuvenation steps in Ex^{2}MCMCand FlEx^{2}MCMCalgorithms, we take every 50th point from MALA samples to compute ESS. We set proposal as standard normal distribution for all target distributions.

Symmetric banana-shaped distribution Following [HST99], we sample from the so-called "Banana-shape" distribution. For $x \in \mathbb{R}^{2d}$, the density of the symmetric banana-shaped distribution is given by

$$p(x) = \frac{1}{Z} \exp\left(\sum_{i=1}^{d} -(x_{2i-1} - x_{2i}^2)^2/\nu - (x_{2i-1} - 1)^2 \right) , \tag{6.138}$$

with Z being a normalizing constant and $\nu > 0$. In our examples we set $\nu = 0.2$.

Asymmetric banana-shaped distribution For $x \in \mathbb{R}^{2d}$, the density of the asymmetric banana-shaped distribution is given by

$$p(x) = \frac{1}{Z} \exp\left(\sum_{i=1}^{d} -(x_{2i-1} - x_{2i}^2)^2/\nu - (x_{2i} - 1)^2 \right) , \tag{6.139}$$

with Z being a normalizing constant and $\nu > 0$. In our examples we set $\nu = 0.2$.

Funnel distribution For $x \in \mathbb{R}^{2d}$ the density of the funnel distribution is given by

$$p(x) = \frac{1}{Z} \exp\left(-\frac{x_1^2}{2a^2} - e^{-2bx_1} \sum_{i=2}^{2d} \{x_i^2 - \log d + 2bx_1\} \right) , \tag{6.140}$$

where Z is a normalizing constant. We set $a = 1$, $b = 0.5$.

Results for both types of banana-shaped distributions and for the funnel distribution are summarized in Figure 6.13 and in Figure 6.11, respectively. We report the values of ESTV, EEMD, and ESS metrics and their dependence on the problem dimension d. Clearly, FlEx^{2}MCMC algorithm outperforms both MALA and Ex^{2}MCMC with independent proposals. We visualize projections of generated samples on first two coordinates for the banana-shaped and funnel examples in Figure 6.12 and Figure 6.10, respectively. They illustrate how well Ex^{2}MCMC and FlEx^{2}MCMC can explore the support compared to MALA.

6.11.7 Bayesian Logistic regression

The training set $\mathcal{D}$ consists of pairs (x, y) where $x = (x^{(0)}, \ldots, x^{(d-1)}) \in \mathbb{R}^d$ and labels $y \in \{-1, 1\}$. In practice, the first coordinate of x represents the bias term, *i.e.*we have $x^{(0)} = 1$. The likelihood for a pair is $p(y \mid x, \theta) = \text{logit}(y \langle x, \theta \rangle)$. Given a prior distibution $p(\theta)$, we sample from the posterior distribution $p(\theta \mid \mathcal{D})$ and compute the predictive posterior $p(y \mid x, \mathcal{D}) = \int p(y \mid x, \theta) p(\theta \mid \mathcal{D}) \mathrm{d}\theta$ for $(x, y) \in \mathcal{D}^{test}$. We approximate $p(y \mid x, \mathcal{D})$ using the Monte Carlo estimate $\frac{1}{n} \sum_{i=1}^{n} p(y \mid x, \mathcal{D}, \theta_i)$, where θ_i is a sample of $p(\theta \mid \mathcal{D})$ obtained using different MCMC samplers. We take a normal prior distribution $p_0(\theta) = \mathcal{N}(0, \sigma^2 \mathrm{Id})$ with $\sigma^2 = 20$. We present results for the following datasets:

Covertype dataset consists of 581k instances of dimension 54, and we arbitrarily classes 3 and 5 from the original 7 classes to build a binary classification task. We used 1.5k steps for burn-in and sampling phases.

EEG dataset consists of 15k instances of dimension 15. We used 1.5k steps for burn-in and sampling phases.

Digits dataset consists of 1.8k instances of dimension 64, and we kept arbitrarily classes 5 and 6 from the original 10 classes for binary classification task. We used 1.5k steps for burn-in and sampling phases.

We randomly take 0.8 of the dataset for the train set and the rest for the test set. We remove outliers from training set using Isolation Forest classifier.

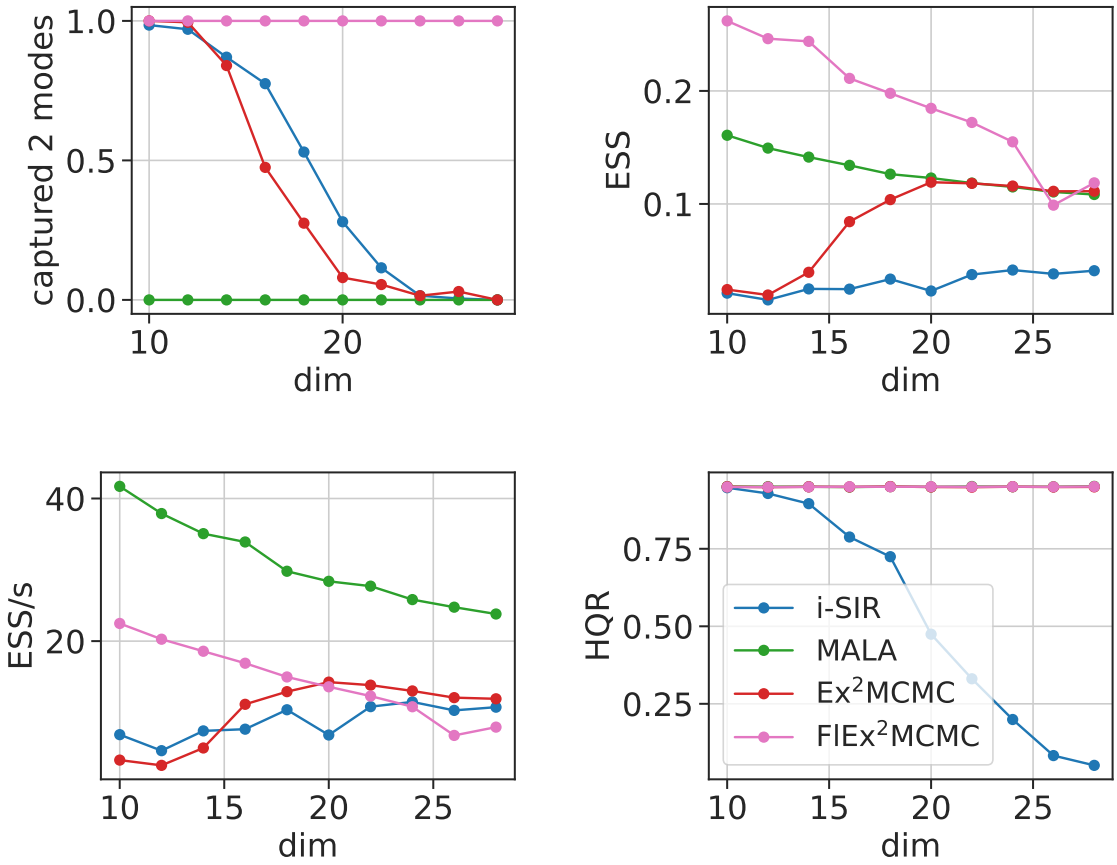


Figure 6.9: Sampling from mixture of two Gaussian distributions $\mathcal{N}(-\mathbf{1.5}, \mathrm{Id})$, $\mathcal{N}(\mathbf{1.5}, \mathrm{Id})$ in high dimensions.

6.11.8 Mixture of two Gaussian distributions

We consider here the task of sampling from a mixture of two Gaussian distributions in different dimensions. The target density is

$$p(x) = \frac{1}{2}\mathrm{g}(x; \mu, \sigma^2 \,\mathrm{Id}_d) + \frac{1}{2}\mathrm{g}(x; -\mu, \sigma^2 \,\mathrm{Id}_d) \;, \text{ where } \mu = (1.5, 1.5, \ldots, 1.5) \in \mathbb{R}^d \;, \sigma^2 = 1 \;,$$

and we set the proposal distribution to be $\mathcal{N}(\mathbf{0}, 2\,\mathrm{Id}_d)$. We perform 200 independent starts for each method for each dimension in $[10, 12, \ldots, 30]$ to compute metrics and average them across different starts. We put more details on hyperparameters used in table 6.6. We compute ESS per second (ESS/s) as a product of ESS and size of obtained sample (900) divided by sampling time in seconds. The result of this experiment is presented in Figure 6.9. Note that, despite the ESS of MALA is good, MALA always explores only one mode of the mixture.

6.11.9 Allen-Cahn equation

In this experiment, we consider the task of sampling from the invariant distribution of the Allen-Cahn stochastic differential equation ([AC75]). We borrow the setting from the paper [GRV21] in which they propose to use normalizing flows to enhance MCMC sampling from distributions with meta-stable states. Allen-Cahn equation is a SDE defined in terms of a random field $\varphi : [0, 1] \to \mathbb{R}$ that satisfies

$$\partial_t \varphi = a\partial_s^2 \varphi + a^{-1}(\varphi - \varphi^3) + \sqrt{2\beta^{-1}}\eta(t, s) \;, \tag{6.141}$$

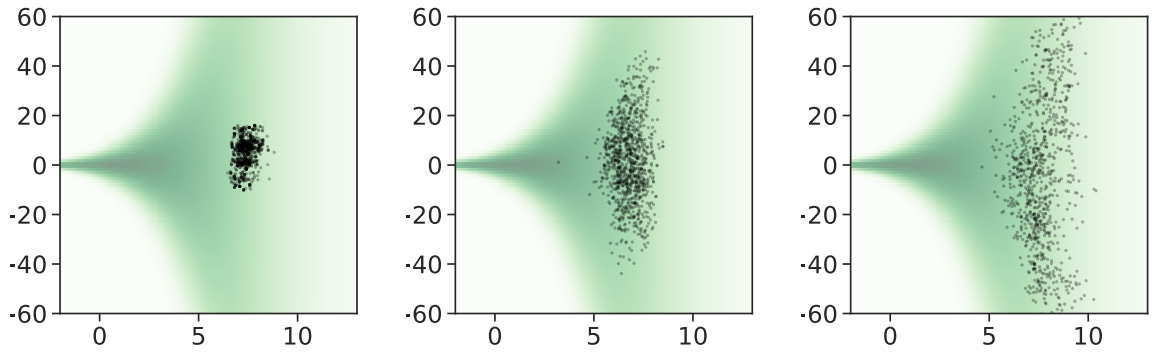


Figure 6.10: Funnel distribution, dim 15: projection of resulted samples on first two coordinates. From left to right: MALA, Ex^{2}MCMC, FlEx^{2}MCMC

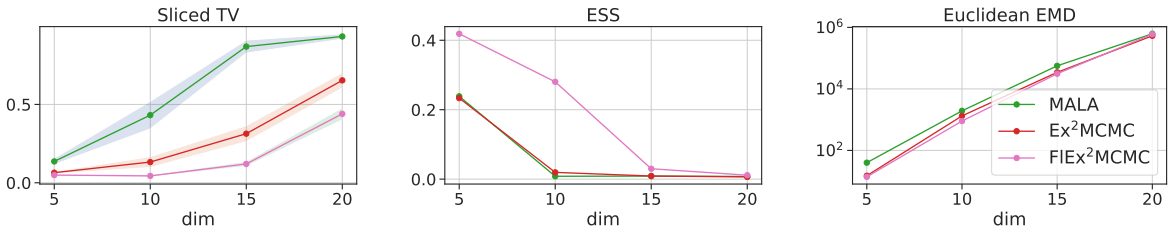


Figure 6.11: Sampling from Funnel distribution.

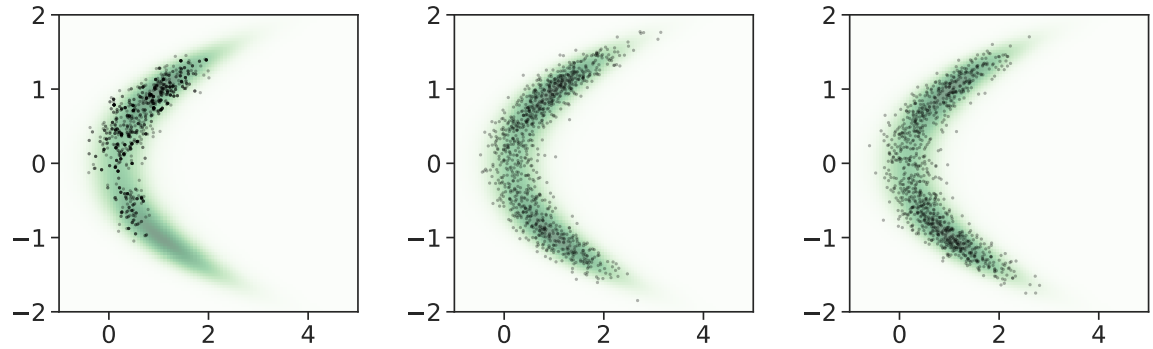


Figure 6.12: Symmetric banana-shaped distribution, dimension $d = 50$: projection of resulted samples on first two coordinates. From left to right: MALA, Ex^{2}MCMC, FlEx^{2}MCMC

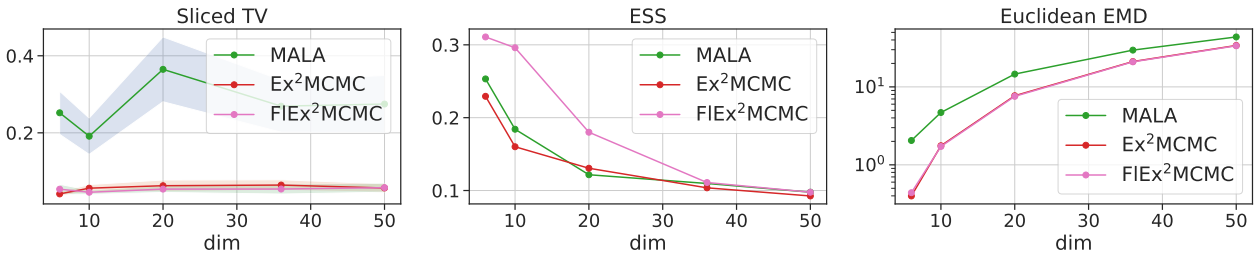


Figure 6.13: Sampling from symmetric banana-shaped distribution.

where $a > 0$, β are parameters, $s \in [0, 1]$, η is a spatio-temporal white noise. Following [GRV21], we impose boundary conditions on $\varphi:\ \varphi(s = 0) = \varphi(s = 1) = 0$. The invariant measure of the stochastic Allen-Cahn equation is the Gibbs measure associated with the Hamiltonian

$$U(\varphi) = \beta \int_0^1 \left[\frac{a}{2} (\partial_s \varphi)^2 + \frac{1}{4a} (1 - \varphi(s)^2)^2 \right] \mathrm{d}s \ . \tag{6.142}$$

The Hamiltonian has two global well separated minima φ^+ and φ^-, making it very challenging to sample from the invariant distribution using MCMC methods. Therefore authors propose to use informed base measure with the following Hamiltonian as a proposal

$$U_B(\varphi) = \beta \int_0^1 \left[\frac{a}{2} (\partial_s \varphi)^2 + \frac{1}{2a} \varphi^2 \right] \mathrm{d}s \ . \tag{6.143}$$

To perform sampling, we discretize the field on a uniform grid taking 100 points of spatial variable s. Therefore the dimension of space we sample on is 100.

We set $a = 0.1,\ \beta = 20$. We use RealNVP with 2 flows and hidden layer size 50. Other details are provided in the Table 6.6.

6.11.10 Sampling from Ill-Conditioned Gaussian distribution

In this experiment, we consider the task of sampling from zero-centered Gaussian distribution with ill-conditioned covariance being a diagonal matrix with elements spaced log-linearly between 10^{-2} and 10^2. We set the dimension to 50. Figure 6.14 shows autocorrelation time versus sampling iterations. We perform 50 independent runs of each method to compute autocorrelations and average accross different chains. The experimental details are provided in the Table 6.6.

6.11.11 Sampling from GAN as an Energy-Based Model

In our experiments, we closely follow the exposition of [Che+20a, Section 5], albeit we were not able to exactly reproduce the numbers reported in this paper. We consider the task of sampling in the latent space of learnt GAN model, as presented in [Che+20a]. The crux of the approach is that under some assumptions on the discriminator being close to the optimal one (given by Bayes rule, see [Che+20a]), sampling in latent space from the distribution $p(z) = \frac{p_0(z)}{Z} \exp\left(\mathrm{logit}(D(G(z)))\right)$ is equivalent to sampling from the data distribution, where $p_0(z)$ is a proposal distribution for Generator, G is a Generator and logit is the inverse sigmoid function.

6.11.12 GANs as energy-based models: artificial datasets

We tackle in this section the task of sampling artificial datasets. We set the proposal distribution to the standard normal distribution for all datasets.

We pay particular attention to mixture of Gaussian distributions and Swissroll examples. Similar setup is considered at [Aza+18; Tur+19b; Tan19]. We follow the same experimental setting as provided at [Tan19] and [Che+20a, Section 5.1]. The main difference compared to [Tan19] is that the prior distribution p_0 is a multivariate Gaussian (instead of a uniform).

Mixture of 25 Gaussian distributions We build a dataset of points sampled from a mixture of 25 Gaussian distributions in dimension 2, as in [Tur+19b]. All Gaussian distributions have scale parameter $\sigma = 0.05$, and the coordinates of the means of the Gaussian distributions $\{\mu_i\}_{i=1}^{25}$ are distributed on a uniform grid: $\mu_i \in \{-2, -1, 0, 1, 2\}^2$. From this dataset, we train a WGAN ([ACB17]). We use 4-layer MLP architectures with hidden dimension 128 and 256 for Generator and Discriminator respectively. We set the dimension of the latent space to $d = 2$. We perform 50 independent runs for each method and average metrics across runs. The other experimental details are presented in the Table 6.6.

Table 6.7: GAN sampling from mixture of Gaussian distributions

	Single chain, 25G			Multiple chains, 25G			Single chain, 243G			Multiple chains, 243G		
Model	std	# modes	EMD	std	# modes	EMD	std	# modes	EMD	std	# modes	EMD
[Goo+14b]	.058	25	.064	.058	25	.064	.040	34.8	3.86	.040	34.8	3.86
[Che+20a]	.036	1	6.97	.043	25	.062	.022	1	30.96	.039	98	4.20
MALA	.050	25	.2	.058	25	.058	.030	3.5	24.03	.040	85.8	3.80
Ex^{2}MCMC	.055	25	.18	.056	25	.18	.030	62.9	3.78	.039	89.4	3.60

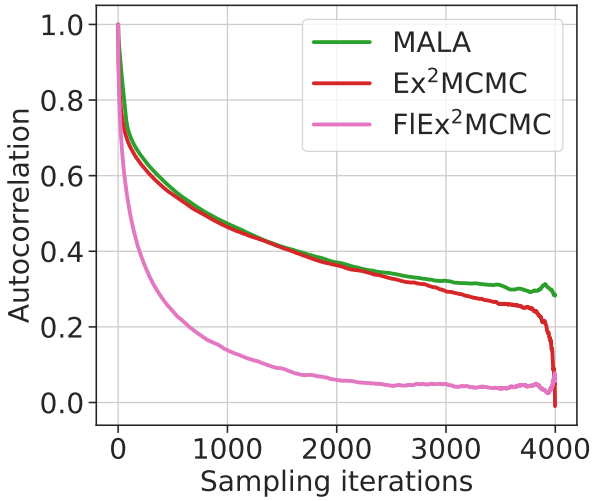


Figure 6.14: Ill-conditioned Gaussian: Autocorrelations vs. sampling iteration.

Mixture of 243 Gaussian distributions To train GAN we construct a dataset from points distributed according to a mixture of 243 Gaussian distributions in dimension 5. All Gaussian distributions have scale parameter $\sigma = 0.05$, and the means of the Gaussian distributions $\{\mu_i\}_{i=1}^{243}$ are distributed on a uniform grid: $\mu_i \in \{-2, 0, 2\}^5$. We set the dimension of the latent space to $d = 5$. We perform 50 independent runs for each method and average metrics across runs. We use 3-layer MLP architectures for both Generator and Discriminator with hidden dimension 256 and 512 respectively.

Swissroll Swissroll is a popular dataset of points in $2d$ distributed along the spiral with some noise. We set the scale factor of noise $\sigma = 0.05$. We set the dimension of the latent space to $d = 2$. To train WGAN we use 3-layer MLP architectures for both Generator and Discriminator with hidden dimension 128 and 256 respectively.

6.11.13 Sampling GANs as energy-based model on CIFAR-10.

The details on experiments with GANs on CIFAR-10 were moved to Section 6.6.

Chapter 7

Approximate Inference in Bayesian Deep Learning

As introduced above, the use of Bayesian neural networks might be crucial for a large scale deployment of deep learning. However, it is not yet clear how to approximate and how to assess the aquality of the approximation of the posterior of a Bayesian neural network.

Recently, [Izm+21] sheds more light on this matter. In this paper, authors propose to launch very long exact HMC chain (thus with a considerable computational budget) on the posterior of different deep neural networks with different datasets, and present criteria to assess convergence of said chains. The idea is to show how effective an exact – or almost exact – approximation of the posterior can be, and how different approximate inference schemes can compare with it, in terms of a few metrics (such as test accuracy, and agreement and total variation or Wasserstein distance of the predictive distributions produced by different competitors compared to this HMC baseline). A NeurIPS 2021 competition (`https://izmailovpavel.github.io/neurips_bdl_competition/`) opened afterwards this comparison to different solutions that could be proposed by competitors participating.

We present here the solution based on Gaussian Stochastic Weight Averaging and Langevin dynamics developed for this competition.

Let us first present a snapshot of different approaches to approximate inference in Bayesian Deep Learning. The base principle is the simple Stochastic Gradient Descent approach. This consists in an optimization run with stochastic mini batching of the loss (2.7). This provides thus stochastic estimates of the gradient of the true loss, and the chain of weights $\{\theta_k\}_{0\leqslant k\leqslant K}$ obtained by such a procedure define a Markov chain.

Considering this optimization sequence as a Markov chain, a natural idea from [Izm+18a] is to consider rather the average of the last points of this sequence as a final sample from the posterior, that is the point $\theta^* \in \mathbb{R}^D$ effectively used is $\theta^* = b^{-1} \sum_{i=K-b+1}^{K} \theta_i$. This leads, as the authors put it, to "wider optima", which ensure a better generalization of the model. Generally, MCMC methods have been used to approximate the posterior of Bayesian neural networks, typically using "full-batch" HMC in the simplest cases [Nea12; CJ21; Wen+20], or stochastic gradient based methods [WT11; ASW14; CFG14; MCF15; AKW12; Din+14; Izm+18b; Zha+19; MHB17; WI20; GF21], typically the stochastic gradient Langevin descent (SGLD) [WT11] or the stochastic gradient Hamiltonian Monte Carlo [CFG14].

An other classical idea is to use Variational inference to approximate the posterior distribution. If one specifies a variational family of distributions $\mathcal{Q}$, we can optimize a lower bound to find the best approximating distribution w.r.t. the KL divergence.

Typically, one chooses $\mathcal{Q}$ to be the class of mean-field Gaussian distributions over the set of weights θ, [Gra11], even though the mean field family can be augmented with normalizing flows, [LW17].

Another practical and popular way to enhance this family distribution is to use dropout, that is, adding a Bernoulli probability r of setting each of the weights in θ to 0 [Sri+14; GG16; KG17; GHK17].

Now, one can thus write the effective variational family as

$$q(\theta) = \prod_{i=1}^{D} \mathrm{Ber}(r)\mathrm{N}(\theta^i; \mu^i, \sigma^i) \ . \tag{7.1}$$

This provides an effective way of approximating the posterior distribution in Bayesian neural networks.

In general, however, one can reinterpret variational inference as a version of stochastic optimization, see [MHB17; Kha+18; HM20].

An other simple yet effective method is given by the Gaussian Stochastic Weight Averaging (SWAG) method [Mad+19]. This method generalizes the Stochastic Weight Averaging which just considers as an effective sample an average of the end of the SGD chain by adding as well an estimate of the covariance at the end of the chain. Typically, given an SGD chain, a point θ^* is sampled according to the Gaussian distribution $\mathrm{N}(\mu^*, \Sigma^*)$, where $\mu^* = b^{-1} \sum_{i=K-b+1}^{K} \theta_i$, and Σ^* is computed as a diagonal + low rank approximation of the empirical covariance matrix of the samples: Write $\sigma^* = (n-1)^{-1} \sum_{i=K-b+1}^{K} (\theta_i - \mu^*)^2$, and for $i \in \{K-b \dots, K\}$, $X_i = (\theta_i - \mu^*) \oslash \sigma^*$, where $\oslash$ denotes the coordinate-wise division, and $\bar{X} = (X_{K-b+1}, \dots, X_K)$ the $D \times b$ matrix.

Let t denote the rank for the low rank approximation of the covariance matrix, $\bar{X} = UDV^T$ the Singular Value Decomposition (SVD), where U is a $D \times D$ matrix, D a $D \times b$ diagonal matrix, and V a $b \times b$ matrix. Note now $\bar{X}_t = U_t D_t V_t^T$ the truncated SVD with the t largest singular values, where U_t is a $D \times t$ matrix, D_t a $t \times t$ matrix and V_t a $b \times t$ matrix.

The approximate covariance is then given by

$$\Sigma^* = \mathrm{diag}\,(\sigma^*) + U_t D_t^2 U_t^T \ ,$$

the diagonal plus low rank approximation of the empirical covariance.

We extend this simple idea for our solution to the NeurIPS competition. First of all, we extend the computation of the covariance matrix by adding three different options. We indeed force the factorization of the covariance matrix of the weights by computing this diagonal plus low rank approximation either for all the weights at once, or for weights layer by layer (thus forcing independence of the different layers), or batching the weights when the layer is large, that is forcing an independence of the weights by blocks of size b_θ.

Our algorithm then consists of three different phases:

- First, run n_1 optimization chains, featuring typically the Adam [KB14] optimizer, targetting the loss (2.5). We use typically a cyclical learning rate here, following [Smi17; Zha+19] to target large valleys of the posterior distribution.
- Then, starting from the n_1 "warm" points obtained by the previous optimization runs, run n_2 SGLD chains with an RMSProp preconditioner [Li+16] (Typically, $n_2 = i \times n_1$, thus starting i chains from each "warm" point). Both those phases are given a similar computational budget.
- Then, compute mean and covariance for the Gaussian distribution approximating each SGLD chain, as described above, for each of those n_2 chains and sample the approximate posterior distribution with the uniform Gaussian mixture with n_2 modes (or even $3n_2$ modes when the mixture is computed with all different options for the covariance matrix).

This algorithm was then evaluated on three different tasks: Classification on the CIFAR-10 and MedMNIST[YSN21] datasets, and regression on the UCI Gap dataset as in [Foo+19; HA15]. A very long run of the HMC algorithm, following [Izm+21], had been done for approximating the posterior distribution of the weights, thus providing a "ground-truth" for the predictive distribution given some test set, which competitors had to approximate. This approximation was then measured using a top-1 agreement metric and a total variation distance for classification task, as well as a wassertein distance computed for regression tasks.

There were two tracks on this competition, a full track featuring all of the tasks described before, and a light one featuring only CIFAR-10 classification task. Our method arrived second on both.

Part III

Contributions: Generative models

Chapter 8

MetFlow: A New Efficient Method for Bridging the Gap between Markov Chain Monte Carlo and Variational Inference

Achille Thin[1], Nikita Kotelevskii[2], Jean-Stanislas Denain[1], Leo Grinsztajn[1], Alain Durmus[3], Maxim Panov[4], Eric Moulines[1]

Abstract

In this contribution, we propose a new computationally efficient method to combine Variational Inference (VI) with Markov Chain Monte Carlo (MCMC). This approach can be used with generic MCMC kernels, but is especially well suited to *MetFlow*, a novel family of MCMC algorithms we introduce, in which proposals are obtained using Normalizing Flows. The marginal distribution produced by such MCMC algorithms is a mixture of flow-based distributions, thus drastically increasing the expressivity of the variational family. Unlike previous methods following this direction, our approach is amenable to the reparametrization trick and does not rely on computationally expensive reverse kernels. Extensive numerical experiments show clear computational and performance improvements over state-of-the-art methods.

8.1 Introduction

One of the biggest computational challenge these days in machine learning and computational statistics is to sample from a complex distribution known up to a multiplicative constant. Indeed, this problem naturally appears in Bayesian inference [Rob07] or generative modeling [KW13b]. Very popular methods to address this problem are Markov Chain Monte Carlo (MCMC) algorithms [Bro+11] and Variational Inference (VI) [WJ+08]. The main contribution of this paper is to present a new methodology to successfully combine these two approaches mitigating their drawbacks and providing the state-of-the-art sampling quality from high dimensional unnormalized distributions.

Starting from a parameterized family of distributions $\mathcal{Q} = \{q_\phi \colon \phi \in \Phi \subset \mathbb{R}^q\}$, VI approximates the intractable distribution with density π on $\mathbb{R}^D$ by maximizing the evidence lower bound (ELBO) defined by

$$\mathcal{L}(\phi) = \int \log\big(\tilde{\pi}(z)/q_\phi(z)\big) q_\phi(z) \mathrm{d}z \;, \tag{8.1}$$

[1]Centre de Mathématiques Appliquées, UMR 7641, Ecole polytechnique, France
[2]CDISE, Skolkovo Institute of Science and Technology, Moscow, Russian Federation
[3]Université Paris-Saclay, ENS Paris-Saclay, CNRS, Centre Borelli, F-91190 Gif-sur-Yvette, France
[4]CS Departement, HSE University, Russian Federation

using an unnormalized version $\tilde{\pi}$ of π, *i.e.* $\pi = \tilde{\pi}/\mathrm{C}_\pi$ setting $\mathrm{C}_\pi = \int_{\mathbb{R}^D} \tilde{\pi}(z)\mathrm{d}z$. Indeed, this approach consists in minimizing $\phi \mapsto \mathrm{KL}(q_\phi|\pi)$ since $\mathcal{L}(\phi) = \log(\mathrm{C}_\pi) - \mathrm{KL}(q_\phi|\pi)$. The design of the family $\mathcal{Q}$ of variational distributions has a huge influence on the overall performance – more flexible families provide better approximations of the target.

Recently, it has been suggested to enrich the traditional mean field variational approximation by combining them with invertible mappings with additional trainable parameters. A popular implementation of this principle is the Normalizing Flows (NFs) approach [DSB16; RM15a; Kin+16b] in which a mean-field variational distribution is deterministically transformed through a fixed-length sequence of parameterized invertible mappings. NFs have received a lot of attention recently and have proven to be very successful for VI and in particular for Variational Auto Encoder; see [KPB19; Pap+19] and the references therein.

The drawback of variational methods is that they only allow the target distribution to be approximated by a parametric family of distributions. On the contrary, MCMC are generic methods which have theoretical guarantees [RC13a]. The basic idea behind MCMC is to design a Markov chain $(z_k)_{k\in\mathbb{N}}$ whose stationary distribution is π. Under mild assumptions, the distribution of z_K converges to the target π as K goes to infinity. Yet, this convergence is in most cases very slow and therefore this class of methods can be prohibitively computationally expensive.

The idea to "bridge the gap" between MCMC and VI was first considered in [SKW15b] and has later been pursued in several works; see [WKS16], [Hof+19] and [CDS18a] and the references therein. In these papers, based on a family of Markov kernels with target distribution π and depending on trainable parameters ϕ, the family $\mathcal{Q}$ consists in the marginal distribution obtained after K iterations of these Markov kernels.

In this paper, we develop a new approach to combine the VI and MCMC approaches. Compared to [SKW15b] and [WKS16], we do not need to extend the variational approximation on the joint distribution of the K samples of the Markov chain and therefore avoid to introduce and learn reverse Markov kernels.

Our main contributions can be summarized as follows:

1) We propose a new computationally tractable ELBO which can be applied to most MCMC algorithms, including Metropolis-Adjusted Langevin Algorithm - MALA - and Hamiltonian Monte Carlo -HMC-. Compared to [Hof17], the Markov kernels can be jointly optimized with the initial distribution q_ϕ^0.

2) We propose an implementation of our approach *MetFlow* using a new family of ergodic MCMC kernels in which the proposal distributions are constructed using Normalizing Flows. Then, combining these Markov kernels with classical mean-field variational initial distributions, we obtain variational distributions with more expressive power than NF models, at the reasonable increase of the computational cost. Moreover, unlike plain NFs, we guarantee that each Markov kernel leaves the target invariant and that each iteration improves the distribution.

3) We present several numerical illustrations to show that our approach allows us to meaningfully trade-off between the approximation of the target distribution and computation, improving over state-of-the-art methods.

Our paper is organized as follows. In Section 8.2, we start by describing our new methodology. Then, in Section 8.3, we introduce MetFlow, a class of "deterministic" MCMC algorithm, taking advantage of the flexibility of Normalizing Flows as proposals. Section 8.4 discusses the related work in more detail. The benefits of our method are illustrated through several numerical experiments in Section 9.4. Finally, we discuss the outcomes of the study and some future research directions in Section 8.6.

8.2 A New Combination Between VI and MCMC

8.2.1 Basics of Metropolis-Hastings

The Metropolis Hastings (MH) algorithm to sample a density π w.r.t. the Lebesgue measure on $\mathbb{R}^D$ defines a Markov chain $(Z_k)_{k\in\mathbb{N}}$ with stationary distribution π as follows. Conditionally to the current state $Z_k \in \mathbb{R}^D$, $k \in \mathbb{N}$, we a proposal $Y_{k+1} = T_\phi(Z_k, U_{k+1})$ is sampled where $(U_k)_{k\in\mathbb{N}^*}$ is a sequence of i.i.d. random variables valued in a measurable space $(\mathsf{U},\mathcal{U})$, with density h w.r.t. to a σ-finite measure μ_{U}, and $T_\phi\colon \mathbb{R}^D \times \mathsf{U} \to \mathbb{R}^D$ is a measurable function, referred to as the *proposal mapping*, and parameterized by $\phi \in \Phi$. In this work, ϕ collectively denotes the parameters used in the proposal distribution, and (U_k) is referred to as the *innovation noise*. Then, Y_{k+1} is accepted with probability $\alpha_\phi^{\mathrm{MH}}\big(Z_k, T_\phi(Z_k, U_{k+1})\big)$, where $\alpha_\phi^{\mathrm{MH}}\colon \mathbb{R}^{2D} \to [0,1]$ is designed so that the resulting Markov kernel, denoted by $M_{\phi,h}$, is reversible w.r.t. π, *i.e.* satisfies the detailed balance $\pi(\mathrm{d}z)M_{\phi,h}(z,\mathrm{d}z') = \pi(\mathrm{d}z')M_{\phi,h}(z',\mathrm{d}z)$. With this notation, $M_{\phi,h}$ can be written, for $z \in \mathbb{R}^D$, $\mathsf{A} \in \mathcal{B}(\mathbb{R}^D)$, as

$$M_{\phi,h}(z,\mathsf{A}) = \int_{\mathsf{U}} h(u) Q_\phi\big((z,u),\mathsf{A}\big)\mu_{\mathsf{U}}(\mathrm{d}u)\ , \tag{8.2}$$

where $\{Q_\phi\colon \phi \in \Phi\}$ is the family of Markov kernels given for any $z \in \mathbb{R}^D$, $u \in \mathsf{U}$, $\mathsf{A} \in \mathcal{B}(\mathbb{R}^D)$, by:

$$Q_\phi((z,u),\mathsf{A}) = \alpha_\phi(z,u)\delta_{T_\phi(z,u)}(\mathsf{A}) + \big\{1-\alpha_\phi(z,u)\big\}\delta_z(\mathsf{A})\ . \tag{8.3}$$

In this definition, δ_z stands for the Dirac measure at z and $\{\alpha_\phi\colon \mathbb{R}^D \times \mathsf{U} \to [0,1]\ ,\ \phi \in \Phi\}$ is a family of acceptance functions related to the MH acceptance probabilities by $\alpha_\phi(z,u) = \alpha_\phi^{\mathrm{MH}}\big(z, T_\phi(z,u)\big)$.

To illustrate this definition, consider first the symmetric Random Walk Metropolis Algorithm (RWM). In such case, $\mathsf{U} = \mathbb{R}^D$, μ_{U} is the Lebesgue measure, and g is the D-dimensional standard normal density. The proposal mapping is

$$T_\phi^{\mathrm{RWM}}(z,u) = z + \Sigma_\phi^{1/2}u, \tag{8.4}$$

where $\{\Sigma_\phi, \phi \in \mathbb{R}^q\}$ is a parametric family of positive definite matrices, and the acceptance function is given by $\alpha_\phi^{\mathrm{RWM}}(z,u) = 1 \wedge \big(\pi(T_\phi^{\mathrm{RWM}}(z,u))/\pi(z)\big)$.

Consider now the Metropolis Adjusted Langevin Algorithm (MALA); see [Bes94b]. Assume that $z \mapsto \log\pi(z)$ is differentiable and denote by $\nabla\log\pi(z)$ its gradient. The proposal mapping and the associated acceptance function for MALA algorithm are given by

$$\begin{aligned} T_\phi^{\mathrm{MALA}}(z,u) &= z + \Sigma_\phi\nabla\log\pi(z) + \sqrt{2}\Sigma_\phi^{1/2}u\ , \\ \alpha_\phi^{\mathrm{MALA}}(z,u) &= 1 \wedge \frac{\pi\big(T_\phi^{\mathrm{MALA}}(z,u)\big)g_\phi\big(T_\phi^{\mathrm{MALA}}(z,u),z\big)}{\pi(z)g_\phi\big(z,T_\phi^{\mathrm{MALA}}(z,u)\big)}\ , \end{aligned} \tag{8.5}$$

where $g_\phi(z_1,z_2) = |\Sigma_\phi|^{-1/2}\mathrm{g}\big(\Sigma_\phi^{-1/2}(z_2 - T_\phi^{\mathrm{MALA}}(z_1,0))\big)$ is the proposal kernel density.

8.2.2 Variational Inference Meets Metropolis-Hastings

Let $K \in \mathbb{N}^*$, $\{\xi_\phi^0\colon \phi \in \Phi\}$ on $\mathbb{R}^D$ a parametric family of distributions and $\{h_i\}_{i=1}^K$ density functions w.r.t μ_{U}. Consider now the following variational family

$$\mathcal{Q} = \{\xi_\phi^K = \xi_\phi^0 M_{\phi,h_1}\cdots M_{\phi,h_K}\colon \phi \in \Phi\}\ , \tag{8.6}$$

obtained by iteratively applying to the initial distribution ξ_ϕ^0 the Markov kernels $(M_{\phi,h_i})_{i=1}^K$.

The objective of VI approach [WJ+08] is to minimize the Kullback-Leibler (KL) divergence $\mathrm{KL}(\xi_\phi^K|\pi)$ w.r.t. the parameter $\phi \in \Phi$ to find the distribution which best fits the target π. For any $\phi \in \Phi$, $u \in \mathsf{U}$ and $z \in \mathbb{R}^D$, denote by $T_{\phi,u}(z) = T_\phi(z,u)$, $\alpha_{\phi,u}(z) = \alpha_\phi(z,u)$ and similarly for any $\mathsf{A} \in \mathcal{B}(\mathbb{R}^D)$, $Q_{\phi,u}(z,\mathsf{A}) = Q_\phi\big((z,u),\mathsf{A}\big)$.

The key assumption in this section is that for any $\phi \in \Phi$ and $u \in \mathsf{U}$, $T_{\phi,u}$ is a C^1 diffeomorphism. This property is satisfied under mild condition on the proposal. In particular, this holds for RWM and MALA associated with the proposal mappings $T^{\mathrm{RWM}}_{\phi,u}$ and $T^{\mathrm{MALA}}_{\phi,u}$. Note that for MALA, $\log \pi$ has to be continuously differentiable with Lipschitz gradient and $\sup_{\phi\in\Phi} \|\Sigma_\phi\|$ has to be small enough see Proposition 69 in the supplement. It holds also for Hamiltonian Monte Carlo at the expense of extending the state space to include a momentum variable, see the supplementary paper Section 8.7.4 and Section 8.8.3. However, it is in general not needed to specify a valid MCMC procedure. For a $\mathrm{C}^1(\mathbb{R}^D, \mathbb{R}^D)$ diffeomorphism ψ, define by $J_\psi(z)$ the absolute value of the Jacobian determinant at $z \in \mathbb{R}^D$.

Lemma 64. *Let $(u, \phi) \in \mathsf{U} \times \Phi$. Assume that ξ^0_ϕ admits a density m^0_ϕ w.r.t. the Lebesgue measure. Assume in addition $T_{\phi,u}$ is a C^1 diffeomorphism. Then, the distribution $\xi^1_\phi(\cdot|u) = \int_{\mathbb{R}^d} m^0_\phi(z_0) Q_{\phi,u}(z_0, \cdot)\mathrm{d}z_0$ has a density w.r.t. the Lebesgue measure given by*

$$m^1_\phi(z|u) = \alpha_{\phi,u}\big(T^{-1}_{\phi,u}(z)\big) m^0_\phi\big(T^{-1}_{\phi,u}(z)\big) J_{T^{-1}_{\phi,u}}(z) + \big\{1 - \alpha_{\phi,u}(z)\big\} m^0_\phi(z) \,, \tag{8.7}$$

and ξ^1_ϕ has a density given by $m^1_\phi(z) = \int m^1_\phi(z|u) h(u) \mu_{\mathsf{U}}(\mathrm{d}u)$.

Proof. Let f be a nonnegative measurable function on $\mathbb{R}^D$. (8.7) follows from the change of variable $z_1 = T_{\phi,u}(z_0)$:

$$\begin{aligned}
&\int_{\mathbb{R}^D} f(z) m^0_\phi(z_0) Q_{\phi,u}(z_0, \mathrm{d}z) \\
&= \int_{\mathbb{R}^D} \Big[m^0_\phi(z_0) \big\{ \alpha_{\phi,u}(z_0) f\big(T_{\phi,u}(z_0)\big) \\
&\qquad + \big(1 - \alpha_{\phi,u}(z_0)\big) f(z_0) \big\} \Big] \mathrm{d}z_0 \,, \\
&= \int_{\mathbb{R}^D} \Big[\{ \alpha_{\phi,u}\big(T^{-1}_{\phi,u}(z_1)\big) m^0_\phi(T^{-1}_{\phi,u}(z_1)) J_{T_u^{-1}}(z_1) \\
&\qquad + (1 - \alpha_{\phi,u}(z_1)) m^0_\phi(z_1) \} f(z_1) \Big] \mathrm{d}z_1 \,.
\end{aligned}$$

□

An induction argument extends this property to the K-th marginal ξ^K_ϕ. Let us define $T^0 = \mathrm{Id}$. For a family $\{T_i\}_{i=1}^K$ of mappings on $\mathbb{R}^D$ and $1 \leqslant i \leqslant k < K$, define $\bigcirc_{j=i}^k T_j = T_i \circ \cdots \circ T_k$ and for a sequence $\{h_i\}_{i=1}^K$ of innovation noise densities w.r.t. μ_{U}, define $h_{1:K}(u_{1:K}) = \prod_{i=1}^K h_i(u_i)$. Finally, set $\alpha^1_{\phi,u}(z) = \alpha_{\phi,u}(z)$ and $\alpha^0_{\phi,u}(z) = 1 - \alpha_{\phi,u}(z)$.

Proposition 65. *Assume that for any $(u, \phi) \in \mathsf{U} \times \Phi$, $T_{\phi,u}$ is a C^1 diffeomorphism and ξ^0_ϕ admits a density m^0_ϕ w.r.t. the Lebesgue measure. For any $\{u_i\}_{i=1}^K \in \mathsf{U}^K$, the distribution $\xi^K_\phi(\cdot \mid u_{1:K}) = \xi^0_\phi Q_{\phi,u_1} \cdots Q_{\phi,u_K}$ has a density m^K_ϕ given by*

$$m^K_\phi(z|u_{1:K}) = \sum_{a_{1:K} \in \{0,1\}^K} m^K_\phi(z, a_{1:K}|u_{1:K}) \,, \tag{8.8}$$

where

$$m^K_\phi(z, a_{1:K}|u_{1:K}) = \prod_{i=1}^K \alpha^{a_i}_{\phi,u_i}\big(\bigcirc_{j=i}^K T^{-a_j}_{\phi,u_j}(z)\big) m^0_\phi\big(\bigcirc_{j=1}^K T^{-a_j}_{\phi,u_j}(z)\big) J_{\bigcirc_{j=1}^K T^{-a_j}_{\phi,u_j}}(z) \,. \tag{8.9}$$

In particular, for a sequence $\{h_i\}_{i=1}^K$ of innovation noise densities, ξ^K_ϕ (8.6) has a density w.r.t. the Lebesgue measure, explicitly given, for any $z \in \mathbb{R}^D$, by

$$m^K_\phi(z) = \int_{\mathsf{U}^K} \big\{ m^K_\phi(z|u_{1:K}) h_{1:K}(u_{1:K}) \big\} \, \mathrm{d}\mu^{\otimes K}_{\mathsf{U}}(u_{1:K}) \,. \tag{8.10}$$

We can now apply the VI approach to the family $\mathcal{Q}$ defined in (8.6). Consider a family of inference function

$$\{\rho(a_{1:K}, u_{1:K} \mid z) \colon z \in \mathbb{R}^D, a_{1:K} \in \{0,1\}^K, u_{1:K} \in \mathsf{U}^K\} \; .$$

This family may depend upon some parameters, implicit in this notation. As shown below, our objective is to take very simple expressions for those functions. We define our ELBO $\mathcal{L}_{\text{aux}}(\phi)$, using now the extended space $(z_K, a_{1:K}, u_{1:K})$, by

$$\mathcal{L}_{\text{aux}}(\phi) = \sum_{a_{1:K} \in \{0,1\}^K} \int h_{1:K}(u_{1:K}) m_\phi^K(z_K, a_{1:K} | u_{1:K}) \\ \times \log\left(\frac{\tilde{\pi}(z_K)\rho(a_{1:K}, u_{1:K}|z_K)}{m_\phi^K(z_K, a_{1:K}|u_{1:K}) h_{1:K}(u_{1:K})} \right) \mathrm{d}z_K \mathrm{d}\mu_{\mathsf{U}}^{\otimes K}(u_{1:K}) \; . \quad (8.11)$$

Note that $\mathcal{L}_{\text{aux}}$ is a lower bound of $\mathcal{L}$ expressed in (8.1) since defining

$$m_\phi^K(z, a_{1:K}, u_{1:K}) = m_\phi^K(z, a_{1:K}|u_{1:K}) h_{1:K}(u_{1:K})$$

and $m_\phi^K(a_{1:K}, u_{1:K}|z) = m_\phi^K(z, a_{1:K}, u_{1:K})/m_\phi^K(z)$, we obtain

$$\mathcal{L}_{\text{aux}}(\phi) = \mathcal{L}(\phi) - \int_{\mathbb{R}^D} m_\phi^K(z_K) D_{\text{KL}} m_\phi^K(\bullet|z_K) \rho(\bullet|z_K) \mathrm{d}z_K \; ,$$

where $D_{\text{KL}} m_\phi^K(\bullet|z_K)\rho(\bullet|z_K)$ denotes the KL divergence between $m_\phi^K(a_{1:K}, u_{1:K}|z_K)$ and $\rho(a_{1:K}, u_{1:K}|z_K)$. We specify the inference functions ρ. In particular, a simple choice is $\rho(a_{1:K}, u_{1:K}|z) = r(a_{1:K}|z, u_{1:K}) h_{1:K}(u_{1:K})$, where r is a similar family of inference function on $\{0,1\}$. This architecture is based on the representation of the Markov kernel (8.3) we built and simplifies our ELBO. In the following, we always assume the form of such inference function. Note here that the key step of our approach for defining $\mathcal{L}_{\text{aux}}$ is to rely on the representation (8.2), allowing us to write explicitly our marginals m_ϕ^K compared to [SKW15b].

The ELBO $\mathcal{L}_{\text{aux}}$ can be optimized w.r.t. ϕ, typically by stochastic gradient methods, which requires an unbiased estimator of the gradient $\nabla \mathcal{L}_{\text{aux}}(\phi)$. Such estimator can be obtained using the reparameterization trick [RMW14]. The implementation of this procedure is a bit more involved in our case, as the integration is done on a mixture of the components $m_\phi^K(z, a_{1:K}|u_{1:K})$. To develop an understanding of the methodology, we consider first the case $K = 1$. Denote by $\boldsymbol{\varphi}$ the density of the D-dimensional standard Gaussian distribution. Suppose for example that we can write $m_\phi^0(z) = \boldsymbol{\varphi}(V_\phi^{-1}(z)) J_{V_\phi^{-1}}(z)$ with $V_\phi(y) = \mu_\phi + \Sigma_\phi^{-1/2} y$. Other parameterization could be handled as well. With two changes of variables, we can integrate w.r.t. $\boldsymbol{\varphi}$, which implies that

$$\mathcal{L}_{\text{aux}}(\phi) = \sum_{a_1 \in \{0,1\}} \int \mathrm{d}y \mathrm{d}\mu_{\mathsf{U}}(u_1) \left[\alpha_{u_1}^{a_1}\big(V_\phi(y)\big) \boldsymbol{\varphi}(y) h(u_1) \log\left(\frac{\tilde{\pi}\big(T_{\phi,u_1}^{a_1}\big(V_\phi(y)\big)\big) r\big(a_1 | T_{\phi,u_1}^{a_1}\big(V_\phi(y)\big), u_1\big)}{m_\phi^1\big(T_{\phi,u_1}^{a_1}\big(V_\phi(y)\big), a_1 | u_1\big)} \right) \right] . \quad (8.12)$$

Justification and extension to the general case $K \in \mathbb{N}^*$ is given in the supplementary paper, see Section 8.8.1. From (**??**), using the general identity $\nabla \alpha = \alpha \nabla \log(\alpha)$, the gradient of $\mathcal{L}_{\text{aux}}$ is given by

$$\nabla \mathcal{L}_{\text{aux}}(\phi) = \sum_{a_1 \in \{0,1\}} \int \mathrm{d}y \mathrm{d}\mu_{\mathsf{U}}(u_1) \alpha_{u_1}^{a_1}\big(V_\phi(y)\big) \boldsymbol{\varphi}(y) h(u_1) \\ \times \left[\nabla \log\left(\frac{\tilde{\pi}(T_{\phi,u_1}^{a_1}\big(V_\phi(y)\big) r\big(a_1 | T_{\phi,u_1}^{a_1}\big(V_\phi(y)\big), u_1\big)}{m_\phi^1\big(T_{\phi,u_1}^{a_1}\big(V_\phi(y)\big), a_1 | u_1\big)} \right) \right. \\ + \nabla \log\big(\alpha_{u_1}^{a_1}\big(V_\phi(y)\big)\big) \\ \left. \times \log\left(\frac{\tilde{\pi}(T_{\phi,u_1}^{a_1}\big(V_\phi(y)\big) r\big(a_1 | T_{\phi,u_1}^{a_1}\big(V_\phi(y)\big), u_1\big)}{m_\phi^1\big(T_{\phi,u_1}^{a_1}\big(V_\phi(y)\big), a_1 | u_1\big)} \right) \right] \; .$$

Therefore, an unbiased estimator of $\nabla\mathcal{L}_{\text{aux}}(\phi)$ can be obtained by sampling independently the proposal innovation $u_1 \sim h_1$ and the starting point $y \sim \mathcal{N}(0, \mathrm{I})$, and then the acceptation $a_1 \sim \mathrm{Ber}\big(\alpha_{u_1}^{a_1}\big(V_\phi(y)\big)\big)$. A complete derivation for the case $K \in \mathbb{N}^*$ is given in the supplementary paper, Section 8.8.2.

8.3 MetFlow: MCMC and Normalizing Flows

In this section, we extend the construction above to a new class of MCMC methods, for which the proposal mappings are Normalizing Flows (NF). Our objective is to capitalize on the flexibility of NF to represent distributions, while keeping the exactness of MCMC. This new class of MCMC are referred to as *MetFlow*, standing for Metropolized Flows.

Consider a flow $T_\phi \colon \mathbb{R}^D \times \mathsf{U} \to \mathbb{R}^D$ parametrized by $\phi \in \Phi$. It is assumed that for any $u \in \mathsf{U}$, $T_{\phi,u} \colon z \mapsto T_\phi(z,u)$ is a C^1 diffeomorphism. Set $\mathsf{V} = \{-1, 1\}$. For any $u \in \mathsf{U}$, consider the involution $\mathring{T}_{\phi,u}$ on $\mathbb{R}^D \times \mathsf{V}$, *i.e.* $\mathring{T}_{\phi,u} \circ \mathring{T}_{\phi,u} = \mathrm{Id}$, defined for $z \in \mathbb{R}^D$, $v \in \{-1, 1\}$ by

$$\mathring{T}_{\phi,u}(z,v) = (T_{\phi,u}^v(z), -v) \; . \tag{8.13}$$

The variable v is called the direction. If $v = 1$ (respectively $v = -1$), the "forward"(resp. "backward") flow $T_{\phi,u}$ (resp. $T_{\phi,u}^{-1}$) is used. For any $z \in \mathbb{R}^D$, $v \in \{-1, 1\}$, $\mathsf{A} \in \mathcal{B}(\mathbb{R}^D)$, $\mathsf{B} \subset \mathsf{V}$, we define the kernel

$$R_{\phi,u}\big((z,v), \mathsf{A} \times \mathsf{B}\big) = \mathring{\alpha}_{\phi,u}(z,v)\delta_{T_{\phi,u}^v(z)}(\mathsf{A}) \otimes \delta_{-v}(\mathsf{B}) + \{1 - \mathring{\alpha}_{\phi,u}(z,v)\}\delta_z(\mathsf{A}) \otimes \delta_v(\mathsf{B}) \; , \tag{8.14}$$

where $\mathring{\alpha}_{\phi,u} \colon \mathbb{R}^D \times \mathsf{V} \to [0,1]$ is the acceptance function.

Proposition 66. *Let ν be a distribution on V, and $(u,\phi) \in \mathsf{U} \times \Phi$. Assume that $\mathring{\alpha}_{\phi,u} \colon \mathbb{R}^D \times \mathsf{V} \to [0,1]$ satisfies for any $(z,v) \in \mathbb{R}^D \times \mathsf{V}$,*

$$\mathring{\alpha}_{\phi,u}(z,v)\pi(z)\nu(v) = \mathring{\alpha}_{\phi,u}\big(\mathring{T}_\phi(z,v)\big)\pi\big(T_{\phi,u}^v(z)\big)\nu(-v)J_{T_{\phi,u}^v}(z) \; . \tag{8.15}$$

Then for any $(u,\phi) \in \mathsf{U} \times \Phi$, $R_{\phi,u}$ defined by (8.14) is reversible with respect to $\pi \otimes \nu$. In particular, if for any $(z,v) \in \mathbb{R}^D \times \mathsf{V}$,

$$\mathring{\alpha}_{\phi,u}(z,v) = \varphi\left(\pi\big(T_{\phi,u}^v(z)\big)\nu(-v)J_{T_{\phi,u}^v}(z)/\pi(z)\nu(v)\right) \; ,$$

for $\varphi \colon \overline{\mathbb{R}}_+ \to \overline{\mathbb{R}}_+$, then (8.15) is satisfied if $\varphi(+\infty) = 1$ and for any $t \in \overline{\mathbb{R}}_+$, $t\varphi(1/t) = \varphi(t)$.

Remark 67. *The condition (8.15) on the acceptance ratio $\mathring{\alpha}_{\phi,u}$ has been reported in [Tie98], Section 2, (see also [AL19a], Proposition 3.5, for extensions to the non reversible case). Standard choices for the acceptance function $\mathring{\alpha}_{\phi,u}$ are the Metropolis-Hastings and Barker ratios which correspond to $\varphi \colon t \mapsto \min(1,t)$ and $t \mapsto t/(1+t)$ respectively.*

If we define for $u \in \mathsf{U}$, $v \in \mathsf{V}$, $z \in \mathbb{R}^D$, $\mathsf{A} \in \mathcal{B}(\mathbb{R}^D)$,

$$Q_{\phi,(u,v)}(z,\mathsf{A}) = R_{\phi,u}((z,v), \mathsf{A} \times \mathsf{V}) = \mathring{\alpha}_{\phi,u}(z,v)\delta_{T_{\phi,u}^v(z)}(\mathsf{A}) + \{1 - \mathring{\alpha}_{\phi,u}(z,v)\}\delta_z(\mathsf{A}) \; , \tag{8.16}$$

we retrieve the framework defined in Section 8.2. In turn, from a distribution ν for the direction, the family $\{Q_{\phi,(u,v)} \colon (u,v) \in \mathsf{U} \times \mathsf{V}\}$ defines a MH kernel, given for $u \in \mathsf{U}$, $z \in \mathbb{R}^D$, $\mathsf{A} \in \mathcal{B}(\mathbb{R}^D)$ by

$$M_{\phi,u,\nu}(z,\mathsf{A}) = \nu(1)Q_{\phi,(u,1)}(z,\mathsf{A}) + \nu(-1)Q_{\phi,(u,-1)}(z,\mathsf{A}) \; .$$

The key result of this section is

Corollary 68. *For any $u \in \mathsf{U}$ and any distribution ν, the kernel $M_{\phi,u,\nu}$ is reversible w.r.t. π.*

Consider for example the MALA proposal mapping T_ϕ^{MALA}. If we set

$$\mathring{\alpha}_{\phi,u}^{\mathrm{MALA}}(z,v) = 1 \wedge \Big\{ \pi\big(T_{\phi,u}^v(z)\big)\nu(-v) J_{T_{\phi,u}^v}(z) / \pi(z)\nu(v) \Big\}$$

with $T_\phi \leftarrow T_\phi^{\mathrm{MALA}}$, then for any $u \in \mathbb{R}^D$ and any distribution ν, $M_{\phi,u,\nu}^{\mathrm{MALA}}$ is reversible w.r.t. π, which is not the case for $Q_{\phi,u}^{\mathrm{MALA}}$ defined in (8.3) with acceptance function $\alpha_\phi^{\mathrm{MALA}}$ given by (8.5). Recall indeed that the reversibility is only satisfied for the kernel $\int Q_{\phi,u}^{\mathrm{MALA}}(z,\mathsf{A})\boldsymbol{\varphi}(u)\mathrm{d}u$.

As the reversibility is satisfied for any $u_{1:K} \in \mathsf{U}^K$, we typically get rid of the integration w.r.t. the innovation noise $h_{1:K}$ and rather consider a fixed sequence $\mathbf{u}_{1:K}$ of proposal noise. For RWM or MALA, this sequence could be a completely uniformly distributed sequence as in Quasi Monte Carlo method for MCMC, see [SC18; CDO+11]. Using definition (8.16) and Proposition 65, we can write the density $m_{\phi,\mathbf{u}_{1:K}}^K(\cdot|v_{1:K})$ of the distribution $\xi_{\phi,\mathbf{u}_{1:K}}^K(\cdot \mid v_{1:K}) = \xi_\phi^0 Q_{\phi,(\mathbf{u}_1,v_1)} \cdots Q_{\phi,(\mathbf{u}_K,v_K)}$. Setting $\alpha_{\phi,u,v}(z) = \mathring{\alpha}_{\phi,u}(z,v)$ as in the previous section, we can write, for any $z \in \mathbb{R}^D$, $a_{1:K} \in \{0,1\}^K$, $v_{1:K} \in \{-1,1\}^K$, $\mathbf{u}_{1:K} \in \mathsf{U}^K$

$$m_{\phi,\mathbf{u}_{1:K}}^K(z,a_{1:K}|v_{1:K}) = m_\phi^0\big(\bigcirc_{j=1}^K T_{\phi,\mathbf{u}_j}^{-v_j a_j}(z)\big) J_{\bigcirc_{j=1}^K T_{\phi,\mathbf{u}_j}^{-v_j a_j}}(z) \prod_{i=1}^K \alpha_{\mathbf{u}_i,v_i}^{a_i}\big(\bigcirc_{j=i}^K T_{\phi,\mathbf{u}_j}^{-v_j a_j}(z)\big) \,. \tag{8.17}$$

Moreover, as reversibility is satisfied for any distribution ν, we could let it depend upon some parameters also denoted ϕ and write $\nu_{\phi,1:K} = \prod_{i=1}^K \nu_{\phi,i}$. Defining an inference function $r_{\mathbf{u}_{1:K}}(a_{1:K}|z,v_{1:K})$, we can thus obtain the lower bound parametrized by the fixed sequence $\mathbf{u}_{1:K}$ and ϕ:

$$\mathcal{L}_{\mathrm{aux}}(\phi;\mathbf{u}_{1:K}) = \int \sum_{v_{1:K}} \sum_{a_{1:K}} m_{\phi,\mathbf{u}_{1:K}}^K(z,a_{1:K}|v_{1:K}) \\ \times \log\left(\frac{\tilde{\pi}(z) r_{\mathbf{u}_{1:K}}(a_{1:K}|z,v_{1:K})}{m_{\phi,\mathbf{u}_{1:K}}^K(z,a_{1:K}|v_{1:K})} \right) \nu_{\phi,1:K}(v_{1:K})\mathrm{d}z \,, \tag{8.18}$$

for which stochastic optimization can be performed using the same reparametrization trick (**??**).

The choice of the transformation T_ϕ is really flexible. Let $\{\mathsf{T}_{\phi,i}\}_{i=1}^K$ be a family of K diffeomorphisms on $\mathbb{R}^D$. A flow model based on $\{\mathsf{T}_{\phi,i}\}_{i=1}^K$ is defined as a composition $\mathsf{T}_{\phi,K} \circ \cdots \circ \mathsf{T}_{\phi,1}$ that pushes an initial distribution ξ_ϕ^0 with density m_ϕ^0 to a more complex target distribution ξ_ϕ^K with density m_ϕ^K, given for any $z \in \mathbb{R}^D$ by $m_\phi^K(z) = m^0\big(\bigcirc_{i=1}^K \mathsf{T}_{\phi,i}^{-1}(z)\big) J_{\bigcirc_{i=1}^K \mathsf{T}_{\phi,i}^{-1}}(z)$, see [TT13; RM15a; KPB19; Pap+19]. We now proceed to the construction of *MetFlow*, based on the same deterministic sequence of diffeomorphisms. A *MetFlow* model is obtained by applying successively the Markov kernels $M_{\phi,1,\nu}, \ldots, M_{\phi,K,\nu}$, written as, for $z \in \mathbb{R}^D$, $\mathsf{A} \in \mathcal{B}(\mathbb{R}^D)$, $i \in \{1,\ldots,K\}$:

$$M_{\phi,i,\nu}(z,\mathsf{A}) = \sum_{v \in \mathsf{V}} \nu(v) \mathring{\alpha}_{\phi,i}(z,v) \delta_{\mathsf{T}_{\phi,i}^v(z)}(\mathsf{A}) + (1 - \sum_{v \in \mathsf{V}} \nu(v) \mathring{\alpha}_{\phi,i}(z,v)) \delta_z(\mathsf{A}) \,.$$

Each of those is reversible w.r.t. the stationary distribution π and thus leaves π invariant. In such a case, the resulting distribution ξ_ϕ^K is a mixture of the pushforward of ξ_ϕ^0 by the flows $\{\mathsf{T}_{\phi,K}^{v_K a_K} \circ \cdots \circ \mathsf{T}_{\phi,1}^{v_1 a_1}$, $v_{1:K} \in \mathsf{V}^K, a_{1:K} \in \{0,1\}^K\}$. The parameters ϕ of the flows $\{\mathsf{T}_{\phi,i}\}_{i=1}^K$ are optimized by maximizing an ELBO similar to (8.18) in which $m_{\phi,\mathbf{u}_{1:K}}^K$ is substituted by $m_{\phi,1:K}^K$ with $T_{\phi,\mathbf{u}_i} \leftarrow \mathsf{T}_{\phi,i}$. The kernel $M_{\phi,i,\nu}$ shares some similarity with Transformation-based MCMC (T-MCMC) introduced in [DB14]. However, the transformations considered in [DB14] and later by [DB+16] are elementary additive or multiplicative transforms acting coordinate-wise. In our contribution, we considered much more sophisticated transformations, inspired by the recent advances on normalizing flows.

Among the different flow models which have been considered recently in the literature [Pap+19], we chose Real-Valued Non-Volume Preserving (RNVP) flows [DSB16] because they are easy to compute and invert. An extension of our work would be to consider other flows, such as Flow++ [Ho+19], which can also be computed and inverted efficiently. We could also use autoregressive models, such as Inverse

Autoregressive Flows [Kin+16b], which have a tractable - albeit non parallelizable - inverse. Even more expressive flows like NAF [Hua+18a], BNAF [CTA19] or UMNN [WL19] could be experimented with. Although they are not invertible analytically, this problem could be solved either by the Distribution Distillation method [Oor+17], or simply by a classic root-finding algorithm: this is theoretically tractable because of the monotonous nature of these flows, and empirically satisfactory [WL19].

8.4 Related Work

In this section, we compare our method with the state-of-the-art for combining MCMC and VI. The first attempt to bridge the gap between MCMC and VI is due to [SKW15b]. The method proposed in [SKW15b] uses a different ELBO, based on the joint distribution of the K steps of the Markov chain $z_{0:K} = (z_0, \dots, z_K)$, whereas MetFlows are based on the marginal distribution of the K-th component. [SKW15b] introduce an auxiliary inference function r and consider the ELBO:

$$\int \log\left(\frac{\tilde{\pi}(z)r(z_{0:K-1}|z_K)}{q_\phi(z_{0:K})}\right) q_\phi(z_{0:K})\mathrm{d}z_{0:K} = \mathcal{L}(\phi) - \int q_\phi^K(z_K) D_{\mathrm{KL}} q_\phi(\cdot \mid z_K) r(\cdot \mid z_K)\mathrm{d}z_K \ , \quad (8.19)$$

where $q_\phi(z_{0:K})$ is the joint distribution of the path $z_{0:K}$ w.r.t. the Lebesgue measure. An optimal choice of the auxiliary inference distribution is $r(z_{0:K-1} \mid z_K) = q_\phi(z_{0:K-1}|z_K)$, the conditional distribution of the Markov chain path $z_{0:K-1}$ given its terminal value z_K, but this distribution is in most cases intractable. [SKW15b; WKS16] discuss several way to construct sensible approximations of $q_\phi(z_{0:K-1} \mid z_K)$ by introducing learnable time inhomogeneous backward Markov kernels. This introduces additional parameters to learn and degrees of freedom in the choice on the reverse kernels which are not easy to handle. On the top of that, this increases significantly the computational budget.

[Hof17] also suggests to build a Markov Chain to enrich the approximation of π. More precisely, [Hof17] optimizes the ELBO with respect to the initial distribution q_ϕ^0, and only uses the MCMC steps to produce "better" samples to the target distribution. However, there is no feedback from the resulting marginal distribution there to optimize the parameters of the variational distribution ϕ. This method does not thus directly and completely unifies VI and MCMC, even though it simplifies the process by avoiding the use of the extended space and the reverse kernels. [RT19] refines [Hof17] by using a contrastive divergence approach; compared to the methodology presented in this paper, [RT19] do not capitalize on the expression of the marginal density.

Another solution to avoid reverse kernels is considered in [CDS18a] which amounts to remove randomness from an Hamiltonian MC algorithm. However, by getting rid of the accept-reject step and the resampling of the momenta in the HMC algorithm, this approach forgoes the guarantees that come with exact MCMC algorithms.

8.5 Experiments

In this section, we illustrate our findings. We present examples of sampling from complex synthetic distributions which are often used to benchmark generative models, such as a mixture of highly separated Gaussians and other non-Gaussian 2D distributions. We also present posterior inference approximations and inpainting experiments on MNIST dataset, in the setting outlined by [LHS17b]. Many more examples are given in the supplementary paper.

We implement the *MetFlow* algorithm described in Section 8.3 to highlight the efficiency of our method. For our learnable transitions T_ϕ, we use RNVP flows.

We consider two settings. In the *deterministic* setting, we use K different RNVP transforms $\{\mathsf{T}_{\phi,i}\}_{i=1}^K$, and the parameters for each individual transform $\mathsf{T}_{\phi,i}$ are different. In the *pseudo-randomized* setting, we define global transformation T_ϕ on $\mathbb{R}^D \times \mathsf{U}$ and set $\mathsf{T}_{\phi,i} = T_\phi(\cdot, \mathbf{u}_i)$, where $\mathbf{u}_{1:K}$ are K independent draws from a standard normal distribution. In such case, the parameters are the same for

the flows $\mathsf{T}_{\phi,i}$, only the innovation noise $\mathbf{u}_i$ differs. Typically, RNVP are encoded by neural networks. In the second setting, the network will thus take as input z and $\mathbf{u}$ stacked together.

In the second setting, once training has been completed and a fit $\hat{\phi}$ of the parameters has been obtained, we can sample additional noise innovations $\mathbf{u}_{(K+1):mK}$. We then consider the distribution given by $\xi_{\hat{\phi}}^{mK} = \xi_{\hat{\phi}}^{K} M_{\hat{\phi},\mathbf{u}_{K+1},\nu}, \ldots, M_{\hat{\phi},\mathbf{u}_{mK},\nu}$ where ν is typically the uniform on $\{-1, 1\}$, as defined as in Section 8.3. mK corresponds to the length of the final Markov chain we consider. In practice, we have found that sampling additional noise innovations this way yields a more accurate approximation of the target, thanks to the asymptotic guarantees of MCMC.

Barker ratios (see Remark 67) have the advantage of being differentiable everywhere. Metropolis-Hastings (MH) ratios are known to more efficient than Barker ratios in the Peskun sense, see [Pes73]. Moreover, although they are not differentiable at every point $z \in \mathbb{R}^D$, differentiating MH ratios is no harder than differentiating a ReLu function. We thus use MH ratios in the following.

All code is written with the Pytorch [Pas+17] and Pyro [Bin+18] libraries, and all our experiments are run on a GeForce GTX 1080 Ti.

8.5.1 Synthetic data. Examples of sampling.

Mixture of Gaussians

The objective is to sample from a mixture of 8 Gaussians in dimension 2, starting from a standard normal prior distribution q^0, and compare MetFlow to RNVP. We are using an architecture of five RNVP flows ($K = 5$), each of which is parametrized by two three-layer fully-connected neural networks with LeakyRelu (0.01) activations. In this example, we consider the pseudo-randomized setting. The results for MetFlow and for RNVPs alone are shown on Figure 8.1. First, we observe that while our method successfully finds all modes of the target distribution, RNVP alone struggles to do the same. Our method is therefore able to approximate multimodal distributions with well separated modes. Here, the mixture structure of the distribution (with potentially $3^5 = 243$ modes) produced by MetFlow is very appropriate to such a problem. On the contrary, classical flows are unable to approximate well separated modes starting from a simple unimodal prior, without much surprise. In particular, mode dropping is a serious issue even in small dimension. Moreover, an other advantage of MetFlow in the pseudo randomized setting is to be able to iterate the learnt kernels which still preserve the target distribution. Iterating MetFlow kernels widens the gap between both approaches, significantly improving the accuracy of our approximation.

Non-Gaussian 2D Distributions

In a second experiment, we sample the non-Gaussian 2D distributions proposed in [RM15a]. Figure 8.2 illustrates the performance of MetFlow compared to RNVP. We are again using 5 RNVPs ($K = 5$) with the architecture described above, and use the pseudo-randomized setting for MetFlow. After only five steps, MetFlow already finds the correct form of the target distribution, while the simple RNVP fails on the more complex distributions. Moreover, iterating again MetFlow kernels allows us to approximate the target distribution with striking precision, after only 50 MCMC steps.

8.5.2 Deep Generative Models

Deep Generative Models (DGM), such as Deep Latent Gaussian Models (see [KW13b; RMW14]) have recently become very popular. The basic assumption in a DGM is that the observed data x is generated by sampling a latent vector z which is used as the input of a deep neural network. This network then outputs the parameters of a family of distributions (e.g., the canonical parameters of exponential family like Bernoulli or Gaussian distributions) from which the data are sampled. Given data generated by a DGM, a classical problem is to approach the conditional distribution $p(z \mid x)$ of the latent variables z given the observation x, using variational inference to construct an amortized approximation.

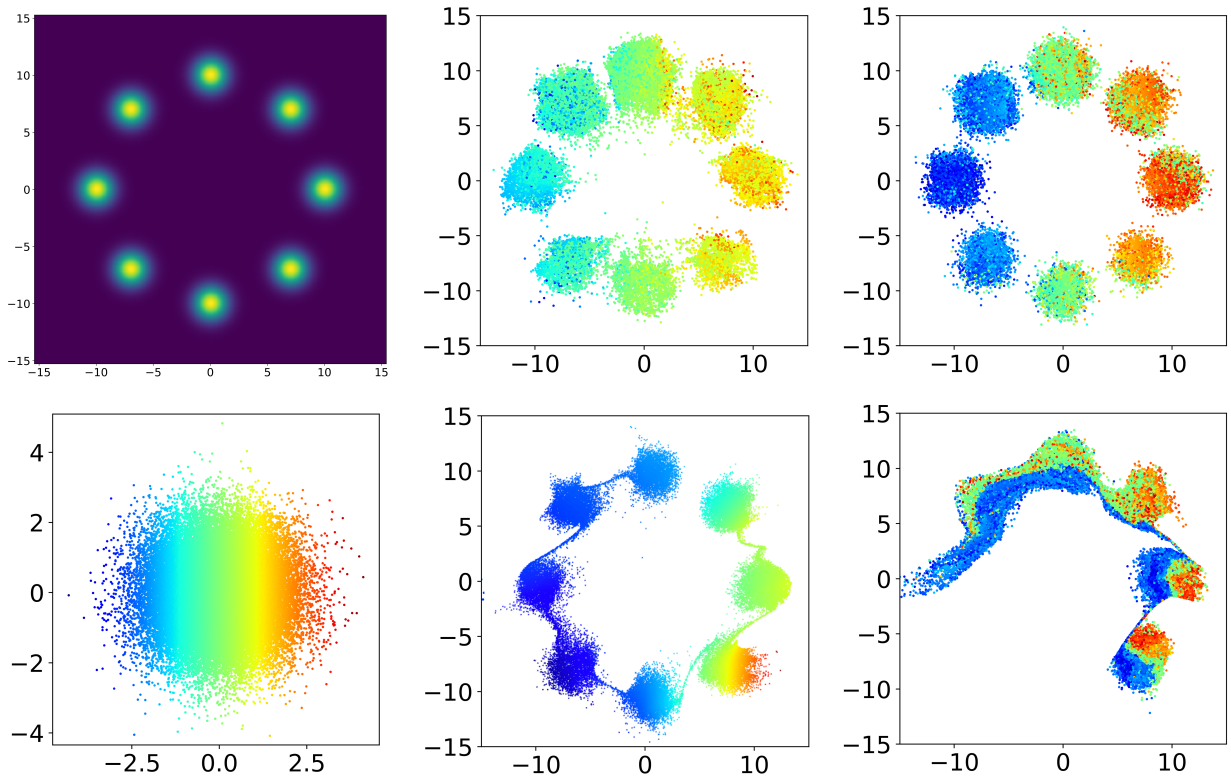


Figure 8.1: Sampling a mixture of 8 Gaussian distributions. Top row from left to right: Target distribution, MetFlow, MetFlow with 145 resampled innovation noise. Bottom row from left to right: Prior distribution, First run of RNVP, Second run of RNVP. MetFlow finds all the modes and improves with more iterations, while RNVP depend on a good initialization to find all the modes and fails to separate them correctly.

We consider the binarized MNIST handwritten digit dataset. The generative model is as follows. The latent variable z is a $l = 64$ dimensional standard normal Gaussian. The observation $x = (x^j)_{j=1}^D$ is a vector of $D = 784$ bits. The bits $(x^j)_{j=1}^D$ are, given the latent variable z, conditionally independent Bernoulli distributed random variables with success probability $p_\theta(z)^j$ where $(p_\theta^j)_{j=1}^D$ is the output of a convolutional neural network. In this framework, p_θ is called the decoder. In the following, we show that our method provides a flexible and accurate variational approximation of the conditional distribution of the latent variable given the observation $p_\theta(z \mid x)$, outperforming mean-field and Normalizing Flows based approaches.

As we are focusing in this paper on the comparison of VI methods to approximate complex distributions and not on learning the Variational Auto Encoder itself, we have chosen to use a fixed decoder for both Normalizing Flows (here, Neural Autoregessive Flows) and MetFlow (with RNVP transforms). The decoder is obtained using state-of-the-art method described in the supplementary paper. We can illustrate the expressivity of MetFlow in two different ways. We first fix L different samples. In this example, we take $L = 3$ images representing the digit "3". We are willing to approximate, for a given decoder p_θ, the posterior distribution $p_\theta(z|(x_i)_{i=1}^L)$. We show in Figure 8.3 the decoded samples corresponding to the following variational approximations of $p_\theta(\cdot|(x_i)_{i=1}^L)$: (i) a NAF trained from the decoder to approximate $p_\theta(\cdot|(x_i)_{i=1}^L)$ and (ii) *MetFlow* in the deterministic setting with $K = 5$ RNVP flows.

Figure 8.3 shows that the samples generated from (i) collapse essentially to one mode corresponding to the first digit. On the contrary, MetFlow is able to capture the three different modes of the posterior and generates much more variability in the decoded samples. The same phenomenon is observed in different settings by varying L and the digits chosen, as illustrated in the supplementary paper.

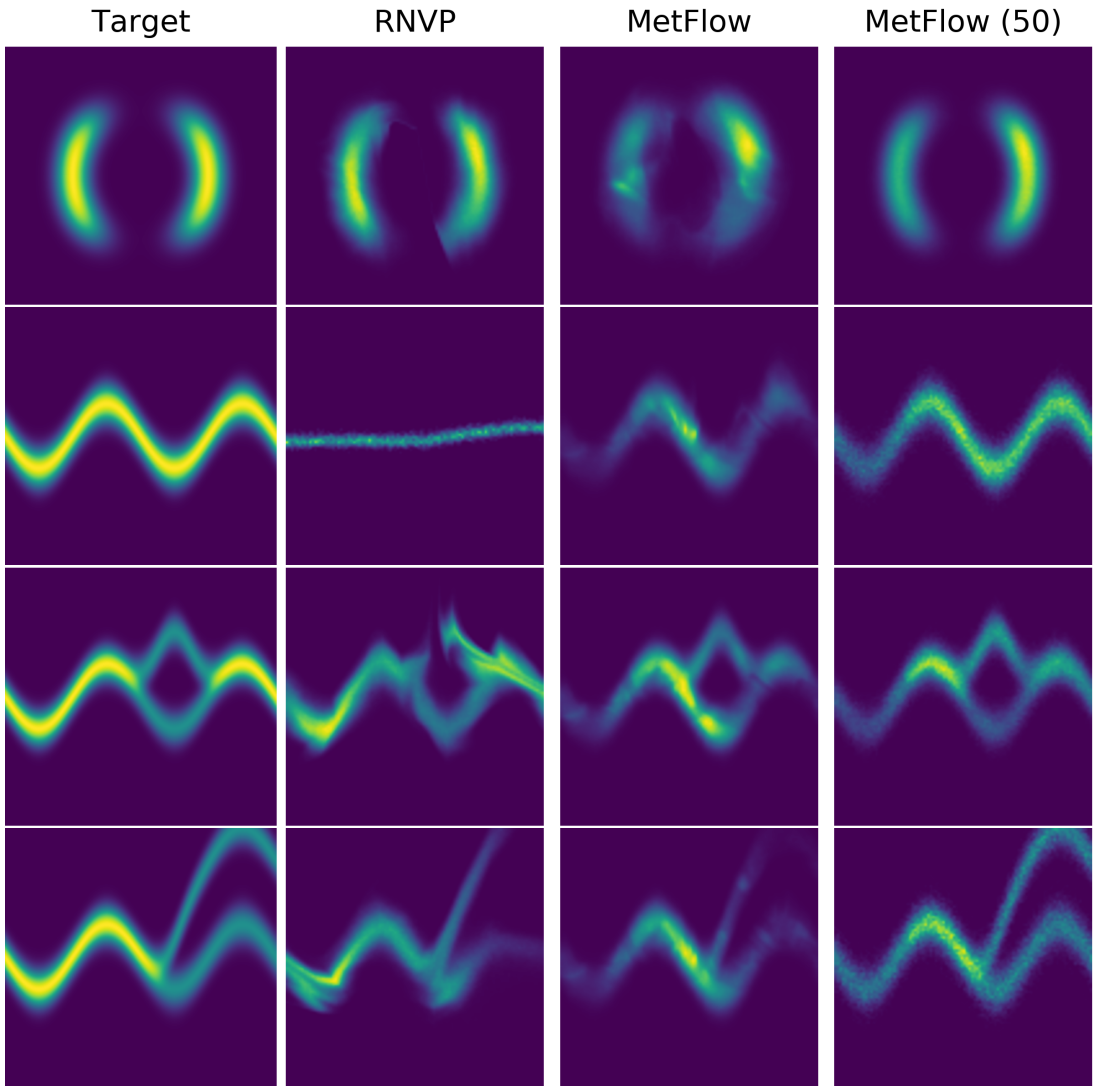


Figure 8.2: Density matching example [RM15a] and comparison between RNVP and MetFlow.

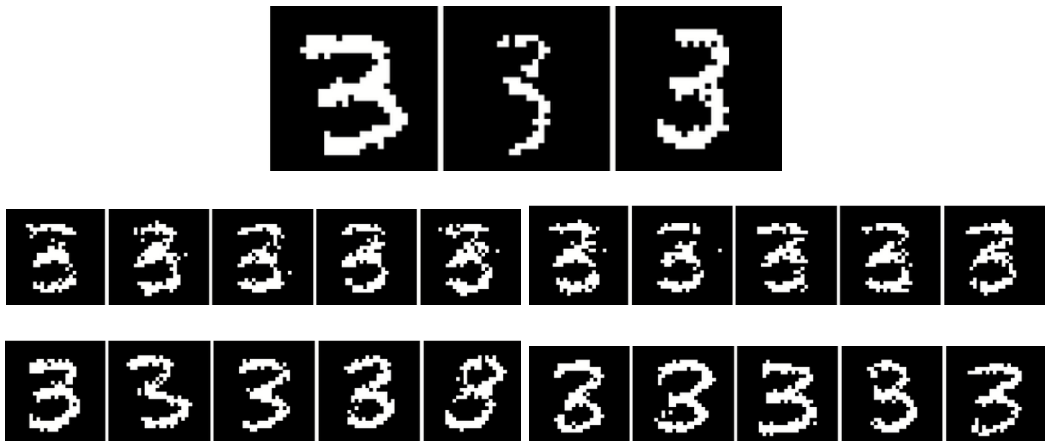


Figure 8.3: Mixture of '3' digits. Top: Fixed digits, Middle: NAF samples, Bottom: MetFlow samples. Compared to NAF, MetFlow is capable to mix better between these modes, while NAF seems to collapse.

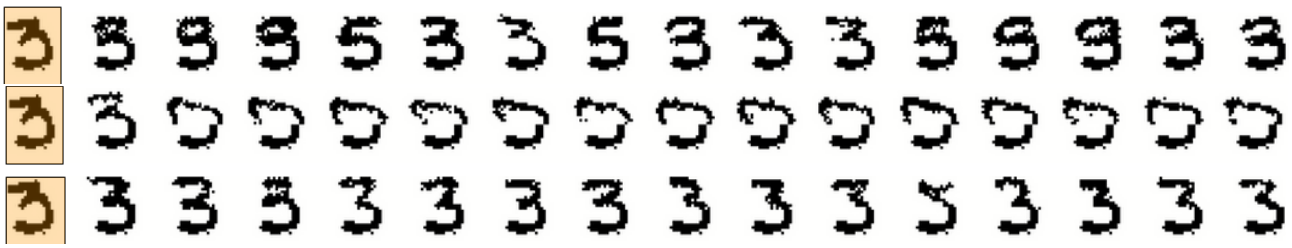


Figure 8.4: Top line: Mean-Field approximation and MetFlow, Middle line: Mean-Field approximation, Bottom line: Mean-Field Approximation and NAF. Orange samples on the left represent the initialization image. We observe that MetFlow easily mixes between the modes while other methods are stuck in one mode.

We now consider the in-painting set-up introduced in [LHS17b], Section 5.2.2. Formally, we in-paint the top of an image using Block Gibbs sampling. Given an image x, we denote x^t, x^b the top and the bottom half pixels. Starting from an image $\mathbf{x_0}$, we sample at each step $z_t \sim p_\theta(z \mid x_t)$ and then $\tilde{x} \sim p_\theta(x \mid z_t)$. We then set $x_{t+1} = (\tilde{x}^t, x_0^b)$. We give the output of this process when sampling from the mean-field approximation of the posterior only, the mean-field pushed by a NAF, or using our method. The result for the experiment can be seen on Figure 8.4.

We can see that MetFlow mixes easily between different modes, and produces sharp images. We recognize furthermore different digits (3,5,9). It is clear from the middle plot that the mean-field approximation is not able to capture the complexity of the distribution $p_\theta(z \mid x)$. Finally, the NAF improves the quality of the samples but does not compare to MetFlow in terms of mixing.

8.6 Conclusions

In this paper, we propose a novel approach to combine MCMC and VI which alleviates the computational bottleneck of previously reported methods. In addition, we design *MetFlow*, a particular class of MH algorithms which fully takes advantage of our new methodology while capitalizing on the great expressivity of NF. Finally, numerical experiments highlight the benefits of our method compared to state-of-the-arts VI.

This work leads to several natural extensions. All NF applications can be adapted with *MetFlow*, which can be seen as a natural extension of a NF framework. *MetFlow* are very appropriate for VAE by amortizing. Due to lack of space, we did not present applications with Forward KL divergence. The mixture structure of the distribution obtained by *MetFlow* suggests the Variational Expectation Maximization (VEM) is a sensible strategy and in particular a chunky version of VEM in the case where the number of steps K is large [VVN03].

Supplementary Material

8.7 Proofs

8.7.1 Proof of Proposition 65

The proof is by induction on $K \in \mathbb{N}^*$. The base case $K = 1$ is given by Lemma 64. Assume now that the statement holds for $K-1 \in \mathbb{N}^*$. Then noticing that $\xi_\phi^K(\cdot|u_{1:K}) = \xi_\phi^{K-1}(\cdot|u_{1:K-1})Q_{\phi,u_K}$, using again Lemma 64 and the induction hypothesis, we get that $\xi_\phi^K(\cdot|u_{1:K})$ admits a density $m_\phi^K(\cdot|u_{1:K})$ w.r.t. the Lebesgue measure given for any $z \in \mathbb{R}^D$ by

$$\begin{aligned} m_\phi^K(z|u_{1:K}) &= \alpha_{\phi,u_K}(T_{\phi,u_K}^{-1}(z)) m_\phi^{K-1}(T_{\phi,u_K}^{-1}(z)|u_{1:K-1}) J_{T_{\phi,u_K}^{-1}}(z) + \{1 - \alpha_{\phi,u_K}(z)\} m_\phi^{K-1}(z|u_{1:K-1}) \,, \\ &= \sum_{a_K \in \{0,1\}} \alpha_{\phi,u_K}^{a_K}(T_{\phi,u_K}^{-a_K}(z)) J_{T_{\phi,u_K}^{-a_K}}(z) m_\phi^{K-1}(T_{\phi,u_K}^{-a_K}(z)|u_{1:K-1}) \,. \end{aligned}$$

Using the induction hypothesis, the density $m_\phi^{K-1}(\cdot|u_{1:K-1})$ w.r.t. the Lebesgue measure of $\xi_\phi^{K-1}(\cdot|u_{1:K-1})$ of the form (8.8). Therefore, we obtain that

$$\begin{aligned} m_\phi^K(z|u_{1:K}) &= \sum_{a_K \in \{0,1\}} \alpha_{\phi,u_K}^{a_K}(T_{\phi,u_K}^{-a_K}(z)) J_{T_{\phi,u_K}^{-a_K}}(z) \left[\sum_{a_{1:K-1} \in \{0,1\}^{K-1}} m_\phi^0(\bigcirc_{j=1}^{K-1} T_{\phi,u_j}^{-a_j}(T_{\phi,u_K}^{-a_K}(z))) \right. \\ &\quad \left. \times J_{\bigcirc_{j=1}^{K-1} T_{\phi,u_j}^{-a_j}}(T_{\phi,u_K}^{-a_K}(z)) \prod_{i=1}^{K-1} \alpha_{\phi,u_i}^{a_i}(\bigcirc_{j=i}^{K-1} T_{\phi,u_j}^{-a_j}(T_{\phi,u_K}^{-a_K}(z))) \right] \\ &= \sum_{a_{1:K} \in \{0,1\}^K} m_\phi^0(\bigcirc_{j=1}^{K} T_{\phi,u_j}^{-a_j}(z)) J_{T_{\phi,u_K}^{-a_K}}(z) J_{\bigcirc_{j=1}^{K-1} T_{\phi,u_j}^{-a_j}}(T_{\phi,u_K}^{-a_K}(z)) \alpha_{\phi,u_K}^{a_K}(T_{\phi,u_K}^{-a_K}(z)) \prod_{i=1}^{K-1} \alpha_{\phi,u_i}^{a_i}(\bigcirc_{j=i}^{K} T_{\phi,u_j}^{-a_j}(z)) \\ &= \sum_{a_{1:K} \in \{0,1\}^K} m_\phi^0(\bigcirc_{j=1}^{K} T_{\phi,u_j}^{-a_j}(z)) J_{\bigcirc_{j=1}^{K} T_{\phi,u_j}^{-a_j}}(z) \prod_{i=1}^{K} \alpha_{\phi,u_i}^{a_i}(\bigcirc_{j=i}^{K} T_{\phi,u_j}^{-a_j}(z)) \,, \end{aligned}$$

where in the last step, we have used that for any differentiable functions $\psi_1, \psi_2 \colon \mathbb{R}^D \to \mathbb{R}^D$, $J_{\psi_1 \circ \psi_2}(z) = J_{\psi_1}(\psi_2(z)) J_{\psi_2}(z)$ for any $z \in \mathbb{R}^D$.

8.7.2 Proof of Proposition 66

Let $(u, \phi) \in \mathsf{U} \times \mathsf{\Phi}$. We want to find a condition such that the kernel $R_{\phi,u}$ defined by (8.14) is reversible w.r.t. $\pi \otimes \nu$ where ν is a distribution on V. This means that for any $\mathsf{A}_1, \mathsf{A}_2 \in \mathcal{B}(\mathbb{R}^D)$, $\mathsf{B}_1, \mathsf{B}_2 \subset \mathsf{V}$,

$$\int_{\mathsf{A}_1 \times \mathsf{A}_2} \sum_{(v,v') \in \mathsf{B}_1 \times \mathsf{B}_2} \pi(z)\nu(v) R_{\phi,u}((z,v), \mathrm{d}z' \times \{v'\}) \mathrm{d}z = \int_{\mathsf{A}_2 \times \mathsf{A}_1} \sum_{(v,v') \in \mathsf{B}_2 \times \mathsf{B}_1} \pi(z)\nu(v) R_{\phi,u}((z,v), \mathrm{d}z' \times \{v'\}) \mathrm{d}z \,. \tag{8.20}$$

By definition of $R_{\phi,u}$ (8.14), the left-hand side simplifies to

$$\begin{aligned} \int_{\mathsf{A}_1 \times \mathsf{A}_2} \sum_{(v,v') \in \mathsf{B}_1 \times \mathsf{B}_2} \pi(z)\nu(v) R_{\phi,u}((z,v), \mathrm{d}z' \times \{v'\}) \mathrm{d}z &= \int_{\mathsf{A}_1} \sum_{v \in \mathsf{B}_1} \pi(z)\nu(v) R_{\phi,u}((z,v), \mathsf{A}_2 \times \mathsf{B}_2) \mathrm{d}z \\ &= \int_{\mathsf{A}_1} \sum_{v \in \mathsf{B}_1} \pi(z)\nu(v) \left\{ \mathring{\alpha}_{\phi,u}(z,v) \delta_{\mathring{T}_{\phi,u}(z,v)}(\mathsf{A}_2 \times \mathsf{B}_2) + \{1 - \mathring{\alpha}_{\phi,u}(z,v)\} \delta_{(z,v)}(\mathsf{A}_2 \times \mathsf{B}_2) \right\} \mathrm{d}z \\ &= \mathfrak{I} + \int_{\mathbb{R}^D} \sum_{v \in \mathsf{V}} \pi(z)\nu(v) \{1 - \mathring{\alpha}_{\phi,u}(z,v)\} \mathbb{1}_{\mathsf{A}_2 \times \mathsf{B}_2}(z,v) \mathbb{1}_{\mathsf{A}_1 \times \mathsf{B}_1}(z,v) \mathrm{d}z \,, \end{aligned} \tag{8.21}$$

where using that $\mathring{T}_{\phi,u}$ is an involution, and for $v \in \mathsf{V}$, the change of variable $\tilde{z} = T^{-v}_{\phi,u}(z)$,

$$\begin{aligned}
\mathfrak{I} &= \int_{\mathbb{R}^D} \sum_{v\in\mathsf{V}} \pi(z)\nu(v)\mathring{\alpha}_{\phi,u}(z,v)\mathbb{1}_{\mathsf{A}_2\times\mathsf{B}_2}(\mathring{T}_{\phi,u}(z,v))\mathbb{1}_{\mathsf{A}_1\times\mathsf{B}_1}(z,v)\mathrm{d}z \\
&= \int_{\mathbb{R}^D} \sum_{v\in\mathsf{V}} \mathring{\alpha}_{\phi,u}(T^v_{\phi,u}(\tilde{z}),v) J_{T^{-v}_{\phi,u}}(\tilde{z})\pi(T^v_{\phi,u}(\tilde{z}))\nu(v)\mathbb{1}_{\mathsf{A}_2\times\mathsf{B}_2}(\tilde{z},-v)\mathbb{1}_{\mathsf{A}_1\times\mathsf{B}_1}(T^v_{\phi,u}(\tilde{z}),v)\mathrm{d}\tilde{z} \\
&= \int_{\mathbb{R}^D} \sum_{\tilde{v}\in\mathsf{V}} \mathring{\alpha}_{\phi,u}(\mathring{T}_{\phi,u}(\tilde{z},\tilde{v})) J_{T^{\tilde{v}}_{\phi,u}}(\tilde{z})\pi(T^{\tilde{v}}_{\phi,u}(\tilde{z}))\nu(-\tilde{v})\mathbb{1}_{\mathsf{A}_2\times\mathsf{B}_2}(\tilde{z},\tilde{v})\mathbb{1}_{\mathsf{A}_1\times\mathsf{B}_1}(\mathring{T}_{\phi,u}(\tilde{z},\tilde{v}))\mathrm{d}\tilde{z}\ , \qquad (8.22)
\end{aligned}$$

where in the last step, we have the change of variable $\tilde{v} = -v$. As for the right-hand side of (8.20), we have by definition (8.14),

$$\begin{aligned}
&\int_{\mathsf{A}_2\times\mathsf{A}_1} \sum_{(v,v')\in\mathsf{B}_2\times\mathsf{B}_1} \pi(z)\nu(v)R_{\phi,u}((z,v),\mathrm{d}z'\times\{v'\})\mathrm{d}z = \int_{\mathsf{A}_2}\sum_{v\in\mathsf{B}_2}\pi(z)\nu(v)R_{\phi,u}((z,v),\mathsf{A}_1\times\mathsf{B}_1)\mathrm{d}z \\
&= \int_{\mathbb{R}^D}\sum_{v\in\mathsf{V}}\pi(z)\nu(v)\left\{\mathring{\alpha}_{\phi,u}(z,v)\mathbb{1}_{\mathsf{A}_1\times\mathsf{B}_1}(\mathring{T}_{\phi,u}(z,v)) + \{1-\mathring{\alpha}_{\phi,u}(z,v)\}\mathbb{1}_{\mathsf{A}_1\times\mathsf{B}_1}(z,v)\right\}\mathbb{1}_{\mathsf{A}_2\times\mathsf{B}_2}(z,v)\mathrm{d}z\ .
\end{aligned}$$

Therefore combining this result with (8.21)-(8.22), we get that if (8.15) holds then $R_{\phi,u}$ is reversible w.r.t. $\pi\otimes\nu$.

Moreover, let us suppose that $\varphi\colon \overline{\mathbb{R}}_+ \to \overline{\mathbb{R}}_+$ is a function satisfying $\varphi(+\infty) = 1$ and for any $t \in \overline{\mathbb{R}}_+$, $t\varphi(1/t) = \varphi(t)$. If for any $z \in \mathbb{R}^D$, $u \in \mathsf{U}$, $v \in \mathsf{V}$,

$$\mathring{\alpha}_{\phi,u}(z,v) = \varphi\left[\frac{\pi(T^v_{\phi,u}(z))\nu(-v)J_{T^v_{\phi,u}}(z)}{\pi(z)\nu(v)}\right]\ ,$$

then

$$\begin{aligned}
\mathring{\alpha}_{\phi,u}(\mathring{T}_\phi(z,v))\pi(T^v_{\phi,u}(z))\nu(-v)J_{T^v_{\phi,u}}(z) &= \varphi\left[\frac{\pi(z)\nu(v)J_{T^{-v}_{\phi,u}}(T^v_{\phi,u}(z))}{\pi(T^v_{\phi,u}(z))\nu(-v)}\right]\pi(T^v_{\phi,u}(z))\nu(-v)J_{T^v_{\phi,u}}(z) \\
&= \varphi\left[\frac{\pi(z)\nu(v)}{\pi(T^v_{\phi,u}(z))\nu(-v)J_{T^v_{\phi,u}}(z)}\right]\frac{\pi(T^v_{\phi,u}(z))\nu(-v)J_{T^v_{\phi,u}}(z)}{\pi(z)\nu(v)}\pi(z)\nu(v) \\
&= \varphi\left[\frac{\pi(T^v_{\phi,u}(z))\nu(-v)J_{T^v_{\phi,u}}(z)}{\pi(z)\nu(v)}\right]\pi(z)\nu(v) = \mathring{\alpha}_{\phi,u}(z,v)\pi(z)\nu(v)\ ,
\end{aligned}$$

which concludes the proof for Proposition 66

8.7.3 Proof of Corollary 68

Let us suppose that for any $u \in \mathsf{U}$, $\mathring{\alpha}_{\phi,u}$ are chosen such that $R_{\phi,u}$ defined by (8.14) is $\pi\otimes\nu$ invariant, where ν is any distribution on V. Then, by definition, for any $\mathsf{A}_1,\mathsf{A}_2 \in \mathcal{B}(\mathbb{R}^D)$, $\mathsf{B}_1,\mathsf{B}_2 \subset \mathsf{V}$, we have:

$$\begin{aligned}
&\int_{\mathsf{A}_1\times\mathsf{A}_2}\sum_{(v,v')\in\mathsf{B}_1\times\mathsf{B}_2}\pi(z)\nu(v)R_{\phi,u}((z,v),\mathrm{d}z'\times\{v'\})\mathrm{d}z = \int_{\mathsf{A}_2\times\mathsf{A}_1}\sum_{(v,v')\in\mathsf{B}_2\times\mathsf{B}_1}\pi(z)\nu(v)R_{\phi,u}((z,v),\mathrm{d}z'\times\{v'\})\mathrm{d}z \\
&\int_{\mathsf{A}_1}\sum_{v\in\mathsf{B}_1}\pi(z)\nu(v)R_{\phi,u}((z,v),\mathsf{A}_2\times\mathsf{B}_2)\mathrm{d}z = \int_{\mathsf{A}_2}\sum_{v\in\mathsf{B}_2}\pi(z)\nu(v)R_{\phi,u}((z,v),\mathsf{A}_1\times\mathsf{B}_1)\mathrm{d}z\ .
\end{aligned}$$

This is true for any $\mathsf{B}_1,\mathsf{B}_2 \subset \mathsf{V}$, hence in particular if $\mathsf{B}_1 = \mathsf{B}_2 = \mathsf{V}$. Then

$$\begin{aligned}
&\int_{\mathsf{A}_1}\sum_{v\in\mathsf{V}}\pi(z)\nu(v)R_{\phi,u}((z,v),\mathsf{A}_2\times\mathsf{V})\mathrm{d}z = \int_{\mathsf{A}_2}\sum_{v\in\mathsf{V}}\pi(z)\nu(v)R_{\phi,u}((z,v),\mathsf{A}_1\times\mathsf{V})\mathrm{d}z \\
&\int_{\mathsf{A}_1}\pi(z)\sum_{v\in\mathsf{V}}\nu(v)Q_{\phi,(u,v)}(z,\mathsf{A}_2)\mathrm{d}z = \int_{\mathsf{A}_2}\pi(z)\sum_{v\in\mathsf{V}}\nu(v)Q_{\phi,(u,v)}(z,\mathsf{A}_1)\mathrm{d}z.
\end{aligned}$$

By definition (8.16). In particular, we obtain exactly, for any $u \in \mathsf{U}$,

$$\int_{\mathsf{A}_1} \pi(z) M_{\phi,u,\nu}(z, \mathsf{A}_2) \mathrm{d}z = \int_{\mathsf{A}_2} \pi(z) M_{\phi,u,\nu}(z, \mathsf{A}_1) \mathrm{d}z.$$

Thus concluding that $M_{\phi,u,\nu}$ is reversible w.r.t. π, for any distribution ν and any $u \in \mathsf{U}$.

8.7.4 Checking the Assumption of Lemma 64 for RWM and MALA algorithms

RWM: For any $u \in \mathbb{R}^D$ and ϕ,

$$T^{\mathrm{RWM}}_{\phi,u}(z) = z + \Sigma_\phi^{1/2} u \ ,$$

which clearly is a $\mathrm{C}^1(\mathbb{R}^D, \mathbb{R}^D)$ diffeomorphism with inverse

$$\{T^{\mathrm{RWM}}_{\phi,u}\}^{-1}(y) = y - \Sigma_\phi^{1/2} u \ .$$

In the simple case where $J_{T^{\mathrm{RWM}}_{\phi,u}}(z) = 1$, using Proposition 65, we get

$$m^K_\phi(z, a_{1:K} | u_{1:K}) = m^0_\phi \left(z - \sum_{j=1}^{k} a_j u_j \right) \prod_{i=1}^{K} \alpha^{a_i}_{\phi,u_i} \left(z - \sum_{j=i}^{k} a_j u_j \right) \tag{8.23}$$

MALA: We prove here that under appropriate conditions, the transformations defined by the Metropolis Adjusted Langevin Algorithm (MALA) are C^1 diffeomorphisms. We consider only the case where $\Sigma_\phi = \gamma \, \mathrm{Id}$. The general case can be easily deduced by a simple adaptation. Remember, for MALA, $\mathcal{T} = \{T_{\gamma,u} \colon z \mapsto z + \gamma \nabla U(z) + \sqrt{2\gamma} u \ : \ u \in \mathbb{R}^D, \gamma > 0\}$, where U is defined as $U(z) = \log(\tilde{\pi}(z))$,

Proposition 69. *Assume the potential U is gradient Lipschitz, that is there exists L in $\mathbb{R}^+$ such that for any $z_1, z_2 \in \mathbb{R}^D$, $\|\nabla U(z_1) - \nabla U(z_2)\| \leqslant L \|z_1 - z_2\|$, and that $\gamma \leqslant 1/(2L)$. Then for any u in $\mathbb{R}^D$, $T_{\gamma,u} \colon z \mapsto z + \gamma \nabla U(z) + \sqrt{2\gamma} u$ is a C^1 diffeomorphism.*

Proof. Let $\gamma \leqslant 1/(2L)$ and $u \in \mathbb{R}^D$. First we show that $T_{\gamma,u}$ is invertible. Consider, for each y in $\mathbb{R}^D$, the mapping $H_{y,u}(z) = y - \sqrt{2\gamma} u - \gamma \nabla U(z)$. We have, for $z_1, z_2 \in \mathbb{R}^D$,

$$\|H_{y,u}(z_1) - H_{y,u}(z_2)\| \leqslant \gamma \|\nabla U(z_1) - \nabla U(z_2)\| \leqslant \gamma L \|z_1 - z_2\| \tag{8.24}$$

and $\gamma L \leqslant 1/2$. Hence $H_{y,u}$ is a contraction mapping and thus has a unique fixed point $z_{y,u}$ and we have:

$$H_{y_u}(z_{y,u}) = z_{y,u} \Rightarrow y = z_{y,u} + \gamma \nabla U(z_{y,u}) + \sqrt{2\gamma} u \tag{8.25}$$

and existence and uniqueness of the fixed point $z_{y,u}$ thus complete the proof for invertibility of $T_{\gamma,u}$. The fact that the inverse of $T_{\gamma,u}$ is C^1 follows from a simple application of the local inverse function theorem. □

Therefore, Proposition 65 can be applied again. Although there is no explicit expression available for m^K_γ because of the intractability of the inverse of $T_{\gamma,u}$, numerical approximations can be used.

8.8 Reparameterization trick and estimator of the gradient

8.8.1 Expression for the reparameterization trick

The goal of the reparameterization trick is to rewrite a distribution depending on some parameters as a simple transformation of a fixed one. The implementation of this procedure is a bit more involved in our case, as the integration is now done on a mixture of the components $m^K_\phi(z, a_{1:K} | u_{1:K})$, for $a_{1:K} \in \{0,1\}^K$. To develop an understanding of the methodology we suggest, we consider first the

case $K = 1$. Recall that $\boldsymbol{\varphi}$ stands for the density of the standard Gaussian distribution over $\mathbb{R}^D$, and suppose here that there exists $V_\phi \colon \mathbb{R}^D \to \mathbb{R}^D$ a C^1 diffeomorphism such that for any $z \in \mathbb{R}^D$, $m^0_\phi(z) = \boldsymbol{\varphi}(V_\phi^{-1}(z)) J_{V_\phi}(V_\phi^{-1}(z))$ which is the basic assumption of the reparameterization trick. With the two changes of variables, $\tilde{z} = T^{-a_1}_{\phi,u_1}(z)$ and $y = V_\phi^{-1}(\tilde{z})$, we get

$$
\begin{aligned}
\mathcal{L}_{\mathrm{aux}}(\phi) &= \int h_1(u_1) m^1_\phi(z, a_1|u_1) \log\left(\frac{\tilde{\pi}(z) r(a_1|z,u_1)}{m^1_\phi(z,a_1|u_1)}\right) \mathrm{d}z \mathrm{d}a_1 \mathrm{d}\mu_{\mathsf{U}}(u_1) \\
&= \sum_{a_1 \in \{0,1\}} \int h_1(u_1) m^0_\phi(T^{-a_1}_{\phi,u_1}(z)) \alpha^{a_1}_{\phi,u_1}(T^{-a_1}_{\phi,u_1}(z)) J_{T^{-a_1}_{\phi,u_1}}(z) \log\left(\frac{\tilde{\pi}(z) r(a_1|z,u_1)}{m^1_\phi(z,a_1|u_1)}\right) \mathrm{d}z \mathrm{d}\mu_{\mathsf{U}}(u_1) \\
&= \sum_{a_1 \in \{0,1\}} \int h_1(u_1) m^0_\phi(\tilde{z}) \alpha^{a_1}_{\phi,u_1}(\tilde{z}) \log\left(\frac{\tilde{\pi}(T^{a_1}_{\phi,u_1}(\tilde{z})) r(a_1|T^{a_1}_{u_1,\phi}(\tilde{z}),u_1)}{m^1_\phi(T^{a_1}_{\phi,u_1}(\tilde{z}),a_1|u_1)}\right) \mathrm{d}\tilde{z} \mathrm{d}\mu_{\mathsf{U}}(u_1) \qquad (8.26) \\
&= \sum_{a_1 \in \{0,1\}} \int h_1(u_1) \boldsymbol{\varphi}(y) \alpha^{a_1}_{\phi,u_1}(V_\phi(y)) \log\left(\frac{\tilde{\pi}(T^{a_1}_{\phi,u_1}(V_\phi(y))) r(a_1|T^{a_1}_{\phi,u_1}(V_\phi(y)),u_1)}{m^1_\phi(T^{a_1}_{\phi,u_1}(V_\phi(y)),a_1|u_1)}\right) \mathrm{d}y \mathrm{d}\mu_{\mathsf{U}}(u_1) \,.
\end{aligned}
$$

This result implies that we can integrate out everything with respect to $\boldsymbol{\varphi}$.

The intuition is the same after K steps, and we can write:

$$
\begin{aligned}
\mathcal{L}_{\mathrm{aux}}(\phi) &= \int h_{1:K}(u_{1:K}) \sum_{a_{1:K} \in \{0,1\}^K} m^K_\phi(z, a_{1:K}|u_{1:K}) \log\left(\frac{\tilde{\pi}(z) r(a_{1:K}|z,u_{1:K})}{m^K_\phi(z,a_{1:K}|u_{1:K})}\right) \mathrm{d}z \mathrm{d}\mu^{\otimes K}_{\mathsf{U}}(u_{1:K}) \\
&= \sum_{a_{1:K} \in \{0,1\}^K} \int h_{1:K}(u_{1:K}) m^0_\phi(\bigcirc_{j=1}^K T^{-a_j}_{\phi,u_j}(z)) J_{\bigcirc_{j=1}^K T^{-a_j}_{\phi,u_j}}(z) \\
&\qquad \times \prod_{i=1}^K \alpha^{a_i}_{\phi,u_i}(\bigcirc_{j=i}^K T^{-a_j}_{\phi,u_j}(z)) \log\left(\frac{\tilde{\pi}(z) r(a_{1:K}|z,u_{1:K})}{m^K_\phi(z,a_{1:K}|u_{1:K})}\right) \mathrm{d}z \mathrm{d}\mu^{\otimes K}_{\mathsf{U}}(u_{1:K}) \\
&= \sum_{a_{1:K} \in \{0,1\}^K} \int h_{1:K}(u_{1:K}) m^0_\phi(\tilde{z}) \prod_{i=1}^K \alpha^{a_i}_{\phi,u_i}(\bigcirc_{j=i-1}^1 T^{a_j}_{\phi,u_j}(\tilde{z})) \\
&\qquad \times \log\left(\frac{\tilde{\pi}(\bigcirc_{j=K}^1 T^{a_j}_{\phi,u_j}(\tilde{z})) r(a_{1:K}|\bigcirc_{j=K}^1 T^{a_j}_{\phi,u_j}(\tilde{z}),u_{1:K})}{m^K_\phi(\bigcirc_{j=K}^1 T^{a_j}_{\phi,u_j}(\tilde{z}),a_{1:K}|u_{1:K})}\right) \mathrm{d}\tilde{z} \mathrm{d}\mu^{\otimes K}_{\mathsf{U}}(u_{1:K}) \\
&= \sum_{a_{1:K} \in \{0,1\}^K} \int h_{1:K}(u_{1:K}) \boldsymbol{\varphi}(y) \prod_{i=1}^K \alpha^{a_i}_{\phi,u_i}(\bigcirc_{j=i-1}^1 T^{a_j}_{\phi,u_j}(V_\phi(y))) \\
&\qquad \times \log\left(\frac{\tilde{\pi}(\bigcirc_{j=K}^1 T^{a_j}_{\phi,u_j}(V_\phi(y))) r(a_{1:K}|\bigcirc_{j=K}^1 T^{a_j}_{\phi,u_j}(V_\phi(y)),u_{1:K})}{m^K_\phi(\bigcirc_{j=K}^1 T^{a_j}_{\phi,u_j}(V_\phi(y)),a_{1:K}|u_{1:K})}\right) \mathrm{d}y \mathrm{d}\mu^{\otimes K}_{\mathsf{U}}(u_{1:K}) \,. \qquad (8.27)
\end{aligned}
$$

8.8.2 Unbiased estimator for the gradient of the objective

We now show in the following that we can also estimate without bias the gradient of the objective starting from the expression (8.27) of the ELBO assuming that we can perform the associated reparameterization trick.

Again, we start by the simple case $K = 1$ to give the reader the main idea of the proof. Using for

any function $f\colon \mathbb{R}^D \to \mathbb{R}_+^*$, $\nabla f = f\nabla\log(f)$, the gradient of our ELBO is given for any $\phi \in \Phi$ by

$$\nabla\mathcal{L}_{\text{aux}}(\phi) = \sum_{a_1\in\{0,1\}} \int h_1(u_1)\boldsymbol{\varphi}(y)\alpha_{\phi,u_1}^{a_1}(V_\phi(y)) \left[\nabla\log\left(\frac{\tilde{\pi}(T_{\phi,u_1}^{a_1}(V_\phi(y))r(a_1|T_{\phi,u_1}^{a_1}(V_\phi(y)),u_1)}{m_\phi^1(T_{\phi,u_1}^{a_1}(V_\phi(y)),a_1|u_1)}\right)\right.$$
$$\left. + \nabla\log[\alpha_{\phi,u_1}^{a_1}(V_\phi(y))]\log\left(\frac{\tilde{\pi}(T_{\phi,u_1}^{a_1}(V_\phi(y))r(a_1|T_{\phi,u_1}^{a_1}(V_\phi(y)),u_1)}{m_\phi^1(T_{\phi,u_1}^{a_1}(V_\phi(y)),a_1|u_1)}\right)\right] \mathrm{d}y\mathrm{d}\mu_{\mathsf{U}}(u_1)\ .$$

Now, this form is particularly interesting, because we can access an unbiased estimator of this sum by sampling $u_1 \sim h_1$ and $y \sim \boldsymbol{\varphi}$, then $a_1 \sim \mathrm{Ber}\{\alpha_{\phi,u_1}^1(V_\phi(y))\}$ and computing the expression between brackets.

The method goes the same way for the K-th step. Indeed for any $\phi \in \Phi$, we have using for any function $f\colon \mathbb{R}^D \to \mathbb{R}_+^*$, $\nabla f = f\nabla\log(f)$,

$$\nabla\mathcal{L}_{\text{aux}}(\phi) = \sum_{a_{1:K}\in\{0,1\}^K} \int h_{1:K}(u_{1:K})\boldsymbol{\varphi}(y)\prod_{i=1}^K \alpha_{\phi,u_i}^{a_i}\left(\bigcirc_{j=i-1}^1 T_{\phi,u_j}^{a_j}(V_\phi(y))\right)$$
$$\times\left[\nabla\log\left(\frac{\tilde{\pi}(\bigcirc_{j=K}^1 T_{\phi,u_j}^{a_j}(V_\phi(y)))r(a_{1:K}|\bigcirc_{j=K}^1 T_{\phi,u_j}^{a_j}(V_\phi(y)),u_{1:K})}{m_\phi^K(\bigcirc_{j=K}^1 T_{\phi,u_j}^{a_j}(V_\phi(y)),a_{1:K}|u_{1:K})}\right)\right.$$
$$+\log\left(\frac{\tilde{\pi}(\bigcirc_{j=K}^1 T_{\phi,u_j}^{a_j}(V_\phi(y)))r(a_{1:K}|\bigcirc_{j=K}^1 T_{\phi,u_j}^{a_j}(V_\phi(y)),u_{1:K})}{m_\phi^K(\bigcirc_{j=K}^1 T_{\phi,u_j}^{a_j}(V_\phi(y)),a_{1:K}|u_{1:K})}\right)$$
$$\left.\times\sum_{i=1}^K \nabla\log[\alpha_{\phi,u_i}^{a_i}\left(\bigcirc_{j=i-1}^1 T_{\phi,u_j}^{a_j}(V_\phi(y))\right)]\right]\mathrm{d}z\mathrm{d}\mu_{\mathsf{U}}^{\otimes K}(u_{1:K})\ . \tag{8.28}$$

And again, this sum can be estimated by sampling $u_{1:K} \sim h_{1:K}$, $y \sim \boldsymbol{\varphi}$, then recursively $a_1 \sim \mathrm{Ber}(\alpha_{\phi,u_1}^1(V_\phi(y))$ and for $i > 1$,

$$a_i \sim \mathrm{Ber}\left(\alpha_{\phi,u_i}^{a_i}\left(\bigcirc_{j=i-1}^1 T_{\phi,u_j}^{a_j}(V_\phi(y))\right)\right)\ .$$

Those variables sampled, the expression between brackets provides an unbiased estimator of the gradient of our objective.

8.8.3 Extension to Hamiltonian Monte-Carlo

Inversibility of kernel (8.3) when the transformations are involutions

The following proposition show a key result for applicability of HMC to our method.

Proposition 70. *Assume that for any $(u,\phi) \in \mathsf{U}\times\Phi$, $T_{\phi,u}$ defines an involution, i.e. $T_{\phi,u}\circ T_{\phi,u} = \mathrm{Id}$. If for any $z \in \mathbb{R}^D$,*

$$\alpha_{\phi,u}(z)\pi(z) = \alpha_{\phi,u}(T_{\phi,u}(z))\pi(T_{\phi,u}(z))J_{T_{\phi,u}}(z)\ ,$$

then the kernel defined by (8.3) and acceptance functions α and transformation T_ϕ is reversible w.r.t. π.

This result is direct consequence of Proposition 66. In particular, the functions φ identified in Proposition 66 are still applicable here.

Application to HMC

An important special example which falls into the setting of Proposition 70 is the Hamiltonian Monte-Carlo algorithm (HMC) [Dua+87; Nea11]. In such a case, the state variable is $z = (q,p) \in \mathbb{R}^{2D}$, where q stands for the position and p the momentum. The unnormalized target distribution is defined as $\tilde{\pi}(q,p) = \tilde{\pi}(q)\boldsymbol{\varphi}(p)$ where $\boldsymbol{\varphi}$ is the density of the D-dimensional standard Gaussian distribution. Define

the potential $U(q) = -\log(\tilde{\pi}(q))$. Hamiltonian dynamics propagates a particle in this extended space according to Hamilton's equation, for any $t \geqslant 0$,

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix} q(t) \\ p(t) \end{bmatrix} = \begin{bmatrix} p(t) \\ -\nabla U(t) \end{bmatrix} .$$

This dynamics preserves the extended target distribution as the flow described above is reversible, symplectic and preserves the Hamiltonian $H(q,p)$, defined as the sum of the potential energy $U(q) = -\log(\tilde{\pi}(q))$ and the kinetic energy $(1/2)p^T p$ (note that we can write $\tilde{\pi}(q,p) \propto \exp(-H(q,p))$), see [BJ18]. It is not however usually possible to compute exactly the solutions of the continuous dynamics described above. However, reversibility and symplectiness can be preserved exactly by discretization using a particular scheme called the leap-frog integrator. Given a stepsize η, the elementary leap-frog $\mathrm{F}_\eta(q_0,p_0)$ for any $(q_0,p_0) \in \mathbb{R}^{2D}$ is given by $\mathrm{F}_\eta(q_0,p_0) = (q_1,p_1)$ where

$$\begin{aligned} p_{1/2} &= p_0 - \eta/2 \nabla U(q_0) \\ q_1 &= q_0 + \eta p_{1/2} \\ p_1 &= p_{1/2} - \eta/2 \nabla U(q_1) . \end{aligned}$$

The N-steps leap-frog integrator with stepsize $\eta > 0$ is defined by $\mathrm{F}_{\eta,N} = \bigcirc_{i=1}^N \mathrm{F}_\eta$ is the N-times composition of F_η. For some parameters $a \in (0,1)$ and $\eta > 0$, consider now the two following transformations:

$$\begin{aligned} T^{\mathsf{ref}}_{\phi,u}(q,p) &= (q, ap + \sqrt{1-a^2}u) , u \in \mathsf{U} = \mathbb{R}^D , \\ T^{\mathrm{F}}_{\phi}(q,p) &= \mathrm{F}_{\eta,N}(q,-p) . \end{aligned}$$

Here, the parameter ϕ stands for the stepsize η and the auto-regressive coefficient a in the momentum refreshment transform. Other parameters could be included as well; see for example [LHS17b]. For any $a \in (0,1)$, $T^{\mathsf{ref}}_{\phi,u}$ is a continuously differentiable diffeomorphism. Then, taking $h = \boldsymbol{\varphi}$ and setting the acceptance ratios to $\alpha^{\mathsf{ref}}_{\phi,u} \equiv 1$, it is easily showed that $M^{\mathsf{ref}}_{\phi,h}$ defined by (8.2) – with $T_{\phi,u} \leftarrow T^{\mathsf{ref}}_{\phi,u}$ – is reversible w.r.t. $\tilde{\pi}$. On the other hand, by composition and a straightforward induction, for any $\phi \in \Phi$, T^{F}_ϕ is continuously differentiable if $\log(\pi)$ is twice continuously differentiable on $\mathbb{R}^D$ and the determinant of its Jacobian is equal to 1 on $\mathbb{R}^D$. In addition, T^{F}_ϕ is an involution by reversibility to the momentum flip operator of the Hamiltonian dynamics [BJ18], Section 6. Indeed, T^{F}_ϕ is here written as the composition of $\mathrm{F}_{\eta,N}$ and the momentum flip operator $S(q,p) = (q,-p)$, $T^{\mathrm{F}}_\phi = \mathrm{F}_{\eta,N} \circ S$. [BJ18] show indeed that $\mathrm{F}_{\eta,N}$ satisfies the following property $\mathrm{F}_{\eta,N} \circ S = S \circ \mathrm{F}^{-1}_{\eta,N}$. As S is also an involution, we can write $S \circ \mathrm{F}_{\eta,N} \circ S = \mathrm{F}^{-1}_{\eta,N}$ and thus $\mathrm{F}_{\eta,N} \circ S \circ \mathrm{F}_{\eta,N} \circ S = \mathrm{Id}$, hence T^{F}_ϕ is an involution. Thus, Proposition 70 applies and the expression given for $\alpha_{\phi,u}$ is the classical acceptance ratio for HMC when $\varphi(t) = \min(1,t)$. HMC algorithm is obtained by alternating repeatedly these two kernels $M^{\mathsf{ref}}_{\phi,h}$ and M^{F}_ϕ. To fall exactly into the framework outlined above, one might consider the extension $\mathsf{U} \times \{0,1\}$, for $(q,p) \in (\mathbb{R}^D)^2$, $(u,v) \in \mathsf{U} \times \{0,1\}$, the transformation

$$T_\phi\big((p,q),(u,v)\big) = \begin{cases} T^{\mathsf{ref}}_{\phi,u}(q,p) \text{ if } v = 1, \\ T^{\mathrm{F}}_\phi(q,p) \text{ if } v = 0, \end{cases} \tag{8.29}$$

and the two densities $h_{\mathsf{ref}}(u,v) = h(u)\mathbb{1}_{\{1\}}(v)$, $h_{\mathrm{F}}(u,v) = f(u)\mathbb{1}_{\{0\}}(v)$ where f is an arbitrary density since T^{F}_ϕ does not depend on u. The kernel defined by (8.2) associated with this transformation and density h_{ref} is $M_{\phi,h_{\mathsf{ref}}} = M^{\mathsf{ref}}_{\phi,h}$, and similarly $M_{\phi,h_{\mathrm{F}}} = M^{\mathrm{F}}_{\phi,g}$. Then, the density after K HMC steps with parameters ϕ can be written as $\xi^K_\phi = \xi^0_\phi (M_{\phi,h_{\mathrm{F}}} M_{\phi,h_{\mathsf{ref}}})^K$.

8.9 Optimization Procedure

8.9.1 Optimization in the general case

We saw in the previous section a way to compute an unbiased estimator of our objective. We will perform gradient ascent in the following, using the estimator provided before.

At each step of optimization, we will sample $u_{1:K} \sim h_{1:K}$, $y \sim \boldsymbol{\varphi}$ and then sequentially $a_1 \sim \mathrm{Ber}(\alpha^1_{\phi,u_1}(V_\phi(y))$ and for $i > 1$, $a_i \sim \mathrm{Ber}(\alpha^{a_i}_{\phi,u_i}(\bigcirc_{j=i-1}^{1} T^{a_j}_{\phi,u_j}(V_\phi(y)))$. With those variables, we can compute the expression between brackets in the formula above (8.28). We then perform a stochastic gradient scheme, repeating the process until convergence of our parameters. A detailed algorithm highlighting the simplicity of our method is presented in 8.9.1. Note that our method "follows" a trajectory, conditioned on noise $u_{1:K}$ and accept/reject Booleans $a_{1:K}$, which is conceptually equivalent to a flow pushforward.

Algorithm 11 Optimization procedure

Input: Transformation T_ϕ, Acceptance function $\alpha_{\phi,u}$, Unnormalized target $\tilde{\pi}(.)$, Variational prior $m^0_\phi(.)$ and reparameterization trick V_ϕ, densities on U $\{h_i, i \in \{1,\dots,K\}\}$ w.r.t. μ_{U}
Input: T optimization steps, schedule $\gamma(t)$
Initialize parameters ϕ;
for $t = 1$ **to** T **do**
  Sample $u_1,\dots,u_K$ from $h_1,\dots,h_K$ innovation noise;
  Sample $y \sim \mathcal{N}(0, \mathrm{I})$ starting point;
  Define current point $z_{aux} \leftarrow V_\phi(y)$;
  $\overline{a} \leftarrow 1$, $S_a \leftarrow 0$ product of the α and sum of the log gradients respectively;
  for $k = 1$ **to** K **do**
    Sample a_k from $\mathrm{Ber}(\alpha^1_{\phi,u_k}(z_{aux}))$;
    $\alpha_{aux} = \alpha^{a_k}_{\phi,u_k}(z_{aux})$;
    Compute $da_k = \nabla_\phi \alpha_{aux}$;
    Update auxiliary variables:
      $\overline{a} \leftarrow \overline{a} \times \alpha_{aux}$, $S_a \leftarrow$, $S_a + da_k/\alpha_{aux}$;
      $z_{aux} \leftarrow T^{a_k}_{\phi,u_k}(z_{aux})$;
  end for
  Compute $dp = \nabla_\phi(\log(\tilde{\pi}(z_{aux}))$;
  Compute $dr = \nabla_\phi(\log(r(a_{1:K} \mid z_{aux}, u_{1:K}))$;
  Compute $dm = \nabla_\phi(|V_\phi|)/|V_\phi| + \nabla_\phi \log(J_{\bigcirc_{j=1}^{K} T^{-a_j}_{\phi,u_j}}(z_{aux})) + S_a$;
  Compute $p = \log(\tilde{\pi}(z_{aux}))$;
  Compute $m = \log\left(\gamma(y)|V_\phi| J_{\bigcirc_{j=1}^{K} T^{-a_j}_{\phi,u_j}}(z_{aux})\overline{a}\right)$;
  Compute $r = \log(r(a_{1:K} \mid z_{aux}, u_{1:K}))$
  Apply gradient update with $\Gamma_\phi = dp + dr - dq + (p + r - q) \times S_a$, schedule $\gamma(t)$;
end for

8.9.2 Optimization for *MetFlow*

We give in the following a detailed algorithm for the optimization procedure for *MetFlow*, highlighting the low computational complexity of our method compared to the previous attempts. Note that the increased complexity is only linear in K the number of steps in our Markov Chain. The functions denoted as acceptance function are for $u \in \mathsf{U}$, $v \in \{-1, 1\}$: $\alpha^1_{u,v}(z) = \mathring{\alpha}_{\phi,u}(z, v)$ and $\alpha^0_{u,v}(z) = 1 - \mathring{\alpha}_{\phi,u}(z, v)$.

Define the Rademacher distribution $\mathrm{Rad}(p)$ on $\{-1, 1\}$ with parameter $p \in [-1, 1]$ by $\mathrm{Rad}(p) = p\delta_1 + (1-p)\delta_{-1}$. The noise v for MetFlow will be sampled using a Rademacher distribution, whose parameter p is let to depend on some parameters ϕ which will be optimized.

Algorithm 12 Optimization procedure for MetFlow

Input: T optimization steps, schedule $\gamma(t)$
Input: Transformation T_ϕ, Acceptance function $\alpha^a_{u,v}$, Unnormalized target $\tilde{\pi}(.)$, Variational prior $m^0_\phi(.)$ and reparameterization trick V_ϕ, densities on U $\{h_i, i \in \{1,\dots,K\}\}$ w.r.t. μ_{U}, probabilities $\{p_{\phi,i}, i \in \{1,\dots,K\}\}$ for Rademacher distributions
Initialize parameters ϕ;
for $t = 1$ **to** T **do**
  Sample $u_1,\dots,u_K$ from $h_1,\dots,h_K$;
  Sample $v_1,\dots,v_K$ from $\mathrm{Rad}(p_{\phi,1}),\dots,\mathrm{Rad}(p_{\phi,K})$;
  Sample $y \sim \mathcal{N}(0,\mathrm{I})$;
  $S_a, S_p \leftarrow 0$;
  $\overline{a} \leftarrow 1$;
  $z_{aux} \leftarrow V_\phi(y)$;
  for $k = 1$ **to** K **do**
    Sample $a_k \sim \mathrm{Ber}(\alpha^1_{u_k,v_k}(z_{aux}))$;
    $\alpha_{aux} = \alpha^{a_k}_{u_k,v_k}(z_{aux})$;
    Compute $da_k = \nabla_\phi \alpha_{aux}$;
    $S_a \leftarrow S_a + da_k/\alpha_{aux}$;
    $S_p \leftarrow S_p + 2^{-1}(1+v_k)\nabla_\phi p_{\phi,k}/p_{\phi,k} - 2^{-1}(1-v_k)\nabla_\phi p_{\phi,k}/(1-p_{\phi,k})$;
    $\overline{a} \leftarrow \overline{a} \times \alpha_{aux}$
    $z_{aux} \leftarrow T^{v_k a_k}_{\phi,u_k}(z_{aux})$;
  end for
  Compute $dp = \nabla_\phi(\log(\tilde{\pi}(z_{aux}))$;
  Compute $dm = \nabla_\phi(|V_\phi|)/|V_\phi| + \nabla_\phi \log(J_{\bigcirc_{j=1}^K T^{-v_j a_j}_{\phi,u_j}}(z_{aux})) + S_a$;
  Compute $dr = \nabla_\phi \log(r(a_{1:K}|z_{aux}, u_{1:K}, v_{1:K}))$
  Compute $p = \log(\tilde{\pi}(z_{aux}))$;
  Compute $m = \log\left(\gamma(y)|V_\phi| J_{\bigcirc_{j=1}^K T^{-v_j a_j}_{\phi,u_j}}(z_{aux})\overline{a}\right)$;
  Compute $r = \log(r(a_{1:K}|z_{aux}, u_{1:K}, v_{1:K}))$;
  Apply gradient update using $\Gamma_\phi = dp + dr - dq + (p + r - q) \times (S_a + S_p)$, schedule $\gamma(t)$;
end for

8.10 Experiments

In all the sampling experiments presented in the main paper (mixture of Gaussians, [RM15a] distributions) as well as for the additional experiments presented here (Neal's funnel distribution, mixture of Gaussians in higher dimensions), the flows used are real non volume preserving (R-NVP) [DSB16] flows. For $\phi \in \Phi$, one R-NVP transform f_ϕ on $\mathbb{R}^D$ is defined as follows, for any $z \in \mathbb{R}^D$, $\tilde{z} = f_\phi(z)$, with $\tilde{z}_{\mathsf{I}} = z_{\mathsf{I}}$ and $\tilde{z}_{\mathsf{I}^c} = z_{\mathsf{I}^c} \odot \exp(s_\phi(z_{\mathsf{I}})) + t_\phi(z_{\mathsf{I}})$, where $\odot$ is the element-wise product, $\mathsf{I} \subset \{1,\dots,D\}$ with cardinal $|\mathsf{I}|$, called an auto regressive mask, and $\mathsf{I}^c = \{1,\dots,D\} \setminus \mathsf{I}$, $s_\phi, t_\phi \colon \mathbb{R}^{|\mathsf{I}|} \to \mathbb{R}^{D-|\mathsf{I}|}$. Typically, the two functions s_ϕ, t_ϕ are parametrized by a neural network and that case ϕ represents the corresponding weights. The use of this kind of functions is justified by the fact that inverses for these flows can be easily computed and have a tractable Jacobian. Indeed, a straightforward calculation leads to for any $z \in \mathbb{R}^D$, $\tilde{z} = f^{-1}_\phi(z)$ with $\tilde{z}_{\mathsf{I}} = z_{\mathsf{I}}$ and $\tilde{z}_{\mathsf{I}^c} = (z_{\mathsf{I}^c} - t_\phi(z_{\mathsf{I}})) \odot \exp(-s_\phi(z_{\mathsf{I}}))$ and $J_{f_\phi}(z) = \exp\{\sum_{j=1}^{D-|\mathsf{I}|}(s_\phi(z_{\mathsf{I}}))_j\}$.

In our experiments, we consider a generalization of this setting which we refer to as latent noisy NVP (LN-NVP) flows defined for $(\phi, u) \in \Phi \times \mathsf{U}$, $f_{\phi,u} \colon \mathbb{R}^D \to \mathbb{R}^D$, of the form for any $z \in \mathbb{R}^D$, $\tilde{z} = f_{\phi,u}(z)$ with $\tilde{z}_{\mathsf{I}} = z_{\mathsf{I}}$, $\tilde{z}_{\mathsf{I}^c} = z_{\mathsf{I}^c} \odot \exp(s_\phi(z_{\mathsf{I}}, u)) + t_\phi(z_{\mathsf{I}}, u)$, where I is an auto-regressive mask and $s_\phi, t_\phi \colon \mathbb{R}^{|\mathsf{I}|} \times \mathsf{U} \to \mathbb{R}^{D-|\mathsf{I}|}$. All the results on R-NVP apply to our LN-NVP, in particular for any $z \in \mathbb{R}^D$, $\tilde{z} = f^{-1}_{\phi,u}(z)$ with $\tilde{z}_{\mathsf{I}} = z_{\mathsf{I}}$ and $\tilde{z}_{\mathsf{I}^c} = (z_{\mathsf{I}^c} - t_\phi(z_{\mathsf{I}}, u)) \odot \exp(-s_\phi(z_{\mathsf{I}}, u))$ and $J_{f_{\phi,u}}(z) = \exp\{\sum_{j=1}^{D-|\mathsf{I}|}(s_\phi(z_{\mathsf{I}}, u))_j\}$.

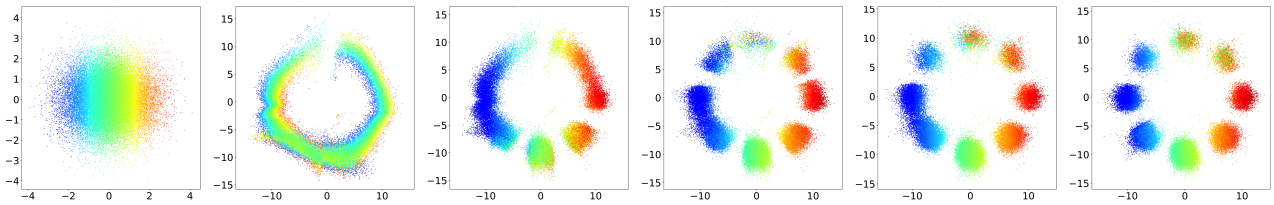


Figure 8.5: Consecutive outputs of each MetFlow kernel. Left: prior normal distribution, then successive effect of the 5 trained MetFlow kernels.

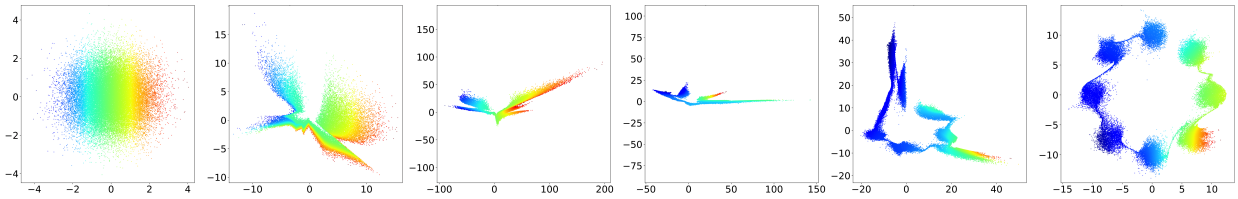


Figure 8.6: Consecutive outputs of each block of R-NVP. Left: prior normal distribution, then successive effect of the 5 trained R-NVP blocks - 6 transforms each.

8.10.1 Mixture of Gaussians: Additional Results

We use MetFlow with the pseudo random setting. Recall that by this, we mean that we define a unique function $(z, u) \to T_\phi(z, u)$ and before training, we sample innovation noise $u_1, \dots, u_K$ that will be "fixed" during all the training process. We then perform optimization on the "fixed" flows $\mathsf{T}_{\phi,i} \leftarrow T_\phi(\cdot, u_i)$, $i \in \{1, \dots, K\}$. The specific setting we consider first is as follows. We compare the sampling based on variational inference with R-NVP Flow and our methodology MetFlow with LN-NVP. In both case, each elementary transformation we consider is the composition of 6 NVP transforms. Here $\mathsf{U} = \mathbb{R}^D$ with $D = 2$ and the target distribution is the one described in Section 8.5.1. Each function t and s in the NVP flows is a neural network with one fully connected hidden layers of size 4 for R-NVP and LN-NVP, and LeakyRelu (0.01) activation functions, and final layers with activation functions tanh for s and identity for t. Each method is trained for 25000 iterations, with ADAM [KB14], with learning rate of 10^{-3} and betas $= (0.9, 0.999)$, and an early stopping criterion of 250 iterations (If within 250 iterations, the objective did not improve, we stop training). The prior distribution is a standard normal Gaussian. Figure 8.5 and Figure 8.6 show the results and the effect of each trained flow. The first pictures are gradient-coloured along the x-axis, with the colour of each point corresponding to its position in the previous image. This helps us to understand how well the processes mixes and transforms an original distribution. It is interesting to note that our method produces flows that are all relevant and all help the distribution to improve at each step. On the contrary, classical R-NVP and flows in general only take interest in what happens at the very last (K-th) step: the pushforward distribution can take any form in between. However, this representation is only interesting when each of the transformations have different sets of parameters, increasing a lot the total numbers of parameters to tune. The evolution of our method can be interpreted as well with the presence of acceptance ratios, that at each step "filter" or "tutor" the distribution, helping it not to get too far from the target. This allows us as well to introduce the setting where we can iterate our flows to refine more and more the approximation we produce in the end. We also show that our method is robust to a change in the prior (even a violent one). In the following figure, we change the standard normal prior to a mixture of two well-separated Gaussian distributions, with a standard deviation of 1 and means $(-50, 0)$ and $(50, 0)$. Figure 8.7 shows that MetFlow still remains efficient after iterating only a few times the learnt kernel, without retraining it at any point. It obviously does not work for R-NVP. Recall that by "iterating a kernel", we mean (as is explained in the main paper) that additional noise $\{u_i\}_{i \geqslant K+1}$ is sampled to define other transformations $\mathsf{T}_{\phi,i} \leftarrow T_\phi(\cdot, u_i)$ and thus additional MetFlow kernels. This particular property could find applications with time-changing data

domains. Indeed, it does not require retraining existing models, while remaining an efficient way to sample from a given distribution at low computational cost.

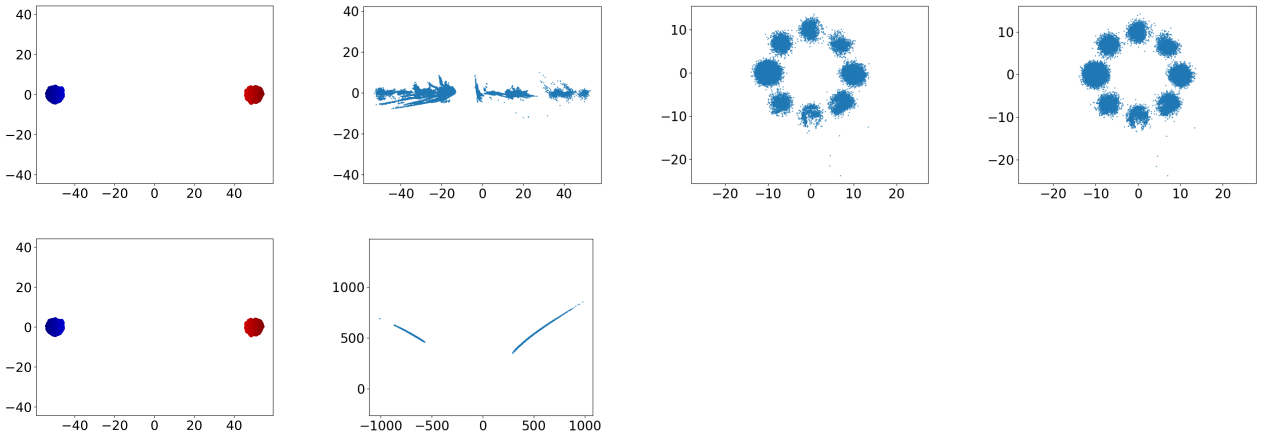


Figure 8.7: Changing the prior to a mixture of two separated Gaussians, having trained the method on a standard normal prior. Top row, from left to right: Subsituted prior, 5 trained MetFlow kernels, re-iteration of 100 MetFlow kernels, 200 MetFlow kernels. Bottom row: Substituted prior, R-NVP flow.

Mode retrieval Challenge

Variational Inference, and especially VAEs, suffer from a known issue: Variational pruning. This happens in particular because the exclusive KL used in VI ($D_{\mathrm{KL}}q\pi = \int q \log(\frac{q}{\pi})$ does not penalize variational distribution when it does not put mass on areas where π is important. This tends to concentrate the variational approximation on a mode of the target and "forget", or "prune", the others. To benchmark our method built to use the exactness of MCMC to tackle this issue, we consider a mixture of Gaussian distributions as our target, in several dimensions. More precisely, in dimension d, our target is a mixture of 8 Gaussians with variance 1, located at the corners of the d-dimensional hypercube. We show in Figure 8.8 the number of modes retrieved by a large variety of state-of-the-art methods, both from a pure VI framework or a pure MCMC framework, as well as performance of our hybrid method. Our methods here are deterministic MetFlow with 7 R-NVP blocks and MetFlow in fully random setting, which will be defined below. Constituents of two t and s being as previously two one-layer fully connected neural networks, input dimension of the dimension considered d (again, we take $\mathsf{U} = \mathbb{R}^d$) and hidden dimension of $2d$. The activation functions stay the same, LeakyRelu (0.01) for all layers except final layer, and final layers with activation functions tanh for s and identity for t. We present as well another setting, referred to as “fully random”, which will be further studied in Section 8.10.4. In this setting, we use LN-NVP blocks for which the innovation noise u is resampled at each epoch. Note that if the R-NVP blocks are all parameterized independently, LN-NVP blocks are jointly parameterized by the same functions t, s, reducing drastically the number of parameters. This introduces additional randomness into our algorithm, which counterfacts the decrease in the number of parameters compared to the deterministic setting on the overall computational complexity. t and s are similar to R-NVP described previously, except that their input layer is now $2d$ twice the dimension, as we consider here noise innovation $u \in \mathbb{R}^d$. We compare our method with MCMC methods, NUTS in this case, with different initializations and number of chains. First, we considered 8 chains of NUTS, ran to match training time of MetFlow. We initialized it either with a wide Gaussian, covering all the modes of the target distribution, or with a Gaussian trained before using Hoffman’s VI procedure [Hof17] (consider mean-field approximation by minimizing KL-divergence between initial distribution and target). Note, that both these priors simplifies sampling drastically, since they transfer an information about target to prior. We also present a result obtained using 8-time longer NUTS chain, with wider prior initialization. Finally, we present results obtained using NAF, a state-of-the-art

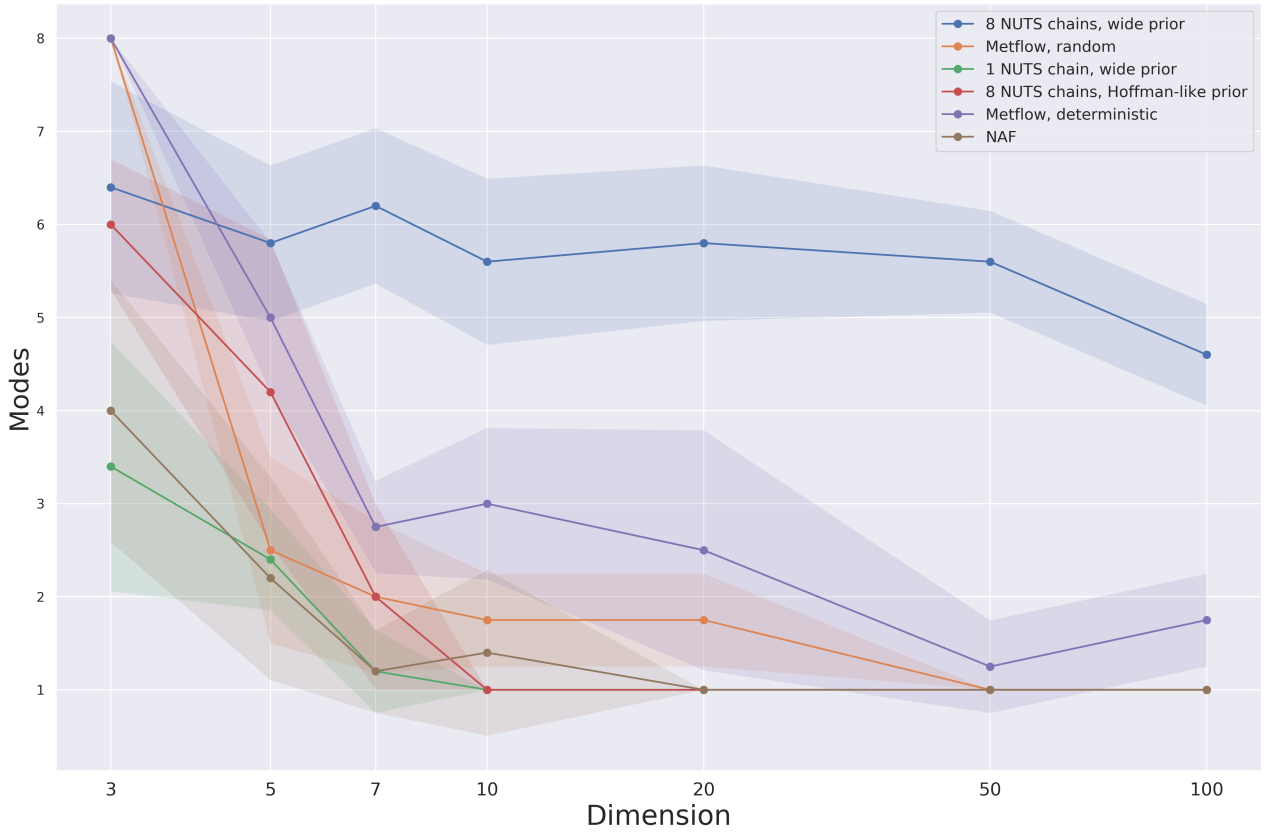


Figure 8.8: Number of modes retrieved by different methods. The target distribution is a mixture of 8 isotropic Gaussian distributions of variance 1 located at the corners of a d-dimensional hypercube. The methods were trained in a way to use the same computational budget, and mode retrieval is computed by counting the number of samples in a ball of radius $2\sqrt{d}$ around the center of a mode. Error bars represent the standard deviation of the mean number of modes retrieved for different runs of the method (different initialization and random seed).

Normalizing Flow.

We can see, that the best result displayed on the plot is 8 chains of NUTS algorithm, initialized from wide prior, which captures all these modes. We stress, that this result is expected, because randomly initialized chains from that prior have more chances to find different modes. While usage of NUTS with target-aware prior leads to spectacular results, our method shows that even without this cheating knowledge we can obtain decent results We emphasize the fact that the time complexity shown here is training time for MetFlow or NAF, as sampling time is negligible once training is complete, while it is really sampling time for MCMC techniques as NUTS. In particular, this explains why MCMC techniques are not used per say in VI and VAEs, or just as some short-run MCMC starting from an informative initial distribution [Hof17]. We show in particular in the following figure that we combine more effectively the exactness of MCMC in a VI setting, for a similar computational cost.

The results show competitiveness of MetFlow method compared to state-of-the-art VI and the hybrid MCMC/VI method described in [Hof17]. The complexity of the distribution obtained after using MetFlow is indeed higher than either NAF or NUTS run after Hoffman initialization, even with a very long chain. Note here that if 8 chains of NUTS run after a wide initial distribution effectively retrieve more modes, in practical applications, such as the VAE, this would be impossible. VAEs would require for each example of the dataset to run a long NUTS chain starting from a wide prior to get relevant samples, which is not realistic in practice (hence the suggestion from [Hof17] to start a short-run MCMC from a "good" initial distribution). Our method thus produces a better approximation of a complex

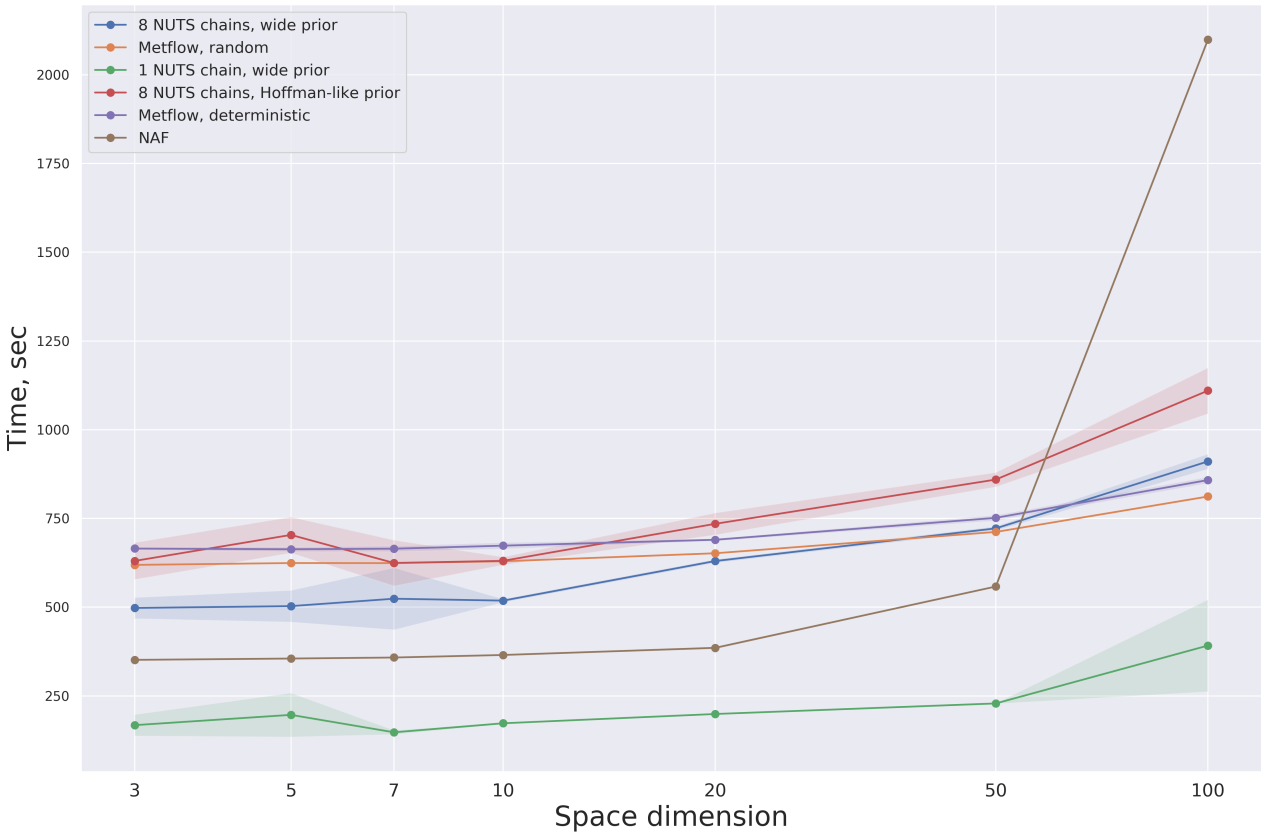


Figure 8.9: The figure demonstrates, that all these methods used approximately the same computational budget.

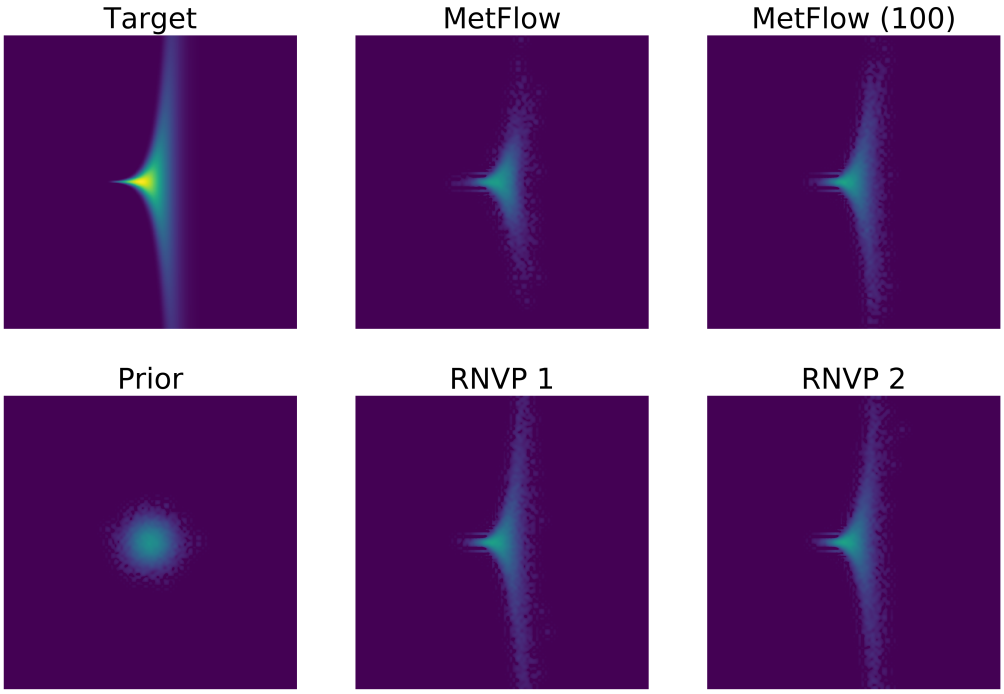


Figure 8.10: Density matching for funnel. Top row: Target distribution, MetFlow with 5 trained kernels, MetFlow with 5 trained kernel iterated 100 times. Bottom row: Prior distribution, First run of 5 R-NVP, second run of 5 R-NVP

multimodal distribution.

8.10.2 Funnel distribution

We now test our approach on an other hard target distribution for MCMC proposed in [Nea03]. The target distribution has density Lebesgue density w.r.t. the Lebesgue measure given for any $z \in \mathbb{R}^2$ by

$$\pi(z) \propto \exp(z_1^2/2) \exp(z_2 e^{z_1}/2) \, . \tag{8.30}$$

We are using exactly the same setting for MetFlow with LN-NVP and R-NVP as for the mixture of Gaussian. We train the flows with the same optimizer, for 25000 iterations again. Figure 8.10 show the results of MetFlow and R-NVP flows. Again, as we are in the pseudo random setting, we sample 95 additional innovation noise $(u_i)_{i\in\{6,\ldots,100\}}$ and show the distribution produced by 100 MetFlow kernels, having only trained the first five transformations $\{\mathsf{T}_{\phi,i} \, , \, i \in \{1, \ldots, 5\}\}$, as described in Section 9.4 of the main document. We can observe that after only five steps, the distribution has been pushed toward the end of the funnel. However, the amplitude is not recovered fully. It can be interpreted in light of the "tutoring" analogy we used previously. As the Accept/Reject control at each step the evolution of the points, if the number of steps is too small, the proposals do not got to the far end of the distribution. However, the plots show that the proposal given by MetFlow are still learnt relevantly, as Figure 8.10 illustrates that iterating only a few more MetFlow kernels matches the target distribution in all its amplitude.

8.10.3 Real-world inference - MNIST

In the experiments ran on MNIST described in the main paper, we fix a decoder p_θ. To obtain our model, we use the method described in [Kin+16b]. The mean-field approximation output by the encoder is here pushforward by a flow. In [Kin+16b], the flows used were Inverse Autoregressive Flows, introduced in the same work. We choose here to use a more flexible class of flows neural auto-regressive flows (NAF), introduced in [Hua+18a], which show better results in terms of log-likelihood. In practice, we use convolutional networks for our encoder and decoder, matching the architecture described in [SKW15b]. The inference network (encoder) consists of three convolutional layers, each with filters of size 5×5 and a stride of 2, and output 16, 32, and 32 feature maps, respectively. The output of the third layer feeds a fully connected layer with hidden dimension 450, which then is fully connected to the outputs, means and standard deviations, of the size of our latent dimension, here 64. Softplus activation functions are used everywhere except before outputting the means. For the decoder, a similar but reversed architecture is used, using upsampling instead of stride, again as described in [SKW15b]. The Neural Autoregressive Flows are given by pyro library. NAF have a hidden layer of 64 units, with an AutoRegressiveNN which is a deep sigmoidal flow [Hua+18a], with input dimension 64 and hidden dimension 128. Our data is the classical stochastic binarization of MNIST [SM08a]. We train our model using Adam optimizer for 2000 epoches, using early stopping if there is no improvement after 100 epoches. The learning rate used is 10^{-3}, and betas $(0.9, 0.999)$. This produces a complex and expressive model for both our decoder and variational approximation.

We show first using mixture experiments that MetFlow can overcome state-of-the-art sampling methods. The mixture experiment described in the main paper goes as follows. We fix L different samples, and wish to approximate the complex posterior $p_\theta(\cdot|(x_i)_{i=1}^L)) \propto p(z) \prod_{i=1}^L p_\theta(x_i|z)$. We give two approximations of this distribution, given by a state-of-the-art method, and MetFlow. The state-of-the-art method is a NAF a hidden layer of 16 units, with an AutoRegressiveNN which is a deep sigmoidal flow [Hua+18a], with input dimension 64 and hidden dimension 128. MetFlow is trained here in the deterministic setting, with 5 blocks of 2 R-NVPs, where again each function t and s is a neural network with one fully connected hidden layers of size 128 and LeakyRelu (0.01) activation functions, and final layers with activation functions tanh for s and identity for t. MetFlow and NAF are optimized using 10000 batches of size 250 and early stopping tolerance of 250, with ADAM, with learning rate of 10^{-3} and betas $= (0.9, 0.999)$. We use Barker ratios as well here. The prior in both cases is a standard 64-dimensional Gaussian.

Figure 8.11: Fixed digits for mixture experiment.

We see on Figure 8.13 that if NAF "collapses" to one fixed digit (thus one specific mode of the posterior), MetFlow Figure 8.12 is able to find diversity and multimodality in the posterior, leading even to "wrong" digits sometimes, showing that it truly explores the complicated latent space.

Moreover, we can compare the variational approximations computed for our VAE (encoder - mean field approximation - and encoder and NAF) to MetFlow approximation. MetFlow used here are the composition of 5 blocks of 2 R-NVPs (deterministic setting), where each function t and s is a neural network with one fully connected hidden layers of size 128 and LeakyRelu (0.01) activation functions, and final layers with activation functions tanh for s and identity for t. MetFlow is optimized using 150 epoches of 192 batches of size 250 over the dataset MNIST, with ADAM, with learning rate of 10^{-4} and betas $= (0.9, 0.999)$ and early stopping with tolerance of 25 (if within 25 epoches the objective did not improve, we stop training). The optimization goes as follows. We fix encoder (mean-field approximation) and decoder (target distribution). For each sample x in a minibatch of the dataset,

Figure 8.12: Mixture of 3, MetFlow approximation.

Figure 8.13: Mixture of 3, NAF approximation.

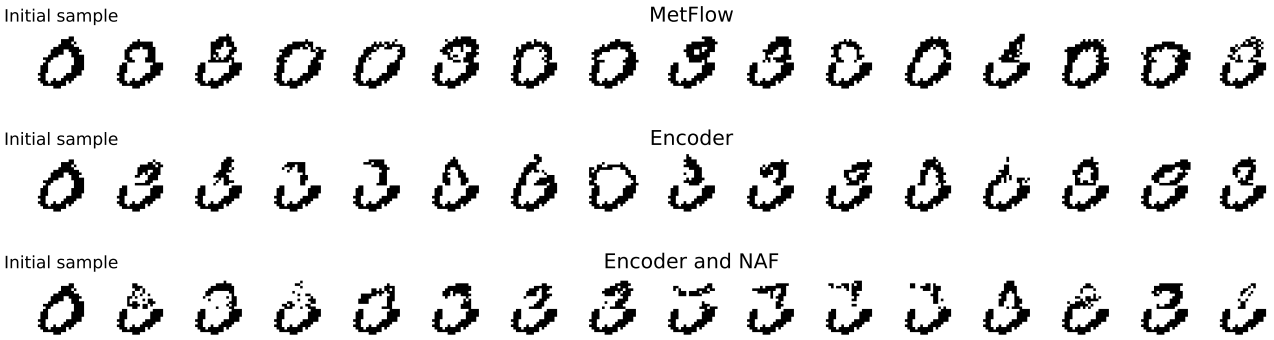


Figure 8.14: Gibbs inpainting experiments starting from digit 0.

we optimize MetFlow starting from the prior given by the encoder (mean-field approximation) and targetting posterior $p_\theta(\cdot|x)$. Note that during all this optimization procedure, the initial distribution of MetFlow is fixed to be the encoder, which corresponds to freeze the corresponding parameters. This is a simple generalization of our method to amortized inference. In the following, we use Barker ratios. The additional results are given by Figures 8.14 to 8.18.

The encoder represents just the mean-field approximation here, while encoder and NAF represent the total variational distribution learnt by the VAE described above.

8.10.4 Additional setting of experiments

So far, we have described two settings. The first one, that we have called deterministic, in which the transformations take no input "innovation noise", but all have different sets of parameters. The second one, pseudo random, defines one global transformation T_ϕ with a unique set of parameters ϕ, considers K initially sampled at random "innovation noise" $u_1, \dots, u_K$, and considers densities $h_{1:K}$ such that the noise resampled at every step of the optimization is "fixed". It then learns the parameters for the global transformation using the considered transformations $\mathsf{T}_{\phi,i} \leftarrow T_\phi(\cdot, u_i)$, $i \in \{1, \dots, K\}$. We can consider here a last setting, *fully random*, on which a global transformation T_ϕ with a unique set of parameters ϕ is considered. However, now, we consider a unique density h (typically Gaussian) to sample the "innovation noise" at every step of the optimization - note that this is still covered by Algorithm 8.9.2. This allow us to consider properly random transformations, and looks more like the classical framework of MCMC. Even if this introduces more noise in our stochastic gradient descent, it encourages MetFlow kernels trained to incorporate a lot more diversity. This can be seen with the two experiments described before, for mixture of different digits in MNIST, or the inpainting experiments.

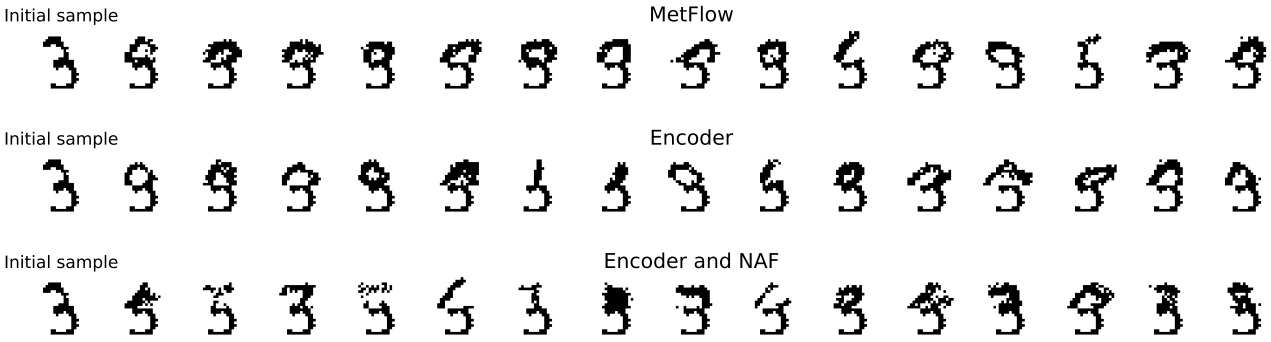


Figure 8.15: Gibbs inpainting experiments starting from digit 3.

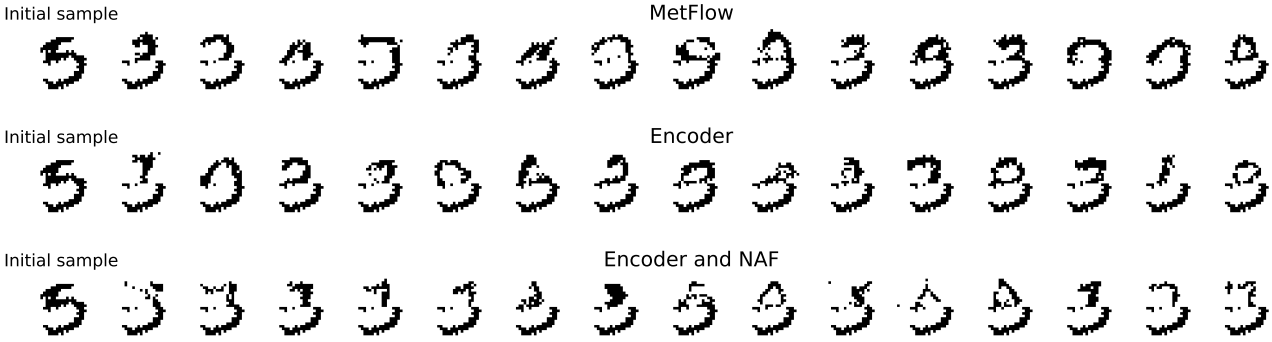


Figure 8.16: Gibbs inpainting experiments starting from digit 9.

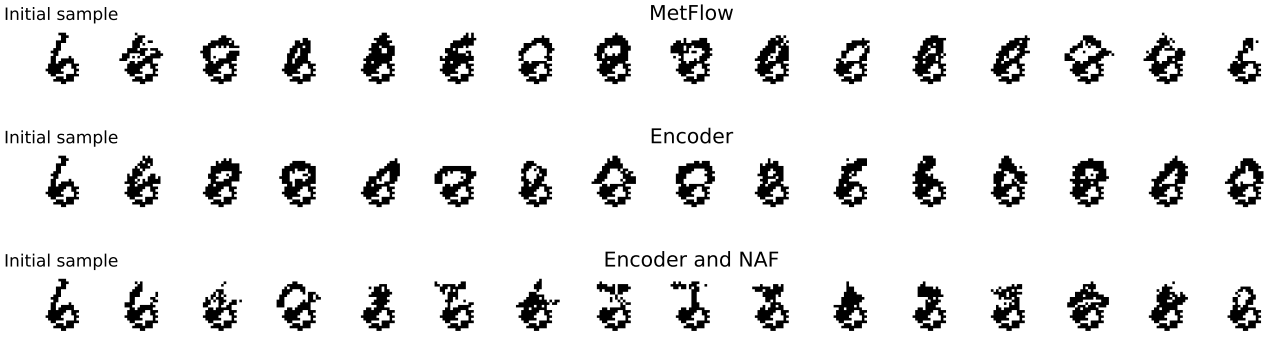


Figure 8.17: Gibbs inpainting experiments starting from digit 6.

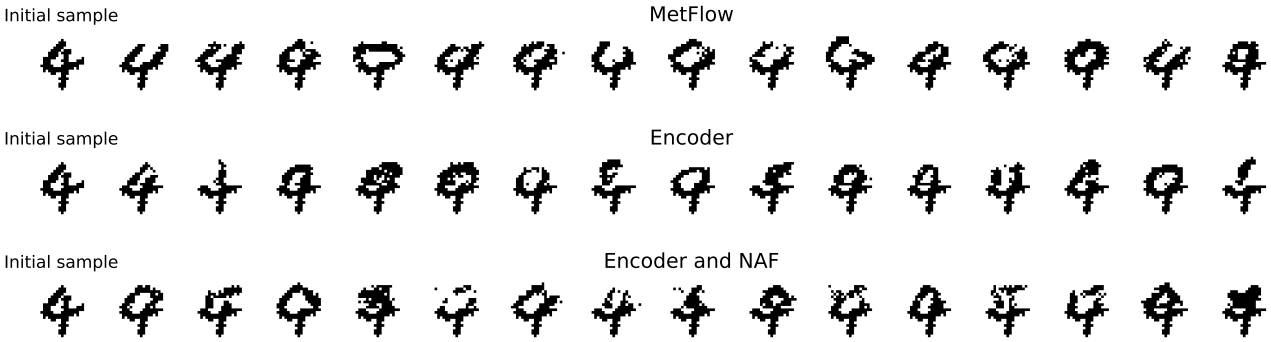


Figure 8.18: Gibbs inpainting experiments starting from digit 4.

Even though experiments can show more diversity, it is important to note that they typically require a longer number of epoches to reach convergence. We give in Section 8.10.4 a comparison of the three settings for a mixture of digits problem. As we can see, the diversity introduced by our method is highest in the fully random setting. We can see a wider variety of 3, even though other digits tend to appear more when we decode them as well.

Figure 8.19: Comparison of the different settings described for a mixture of digits experiment. From left to right, deterministic setting, pseudo-random setting, fully random setting.

Chapter 9

Monte Carlo Variational Auto Encoders

Achille Thin[1], Nikita Kotelevskii[2], Arnaud Doucet[3], Alain Durmus[4], Eric Moulines[1], Maxim Panov[5]

Abstract

Variational auto-encoders (VAE) are popular deep latent variable models which are trained by maximizing an Evidence Lower Bound (ELBO). To obtain tighter ELBO and hence better variational approximations, it has been proposed to use importance sampling to get a lower variance estimate of the evidence. However, importance sampling is known to perform poorly in high dimensions. While it has been suggested many times in the literature to use more sophisticated algorithms such as Annealed Importance Sampling (AIS) and its Sequential Importance Sampling (SIS) extensions, the potential benefits brought by these advanced techniques have never been realized for VAE: the AIS estimate cannot be easily differentiated, while SIS requires the specification of carefully chosen backward Markov kernels. In this paper, we address both issues and demonstrate the performance of the resulting Monte Carlo VAEs on a variety of applications.

9.1 Introduction

Variational Auto-Encoders (VAE) introduced by [KW13a] are a very popular class of methods in unsupervised learning and generative modelling. These methods aim at finding a parameter θ maximizing the marginal log-likelihood $p_\theta(x) = \int p_\theta(x, z)\mathrm{d}z$ where $x \in \mathbb{R}^N$ is the observation and $z \in \mathbb{R}^d$ is the latent variable. They rely on the introduction of an additional parameter ϕ and a family of variational distributions $q_\phi(z|x)$. The joint parameters $\{\theta, \phi\}$ are then inferred through the optimization of the Evidence Lower Bound (ELBO) defined as

$$\mathcal{L}(\theta, \phi) = \int \log\left(\frac{p_\theta(x, z)}{q_\phi(z|x)}\right) q_\phi(z|x)\mathrm{d}z \; = \log p_\theta(x) - D_{\mathrm{KL}}\big(q_\phi(z|x) \parallel p_\theta(z|x)\big) \leqslant \log p_\theta(x) \; .$$

The design of expressive variational families has been the topic of many works and is a core ingredient in the efficiency of VAE [RM15b; Kin+16a]. Another line of research consists in using positive unbiased estimators $\hat{p}_\theta(x)$ of the loglikelihood $p_\theta(x)$ for q_ϕ, *i.e.* $\mathbb{E}_{q_\phi}[\hat{p}_\theta(x)] = p_\theta(x)$. Indeed, as noted in [MR16], it follows from Jensen's inequality that

$$\mathcal{L}(\theta, \phi) = \mathbb{E}_{q_\phi}[\log \hat{p}_\theta(x)] \leqslant \log p_\theta(x) \; . \tag{9.1}$$

[1]Centre de Mathématiques Appliquées, UMR 7641, Ecole polytechnique, France
[2]CDISE, Skolkovo Institute of Science and Technology, Moscow, Russian Federation
[3]Department of Statistics, University of Oxford
[4]Université Paris-Saclay, ENS Paris-Saclay, CNRS, Centre Borelli, F-91190 Gif-sur-Yvette, France
[5]CS Departement, HSE University, Russian Federation

A Taylor expansion shows that

$$\mathcal{L}(\theta, \phi) \approx \log p_\theta(x) - \frac{1}{2}\mathrm{var}_{q_\phi}\left[\frac{\hat{p}_\theta(x)}{p_\theta(x)}\right] ;$$

see e.g. [Mad+17; DS18] for formal results. Hence the ELBO becomes tighter as the variance of the estimator decreases.

A common method to obtain an unbiased estimate is built on importance sampling; *i.e.* $\hat{\mathrm{Z}}(=)n^{-1}\sum_{i=1}^n [p_\theta(x, z_i)/q_\phi(z_i|x)]$ for $z_i \overset{\text{i.i.d.}}{\sim} q_\phi(\cdot|x)$. In particular, combined with (9.1), we obtain the popular Importance Weighted Auto Encoder (IWAE) proposed by [BGS15]. However, it is expected that the relative variance of this importance-sampling based estimator typically increases with the dimension of the latent z. To circumvent this issue, we suggest in this paper to consider other estimates of the evidence which have shown great success in the Monte Carlo literature. In particular, Annealed Importance Sampling (AIS) [Nea01b; Wu+16], and its Sequential Importance Sampling (SIS) extensions [DDJ06b] define state-of-the-art estimators of the evidence. These algorithms rely on an extended target distribution for which an efficient importance distribution can be defined using non-homogeneous Markov kernels.

It has been suggested in various contributions that AIS could be useful to train VAE [SKW15a; Wu+16; Mad+17; WKN20b]. However, to the authors knowledge, no contribution discusses how an unbiased gradient of the resulting ELBO could be obtained. Indeed, the main difficulty in this computation arises from the MCMC transitions in AIS. As a result, when this estimator is used, alternatives to the ELBO (9.1) have often been considered, see e.g. [DF19].

Whereas AIS requires using MCMC transition kernels, SIS is more flexible and can exploit Markov transition kernels which are only approximately invariant w.r.t. to a given target distribution, e.g. unadjusted Langevin kernels. In this case, the construction of the augmented target distribution, which is at the core of the estimator, requires the careful selection of a class of auxiliary 'backward' Markov kernels. [SKW15a; RTB16; Maa+16; Goy+17; Hua+18b] propose to learn such auxiliary kernels parameterized with neural networks through the ELBO. However, as illustrated in our simulations, this comes at an increase in the overall computational cost.

Our contributions. The contributions of this paper are as follows:

(i) We show how to obtain an SIS-based ELBO relying on undadjusted Langevin dynamics which, contrary to [SKW15a; Goy+17; Hua+18b], does not require introducing and optimizing backward Markov kernels. In addition, an unbiased gradient estimate of the resulting ELBO which exploits the reparameterization trick is derived.

(ii) We propose an unbiased gradient estimate of the ELBO computed based on AIS. At the core of this estimate is a non-standard representation of Metropolis–Hastings type kernels which allows us to differentiate them. This is combined to a variance reduction technique for improved efficiency.

(iii) We apply these new methods to build novel Monte Carlo VAEs, and show their efficiency on real-world datasets.

All the theoretical results are detailed in the supplementary material.

9.2 Variational Inference via Sequential Importance Sampling

9.2.1 SIS estimator

The design of efficient proposal importance distributions has been proposed in [Cro98; Nea01b] starting from an annealing schedule, and was later extended by [DDJ06b]. Let $\{\gamma_k\}_{k=0}^K$ be a sequence of

unnormalized "bridge" densities satisfying $\gamma_0(z) = q_\phi(z|x)$, $\gamma_K(z) = p_\theta(x, z)$. Note that γ_K is not normalized, in contrast to γ_0. Here we set

$$\gamma_k(z) = q_\phi(z|x)^{1-\beta_k} p_\theta(x, z)^{\beta_k} \tag{9.2}$$

for an annealing schedule $0 = \beta_0 < \cdots < \beta_K = 1$, but alternative sequences of intermediate densities could be considered. At each iteration, SIS samples a non-homogeneous Markov chain use the transition kernels $\{M_k\}_{k=1}^K$. In this section, we assume that M_k admits a positive transition density m_k, such that m_k leaves γ_k invariant, *i.e.* $\int \gamma_k(z) m_k(z, z') \mathrm{d}z = \gamma_k(z')$, or is only approximately invariant. In particular, m_k typically depends on the data x. However, to simplify notation, this dependence is omitted.

Based on this sequence of transition densities, SIS considers the joint density on the path space $z_{0:K} \in \mathbb{R}^{d(K+1)}$

$$q_\phi^K(z_{0:K}|x) = q_\phi(z_0|x) \prod_{k=1}^K m_k(z_{k-1}, z_k) \ , \tag{9.3}$$

where $z_{i:j} = (z_i, \ldots, z_j)$ for $0 \leqslant i \leqslant j$. By construction, we expect the K-th marginal $q_\phi^K(z_K|x) = \int q_\phi^K(z_{0:K}|x) \mathrm{d}z_{0:K-1}$ to be close to $p_\theta(z_K|x)$. However, we cannot use importance sampling to correct for the discrepancy between these two densities, as $q_\phi^K(z_K|x)$ is typically intractable. To bypass this problem, we introduce another density on the path space

$$p_\theta^K(x, z_{0:K}) = p_\theta(x, z_K) \prod_{k=K-1}^0 \ell_k(z_{k+1}, z_k) \ , \tag{9.4}$$

where $\{\ell_k\}_{k=0}^{K-1}$ is a sequence of auxiliary positive transition densities. Note that in this case, the K-th marginal $\int p_\theta^K(x, z_{0:K}) \mathrm{d}z_{0:K-1}$ is exactly $p_\theta(x, z_K)$. Using now importance sampling on the path space, we obtain the following unbiased SIS estimator [DDJ06b; SKW15a] by sampling independently $z_{0:K}^i \sim q_\phi^K(\cdot|x)$ and computing

$$\hat{\mathrm{Z}}(=) \frac{1}{n} \sum_{i=1}^n w^K(z_{0:K}^i) \ , \ w^K(z_{0:K}) = \frac{p_\theta^K(x, z_{0:K})}{q_\phi^K(z_{0:K}|x)}. \tag{9.5}$$

9.2.2 AIS estimator

The selection of the kernel $\{\ell_k\}_{k=0}^{K-1}$ has a large impact on the variance of the estimator. The optimal reverse kernels ℓ_k minimizing the variance of $\hat{\mathrm{Z}}()$ are given by

$$\ell_{k-1}(z_k, z_{k-1}) = \frac{q_{\phi,k-1}(z_{k-1}|x) m_k(z_{k-1}, z_k)}{q_{\phi,k}(z_k|x)} \ , \tag{9.6}$$

where $q_{\phi,k}(z_k|x) = \int q_\phi^K(z_{0:K}|x) \mathrm{d}z_{0:K}^{-k}$ is the k-th marginal of q_ϕ^K; see [DDJ06b]. However, the resulting estimator $\hat{\mathrm{Z}}()$ is usually intractable. An approximation to (9.6) leading to a tractable estimator is provided by AIS [Cro98; Nea01b]. When m_k is γ_k-invariant, then, by assuming that $q_{\phi,k} \approx q_{\phi,k-1} \approx \gamma_k$ in (9.6), we obtain

$$\ell_{k-1}(z_k, z_{k-1}) = \frac{\gamma_k(z_{k-1}) m_k(z_{k-1}, z_k)}{\gamma_k(z_k)} \ . \tag{9.7}$$

We refer to this kernel as the *reversal* of m_k. In particular, if m_k is γ_k-reversible, *i.e.* $\gamma_k(z_{k-1}) m_k(z_{k-1}, z_k) = \gamma_k(z_k) m_k(z_k, z_{k-1})$, then $\ell_{k-1} = m_k$. Note that the weights (9.5) can be computed using the decomposition $w^K(z_{0:K}) = \prod_{k=1}^K w_k(z_{k-1}, z_k)$ with

$$w_k(z_{k-1}, z_k) = \frac{\gamma_k(z_k) \ell_{k-1}(z_k, z_{k-1})}{\gamma_{k-1}(z_{k-1}) m_k(z_{k-1}, z_k)} \ , \tag{9.8}$$

which simplifies when using (9.7) as

$$w_k(z_{k-1}, z_k) = \gamma_k(z_{k-1})/\gamma_{k-1}(z_{k-1}) \; . \tag{9.9}$$

In contrast to previous works, we will consider in the next section transition kernels m_k which are only approximately γ_k-invariant but still build ℓ_{k-1} in the spirit of (9.7).

9.2.3 SIS-ELBO using unadjusted Langevin

For any k, we consider the Langevin dynamics associated to γ_k and the corresponding Euler-Maruyama discretization. Then, we choose for m_k the transition density associated to this discretization

$$m_k(z_{k-1}, z_k) = \mathrm{N}(z_k; z_{k-1} + \eta \nabla \log \gamma_k(z_{k-1}), 2\eta \, \mathrm{Id}) \; . \tag{9.10}$$

Note that sampling q_ϕ^K boils down to sampling $Z_0 \sim q_\phi$ and defining the Markov chain $\{Z_k\}_{k \leqslant K}$ recursively by

$$Z_k = Z_{k-1} + \eta \nabla \log \gamma_k(Z_{k-1}) + \sqrt{2\eta} U_k \; , \tag{9.11}$$

where $U_k \sim \mathrm{N}(0, \mathrm{Id}_d)$. Moreover, as the continuous Langevin dynamics is reversible w.r.t. γ_k, this suggests that we can approximate the reversal ℓ_{k-1} of m_k by m_k directly as in AIS and thus select for any k, $\ell_{k-1} = m_k$ as done in [Hen+20]. However, in this case, the weights $w_k(z_{k-1}, z_k)$ do not simplify to $\gamma_k(z_{k-1})/\gamma_{k-1}(z_{k-1})$ as m_k is not exactly γ_k-invariant and we need to rely on the general expression given in (9.8). This approach was concurrently and independently proposed in [WKN20b] but they do not discuss gradient computations therein.

Based on (9.1) and (9.5), we introduce

$$\mathcal{L}_{\mathrm{SIS}} = \int \log \left(\prod_{k=1}^{K} w_k(z_{k-1}, z_k) \right) q_\phi^K(z_{0:K}|x) \mathrm{d}z_{0:K} \; . \tag{9.12}$$

We consider a reparameterization of (9.12) based on the Langevin mappings associated with target γ_k given by

$$\mathrm{T}_{k,u}(z_{k-1}) = z_{k-1} + \eta \nabla \log \gamma_k(z_{k-1}) + \sqrt{2\eta} u \; . \tag{9.13}$$

An easy change of variable based on the identity $Z_k = \mathrm{T}_{k,U_k}(Z_{k-1})$ in (9.11) shows that $\mathcal{L}_{\mathrm{SIS}}$ can be written as

$$\int \log \left(\prod_{k=1}^{K} w_k(z_{k-1}, z_k) \right) q_\phi(z_0|x) \varphi_{d,K}(u_{1:K}) \mathrm{d}z_0 \mathrm{d}u_{1:K},$$

where φ_d stands for the density of the d-dimensional standard Gaussian distribution, $\varphi_{d,K}(u_{1:K}) = \prod_{i=1}^K \varphi_d(u_i)$, and we write $z_k = \mathrm{T}_{k,u_k} \circ \cdots \circ \mathrm{T}_{1,u_1}(z_0) = \bigcirc_{i=1}^k \mathrm{T}_{i,u_i}(z_0)$. By (9.5) and as $\ell_{k-1} = m_k$, our objective thus finally writes

$$\begin{aligned} \mathcal{L}_{\mathrm{SIS}}(\theta, \phi; x) = \int & q_\phi(z_0|x) \varphi_{d,K}(u_{1:K}) \\ & \times \log \left(\frac{p_\theta(x, z_K) \prod_{k=1}^K m_k(z_k, z_{k-1})}{q_\phi(z_0|x) \prod_{k=1}^K m_k(z_{k-1}, z_k)} \right) \mathrm{d}z_0 \mathrm{d}u_{1:K} \; . \end{aligned} \tag{9.14}$$

This defines the reparameterizable Langevin Monte Carlo VAE (L-MCVAE). The algorithm to obtain an unbiased SIS estimate of $p_\theta(x)$ is described in Algorithm 13. This estimate is related to the one presented in [CDS18b], however this work builds on a deterministic dynamics which limits the expressivity of the induced variational approximation. In contrast, we rely here on a stochastic dynamics. While we limit ourselves here to undajusted (overdamped) Langevin dynamics, this could also be easily extended to unajusted underdamped Langevin dynamics [Mon20].

Algorithm 13 Langevin Monte Carlo VAE

Input: Number of steps K, initial distribution q_ϕ, unnormalized target distribution p_θ, step-size η, annealing schedule $\{\beta_k\}_{k=0}^K$.
Output: SIS estimator W of $\log p_\theta(x)$.

Draw $z_0 \sim q_\phi(\cdot|x)$;
Set $W = -\log q_\phi(z_0|x)$;
for $k = 1$ **to** K **do**
  Draw $u_k \sim \varphi_d$;
  Set $\gamma_k(\cdot) = \beta_k \log p_\theta(x, \cdot) + (1 - \beta_k) \log q_\phi(\cdot|x)$;
  Set $z_k = z_{k-1} + \eta \nabla \log \gamma_k(z_k) + \sqrt{2\eta} u_k$;
  Set $W = W + \log m_k(z_k, z_{k-1}) - \log m_k(z_{k-1}, z_k)$;
end for
Set $W = W + \log p_\theta(x, z_K)$;
Return W

9.3 Variational Inference via Annealed Importance Sampling

In Section 9.2.3, we derived the SIS estimate of the evidence computed using ULA kernels M_k, whose invariant distribution approximates γ_k. We can differentiate the resulting ELBO and exploit the reparametrization trick as these kernels admit a density m_k w.r.t. Lebesgue measure. When computing the AIS estimates, we need Markov kernels M_k that are γ_k-invariant. Such Markov kernels M_k most often rely on a Metropolis–Hastings (MH) step and therefore do not typically admit a transition density. While this does not invalidate the expression of the AIS estimate presented earlier, it complicates significantly the computation of an unbiased gradient estimate of the corresponding ELBO. In this section, we propose a way to compute an unbiased estimator of the ELBO gradient for MH Markov kernels. We use here elementary measure-theoretical notations which are recalled in Section 9.6

9.3.1 Differentiating Markov kernels

Markov kernel. Let $\mathcal{B}(\mathbb{R}^d)$ denote the Borelian σ-field associated to $\mathbb{R}^d$. A Markov kernel M is a function defined on $\mathbb{R}^d \times \mathcal{B}(\mathbb{R}^d)$, such that for any $z \in \mathbb{R}^d$, $M(z, \cdot)$ is a probability distribution on $\mathcal{B}(\mathbb{R}^d)$, *i.e.* $M(z, A)$ is the probability starting from z to hit the set $A \subset \mathbb{R}^d$. The simplest is "deterministic", in which case $Q(z, A) = \delta_{\mathrm{T}(z)}(A)$, where T is a measurable mapping on $\mathbb{R}^d$ and δ_y is the Dirac mass at y. Instead of a single mapping T, we can consider a family of "indexed" mappings $\left\{\mathrm{T}_u \,:\, u \in \mathbb{R}^{d_u}\right\}$. If g is a p.d.f on $\mathbb{R}^{d_U}$, we consider

$$M(z, A) = \int \mathbb{1}_A(\mathrm{T}_u(z))\mathrm{g}(u)\mathrm{d}u \; .$$

To sample $M(z, \cdot)$, we first sample $u \sim \mathrm{g}$ and then apply the mapping $\mathrm{T}_u(z)$. If $d = d_u$, we consider the function $G_z : \mathbb{R}^d \mapsto \mathbb{R}^d$ defined for all $z \in \mathbb{R}^d$ by

$$G_z : u \mapsto G_z(u) = \mathrm{T}_u(z) \tag{9.15}$$

If G_z is a diffeomorphism, then by applying the change of variables formula, we obtain

$$\begin{aligned} M(z, A) &= \textstyle\int \mathbb{1}_A\big(G_z(u)\big)\mathrm{g}(u)\mathrm{d}u \\ &= \textstyle\int \mathbb{1}_A(z')m(z, z')\mathrm{d}z' \; , \end{aligned} \tag{9.16}$$

where, denoting $\mathrm{J}_{G_z^{-1}}(z')$ is the absolute value of the Jacobian determinant of G_z^{-1} evaluated at z', we set

$$m(z, z') = \mathrm{J}_{G_z^{-1}}(z')\mathrm{g}\big(G_z^{-1}(z')\big) \; . \tag{9.17}$$

In this case, the Markov kernel has a transition density $m(z, z')$. This is the setting considered in the previous section.

Finally, some Markov kernels have both a deterministic and a continuous component. This is for example the case of Metropolis–Hastings (MH) kernels:

$$M(z, \mathsf{A}) = \int_{\mathsf{U}} Q_u(z, \mathsf{A}) \mathrm{g}(u) \mathrm{d}u \ , \text{ where} \tag{9.18}$$
$$Q_u\big(z, \mathsf{A}\big) = \alpha_u(z) \delta_{\mathrm{T}_u(z)}(\mathsf{A}) + \big\{1 - \alpha_u(z)\big\} \delta_z(\mathsf{A}) \ ,$$

with $\alpha_u(z) = \alpha\big(z, \mathrm{T}_u(z)\big)$ is the *acceptance function* and $\mathrm{T}_u \colon \mathbb{R}^d \to \mathbb{R}^d$ is the *proposal mapping*. In the sequel, we denote $\alpha_u^1(z) = \alpha_u(z)$ and $\alpha_u^0(z) = 1 - \alpha_u(z)$, and set $\mathrm{T}_u^0(z) = z$. With these notations, (9.18) can be rewritten in a more concise way

$$Q_u(z, \mathsf{A}) = \sum_{a=0}^{1} \alpha_u^a(z) \delta_{\mathrm{T}_u^a(z)}(\mathsf{A}) \ . \tag{9.19}$$

To sample $M(z, \cdot)$, we first draw $u \sim \mathrm{g}$ and then compute the proposal $y = \mathrm{T}_u(z)$. With probability $\alpha_u(z)$, the next state of the chain is set to $z' = y$, and $z' = z$ otherwise. If G_z defined in (9.15) is a diffeomorphism, then the Metropolis–Hasting kernel may be expressed as

$$M(z, A) = \int \alpha(z, z') m(z, z') \mathbb{1}_A(z') \mathrm{d}z' + \big(1 - \bar{\alpha}(z)\big) \delta_z(A),$$

where $\bar{\alpha}(z) = \int \alpha(z, z') m(z, z') \mathrm{d}z'$ is the mean acceptance probability at z (the probability of accepting a move) and $m(z, z')$ is defined as in (9.17). In MH-algorithms, the acceptance function $\alpha \colon \mathbb{R}^{2d} \to [0, 1]$ is chosen so that M is π-reversible $\pi(\mathrm{d}z) M(z, \mathrm{d}z') = \pi(\mathrm{d}z') M(z', \mathrm{d}z)$, where π is the target distribution. This implies, in particular, that M is π-invariant. Standard MH algorithms use $\alpha(z, z') = 1 \wedge \pi(z') m(z', z) / \pi(z) m(z, z')$; see [Tie94].

To illustrate these definitions and constructions, consider first the symmetric Random Walk Metropolis Algorithm (RWM). In this case, $d_U = d$ and $\mathrm{g} \leftarrow \varphi_d$, where φ_d is the d-dimensional standard Gaussian density. The proposal mapping is given by

$$G_z^{\mathrm{RWM}}(u) = \mathrm{T}_u^{\mathrm{RWM}}(z) = z + \Sigma^{1/2} u \ ,$$

where Σ is a positive definite matrix, and the acceptance function is given by $\alpha_u^{\mathrm{RWM}}(z) = 1 \wedge \big(\pi\big(T_u^{\mathrm{RWM}}(z)\big) / \pi(z)\big)$.

Consider now the Metropolis Adjusted Langevin Algorithm (MALA); see [Bes94a]. Assume that $z \mapsto \log \pi(z)$ is differentiable and denote by $\nabla \log \pi(z)$ its gradient. The Langevin proposal mapping $\mathrm{T}_u^{\mathrm{MALA}}$ is defined by

$$G_z^{\mathrm{MALA}}(u) = \mathrm{T}_u^{\mathrm{MALA}}(z) = z + \eta \nabla \log \pi(z) + \sqrt{2\eta} u \ . \tag{9.20}$$

We set $\mathrm{g} \leftarrow \varphi_d$ and $\alpha_u^{\mathrm{MALA}}(z)$ is the acceptance given by

$$\alpha_u^{\mathrm{MALA}}(z) = 1 \wedge \frac{\pi\big(\mathrm{T}_u^{\mathrm{MALA}}(z)\big) m_\eta\big(\mathrm{T}_u^{\mathrm{MALA}}(z), z\big)}{\pi(z) m_\eta\big(z, \mathrm{T}_u^{\mathrm{MALA}}(z)\big)} \ , \tag{9.21}$$

where $m_\eta(z, z') = \eta^{-1/2} \varphi_d\big(\eta^{-1/2} \{z' - \mathrm{T}_0^{\mathrm{MALA}}(z)\}\big)$, similarly to (9.10).

Lemma 71. *For all $z \in \mathbb{R}^d$, G_z^{MALA} is a C^1-diffeomorphism. Moreover, assume that $\log \pi$ is L-smooth with $\mathrm{L} > 0$, i.e. for $z, z' \in \mathbb{R}^d$, $\|\nabla \log \pi(z') - \nabla \log \pi(z)\| \leqslant \mathrm{L} \|z' - z\|$. Then, if $0 \leqslant \eta < L^{-1}$, for all $u \in \mathbb{R}^d$, $\mathrm{T}_u^{\mathrm{MALA}}$ defined in (9.20) is a C^1-diffeomorphism.*

The proof of Lemma 71 is postponed to Section 9.8.

9.3.2 Differentiable AIS-based ELBO

We now generalize the derivation of Section 9.2.1 to handle the case where M_k and its reversal do not admit transition densities. In this case, the proposal and unnormalized target distributions are defined by

$$\mathrm{Q}_\phi^K(\mathrm{d}z_{0:K}|x) = q_\phi(z_0|x)\mathrm{d}z_0 \prod_{k=1}^K M_k(z_{k-1}, \mathrm{d}z_k) \;, \tag{9.22}$$

$$\mathrm{P}_\theta^{\mathrm{un}}(x, \mathrm{d}z_{0:K}) = p_\theta(x, z_K)\mathrm{d}z_K \prod_{k=K}^1 L_{k-1}(z_k, \mathrm{d}z_{k-1}) \;,$$

where we define the reversal Markov kernel L_{k-1} by

$$\gamma_k(z_{k-1})\mathrm{d}z_{k-1}M_k(z_{k-1}, \mathrm{d}z_k) = \gamma_k(z_k)\mathrm{d}z_k L_{k-1}(z_k, \mathrm{d}z_{k-1}) \;.$$

We consider then the AIS estimator presented in Section 9.2.1 in (9.5) by sampling independently $z_{0:K}^i \sim \mathrm{Q}_\phi^K(\cdot|x)$ and with $w^K(z_{0:K}) = \prod_{k=1}^K w_k(z_{k-1})$, $w_k(z_{k-1}) = \gamma_k(z_{k-1})/\gamma_{k-1}(z_{k-1})$. A rigorous proof of the unbiasedness of the resulting estimator can be found in the Supplementary Material and is based on the formula

$$\mathrm{Z} = \int w^K(z_{0:K})\mathrm{Q}_\phi^K(\mathrm{d}z_{0:K}|x) \;. \tag{9.23}$$

In the sequel, we use Metropolis Adjusted Langevin Algorithm (MALA) kernels $\{M_k\}_{k=1}^K$ targeting γ_k for each $k \in \{1, \ldots, K\}$. By construction, the Markov kernel M_k is reversible w.r.t. γ_k and we set for the reversal kernel $L_{k-1} = M_k$. Note that we could easily generalize to other cases, especially inspired by recent works on non-reversible MCMC algorithms [Thi+20a].

For $k \in \{1, \ldots, K\}$, we use the representation of MALA kernel M_k outlined in (9.18) with proposal mapping $\mathrm{T}_{k,u}$ and acceptance function $\alpha_{k,u}$ defined as in (9.20) and (9.21) with $\pi \leftarrow \gamma_k$. We set

$$Q_{k,u}(z, \mathrm{d}z') = \sum_{a=0}^1 \alpha_{k,u}^a(z)\delta_{\mathrm{T}_{k,u}^a(z)}(\mathrm{d}z') \;. \tag{9.24}$$

By construction, the MALA kernel M_k (see (9.18)) writes $M_k(z, \mathrm{d}z') = \int Q_{k,u}(z, \mathrm{d}z')\varphi_d(u)\mathrm{d}u$. Plugging this representation into (9.22), we get

$$\mathrm{Q}_\phi^K(\mathrm{d}z_{0:K}|x) = \int \prod_{k=1}^K Q_{k,u_k}(z_{k-1}, \mathrm{d}z_k) q_\phi(z_0|x)\varphi_{d,K}(u_{1:K})\mathrm{d}z_0\mathrm{d}u_{1:K} \;,$$

writing $\varphi_{d,K}(u_{1:K}) = \prod_{i=1}^K \varphi_d(u_i)$. Eq. (9.24) suggests to consider the extended distribution on $(z_{0:K}, a_{1:K}, u_{1:K})$:

$$\mathrm{Q}_\phi^K(\mathrm{d}z_{0:K}, a_{1:K}, \mathrm{d}u_{1:K}|x) = q_\phi(z_0|x) \prod_{k=1}^K \alpha_{k,u_k}^{a_k}(z_{k-1}) \prod_{k=1}^K \delta_{\mathrm{T}_{k,u_k}^{a_k}(z_{k-1})}(\mathrm{d}z_k)\varphi_{d,K}(u_{1:K})\mathrm{d}z_0\mathrm{d}u_{1:K} \;,$$

which admits again as a marginal $\mathrm{Q}_\phi^K(\mathrm{d}z_{0:K}|x)$. Note that the variables $a_{1:K}$ correspond to the binary outcomes of the K A/R steps. By construction, $z_{1:K}$ are *deterministic functions* of z_0, $a_{1:K}$ and $u_{1:K}$: for each $k \in \{1, \ldots, K\}$, $z_k = \bigcirc_{i=1}^k \mathrm{T}_{i,u_i}^{a_i}(z_0)$.

Set $w^K(z_{0:K}) = \prod_{k=1}^K w_k(z_{k-1})$, and

$$A(z_0, a_{1:K}, u_{1:K}) = \prod_{k=1}^K \alpha_{k,u_k}^{a_k}\big(\bigcirc_{i=1}^{k-1} \mathrm{T}_{i,u_i}^{a_i}(z_0)\big) \;,$$

$$W(z_0, a_{1:K}, u_{1:K}) = \log \prod_{k=1}^K w_k\Big(\bigcirc_{i=1}^{k-1} \mathrm{T}_{i,u_i}^{a_i}(z_0)\Big) \;.$$

Given z_0 and $u_{1:K}$, $A(z_0, a_{1:K}, u_{1:K})$ is the conditional distribution of the A/R random variables $a_{1:K}$. It is easy to sample this distribution recursively: for each $k \in \{1, \ldots, K\}$ we sample a_k from a Bernoulli distribution of parameter $\alpha_{k,u_k}(z_{k-1})$ and we set $z_k = \mathrm{T}_{k,u_k}^{a_k}(z_{k-1})$. Eqs. (9.5) and (9.9) naturally lead us to consider the ELBO

$$\begin{aligned}\mathcal{L}_{\mathrm{AIS}} &= \int \sum_{a_{1:K}} \mathrm{Q}_\phi^K(\mathrm{d}z_{0:K}, a_{1:K}, \mathrm{d}u_{1:K}|x) \log\left[w^K(z_{0:K})\right] \\ &= \int \sum_{a_{1:K}} q_\phi(z_0|x) A(z_0, a_{1:K}, u_{1:K}) \\ &\qquad \times W(z_0, a_{1:K}, u_{1:K})\, \varphi_{d,K}(u_{1:K}) \mathrm{d}z_0 \mathrm{d}u_{1:K} \ ,\end{aligned}$$

where in the last line we have integrated w.r.t. $z_{1:K}$. The fact that $\mathcal{L}_{\mathrm{AIS}}$ is an ELBO stems immediately from (9.23), by applying Jensen's inequality. We can optimize this ELBO w.r.t. the different parameters at stake, even possibly the parameters of the proposal mappings.

We reparameterize the latent variable distribution $q_\phi(z_0|x)$ in terms of a known base distribution and a differentiable transformation (such as a location-scale transformation). Assuming for simplicity that $q_\phi(z_0|x)$ is a Gaussian distribution $\mathrm{N}(z_0; \mu_\phi(x), \sigma_\phi^2(x))$, then the location-scale transformation using the standard Normal as a base distribution allows us to reparameterize z_0 as $z_0 = \mu_\phi(x) + \sigma_\phi(x) \cdot u_0 = V_{\phi,x}(u_0)$, with $u_0 \sim \varphi_d$. Using this reparameterization trick, we can write $\nabla\mathcal{L}_{\mathrm{AIS}} = \nabla\mathcal{L}_{\mathrm{AIS}1} + \nabla\mathcal{L}_{\mathrm{AIS}2}$ with

$$\begin{aligned}\nabla\mathcal{L}_{\mathrm{AIS}1} &= \int \sum_{a_{1:K}} A(V_{\phi,x}(u_0), a_{1:K}, u_{1:K}) \\ &\times \nabla W(V_{\phi,x}(u_0), a_{1:K}, u_{1:K}) \varphi_{d,K+1}(u_{0:K}) \mathrm{d}u_{0:K} \ , \\ \nabla\mathcal{L}_{\mathrm{AIS}2} &= \int \sum_{a_{1:K}} A(V_{\phi,x}(u_0), a_{1:K}, u_{1:K}) W(V_{\phi,x}(u_0), a_{1:K}, u_{1:K}) \\ &\times \nabla \log A(V_{\phi,x}(u_0), a_{1:K}, u_{1:K}) \varphi_{d,K+1}(u_{0:K}) \mathrm{d}u_{0:K} \ . \end{aligned} \tag{9.25}$$

The estimation of $\nabla\mathcal{L}_{\mathrm{AIS}1}$ is straightforward. We sample n independent samples $u_{0:K}^{1:n} \sim \varphi_{d,K+1}$ and, for $i \in \{1, \ldots, n\}$, we set $z_0^i = V_{\phi,x}(u_0^i)$ and then, for $k \in \{1, \ldots, K\}$, we sample the A/R variable $a_k^i \sim \mathrm{Ber}\{\alpha_{k,u_k^i}^1(z_{k-1}^i)\}$ and set $z_k^i = \mathrm{T}_{k,u_k^i}^{a_k^i}(z_{k-1}^i)$, see Algorithm 14. Similarly, we then compute

$$\widehat{\nabla\mathcal{L}_{\mathrm{AIS}2,n}} = n^{-1} \sum_{i=1}^n \nabla W(V_{\phi,x}(u_0^i), a_{1:K}^i, u_{1:K}^i) \ .$$

The expression $\nabla\mathcal{L}_{\mathrm{AIS}2}$ is the REINFORCE gradient estimator [Wil92] for the A/R probabilities. Indeed, we have to compute the gradient of the conditional distribution of the A/R variables given $(z_0, u_{0:K})$, and there is no obvious reparametrization for such purpose (see however [MMT16] for a possible solution to the problem; this solution was not investigated in this work). To reduce the variance of the REINFORCE estimator, we rely on control variates, in the spirit of [MR16]. For $i \in \{1, \ldots, n\}$, we define

$$\tilde{W}_{n,i} = \frac{1}{n-1} \sum_{j \neq i} W(V_{\phi,x}(u_0^j), a_{1:K}^j, u_{1:K}^j) \ ,$$

which is independent of $W(V_{\phi,x}(u_0^i), a_{1:K}^i, u_{1:K}^i)$ and $\nabla \log A(V_{\phi,x}(u_0^i), a_{1:K}^i, u_{1:K}^i)$ by construction. This provides the new unbiased estimator of the gradient using

$$\widehat{\nabla\mathcal{L}_{\mathrm{AIS}2,n}} = n^{-1} \sum_{i=1}^n \left[W(V_{\phi,x}(u_0^i), a_{1:K}^i, u_{1:K}^i) - \tilde{W}_{n,i} \right] \nabla \log A(V_{\phi,x}(u_0^i), a_{1:K}^i, u_{1:K}^i) \ . \tag{9.26}$$

Algorithm 14 shows how to compute W and $\log A$.

Algorithm 14 Annealed Importance Sampling VAE

Input: Number of steps K, proposal mappings $\{\mathrm{T}_{k,u}\}_{k\leqslant K,u\in\mathsf{U}}$, acceptance functions $\{\alpha_{k,u}\}_{k\leqslant K,u\in\mathsf{U}}$, initial distribution q_ϕ, unnormalized target distribution p_θ, annealing schedule $\{\beta_k\}_{k=0}^K$.
Output: AIS estimator W of $\log p_\theta(x)$, sum $\log A$ of the A/R log probabilities.

Draw $z_0 \sim q_\phi(\cdot|x)$;
Set $W = 0$;
Set $\log A = 0$;
for $k = 1$ **to** K **do**
  Draw $u_k \sim \varphi_d$;
  Draw $a_k \sim \mathrm{Ber}(\alpha_{k,u_k}(z_{k-1}))$;
  if $a_k = 1$ (Accept) **then**
    Set $z_k = z_{k-1} + \eta\nabla \log \gamma_k(z_k) + \sqrt{2\eta}u_k$;
  else
    $z_k = z_{k-1}$;
  end if
  Compute $\log w_k(z_{k-1}) =$
    $(\beta_k - \beta_{k-1})\big(\log p_\theta(x, z_{k-1}) - \log q_\phi(z_{k-1}|x)\big)$;
  Set $W = W + \log w_k(z_{k-1})$;
  Set $\log A = \log A + \log \alpha_{k,u_k}^{a_k}(z_{k-1})$;
end for
Return $W, \log A$

9.4 Experiments

9.4.1 Methods and practical guidelines

In what follows, we consider two sets of experiments[1]. In the first one, we aim at illustrating the expressivity and the efficiency of our estimator for VI. In the second, we tackle the problem of learning VAE: (a) Classical VAE based on mean-field approximation [KW13a]; (b) Importance-weighted Autoencoder (IWAE, [BGS15]); (c) L-MCVAE given by Algorithm 13; (d) A-MCVAE given by Algorithm 14. We provide in the following some guidelines on how to tune the step sizes and the annealing schedules in Algorithm 13 and Algorithm 14.

A crucial hyperparameter of our method is the step size η. In principle, it could be learned by including it as an additional inference parameter ϕ and by maximizing the ELBO. However, it is then difficult to find a good trade-off between having a high A/R ratio and a large step size η at the same time. Instead, we suggest adjusting η by targeting a fixed A/R ratio ρ. It has proven effective to use a preconditioned version of (9.11), *i.e.* $Z_k = Z_{k-1} + \eta \odot \nabla \log \gamma_k(Z_{k-1}) + \sqrt{2\eta} \odot U_k$ with $\eta \in \mathbb{R}^p$, where we adapt each component of η using the following rule

$$\eta^{(i)} = 0.9\eta^{(i)} + 0.1\eta_0 / \big(\epsilon + \mathrm{std}[\partial_{z^{(i)}} \log p_\theta(x, z)]\big) \; . \tag{9.27}$$

Here std denotes the standard deviation over the batch x of the quantity $\partial_{z^{(i)}} \log p_\theta(x, z)$, and $\epsilon > 0$. The scalar η_0 is a tuning parameter which is adjusted to target the A/R ratio ρ. This strategy follows the same heuristics as Adam [KB14]. In the following ρ is set to 0.8 for A-MCVAE and 0.9 for L-MCVAE (keeping it high for L-MCVAE ensures that the Langevin dynamics stays "almost reversible", thus keeping a low variance SIS estimator).

An optimal choice of the temperature schedule $\{\beta_k\}_{k=0}^K$ for SIS and AIS is a difficult problem. We have focused in our experiments on three different settings. First, we consider the temperature schedule fixed and regularly spaced between 0 and 1. Following [GGA15], the second option is the sigmoidal tempering scheme where $\beta_k = (\tilde{\beta}_k - \tilde{\beta}_1)/(\tilde{\beta}_K - \tilde{\beta}_1)$ with, $\tilde{\beta}_k = \sigma\big(\delta(2k/K - 1)\big)$, σ is the sigmoid function

[1] The code to reproduce all of the experiments is available online at `https://github.com/stat-ml/mcvae/`.

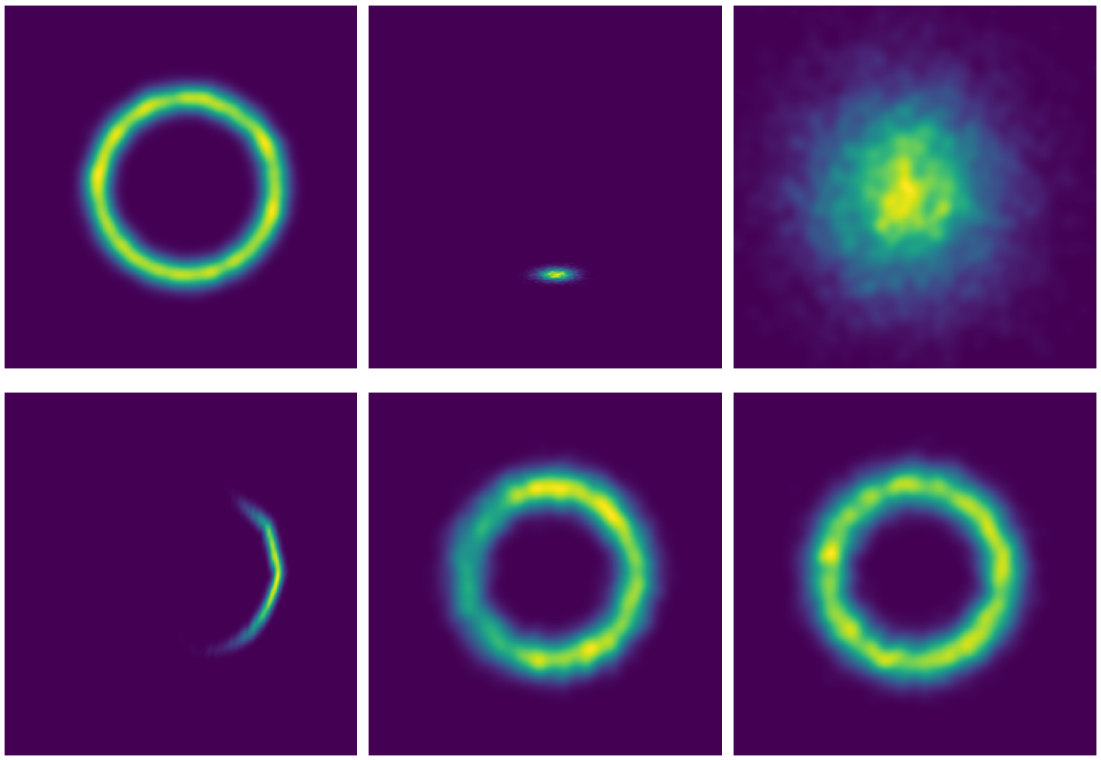


Figure 9.1: Visualization of the posterior approximation given after optimization of different bounds for toy generation process. Top row, from left to right: True posterior, VAE posterior, IWAE posterior. Bottom row, from left to right: VI with RealNVP posterior, A-MCVAE posterior, L-MCVAE posterior.

and $\delta > 0$ is a parameter that we optimize during the training phase. The last schedule consists in learning the temperatures $\{\beta_k\}_{k=0}^K$ directly as additional inference parameters ϕ.

9.4.2 Toy 2D example and Probabilistic Principal Component Analysis

In the following two examples, we fix the parameters θ of the likelihood model and apply Algorithm 13 and Algorithm 14 to perform VI to sample from $z \mapsto p_\theta(z|x)$. Consider first a toy hierarchical example where we generate some i.i.d. data $x = (x_i)_{i=1}^N \in \mathbb{R}^N$ from the i.i.d. latent variables $z = (z_i)_{i=1}^N \in \mathbb{R}^{2N}$ as follows for $\xi > 0$ $x_i|z_i \sim \mathrm{N}(\xi \cdot (\|z_i\|^2 + \zeta), \sigma^2) = p_\theta(x_i|z_i)$. We consider the variational approximation as $q_\phi(z|x) = \mathrm{N}\big(z; \mu_\phi(x), \sigma_\phi(x)^2 \,\mathrm{Id}\big)$, where $\mu_\phi(x), \sigma_\phi(x) \in \mathbb{R}^d$ are the outputs of a fully connected neural network with parameters ϕ. We compare these algorithms to VI using Real-valued Non-Volume Preserving transformation (RealNVP, [DSB16]).

Figure 9.1 displays the VI posterior approximations corresponding to the different schemes for a given observation x. It can be observed that MCVAE benefit from flexible variational distributions compared to other classical schemes, which mostly fail to recover the true posterior. Additional results on the estimation of the parameters ξ, ζ, given in the supplementary material, further support our claims; see Section 9.7.1.

We now illustrate the performance of MCVAE on a probabilistic principal component analysis problem applied to MNIST [SM08b], as we can access in this case the exact likelihood and its gradient. We follow the set-up of [Rui+21], Section 6.1. We consider here a batch of size $N = 100$ for the model $p_\theta(x, z) = \mathrm{N}(z; 0, \mathrm{Id}_d)\mathrm{N}(x; \theta_0 + \theta_1 z, \sigma^2 \,\mathrm{Id}_p)$, with $d = 100$ and $p = 784$. We fix arbitrarily θ_0, θ_1, and fit an amortized variational distribution $q_\phi(z|x)$ by maximizing the IWAE bound w.r.t. ϕ with $K = 100$ importance samples for a large number of epochs. The distribution $q_\phi(z|x) = \mathrm{N}(z; \mu_\phi(x), \mathrm{diag}(\sigma_\phi^2(x)))$ is a mean-field Gaussian distribution where μ_ϕ, σ_ϕ are linear functions of the observation x.

We compare the Langevin SIS estimator (L-MCVAE) of the log evidence $\log p_\theta(x)$ with Langevin

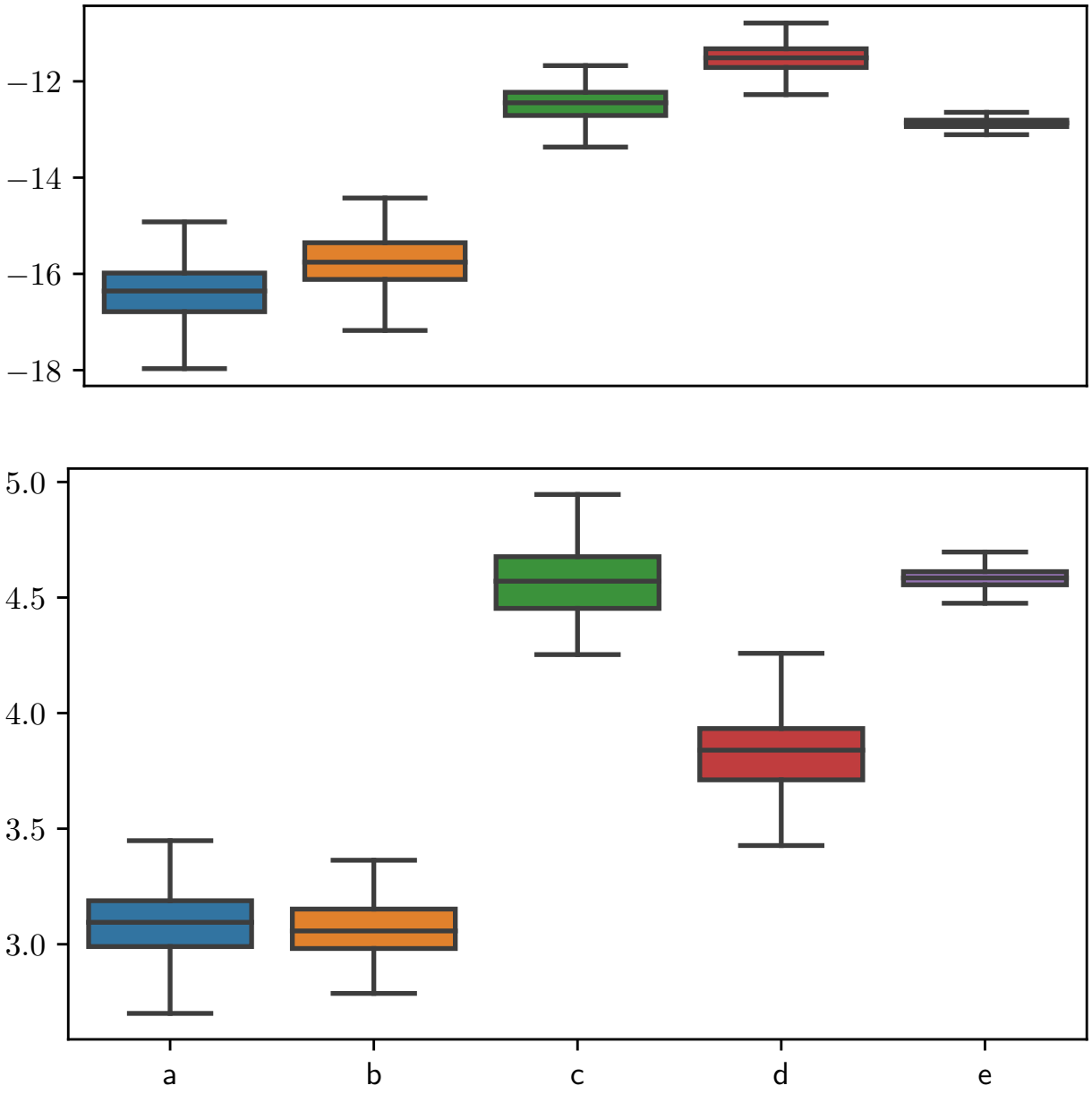


Figure 9.2: Representation of the different estimators (top) and their gradient (bottom) of the true log likelihood. From left to right, a/ L-MCVAE, $K = 5$, b/ L-MCVAE, $K = 10$, c/ A-MCVAE, $K = 5$, d/ A-MCVAE, $K = 10$, e/ A-MCVAE, $K = 5$ with control variates.

auxiliary kernels as described in Section 9.2.3, and the Langevin AIS estimate (A-MCVAE). Moreover, we also compare the gradients of these quantities w.r.t. the parameters θ_0, θ_1.

Figure 9.2 summarises the results with boxplots computed over 200 independent samples of each estimator. The quantity reported on the first boxplot corresponds to the Monte Carlo samples of $\log \hat{p}_\theta(x) - \log p_\theta(x)$. One the one hand, we note that the SIS estimator has larger variance than AIS, and that the latter achieves a better ELBO. Moreover, in both cases, increasing the number of steps K tightens the bound. On the other hand, the estimator of the gradient of AIS is noisier than that of SIS, even though variance reduction techniques allows us to recover a similar variance. We also present in the supplementary material the Langevin SIS estimator using auxiliary backward kernels learnt with neural networks (as done in previous contributions); see Section 9.7.2. The auxiliary neural

backward kernels are set as $l(z, z') = \mathrm{N}(z'; \mu_\psi(z), \mathrm{diag}(\sigma^2_\psi(z)))$, $\mu_\psi, \sigma_\psi \in \mathbb{R}^d$, where the parameters ψ are learnt through the SIS ELBO, similarly to [Hua+18b]. The variance of the associated estimator and their gradients are larger than that of SIS using the approximate reversals as backward kernels; i.e. $\ell_{k-1} = m_k$.

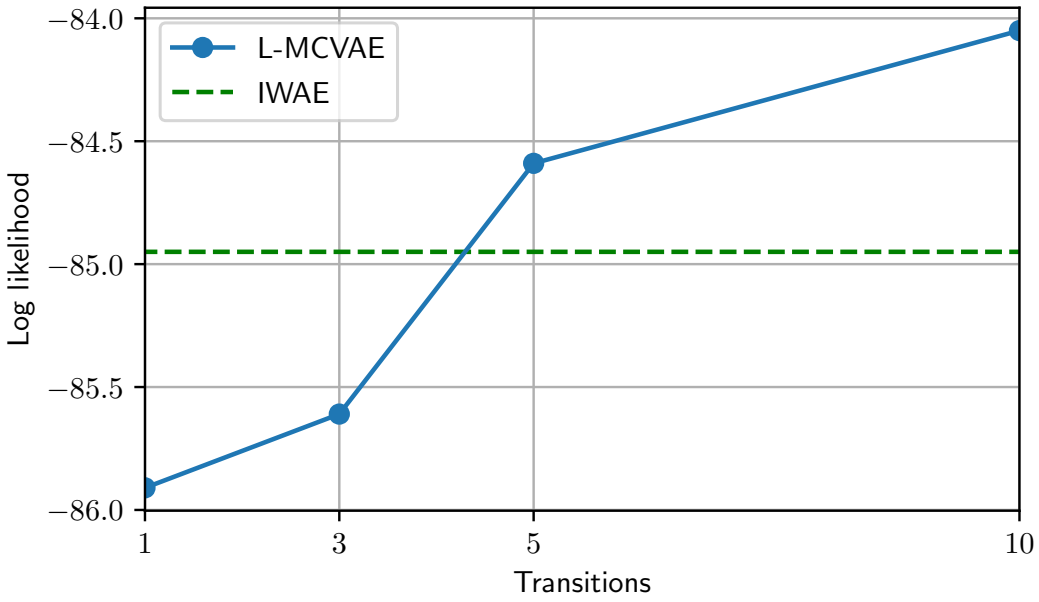


Figure 9.3: Log-likelihood of L-MCVAE depending on the number of Langevin steps K. Increasing K improves performance, however at the expense of the computational complexity.

9.4.3 Numerical results for image datasets

Table 9.1: Results of the different models on MNIST. A more detailed version of this table is included in the supplementary material.

	negative ELBO estimate			*NLL estimate*		
n. of epochs	*10*	*30*	*100*	*10*	*30*	*100*
VAE	95.26 ± 0.49	91.58 ± 0.27	89.70 ± 0.19	89.83 ± 0.59	86.86 ± 0.26	85.22 ± 0.07
IWAE, $K = 10$	91.42 ± 0.21	88.56 ± 0.07	87.16 ± 0.19	88.54 ± 0.27	86.07 ± 0.1	84.82 ± 0.1
IWAE, $K = 50$	90.34 ± 0.27	87.5 ± 0.16	86.05 ± 0.11	89.4 ± 0.25	86.54 ± 0.15	85.05 ± 0.1
L-MCVAE, $K = 5$	96.62 ± 3.24	88.58 ± 0.75	87.51 ± 0.41	90.59 ± 2.01	85.68 ± 0.49	84.92 ± 0.24
L-MCVAE, $K = 10$	96.78 ± 1.06	87.99 ± 0.71	86.8 ± 0.66	91.33 ± 0.61	85.47 ± 0.46	**84.58** ± 0.39
A-MCVAE, $K = 3$	96.21 ± 3.43	88.64 ± 0.78	87.63 ± 0.42	90.42 ± 2.34	85.77 ± 0.65	85.02 ± 0.37
A-MCVAE, $K = 5$	95.55 ± 2.96	87.99 ± 0.57	87.03 ± 0.27	90.39 ± 2.21	85.6 ± 0.67	**84.84** ± 0.38
VAE with RealNVP	95.23 ± 0.33	91.69 ± 0.15	89.62 ± 0.17	89.98 ± 0.24	86.88 ± 0.05	85.23 ± 0.18

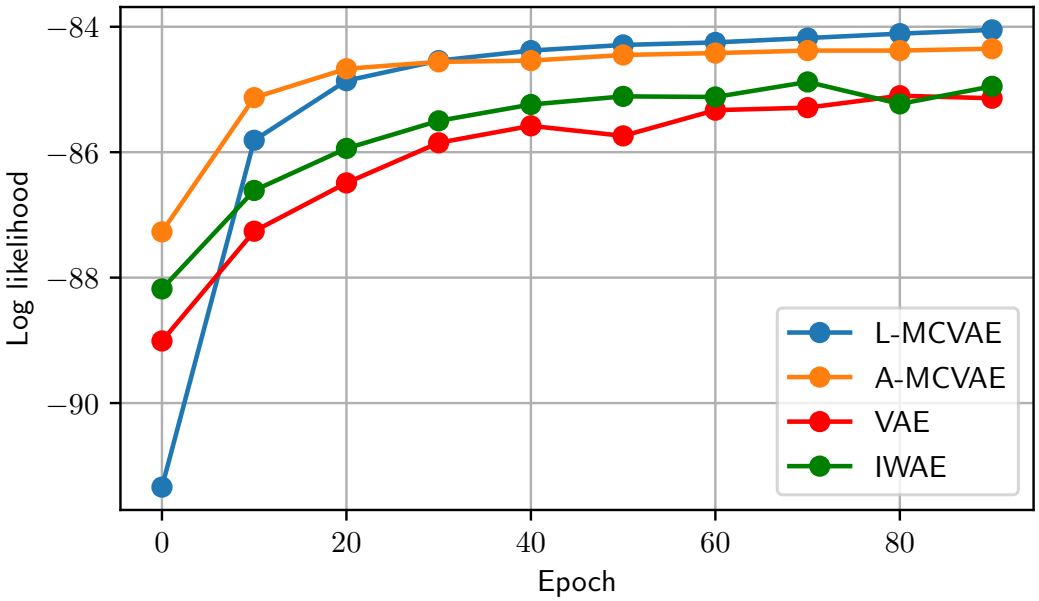


Figure 9.4: Evolution of the held-out loglikelihood during training for A-MCVAE, L-MCVAE, IWAE and VAE on MNIST.

Table 9.2: Results of the different models on CelebA. A more detailed version of this table is included in the supplementary material. 11400 must be added to all scores in this table.

	negative ELBO - 11400+			*NLL* - 11400+		
n. of epochs	*10*	*30*	*100*	*10*	*30*	*100*
VAE	23.78 ± 1.95	17.99 ± 0.4	14.72 ± 0.16	17.35 ± 1.7	12.68 ± 0.62	10.11 ± 0.32
IWAE, $K = 10$	20.59 ± 0.71	15.45 ± 0.52	12.2 ± 0.3	18.25 ± 0.6	13.18 ± 0.42	10.14 ± 0.31
IWAE, $K = 50$	19.05 ± 0.39	13.59 ± 0.5	10.48 ± 0.89	19.08 ± 0.42	13.17 ± 0.54	10.12 ± 0.86
L-MCVAE, $K = 5$	21.61 ± 1.48	12.72 ± 0.43	11.6 ± 0.37	16.42 ± 1.47	9.62 ± 0.47	8.72 ± 0.4
L-MCVAE, $K = 10$	20.7 ± 1.15	11.81 ± 0.34	10.6 ± 0.23	17.0 ± 1.87	9.29 ± 0.73	**8.24** ± 0.52
A-MCVAE, $K = 3$	21.59 ± 1.5	13.94 ± 0.42	12.84 ± 0.3	16.64 ± 1.37	10.98 ± 0.48	9.95 ± 0.3
A-MCVAE, $K = 5$	20.95 ± 1.18	12.42 ± 0.42	11.13 ± 0.37	17.42 ± 1.49	9.97 ± 0.59	**8.82** ± 0.57
VAE with RealNVP	15.12 ± 0.48	13.63 ± 0.27	12.58 ± 0.61	10.42 ± 0.33	9.04 ± 0.26	8.98 ± 0.2

Following [Wu+16], we propose to evaluate our models using AIS (not to be confused with the proposed AIS-based VI approach) to get an estimation of the negative log-likelihood. The base distribution is the distribution output by the encoder, and we perform K steps of annealing to compute the estimator of the likelihood, as given by (9.5). In practice, we use $K = 5$ HMC steps with 3 leapfrogs for evaluating our models.

We evaluate our models on three different datasets: MNIST, CIFAR-10 and CelebA. All the models we compare share the same architecture: the inference network q_ϕ is given by a convolutional network with 8 convolutional layers and one linear layer, which outputs the parameters $\mu_\phi(x), \sigma_\phi(x) \in \mathbb{R}^d$ of a

Table 9.3: Results of the different models on CIFAR. A more detailed version of this table is included in the supplementary material. 2800 must be added to all scores in this table.

	negative ELBO - 2800+			*NLL* - 2800+		
n. of epochs	*10*	*30*	*100*	*10*	*30*	*100*
VAE	69.57 ± 0.08	69.55 ± 0.51	68.84 ± 0.06	68.51 ± 0.07	68.41 ± 0.33	67.9 ± 0.03
IWAE, $K = 10$	69.82 ± 0.03	69.35 ± 0.03	69.36 ± 0.36	68.56 ± 0.03	68.0 ± 0.03	68.02 ± 0.4
IWAE, $K = 50$	69.94 ± 0.08	69.55 ± 0.04	69.43 ± 0.03	69.15 ± 0.15	68.37 ± 0.18	67.93 ± 0.02
L-MCVAE, $K = 5$	70.62 ± 0.41	68.55 ± 0.18	68.09 ± 0.1	69.15 ± 0.38	67.73 ± 0.07	**67.5** ± 0.07
L-MCVAE, $K = 10$	70.99 ± 0.59	68.36 ± 0.04	68.03 ± 0.0	69.8 ± 0.67	67.76 ± 0.04	**67.51** ± 0.03
A-MCVAE, $K = 3$	69.97 ± 0.99	68.48 ± 0.29	68.18 ± 0.16	69.26 ± 0.76	67.77 ± 0.18	67.55 ± 0.1
A-MCVAE, $K = 5$	70.1 ± 0.89	68.28 ± 0.2	68.01 ± 0.08	69.23 ± 0.75	67.71 ± 0.15	**67.5** ± 0.07
VAE with RealNVP	70.01 ± 0.12	69.51 ± 0.07	69.19 ± 0.13	68.73 ± 0.05	68.35 ± 0.05	68.05 ± 0.02

factorized Gaussian distribution, while the generative model $p_\theta(\cdot|z)$ is given by another convolutional network π_θ, where we use nearest neighbor upsamplings. This outputs the parameters for the factorized Bernoulli distribution (for MNIST dataset), that is

$$p_\theta(x|z) = \prod_{i=1}^{N} \mathrm{Ber}\Big(x^{(i)}|\big(\pi_\theta(z)\big)^{(i)}\Big)$$

and similarly the mean of the Gaussian distributions for colored datasets (CIFAR-10, Celeba). We compare A-MCVAE, L-MCVAE, IWAE, and VAE with different settings. All the models are implemented using PyTorch [Pas+19] and optimized using the Adam optimizer [KB14] for 100 epochs each. The training process is using PyTorch Lightning toolkit [Fal19].

First, consider dynamically binarized MNIST dataset [SM08b]. In this case, the latent dimension is set to $d = 64$. We present in Table 9.1 the results of the different models at different stages of the optimization. Moreover, we show on Figure 9.3 the performance of L-MCVAE for different values of K compared to IWAE baseline. In particular, we see that increasing K increases the performance of our VAE, however at the expense of an increase in computational cost. We also display on Figure 9.4 the evolution of the held-out loglikelihood for various objectives during training. Adding Langevin transitions appears to help convergence of the models.

Second, we compare similarly the different models on CelebA and CIFAR, see Table 9.2 and Table 9.3. In this case, the latent dimension is chosen to be $d = 128$. Increasing the number of MCMC steps seems again to improve both the ELBO and the final loglikelihood estimate. In each case, all models are run with 5 different seeds to compute the presented empirical standard deviation.

9.5 Discussion

We have shown in this article how one can leverage state-of-the-art Monte Carlo estimators of the evidence to develop novel competitive VAEs by developing novel gradient estimates of the corresponding ELBOs.

For a given computational complexity, AIS based on MALA provides ELBO estimates which are typically tighter than SIS estimates based on ULA. However, the variance of the gradient estimates of

the AIS-based ELBO (A-MCVAE) is also significantly larger than for the SIS-based ELBO (L-MCVAE) as it has to rely on REINFORCE gradient estimates. While control variates can be considered to reduce the variance, this comes at a significant increase in computational cost.

Empirically, L-MCVAE should thus be favoured as it provides both a tighter ELBO than standard techniques and low variance gradient estimates.

Acknowledgements

The work was partly supported by ANR-19-CHIA-0002-01 "SCAI" and EPSRC CoSInES grant EP/R034710/1. It was partly carried out under the framework of HSE University Basic Research Program. The development of a software system for the experimental study of VAEs and its application to computer vision problems (Section 4) was supported by the Russian Science Foundation grant 20-71-10135. Part of this research has been carried out under the auspice of the Lagrange Center for Mathematics and Computing.

Supplementary material

9.6 Notations and definitions

Let $(\mathbb{X}, \mathcal{X})$ be a measurable space. A *Markov kernel* N on $\mathbb{X} \times \mathcal{X}$ is a mapping $N : \mathbb{X} \times \mathcal{X} \to [0, 1]$ satisfying the following conditions:

(i) for every $x \in \mathbb{X}$, the mapping $N(x, \cdot) : A \mapsto N(x, A)$ is a probability of on $\mathcal{X}$,

(ii) for every $A \in \mathcal{X}$, the mapping $N(\cdot, A) : x \mapsto N(x, A)$ is a measurable function from $(\mathbb{X}, \mathcal{X})$ to $([0, 1], \mathcal{B}([0, 1]))$, where $\mathcal{B}([0, 1])$ denotes the borelian sets of $[0, 1]$.

Let λ be a positive σ-finite measure on $(\mathbb{X}, \mathcal{X})$ and $n : \mathbb{X} \times \mathbb{X} \to \mathbb{R}_+$ be a nonnegative function, measurable with respect to the product σ-field $\mathcal{X} \otimes \mathcal{X}$. Then, the application N defined on $\mathbb{X} \times \mathcal{X}$ by

$$N(x, A) \ = \int_A n(x, y)\lambda(\mathrm{d}y) \ ,$$

is a kernel. The function n is called the density of the kernel N w.r.t. the measure λ. The kernel N is Markovian if and only if $\int_{\mathbb{X}} n(x, y)\lambda(\mathrm{d}y) = 1$ for all $x \in \mathbb{X}$.

Let N be a kernel on $\mathbb{X} \times \mathcal{X}$ and f be a nonnegative function. A function $Nf : \mathbb{X} \to \mathbb{R}_+$ is defined by setting, for $x \in \mathbb{X}$,

$$Nf(x) = \int_{\mathbb{X}} N(x, \mathrm{d}y) f(y) \ .$$

Let μ be a probability on $(\mathbb{X}, \mathcal{X})$. For $A \in \mathcal{X}$, define

$$\mu N(A) = \int_{\mathbb{X}} \mu(\mathrm{d}x) \ N(x, \ A) \ .$$

If N is Markovian, then μN is a probability on $(\mathbb{X}, \mathcal{X})$.

9.7 Experiences

9.7.1 Toy example

We first describe additional experiments on the toy dataset introduced in Section 9.4.2.

Recall that we generate some i.i.d. data $x = (x_i)_{i=1}^N \in \mathbb{R}^N$ from the i.i.d. latent variables $z = (z_i)_{i=1}^N \in \mathbb{R}^{2N}$ as follows for $\eta > 0$: $z_i \sim \mathrm{N}(0; \mathrm{Id})$ and $x_i \mid z_i \sim \mathrm{N}(\eta \cdot (\|z_i\| + \zeta), \sigma^2) = p_\theta(x_i \mid z_i)$.

This example, presented for $z \in \mathbb{R}^2$, easily extends to the case where z lies in $\mathbb{R}^d$, with d increasing from 2 to 300. We tackle here the problem at estimating the parameter $\theta = (\eta, \zeta)$ when d varies.

We show in Figure S5 the error $\|\hat{\theta} - \theta\|^2$ for the different methods. The increased flexibility of the posterior proves more effective for estimating the true parameters of the generative model.

9.7.2 Probabilistic Principal Component Analysis

We detail the impact of the learnable reverse kernels on the variance of the estimator and looseness of the ELBO. In our experiments, reverse kernels were given by fully-connected neural networks. We train K different reverse kernels $\{l_k\}_{k=0}^{K-1}$ for the K transitions, each given by a separate neural network, and amortized over the observation x, similarly to [SKW15a; Hua+18b]. Given the parameters (θ, ϕ), we train these kernels for a large number of epochs using the SIS objective (9.14) and the Adam optimizer [KB14]. In particular, we display in Figure S6 the different estimators to be compared. It is easily seen that reverse kernels can not provide reasonable and stable density estimates. At the same time, we observe the variance of the gradient is higher in those models than in the ones we present in the main text. This motivates our approach bypassing the optimization of the reverse kernels.

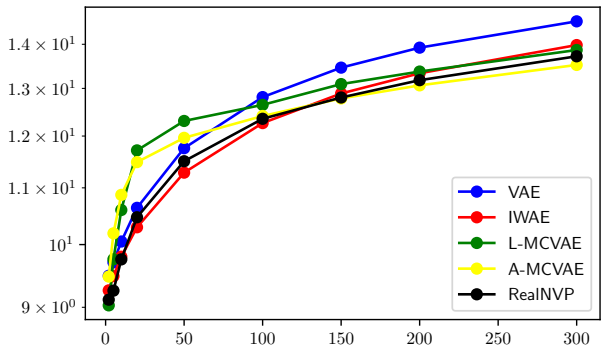


Figure S5: Squared error for parameter's estimates, obtained using different models.

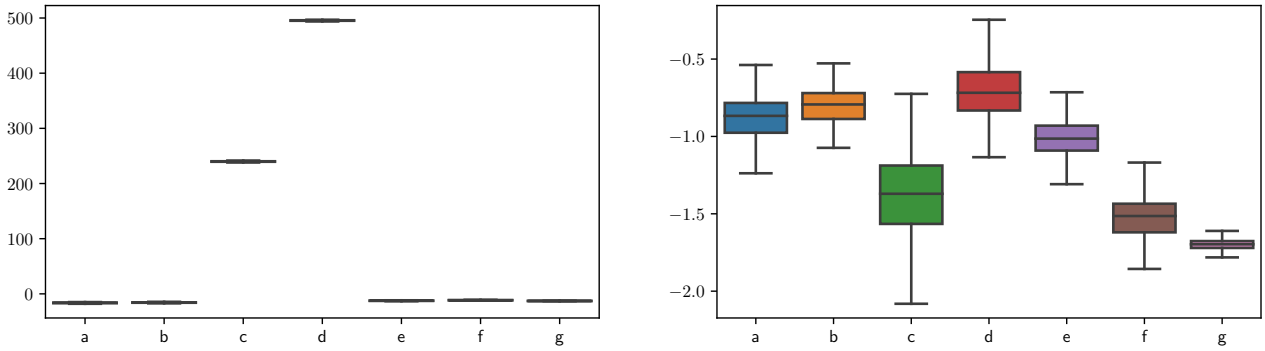


Figure S6: Representation of the different estimators (left) and their gradient (right) of the true log likelihood. From left to right, a/ L-MCVAE, $K = 5$, b/ L-MCVAE, $K = 10$, c/ L-MCVAE, $K = 1$, learnable reverse, d/ L-MCVAE, $K = 2$ learnable reverse, e/ A-MCVAE, $K = 5$, f/ A-MCVAE, $K = 10$, g/ A-MCVAE, $K = 5$ with control variates.

9.7.3 Additional experimental results

We display in this section the full results on MNIST, CelebA and CIFAR respectively of the different models as well as the effect of the different annealing schemes (respectively in Table 9.4, Table 9.5 and 9.6).

9.8 Proofs

9.8.1 Proof of SIS and AIS Identities

Proposition S72. *Let $\{\Gamma_k\}_{k=0}^K$ be a sequence of distributions on $(\mathbb{R}^d, \mathcal{B}(\mathbb{R}^d))$, $\{M_k\}_{k=1}^K$ and $\{L_k\}_{k=0}^{K-1}$ be Markov kernels. Assume that for each $k \in \{0, \ldots, K-1\}$, there exists a positive measurable function $w_k : \mathbb{R}^d \times \mathbb{R}^d \mapsto \mathbb{R}_+$ such that*

$$\Gamma_k(\mathrm{d}z_k) L_{k-1}(z_k, \mathrm{d}z_{k-1}) = \Gamma_{k-1}(\mathrm{d}z_{k-1}) M_k(z_{k-1}, \mathrm{d}z_k) w_k(z_{k-1}, z_k) \,. \tag{S28}$$

Then,

$$\Gamma_0(\mathrm{d}z_0) \prod_{k=1}^{K} M_k(z_{k-1}, \mathrm{d}z_k) \prod_{k=1}^{K} w_k(z_{k-1}, z_k) = \Gamma_K(\mathrm{d}z_K) \prod_{k=K}^{1} L_{k-1}(z_k, \mathrm{d}z_{k-1}) \,. \tag{S29}$$

Proof. We prove by induction that for $k \in \{1, \ldots, K\}$,

$$\Gamma_0(\mathrm{d}z_0) \prod_{i=1}^{k} M_i(z_{i-1}, \mathrm{d}z_i) \prod_{i=1}^{k} w_i(z_{i-1}, z_i) = \Gamma_k(\mathrm{d}z_k) \prod_{i=k}^{1} L_{i-1}(z_i, \mathrm{d}z_{i-1}) \,. \tag{S30}$$

Table 9.4: Results of the different models on MNIST with different annealing schemes.

number of epoches	*ELBO:* 10	30	100	*NLL:* 10	30	100
VAE	95.26 ± 0.5	91.58 ± 0.27	89.7 ± 0.19	89.83 ± 0.59	86.86 ± 0.26	85.22 ± 0.07
IWAE, $K = 10$	91.42 ± 0.21	88.56 ± 0.07	87.17 ± 0.19	88.54 ± 0.27	86.07 ± 0.1	84.82 ± 0.1
IWAE, $K = 50$	90.34 ± 0.27	87.5 ± 0.16	86.05 ± 0.11	89.4 ± 0.25	86.54 ± 0.15	85.05 ± 0.1
L-MCVAE Fixed, $K = 5$	96.6 ± 3.51	88.8 ± 0.46	87.77 ± 0.12	90.63 ± 2.19	85.85 ± 0.27	85.07 ± 0.04
L-MCVAE Sigmoidal, $K = 5$	95.48 ± 2.29	88.87 ± 0.82	87.81 ± 0.53	90.05 ± 1.63	85.92 ± 0.62	85.16 ± 0.38
L-MCVAE All learnable, $K = 5$	96.62 ± 3.24	88.58 ± 0.75	87.51 ± 0.41	90.59 ± 2.01	85.68 ± 0.49	84.92 ± 0.24
L-MCVAE Fixed, $K = 10$	95.98 ± 3.91	88.36 ± 0.7	87.38 ± 0.35	90.5 ± 2.23	85.75 ± 0.33	85.0 ± 0.11
L-MCVAE Sigmoidal, $K = 10$	96.78 ± 0.47	88.35 ± 0.63	87.17 ± 0.52	91.13 ± 0.27	85.72 ± 0.31	84.84 ± 0.26
L-MCVAE All learnable, $K = 10$	96.78 ± 1.06	87.99 ± 0.71	86.8 ± 0.66	91.33 ± 0.61	85.47 ± 0.46	84.58 ± 0.39
A-MCVAE Fixed, $K = 3$	96.21 ± 3.43	88.64 ± 0.78	87.63 ± 0.42	90.42 ± 2.34	85.77 ± 0.65	85.02 ± 0.37
A-MCVAE Sigmoidal, $K = 3$	96.59 ± 2.31	88.96 ± 0.4	87.86 ± 0.06	90.85 ± 1.62	85.97 ± 0.34	85.17 ± 0.1
A-MCVAE All learnable, $K = 3$	95.44 ± 2.68	88.79 ± 0.63	87.78 ± 0.37	89.9 ± 1.68	85.96 ± 0.59	85.23 ± 0.41
A-MCVAE Fixed, $K = 5$	95.55 ± 2.96	87.99 ± 0.57	87.03 ± 0.27	90.39 ± 2.21	85.6 ± 0.67	84.84 ± 0.38
A-MCVAE Sigmoidal, $K = 5$	96.56 ± 2.02	88.51 ± 0.31	87.46 ± 0.48	91.62 ± 1.55	85.96 ± 0.06	85.15 ± 0.21
A-MCVAE All learnable, $K = 5$	95.81 ± 1.72	88.11 ± 0.13	87.14 ± 0.18	90.79 ± 1.14	85.71 ± 0.28	84.95 ± 0.04
VAE with RealNVP	95.23 ± 0.33	91.69 ± 0.15	89.62 ± 0.17	89.98 ± 0.24	86.88 ± 0.05	85.23 ± 0.18

Table 9.5: Full results of the different models on CelebA. All scores must be added 11400 in this table.

number of epoches	*ELBO:* 10	30	100	*NLL:* 10	30	100
VAE	23.78 ± 1.95	17.99 ± 0.4	14.72 ± 0.16	17.35 ± 1.7	12.68 ± 0.62	10.11 ± 0.32
IWAE, $K = 10$	20.59 ± 0.71	15.45 ± 0.52	12.2 ± 0.3	18.25 ± 0.6	13.18 ± 0.42	10.14 ± 0.31
IWAE, $K = 50$	19.05 ± 0.39	13.59 ± 0.5	10.48 ± 0.89	19.08 ± 0.42	13.17 ± 0.54	10.12 ± 0.86
L-MCVAE Fixed, $K = 5$	21.93 ± 1.34	13.12 ± 1.27	12.03 ± 1.21	16.65 ± 1.55	10.12 ± 1.38	9.14 ± 1.27
L-MCVAE Sigmoidal, $K = 5$	21.61 ± 1.48	12.72 ± 0.43	11.6 ± 0.37	16.42 ± 1.47	9.62 ± 0.47	8.72 ± 0.4
L-MCVAE All learnable, $K = 5$	20.75 ± 0.65	12.99 ± 0.7	11.91 ± 0.61	16.16 ± 0.93	10.01 ± 0.72	9.03 ± 0.64
L-MCVAE Fixed, $K = 10$	21.49 ± 0.03	12.83 ± 0.57	11.76 ± 0.56	17.67 ± 0.75	10.26 ± 0.9	9.24 ± 0.79
L-MCVAE Sigmoidal, $K = 10$	19.44 ± 0.82	11.81 ± 0.45	10.7 ± 0.4	15.67 ± 1.48	9.24 ± 0.8	8.24 ± 0.73
L-MCVAE All learnable, $K = 10$	20.7 ± 1.15	11.81 ± 0.34	10.6 ± 0.23	17.0 ± 1.87	9.29 ± 0.73	8.26 ± 0.52
A-MCVAE Fixed, $K = 3$	21.59 ± 1.5	13.94 ± 0.42	12.84 ± 0.3	16.64 ± 1.37	10.98 ± 0.48	9.95 ± 0.3
A-MCVAE Sigmoidal, $K = 3$	23.63 ± 1.19	14.17 ± 0.26	12.96 ± 0.18	18.0 ± 0.54	11.09 ± 0.2	10.11 ± 0.13
A-MCVAE All learnable, $K = 3$	22.11 ± 1.66	14.62 ± 0.35	13.54 ± 0.18	17.38 ± 1.54	11.68 ± 0.33	10.67 ± 0.16
A-MCVAE Fixed, $K = 5$	20.13 ± 1.11	13.11 ± 0.38	11.99 ± 0.56	16.71 ± 1.47	10.64 ± 0.24	9.63 ± 0.32
A-MCVAE Sigmoidal, $K = 5$	20.95 ± 1.18	12.42 ± 0.42	11.13 ± 0.37	17.42 ± 1.49	9.97 ± 0.59	8.82 ± 0.57
A-MCVAE All learnable, $K = 5$	22.17 ± 0.17	12.73 ± 0.09	11.46 ± 0.15	18.97 ± 1.04	10.41 ± 0.28	9.22 ± 0.16
VAE with RealNVP	15.56 ± 0.29	13.60 ± 0.35	12.21 ± 0.27	10.69 ± 0.19	9.09 ± 0.26	8.98 ± 0.2

Table 9.6: Results of the different models on CIFAR-10 with different annealing schemes. All scores must be added 2800 in this table.

number of epoches	*ELBO:* 10	30	100	*NLL:* 10	30	100
VAE	69.57 ± 0.08	69.55 ± 0.51	68.84 ± 0.06	68.51 ± 0.07	68.41 ± 0.33	67.9 ± 0.03
IWAE, $K = 10$	69.82 ± 0.03	69.35 ± 0.03	69.36 ± 0.36	68.56 ± 0.03	68.0 ± 0.03	68.02 ± 0.4
IWAE, $K = 50$	69.94 ± 0.08	69.55 ± 0.04	69.43 ± 0.03	69.15 ± 0.15	68.37 ± 0.18	67.93 ± 0.02
L-MCVAE Fixed, $K = 5$	70.86 ± 0.53	68.44 ± 0.18	68.12 ± 0.11	69.37 ± 0.37	67.78 ± 0.1	67.53 ± 0.07
L-MCVAE Sigmoidal, $K = 5$	70.9 ± 0.59	68.46 ± 0.13	68.12 ± 0.11	69.42 ± 0.39	67.77 ± 0.11	67.51 ± 0.08
L-MCVAE All learnable, $K = 5$	70.62 ± 0.41	68.55 ± 0.18	68.09 ± 0.1	69.15 ± 0.38	67.73 ± 0.07	67.5 ± 0.07
L-MCVAE Fixed, $K = 10$	70.67 ± 0.42	68.37 ± 0.06	69.07 ± 1.49	69.62 ± 0.54	67.78 ± 0.06	67.51 ± 0.03
L-MCVAE Sigmoidal, $K = 10$	70.99 ± 0.59	68.36 ± 0.04	68.03 ± 0.0	69.8 ± 0.67	67.76 ± 0.04	67.51 ± 0.03
L-MCVAE All learnable, $K = 10$	71.19 ± 0.79	68.36 ± 0.03	68.01 ± 0.04	69.95 ± 0.62	67.78 ± 0.07	67.5 ± 0.05
A-MCVAE Fixed, $K = 3$	69.97 ± 0.99	68.48 ± 0.29	68.18 ± 0.16	69.26 ± 0.76	67.77 ± 0.18	67.55 ± 0.1
A-MCVAE Sigmoidal, $K = 3$	70.5 ± 1.18	68.45 ± 0.28	68.19 ± 0.18	69.18 ± 0.8	67.77 ± 0.19	67.56 ± 0.11
A-MCVAE All learnable, $K = 3$	70.69 ± 1.23	68.44 ± 0.3	68.17 ± 0.18	69.36 ± 0.89	67.76 ± 0.2	67.55 ± 0.11
A-MCVAE Fixed, $K = 5$	70.37 ± 1.04	68.31 ± 0.21	68.04 ± 0.1	69.36 ± 0.87	67.73 ± 0.17	67.51 ± 0.08
A-MCVAE Sigmoidal, $K = 5$	70.89 ± 0.38	68.4 ± 0.05	68.07 ± 0.04	69.71 ± 0.33	67.8 ± 0.04	67.53 ± 0.02
A-MCVAE All learnable, $K = 5$	70.1 ± 0.89	68.28 ± 0.2	68.01 ± 0.08	69.23 ± 0.75	67.71 ± 0.15	67.5 ± 0.07
VAE with RealNVP	70.01 ± 0.12	69.51 ± 0.07	69.19 ± 0.13	68.73 ± 0.05	68.35 ± 0.05	68.05 ± 0.02

Eq. (S30) is satisfied for $k = 1$ by (S28). Assume that (S30) is satisfied for $k \leqslant K - 1$. By (S28),

$$\begin{aligned}\Gamma_{k+1}(\mathrm{d}z_{k+1}) \prod_{i=k+1}^{1} L_{i-1}(z_i, \mathrm{d}z_{i-1}) &= \Gamma_{k+1}(\mathrm{d}z_{k+1}) L_k(z_{k+1}, \mathrm{d}z_k) \prod_{i=k}^{1} L_{i-1}(z_i, \mathrm{d}z_{i-1}) \\ &= \Gamma_k(\mathrm{d}z_k) M_{k+1}(z_k, \mathrm{d}z_{k+1}) w_{k+1}(z_k, z_{k+1}) \prod_{i=k}^{1} L_{i-1}(z_i, \mathrm{d}z_{i-1}) \\ &= M_{k+1}(z_k, \mathrm{d}z_{k+1}) w_{k+1}(z_k, z_{k+1}) \Gamma_0(\mathrm{d}z_0) \prod_{i=1}^{k} M_i(z_{i-1}, \mathrm{d}z_i) \prod_{i=1}^{k} w_i(z_{i-1}, z_i)\end{aligned}$$

which concludes the proof. □

We now highlight conditions under which (S28) is satisfied.

1. Assume that $\{\Gamma_k\}_{k=0}^K$ have positive densities w.r.t. to the Lebesgue measure, *i.e.* $\Gamma_k(\mathrm{d}z_k) = \Gamma_k(z_k)\mathrm{d}z_k$ and that the kernels $\{M_k\}_{k=1}^K$ and $\{L_k\}_{k=0}^{K-1}$ have positive transition densities $M_k(z_{k-1}, \mathrm{d}z_k) = m_k(z_{k-1}, z_k)\mathrm{d}z_k$ and $L_{k-1}(z_k, \mathrm{d}z_{k-1}) = \ell_{k-1}(z_k, z_{k-1})\mathrm{d}z_{k-1}$, $k \in \{1, \ldots, K\}$. Then,

$$w_k(z_{k-1}, z_k) = \frac{\gamma_k(z_k)\ell_{k-1}(z_k, z_{k-1})}{\gamma_{k-1}(z_{k-1})m_k(z_{k-1}, z_k)}$$

2. Assume that for $k \in \{1, \ldots, K\}$, $\Gamma_k(\mathrm{d}z_{k-1})M_k(z_{k-1}, \mathrm{d}z_k) = \Gamma_k(\mathrm{d}z_k)L_{k-1}(z_k, \mathrm{d}z_{k-1})$, and that there exists a positive measurable function such that $\Gamma_k(\mathrm{d}z_{k-1}) = \tilde{w}_k(z_{k-1})\Gamma_{k-1}(\mathrm{d}z_{k-1})$. Then,

$$\Gamma_k(\mathrm{d}z_k)L_{k-1}(z_k, \mathrm{d}z_{k-1}) = \Gamma_k(\mathrm{d}z_{k-1})M_k(z_{k-1}, \mathrm{d}z_k) = \tilde{w}_k(z_{k-1})\Gamma_{k-1}(\mathrm{d}z_{k-1})M_k(z_{k-1}, \mathrm{d}z_k) \; .$$

   Hence, (S28) is satisfied with $w_k(z_{k-1}, z_k) = \tilde{w}_k(z_{k-1})$. In particular, if for all $k \in \{0, \ldots, K\}$, $\Gamma_k(z_k) = \gamma_k(z_k)\mathrm{d}z_k$, where γ_k is a positive p.d.f., then $\tilde{w}_k(z_k) = \gamma_k(z_k)/\gamma_{k-1}(z_{k-1})$.

3. Assume that for $k \in \{1, \ldots, K\}$, M_k is reversible w.r.t. Γ_k, *i.e.* $\Gamma_k(\mathrm{d}z_{k-1})M_k(z_{k-1}, \mathrm{d}z_k) = \Gamma_k(\mathrm{d}z_k)M_k(z_k, \mathrm{d}z_{k-1})$, and that there exists a positive measurable function such that $\Gamma_k(\mathrm{d}z_{k-1}) = \tilde{w}_k(z_{k-1})\Gamma_{k-1}(\mathrm{d}z_{k-1})$. Then, setting $L_{k-1} = M_k$, (S28) is satisfied.

9.8.2 Proof of (9.14)

For $k \in \{1, \ldots, K\}$, $z_{k-1} \in \mathbb{R}^d$, denote by $G_{k,z_{k-1}}$ the mapping $u_k \mapsto \mathrm{T}_{u_k}(z_{k-1})$. Our derivation below rely on the fact that for $k \in \{1, \ldots, K\}$, $z_{k-1} \in \mathbb{R}^d$, $G_{k,z_{k-1}}$ is a C^1-diffeomorphism. This is the case for the Langevin mappings. Note, similarly to the density considered in Section 9.3, that $m_k(z_{k-1}, z_k) = \varphi(G_{k,z_{k-1}}^{-1}(z_k))\mathrm{J}_{G_{k,z_{k-1}}^{-1}}(z_k)$. When $K = 1$, we have

$$\begin{aligned}\int \log\big(w_1(z_0, z_1)\big) q_\phi^1(z_{0:1} \mid x)\mathrm{d}z_{0:1} &= \int \log\big(w_1(z_0, z_1)\big) q_\phi(z_0 \mid x)\mathrm{J}_{G_{1,z_0}^{-1}}(z_1)\varphi\big(G_{1,z_0}^{-1}(z_1)\big)\mathrm{d}z_{0:1} \\ &= \int \log\big(w_1(z_0, \mathrm{T}_{1,u_1}(z_0))\big) q_\phi(z_0 \mid x)\varphi(u_1)\mathrm{d}z_0\mathrm{d}u_1 \; ,\end{aligned}$$

where we have performed the change of variables $u_1 = G_{1,z_0}^{-1}(z_1)$, hence $z_1 = G_{1,z_0}(u_1) = \mathrm{T}_{1,u_1}(z_0)$. Let now K be in $\mathbb{N}^*$. In general, we write

$$\begin{aligned}
\mathcal{L}_{\mathrm{SIS}} &= \int \log\left(\prod_{k=1}^{K} w_k(z_{k-1}, z_k)\right) q_\phi^K(z_{0:K} \mid x)\mathrm{d}z_{0:K} \\
&= \int \log\left(\prod_{k=1}^{K} w_k(z_{k-1}, z_k)\right) q_\phi(z_0 \mid x) \prod_{k=1}^{K} m_k(z_{k-1}, z_k)\mathrm{d}z_{0:K-1}\mathrm{d}z_K \\
&= \int \log\left(\prod_{k=1}^{K} w_k(z_{k-1}, z_k)\right) q_\phi(z_0 \mid x) \prod_{k=1}^{K-1} m_k(z_{k-1}, z_k)\varphi(G_{K,z_{K-1}}^{-1}(z_K))\mathrm{J}_{G_{K,z_{K-1}}^{-1}}(z_K)\mathrm{d}z_{0:K-1}\mathrm{d}z_K \\
&= \int \log\left(\prod_{k=1}^{K-1} w_k(z_{k-1}, z_k) w_K(z_{K-1}, \mathrm{T}_{u_K}(z_{K-1}))q_\phi(z_0 \mid x)\right) \prod_{k=1}^{K-1} m_k(z_{k-1}, z_k)\varphi(u_K)\mathrm{d}z_{0:K-1}\mathrm{d}u_K
\end{aligned}$$

using the change of variables $u_K = G_{K,z_{K-1}}^{-1}(z_K)$. By an immediate backwards induction, we write

$$\mathcal{L}_{\mathrm{SIS}} = \int \log\left(\prod_{k=1}^{K} w_k(\bigcirc_{i=1}^{k-1} \mathrm{T}_{i,u_i}(z_0), \bigcirc_{i=1}^{k} \mathrm{T}_{i,u_i}(z_0))\right) q_\phi(z_0 \mid x)\varphi(u_{1:K})\mathrm{d}z_0\mathrm{d}u_{1:K} \, .$$

9.8.3 Proof of Lemma 71

Let $\eta < \mathrm{L}^{-1}$ and $u \in \mathbb{R}^D$. First we show that $\mathrm{T}_u^{\mathrm{MALA}}$ is invertible. Consider, for each $(y, u) \in \mathbb{R}^{2d}$, the mapping $H_{y,u}(z) = y - \sqrt{2\eta}u - \eta\nabla \log \pi(z)$. We have, for $z_1, z_2 \in \mathbb{R}^d$,

$$\|H_{y,u}(z_1) - H_{y,u}(z_2)\| \leqslant \eta\|\nabla \log \pi(z_1) - \nabla \log \pi(z_2)\| \leqslant \eta\mathrm{L}\|z_1 - z_2\|$$

and $\eta\mathrm{L} < 1$. Hence $H_{y,u}$ is a contraction mapping and thus has a unique fixed point $z_{y,u}$. Hence, for all $(y, u) \in \mathbb{R}^{2d}$ there exists a unique $z_{y,u}$ satisfying

$$H_{y,u}(z_{y,u}) = z_{y,u} \Rightarrow y = z_{y,u} + \eta\nabla \log \pi(z_{y,u}) + \sqrt{2\eta}u = \mathrm{T}_u^{\mathrm{MALA}}(z_{y,u}).$$

This establishes the invertibility of $\mathrm{T}_u^{\mathrm{MALA}}$. The fact that the inverse of $\mathrm{T}_u^{\mathrm{MALA}}$ is C^1 follows from a simple application of the local inverse function theorem.

9.9 ELBO AIS

9.9.1 Construction of the control variates

We prove in this section that the variance reduced objective we consider is valid. Sample now n samples $u_{0:K}^{1:n} \overset{\text{i.i.d.}}{\sim} \varphi_{d,K+1}$. For an index $i \in \{1, \ldots, n\}$, given the initial point $z_0^i = V_{\phi,x}(u_0^i)$ and the innovation noise $u_{1:K}^i$, we sample the A/R booleans $a_{1:K}^i$. We introduce, in the main text, for $i \in \{1, \ldots, n\}$

$$\tilde{W}_{n,i} = \frac{1}{n-1}\sum_{j \neq i} W(V_{\phi,x}(u_0^j), a_{1:K}^j, u_{1:K}^j) \, ,$$

$\tilde{W}_i$ provides a reasonable estimate of the AIS ELBO but is independent from the i-th trajectory. We use this quantity as a control variate to reduce the variance of our gradient estimator by introducing

$$\begin{aligned}
\widehat{\nabla\mathcal{L}_{\mathrm{AIS}}}_n &= n^{-1}\sum_{i=1}^{n} \nabla W(V_{\phi,x}(u_0^i), a_{1:K}^i, u_{1:K}^i) \\
&\quad + n^{-1}\sum_{i=1}^{n} \left[W(V_{\phi,x}(u_0^i), a_{1:K}^i, u_{1:K}^i) - \tilde{W}_{n,i}\right] \\
&\quad \times \nabla \log A(V_{\phi,x}(u_0^i), a_{1:K}^i, u_{1:K}^i) \, .
\end{aligned} \tag{S31}$$

Proving its unbiasedness boils down to proving that the term $n^{-1}\sum_{i=1}^n \tilde{W}_{n,i}\nabla \log A(V_{\phi,x}(u_0^i), a_{1:K}^i, u_{1:K}^i)$ has expectation zero. Let us compute for $i \in \{1\dots,n\}$,

$$\int \sum_{a_{1:K}^i} \varphi_{d,K+1}(u_{0:K}^i) A(V_{\phi,x}(u_0^i), a_{1:K}^i, u_{1:K}^i) \tilde{W}_{n,i} \nabla \log A(V_{\phi,x}(u_0^i), a_{1:K}^i, u_{1:K}^i) \mathrm{d}u_{0:K}^i =$$

$$\int \sum_{a_{1:K-1}^i} \varphi_{d,K+1}(u_{0:K}^i) \prod_{k=1}^K \alpha_{k,u_k^i}^{a_k^i}(z_{k-1}^i) \tilde{W}_{n,i} \left[\nabla \sum_{k=1}^{K-1} \log \alpha_{k,u_k^i}^{a_k^i}(z_{k-1}^i) + \sum_{a_K^i} \nabla \log \alpha_{K,u_K^i}^{a_K^i}(z_{K-1}^i) \right] \mathrm{d}u_{0:K}^i \ ,$$

denoting $z_0^j = V_{\phi,x}(u_0^j)$, $z_k^j = \bigcirc_{i=1}^k \mathrm{T}_{i,u_i^j}^{a_i^j}(z_0^j)$ by simplicity of notation. Yet, $\sum_{a_K^i} \alpha_{K,u_K^i}^{a_K^i}(z_{K-1}^i) = 1$ exactly, thus $\sum_{a_K^i} \alpha_{K,u_K^i}^{a_K^i}(z_{K-1}^l) \nabla \log \alpha_{K,u_K^i}^{a_K^i}(z_{K-1}^i) = 0$. We can thus show by an immediate induction that $\int \sum_{a_{1:K}^i} \varphi_d(u_{0:K}^i) \tilde{W}_{n,i} \nabla \log A(V_{\phi,x}(u_0^i), a_{1:K}^i, u_{1:K}^i) \mathrm{d}u_{i:K}^i = 0$, as $\tilde{W}_{n,i}$ is a constant in that integral by independence of the samples for $i \in \{1\dots,n\}$. Moreover, as

$$\int \sum_{a_{1:K}^{1:n}} \sum_{i=1}^n \tilde{W}_{n,i} \nabla \log A(V_{\phi,x}(u_0^i), a_{1:K}^i, u_{1:K}^i) \prod_{\ell=1}^n \varphi_{d,K+1}(u_{0:K}^\ell) \mathrm{d}u_{0:K}^{1:n} =$$

$$\int \sum_{i=1}^n \sum_{a_{1:K}^{-i}} \left[\int \sum_{a_{1:K}^i} \tilde{W}_{n,i} \nabla \log A(V_{\phi,x}(u_0^i), a_{1:K}^i, u_{1:K}^i) \varphi_{d,K+1}(u_{0:K}^i) \mathrm{d}u_{0:K}^i \right] \prod_{\ell \neq i} \varphi_{d,K+1}(u_{0:K}^\ell) \mathrm{d}u_{1:K}^{-i} \ ,$$

then $n^{-1}\sum_{i=1}^n \tilde{W}_{n,i} \nabla \log A(V_{\phi,x}(u_0^i), a_{1:K}^i, u_{1:K}^i)$ is of zero expectation, and (S31) is an unbiased estimator of the gradient.

9.9.2 Discussion of [WKN20b]

In [WKN20b], authors consider a MCMC VAE inspired by AIS. The model used however is quite different in spirit to what is performed in this work. [WKN20b] use Langevin mappings and accept reject steps in their VAE. Note however that the A/R probabilities defined are written as

$$\alpha(x,y) = 1 \wedge \pi(y)/\pi(x) \ ,$$

different from (9.21). Moreover, even though accept/reject steps are considered, the score function estimator (9.25) is not taken into account.

Finally, the initial density of the sequence is not taken to be some variational mean field initialization but directly the prior in the latent space. As a result, the scores obtained by the MCMC VAE are less competitive than that of the RNVP VAE presented in [WKN20b], Table 3., contrary to what is presented here.

References

[AB04] Felix V Agakov and David Barber. "An auxiliary variational method". In: *International Conference on Neural Information Processing*. Springer. 2004, pp. 561–566.

[AC75] Samuel M Allen and John W Cahn. "Coherent and incoherent equilibria in iron-rich iron-aluminum alloys". In: *Acta Metallurgica* 23.9 (1975), pp. 1017–1026. ISSN: 0001-6160. DOI: https://doi.org/10.1016/0001-6160(75)90106-6. URL: https://www.sciencedirect.com/science/article/pii/0001616075901066.

[ACB17] Martin Arjovsky, Soumith Chintala, and Léon Bottou. "Wasserstein generative adversarial networks". In: *International conference on machine learning*. PMLR. 2017, pp. 214–223.

[ADH10] Christophe Andrieu, Arnaud Doucet, and Roman Holenstein. "Particle Markov chain Monte Carlo methods". In: *Journal of the Royal Statistical Society: Series B* 72.3 (2010), pp. 269–342.

[Aga+17] Sergios Agapiou, Omiros Papaspiliopoulos, Daniel Sanz-Alonso, and AM Stuart. "Importance sampling: Intrinsic dimension and computational cost". In: *Statistical Science* (2017), pp. 405–431.

[AKS19] MS Albergo, G Kanwar, and PE Shanahan. "Flow-based generative models for Markov chain Monte Carlo in lattice field theory". In: *Physical Review D* 100.3 (2019), p. 034515.

[AKW12] Sungjin Ahn, Anoop Korattikara, and Max Welling. "Bayesian posterior sampling via stochastic gradient Fisher scoring". In: *arXiv preprint arXiv:1206.6380* (2012).

[AL19a] C. Andrieu and S. Livingstone. "Peskun-Tierney ordering for Markov chain and process Monte Carlo: beyond the reversible scenario". In: *arXiv preprint arXiv:1906.06197* (2019).

[AL19b] Christophe Andrieu and Samuel Livingstone. "Peskun-Tierney ordering for Markov chain and process Monte Carlo: beyond the reversible scenario". In: *arXiv preprint arXiv:1906.06197* (2019).

[ALV+18] Christophe Andrieu, Anthony Lee, Matti Vihola, et al. "Uniform ergodicity of the iterated conditional SMC and geometric ergodicity of particle Gibbs samplers". In: *Bernoulli* 24.2 (2018), pp. 842–872.

[AM21] Ömer Deniz Akyildiz and Joaquín Mıéguez. "Convergence rates for optimised adaptive importance samplers". In: *Statistics and Computing* 31.2 (2021), pp. 1–17.

[And+18] Christophe Andrieu, Arnaud Doucet, Sinan Yıldırım, and Nicolas Chopin. "On the utility of Metropolis-Hastings with asymmetric acceptance ratio". In: *arXiv preprint arXiv:1803.09527* (2018).

[Ard+19] Lynton Ardizzone, Carsten Lüth, Jakob Kruse, Carsten Rother, and Ullrich Köthe. *Guided Image Generation with Conditional Invertible Neural Networks*. 2019. arXiv: 1907.02392 [cs.CV].

[ASW14] Sungjin Ahn, Babak Shahbaba, and Max Welling. "Distributed stochastic gradient MCMC". In: *International conference on machine learning*. PMLR. 2014, pp. 1044–1052.

[AT08] Christophe Andrieu and Johannes Thoms. "A tutorial on adaptive MCMC". In: *Statistics and computing* 18.4 (2008), pp. 343–373.

[Aza+18] Samaneh Azadi, Catherine Olsson, Trevor Darrell, Ian Goodfellow, and Augustus Odena. "Discriminator Rejection Sampling". In: *International Conference on Learning Representations*. 2018.

[Aza+19] Samaneh Azadi, Catherine Olsson, Trevor Darrell, Ian Goodfellow, and Augustus Odena. "Discriminator Rejection Sampling". In: *arXiv:1810.06758* (2019). eprint: 1810.06758 (stat.ML).

[BDH17] Rémi Bardenet, Arnaud Doucet, and Christopher C Holmes. "On Markov chain Monte Carlo methods for tall data". In: *Journal of Machine Learning Research* 18.47 (2017).

[BDM12] Mylène Bédard, Randal Douc, and Eric Moulines. "Scaling analysis of multiple-try MCMC methods". In: *Stochastic Processes and their Applications* 122.3 (2012), pp. 758–786.

[BDM18] Nicolas Brosse, Alain Durmus, and Eric Moulines. "The promises and pitfalls of stochastic gradient Langevin dynamics". In: *Advances in Neural Information Processing Systems* 31 (2018).

[Bes+13] Alexandros Beskos, Natesh Pillai, Gareth Roberts, Jesus-Maria Sanz-Serna, and Andrew Stuart. "Optimal tuning of the hybrid Monte Carlo algorithm". In: *Bernoulli* 19.5A (2013), pp. 1501–1534.

[Bes94a] J.E. Besag. "Comments on "Representations of knowledge in complex systems" by U. Grenander and M. Miller". In: *J. Roy. Statist. Soc. Ser. B* 56 (1994), pp. 591–592.

[Bes94b] J.E. Besag. "Comments on "Representations of knowledge in complex systems" by U. Grenander and M. Miller". In: *J. Roy. Statist. Soc. Ser. B* 56 (1994), pp. 591–592.

[BGS15] Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. "Importance weighted autoencoders". In: *arXiv preprint arXiv:1509.00519* (2015).

[Bin+18] Eli Bingham, Jonathan P. Chen, Martin Jankowiak, Fritz Obermeyer, Neeraj Pradhan, Theofanis Karaletsos, Rohit Singh, Paul Szerlip, Paul Horsfall, and Noah D. Goodman. *Pyro: Deep Universal Probabilistic Programming*. 2018. arXiv: 1810.09538 [cs.LG].

[Bin+19] Eli Bingham, Jonathan P Chen, Martin Jankowiak, Fritz Obermeyer, Neeraj Pradhan, Theofanis Karaletsos, Rohit Singh, Paul Szerlip, Paul Horsfall, and Noah D Goodman. "Pyro: Deep universal probabilistic programming". In: *Journal of Machine Learning Research* 20.1 (2019), pp. 973–978.

[BJ18] N. Bou-Rabee and S.-S. Jesús Marıéa. "Geometric Integrators and the Hamiltonian Monte Carlo method". In: *Acta Numerica* (2018), pp. 1–92.

[Bon+11] Nicolas Bonneel, Michiel Van De Panne, Sylvain Paris, and Wolfgang Heidrich. "Displacement interpolation using Lagrangian mass transport". In: *Proceedings of the 2011 SIGGRAPH Asia Conference*. 2011, pp. 1–12.

[BR17] Joris Bierkens and Gareth Roberts. "A piecewise deterministic scaling limit of lifted Metropolis–Hastings in the Curie–Weiss model". In: *Ann. Appl. Probab.* 27.2 (Apr. 2017), pp. 846–882. DOI: 10.1214/16-AAP1217. URL: https://doi.org/10.1214/16-AAP1217.

[Bro+11] Steve Brooks, Andrew Gelman, Galin Jones, and Xiao-Li Meng. *Handbook of Markov chain Monte Carlo*. CRC press, 2011.

[BZM20] Ricardo Baptista, Olivier Zahm, and Youssef Marzouk. "An adaptive transport framework for joint and conditional density estimation". In: *arXiv preprint arXiv:2009.10303* (2020).

[CDO+11] Su Chen, Josef Dick, Art B Owen, et al. "Consistency of Markov chain quasi-Monte Carlo on continuous state spaces". In: *The Annals of Statistics* 39.2 (2011), pp. 673–701.

[CDS18a] Anthony L Caterini, Arnaud Doucet, and Dino Sejdinovic. "Hamiltonian variational autoencoder". In: *Advances in Neural Information Processing Systems*. 2018, pp. 8167–8177.

[CDS18b] Anthony L Caterini, Arnaud Doucet, and Dino Sejdinovic. "Hamiltonian variational auto-encoder". In: *Advances in Neural Information Processing Systems*. 2018, pp. 8167–8177.

[CFG14] Tianqi Chen, Emily Fox, and Carlos Guestrin. "Stochastic gradient Hamiltonian Monte Carlo". In: *International conference on machine learning*. PMLR. 2014, pp. 1683–1691.

[Cha+18] Niladri Chatterji, Nicolas Flammarion, Yian Ma, Peter Bartlett, and Michael Jordan. "On the theory of variance reduction for stochastic gradient Monte Carlo". In: *International Conference on Machine Learning*. PMLR. 2018, pp. 764–773.

[Che+20a] Tong Che, Ruixiang ZHANG, Jascha Sohl-Dickstein, Hugo Larochelle, Liam Paull, Yuan Cao, and Yoshua Bengio. "Your GAN is Secretly an Energy-based Model and You Should Use Discriminator Driven Latent Sampling". In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin. Vol. 33. Curran Associates, Inc., 2020, pp. 12275–12287. URL: https://proceedings.neurips.cc/paper/2020/file/90525e70b7842930586545c6f1c9310c-Paper.pdf.

[Che+20b] Tong Che, Ruixiang Zhang, Jascha Sohl-Dickstein, Hugo Larochelle, Liam Paull, Yuan Cao, and Yoshua Bengio. "Your GAN is Secretly an Energy-based Model and You Should use Discriminator Driven Latent Sampling". In: *arXiv preprint arXiv:2003.06060* (2020).

[CJ21] Adam D Cobb and Brian Jalaian. "Scaling Hamiltonian Monte Carlo inference for Bayesian neural networks with symmetric splitting". In: *Uncertainty in Artificial Intelligence*. PMLR. 2021, pp. 675–685.

[CL07a] Radu V Craiu and Christiane Lemieux. "Acceleration of the multiple-try Metropolis algorithm using antithetic and stratified sampling". In: *Statistics and Computing* 17.2 (2007), p. 109.

[CL07b] Radu V. Craiu and Christiane Lemieux. "Acceleration of the Multiple-Try Metropolis algorithm using antithetic and stratified sampling". In: *Statistics and Computing* 17.2 (2007), p. 109. ISSN: 1573-1375. DOI: 10.1007/s11222-006-9009-4. URL: https://doi.org/10.1007/s11222-006-9009-4.

[CLP99] F. Chen, L. Lovász, and I. Pak. "Lifting Markov chains to speed up mixing". In: *Annual ACM Symposium on Theory of Computing (Atlanta, GA, 1999)*. ACM, New York, 1999, pp. 275–281. DOI: 10.1145/301250.301315. URL: https://doi.org/10.1145/301250.301315.

[CMD17] Chris Cremer, Quaid Morris, and David Duvenaud. "Reinterpreting importance-weighted autoencoders". In: *arXiv preprint arXiv:1704.02916* (2017).

[Cor+19] Rob Cornish, Anthony L Caterini, George Deligiannidis, and Arnaud Doucet. "Relaxing bijectivity constraints with continuously indexed normalising flows". In: *arXiv preprint arXiv:1909.13833* (2019).

[Cor+20] Rob Cornish, Anthony Caterini, George Deligiannidis, and Arnaud Doucet. "Relaxing bijectivity constraints with continuously indexed normalising flows". In: *International conference on machine learning*. PMLR. 2020, pp. 2133–2143.

[CR10] N. Chopin and C. P. Robert. "Properties of nested sampling". In: *Biometrika* 97.3 (2010), pp. 741–755.

[Cro98] Gavin E Crooks. "Nonequilibrium measurements of free energy differences for microscopically reversible Markovian systems". In: *Journal of Statistical Physics* 90.5-6 (1998), pp. 1481–1487.

[CS97] Ming-Hui Chen and Qi-Man Shao. "On Monte Carlo Methods for Estimating Ratios of Normalizing Constants". In: *Annals of Statistics* 25 (Aug. 1997). DOI: 10.1214/aos/1031594732.

[CTA19] Nicola De Cao, Ivan Titov, and Wilker Aziz. "Block Neural Autoregressive Flow". In: *UAI*. 2019.

[Dal17] Arnak S. Dalalyan. "Theoretical guarantees for approximate sampling from smooth and log-concave densities". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 79.3 (2017), pp. 651–676. DOI: 10.1111/rssb.12183. eprint: https://rss.onlinelibrary.wiley.com/doi/pdf/10.1111/rssb.12183. URL: https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/rssb.12183.

[Dax+20] Erik Daxberger, Eric Nalisnick, James Allingham, Javier Antorán, and José Miguel Hernández-Lobato. "Expressive yet tractable Bayesian deep learning via subnetwork inference". In: (2020).

[DB+16] Kushal Kr Dey, Sourabh Bhattacharya, et al. "On geometric ergodicity of additive and multiplicative transformation-based Markov Chain Monte Carlo in high dimensions". In: *Brazilian Journal of Probability and Statistics* 30.4 (2016), pp. 570–613.

[DB14] Somak Dutta and Sourabh Bhattacharya. "Markov chain Monte Carlo based on deterministic transformations". In: *Statistical Methodology* 16 (Jan. 2014), pp. 100–116. ISSN: 1572-3127. DOI: 10.1016/j.stamet.2013.08.006.

[DDJ06a] P. Del Moral, A. Doucet, and A. Jasra. "Sequential Monte Carlo samplers". In: *Journal of the Royal Statistical Society: Series B* 68.3 (2006), pp. 411–436.

[DDJ06b] Pierre Del Moral, Arnaud Doucet, and Ajay Jasra. "Sequential Monte Carlo samplers". In: *Journal of the Royal Statistical Society: Series* B 68.3 (2006), pp. 411–436.

[DF19] Xinqiang Ding and David J Freedman. "Learning Deep Generative Models with Annealed Importance Sampling". In: *arXiv preprint arXiv:1906.04904* (2019).

[DHN00] Persi Diaconis, Susan Holmes, and Radford M. Neal. "Analysis of a nonreversible Markov chain sampler". In: *Ann. Appl. Probab.* 10.3 (2000), pp. 726–752. ISSN: 1050-5164. DOI: 10.1214/aoap/1019487508. URL: https://doi.org/10.1214/aoap/1019487508.

[Din+14] Nan Ding, Youhan Fang, Ryan Babbush, Changyou Chen, Robert D Skeel, and Hartmut Neven. "Bayesian sampling using stochastic gradient thermostats". In: *Advances in neural information processing systems* 27 (2014).

[DK16] Samuel Dodge and Lina Karam. "Understanding how image quality affects deep neural networks". In: *2016 eighth international conference on quality of multimedia experience (QoMEX)*. IEEE. 2016, pp. 1–6.

[DK19] Arnak S Dalalyan and Avetik Karagulyan. "User-friendly guarantees for the Langevin Monte Carlo with inaccurate gradient". In: *Stochastic Processes and their Applications* 129.12 (2019), pp. 5278–5311.

[DM17] Alain Durmus and Eric Moulines. "Nonasymptotic convergence analysis for the unadjusted Langevin algorithm". In: *The Annals of Applied Probability* 27.3 (2017), pp. 1551–1587.

[DMS17] Alain Durmus, Eric Moulines, and Eero Saksman. "On the convergence of Hamiltonian Monte Carlo". In: *Accepted for publication in Ann. Statist.* (2017).

[Dou+11] Randal Douc, Aurélien Garivier, Eric Moulines, Jimmy Olsson, et al. "Sequential Monte Carlo smoothing for general state space hidden Markov models". In: *Annals of Applied Probability* 21.6 (2011), pp. 2109–2145.

[Dou+15] Arnaud Doucet, Michael K Pitt, George Deligiannidis, and Robert Kohn. "Efficient implementation of Markov chain Monte Carlo when using an unbiased likelihood estimator". In: *Biometrika* 102.2 (2015), pp. 295–313.

[Dou+18] R. Douc, E. Moulines, P. Priouret, and P. Soulier. *Markov chains*. Springer Series in Operations Research and Financial Engineering. Springer, Cham, 2018, pp. xviii+757. ISBN: 978-3-319-97703-4; 978-3-319-97704-1. DOI: 10.1007/978-3-319-97704-1. URL: https://doi.org/10.1007/978-3-319-97704-1.

[DS18] Justin Domke and Daniel R Sheldon. "Importance weighting and variational inference". In: *Advances in neural information processing systems*. 2018, pp. 4470–4479.

[DSB16] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. "Density estimation using Real NVP". In: *arXiv preprint arXiv:1605.08803* (2016).

[DSB17] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. *Density estimation using Real NVP*. 2017. arXiv: 1605.08803 [cs.LG].

[Dua+87] Simon Duane, A.D. Kennedy, Brian J. Pendleton, and Duncan Roweth. "Hybrid Monte Carlo". In: *Physics Letters B* 195.2 (1987), pp. 216–222. ISSN: 0370-2693. DOI: https://doi.org/10.1016/0370-2693(87)91197-X. URL: http://www.sciencedirect.com/science/article/pii/037026938791197X.

[Dus+20] Michael Dusenberry, Ghassen Jerfel, Yeming Wen, Yian Ma, Jasper Snoek, Katherine Heller, Balaji Lakshminarayanan, and Dustin Tran. "Efficient and scalable Bayesian neural nets with rank-1 factors". In: *International conference on machine learning*. PMLR. 2020, pp. 2782–2792.

[EM12] Tarek A El Moselhy and Youssef M Marzouk. "Bayesian inference with optimal maps". In: *Journal of Computational Physics* 231.23 (2012), pp. 7815–7850.

[Fal19] WA Falcon. "PyTorch Lightning". In: *GitHub. Note: https://github.com/PyTorchLightning/pytorch-lightning* 3 (2019).

[Foo+19] Andrew YK Foong, Yingzhen Li, José Miguel Hernández-Lobato, and Richard E Turner. "'In-Between'Uncertainty in Bayesian Neural Networks". In: *arXiv preprint arXiv:1906.11537* (2019).

[Foo+20] Andrew Foong, David Burt, Yingzhen Li, and Richard Turner. "On the expressiveness of approximate inference in Bayesian neural networks". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 15897–15908.

[Fra+20] Guilherme Franca, Jeremias Sulam, Daniel P Robinson, and René Vidal. "Conformal symplectic and relativistic optimization". In: *Journal of Statistical Mechanics: Theory and Experiment* 2020.12 (2020), p. 124008.

[FSG20] Sebastian Farquhar, Lewis Smith, and Yarin Gal. "Liberty or depth: Deep Bayesian neural nets do not need complex weight posterior approximations". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 4346–4357.

[FSS14] Youhan Fang, Jesus-Maria Sanz-Serna, and Robert D Skeel. "Compressible generalized hybrid Monte Carlo". In: *The Journal of chemical physics* 140.17 (2014), p. 174108.

[FW12] Nial Friel and Jason Wyse. "Estimating the evidence–a review". In: *Statistica Neerlandica* 66.3 (2012), pp. 288–308.

[Gal16] Yarin Gal. "Uncertainty in Deep Learning". In: 2016.

[Gaw+21] Jakob Gawlikowski, Cedrique Rovile Njieutcheu Tassi, Mohsin Ali, Jongseok Lee, Matthias Humt, Jianxiang Feng, Anna Kruspe, Rudolph Triebel, Peter Jung, Ribana Roscher, et al. "A survey of uncertainty in deep neural networks". In: *arXiv preprint arXiv:2107.03342* (2021).

[GBC16] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[GC11] Mark Girolami and Ben Calderhead. "Riemann manifold Langevin and Hamiltonian Monte Carlo methods". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 73.2 (2011), pp. 123–214.

[GF21] Adrià Garriga-Alonso and Vincent Fortuin. "Exact Langevin dynamics with stochastic gradients". In: *arXiv preprint arXiv:2102.01691* (2021).

[GG16] Yarin Gal and Zoubin Ghahramani. "Dropout as a Bayesian approximation: Representing model uncertainty in deep learning". In: *international conference on machine learning*. PMLR. 2016, pp. 1050–1059.

[GGA15] Roger B Grosse, Zoubin Ghahramani, and Ryan P Adams. "Sandwiching the marginal likelihood using bidirectional Monte Carlo". In: *arXiv preprint arXiv:1511.02543* (2015).

[GGR97] Andrew Gelman, Walter R Gilks, and Gareth O Roberts. "Weak convergence and optimal scaling of random walk Metropolis algorithms". In: *The annals of applied probability* 7.1 (1997), pp. 110–120.

[GHK17] Yarin Gal, Jiri Hron, and Alex Kendall. "Concrete dropout". In: *Advances in neural information processing systems* 30 (2017).

[GM98] Andrew Gelman and Xiao-Li Meng. "Simulating normalizing constants: From importance sampling to bridge sampling to path sampling". In: *Statistical science* (1998), pp. 163–185.

[Goo+14a] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. "Generative adversarial nets". In: *Advances in neural information processing systems* 27 (2014).

[Goo+14b] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. "Generative Adversarial Nets". In: *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*. NIPS'14. Montreal, Canada: MIT Press, 2014, pp. 2672–2680.

[Goy+17] Anirudh Goyal Alias Parth Goyal, Nan Rosemary Ke, Surya Ganguli, and Yoshua Bengio. "Variational walkback: Learning a transition operator as a stochastic recurrent net". In: *Advances in Neural Information Processing Systems*. 2017, pp. 4392–4402.

[Gra11] Alex Graves. "Practical variational inference for neural networks". In: *Advances in neural information processing systems* 24 (2011).

[GRV21] Marylou Gabrié, Grant M. Rotskoff, and Eric Vanden-Eijnden. "Adaptive Monte Carlo augmented with normalizing flows". In: *arXiv preprint arXiv:2105.12603* (2021).

[Gus98] Paul Gustafson. "A guided walk Metropolis algorithm". In: *Statistics and computing* 8.4 (1998), pp. 357–364.

[HA15] José Miguel Hernández-Lobato and Ryan Adams. "Probabilistic backpropagation for scalable learning of Bayesian neural networks". In: *International conference on machine learning*. PMLR. 2015, pp. 1861–1869.

[HAB19] Matthias Hein, Maksym Andriushchenko, and Julian Bitterwolf. "Why relu networks yield high-confidence predictions far away from the training data and how to mitigate the problem". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 41–50.

[Har82] P. Hartman. *Ordinary Differential Equations: Second Edition*. Classics in Applied Mathematics. Society for Industrial and Applied Mathematics (SIAM, 3600 Market Street, Floor 6, Philadelphia, PA 19104), 1982. ISBN: 9780898719222. URL: `https://books.google.fr/books?id=NEkkJ93O9okC`.

[Hen+20] Jeremy Heng, Adrian N Bishop, George Deligiannidis, and Arnaud Doucet. "Controlled sequential Monte Carlo". In: *The Annals of Statistics* 48.5 (2020), pp. 2904–2929.

[Heu+17] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. "GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett. Vol. 30. Curran Associates, Inc., 2017. URL: `https://proceedings.neurips.cc/paper/2017/file/8a1d694707eb0fefe65871369074926d-Paper.pdf`.

[HG+14a] Matthew D Hoffman, Andrew Gelman, et al. "The No-U-Turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo." In: *J. Mach. Learn. Res.* 15.1 (2014), pp. 1593–1623.

[HG+14b] Matthew D Hoffman, Andrew Gelman, et al. "The No-U-Turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo." In: *J. Mach. Learn. Res.* 15.1 (2014), pp. 1593–1623.

[HH80] P. Hall and C. Heyde. *Martingale Limit Theory and Its Application*. Academic Press, 1980.

[HHS21] Paul Hagemann, Johannes Hertrich, and Gabriele Steidl. "Stochastic normalizing flows for inverse problems: a Markov Chains viewpoint". In: *arXiv preprint arXiv:2109.11375* (2021).

[HM20] Matthew Hoffman and Yian Ma. "Black-box variational inference as a parametric approximation to Langevin dynamics". In: *International Conference on Machine Learning*. PMLR. 2020, pp. 4324–4341.

[Ho+19] Jonathan Ho, Xi Chen, Aravind Srinivas, Yan Duan, and Pieter Abbeel. "Flow++: Improving flow-based generative models with variational dequantization and architecture design". In: *arXiv preprint arXiv:1902.00275* (2019).

[Hod+20] Liam Hodgkinson, Chris van der Heide, Fred Roosta, and Michael W Mahoney. "Stochastic normalizing flows". In: *arXiv preprint arXiv:2002.09547* (2020).

[Hof+19] Matthew Hoffman, Pavel Sountsov, Joshua V Dillon, Ian Langmore, Dustin Tran, and Srinivas Vasudevan. "NeuTra-lizing Bad Geometry in Hamiltonian Monte Carlo Using Neural Transport". In: *arXiv preprint arXiv:1903.03704* (2019).

[Hof17] Matthew D. Hoffman. "Learning Deep Latent Gaussian Models with Markov Chain Monte Carlo". In: *Proceedings of the 34th International Conference on Machine Learning*. Ed. by Doina Precup and Yee Whye Teh. Vol. 70. Proceedings of Machine Learning Research. International Convention Centre, Sydney, Australia: PMLR, June 2017, pp. 1510–1519.

[Hor91] Alan M Horowitz. "A generalized guided Monte Carlo algorithm". In: *Physics Letters B* 268.2 (1991), pp. 247–252.

[HS13] K Hukushima and Y Sakai. "An irreversible Markov-chain Monte Carlo method with skew detailed balance conditions". In: *Journal of Physics: Conference Series*. Vol. 473. 1. IOP Publishing. 2013, p. 012012.

[HST99] Heikki Haario, Eero Saksman, and Johanna Tamminen. "Adaptive proposal distribution for random walk Metropolis algorithm". English. In: *Computational Statistics* 14.3 (1999), pp. 375–395. ISSN: 0943-4062.

[Hua+18a] Chin-Wei Huang, David Krueger, Alexandre Lacoste, and Aaron Courville. "Neural autoregressive flows". In: *arXiv preprint arXiv:1804.00779* (2018).

[Hua+18b] Chin-Wei Huang, Shawn Tan, Alexandre Lacoste, and Aaron C Courville. "Improving Explorability in Variational Inference with Annealed Variational Objectives". In: *Advances in Neural Information Processing Systems*. 2018, pp. 9701–9711.

[Izm+18a] Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. "Averaging weights leads to wider optima and better generalization". In: *34th Conference on Uncertainty in Artificial Intelligence 2018, UAI 2018*. Association For Uncertainty in Artificial Intelligence (AUAI). 2018, pp. 876–885.

[Izm+18b] Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. "Averaging weights leads to wider optima and better generalization". In: *arXiv preprint arXiv:1803.05407* (2018).

[Izm+20] Pavel Izmailov, Wesley J Maddox, Polina Kirichenko, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. "Subspace inference for Bayesian deep learning". In: *Uncertainty in Artificial Intelligence*. PMLR. 2020, pp. 1169–1179.

[Izm+21] Pavel Izmailov, Sharad Vikram, Matthew D Hoffman, and Andrew Gordon Gordon Wilson. "What are Bayesian neural network posteriors really like?" In: *International Conference on Machine Learning*. PMLR. 2021, pp. 4629–4640.

[JS20] He Jia and Uros Seljak. "Normalizing constant estimation with Gaussianized bridge sampling". In: *Symposium on Advances in Approximate Bayesian Inference*. PMLR. 2020, pp. 1–14.

[KB14] Diederik P Kingma and Jimmy Ba. "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980* (2014).

[KG17] Alex Kendall and Yarin Gal. "What uncertainties do we need in Bayesian deep learning for computer vision?" In: *Advances in neural information processing systems* 30 (2017).

[Kha+18] Mohammad Khan, Didrik Nielsen, Voot Tangkaratt, Wu Lin, Yarin Gal, and Akash Srivastava. "Fast and scalable Bayesian deep learning by weight-perturbation in adam". In: *International Conference on Machine Learning*. PMLR. 2018, pp. 2611–2620.

[Kin+16a] Diederik P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. "Improving variational inference with inverse autoregressive flow". In: *arXiv preprint arXiv:1606.04934* (2016).

[Kin+16b] Durk P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. "Improved variational inference with inverse autoregressive flow". In: *Advances in neural information processing systems*. 2016, pp. 4743–4751.

[Kis65] Leslie Kish. *Survey sampling*. English. Chichester : Wiley New York, 1965, xvi, 643 p. : ISBN: 0471109495.

[Kol+19] Soheil Kolouri, Kimia Nadjahi, Umut Simsekli, Roland Badeau, and K Gustavo. "Generalized Sliced Wasserstein Distances". In: *NeurIPS 2019*. 2019.

[KP18] Andreas Kamilaris and Francesc X Prenafeta-Boldú. "Deep learning in agriculture: A survey". In: *Computers and electronics in agriculture* 147 (2018), pp. 70–90.

[KPB19] Ivan Kobyzev, Simon Prince, and Marcus A Brubaker. "Normalizing flows: Introduction and ideas". In: *arXiv preprint arXiv:1908.09257* (2019).

[KPB20] Ivan Kobyzev, Simon JD Prince, and Marcus A Brubaker. "Normalizing flows: An introduction and review of current methods". In: *IEEE transactions on pattern analysis and machine intelligence* 43.11 (2020), pp. 3964–3979.

[KW13a] Diederik P Kingma and Max Welling. "Auto-encoding variational bayes". In: *arXiv preprint arXiv:1312.6114* (2013).

[KW13b] Diederik P Kingma and Max Welling. "Auto-encoding variational bayes". In: *arXiv preprint arXiv:1312.6114* (2013).

[KW14] Diederik Kingma and Max Welling. "Auto-Encoding Variational Bayes". In: *International Conference on Learning Representations*. 2014.

[KW19] Diederik P Kingma and Max Welling. "An introduction to variational autoencoders". In: *arXiv preprint arXiv:1906.02691* (2019).

[Lag+21] Evgeny Lagutin, Daniil Selikhanovych, Achille Thin, Sergey Samsonov, Alexey Naumov, Denis Belomestny, Maxim Panov, and Eric Moulines. "Ex²MCMC: Sampling through Exploration Exploitation". In: *arXiv preprint arXiv:2111.02702* (2021).

[Law+19] Dieterich Lawson, George Tucker, Bo Dai, and Rajesh Ranganath. "Energy-inspired models: Learning with sampler-induced distributions". In: *arXiv preprint arXiv:1910.14265* (2019).

[LBH15] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. "Deep learning". In: *nature* 521.7553 (2015), pp. 436–444.

[LDM15] Fredrik Lindsten, Randal Douc, and Éric Moulines. "Uniform ergodicity of the particle Gibbs sampler". In: *Scandinavian Journal of Statistics* 42.3 (2015), pp. 775–797.

[Lee+10] Anthony Lee, Christopher Yau, Michael B Giles, Arnaud Doucet, and Christopher C Holmes. "On the utility of graphics cards to perform massively parallel simulation of advanced Monte Carlo methods". In: *Journal of computational and graphical statistics* 19.4 (2010), pp. 769–789.

[Lee11] Anthony Lee. "On auxiliary variables and many-core architectures in computational statistics". PhD thesis. University of Oxford, 2011.

[LFR19] Samuel Livingstone, Michael F Faulkner, and Gareth O Roberts. "Kinetic energy choice in Hamiltonian/hybrid Monte Carlo". In: *Biometrika* 106.2 (2019), pp. 303–319.

[LHS17a] Daniel Levy, Matthew D Hoffman, and Jascha Sohl-Dickstein. "Generalizing Hamiltonian Monte Carlo with neural networks". In: *arXiv preprint arXiv:1711.09268* (2017).

[LHS17b] Daniel Levy, Matthew D Hoffman, and Jascha Sohl-Dickstein. "Generalizing Hamiltonian Monte Carlo with neural networks". In: *arXiv preprint arXiv:1711.09268* (2017).

[LHS18] Daniel Levy, Matthew D. Hoffman, and Jascha Sohl-Dickstein. "Generalizing Hamiltonian Monte Carlo with Neural Networks". In: *International Conference on Learning Representations*. 2018.

[Li+16] Chunyuan Li, Changyou Chen, David Carlson, and Lawrence Carin. "Preconditioned stochastic gradient Langevin dynamics for deep neural networks". In: *Thirtieth AAAI Conference on Artificial Intelligence*. 2016.

[Liu+18] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. "Large-scale celebfaces attributes (celeba) dataset". In: *Retrieved August* 15.2018 (2018), p. 11.

[LLC11] Faming Liang, Chuanhai Liu, and Raymond Carroll. *Advanced Markov chain Monte Carlo methods: learning from past samples*. Vol. 714. John Wiley & Sons, 2011.

[LLW00] Jun S Liu, Faming Liang, and Wing Hung Wong. "The multiple-try method and local optimization in Metropolis sampling". In: *Journal of the American Statistical Association* 95.449 (2000), pp. 121–134.

[LM00] B. Laurent and P. Massart. "Adaptive estimation of a quadratic functional by model selection". In: *Ann. Statist.* 28.5 (Oct. 2000), pp. 1302–1338. DOI: 10.1214/aos/1015957395. URL: https://doi.org/10.1214/aos/1015957395.

[LS13] Fredrik Lindsten and Thomas B Schön. "Backward simulation methods for Monte Carlo statistical inference". In: *Foundations and Trends® in Machine Learning* 6.1 (2013), pp. 1–143.

[Lu+17] Xiaoyu Lu, Valerio Perrone, Leonard Hasenclever, Yee Whye Teh, and Sebastian Vollmer. "Relativistic Monte Carlo". In: *Artificial Intelligence and Statistics*. PMLR. 2017, pp. 1236–1245.

[LW17] Christos Louizos and Max Welling. "Multiplicative normalizing flows for variational Bayesian neural networks". In: *International Conference on Machine Learning*. PMLR. 2017, pp. 2218–2227.

[Ma+16] Yi-An Ma, Tianqi Chen, Lei Wu, and Emily B Fox. "A unifying framework for devising efficient and irreversible MCMC samplers". In: *arXiv preprint arXiv:1608.05973* (2016).

[Maa+16] Lars Maaløe, Casper Kaae Sønderby, Søren Kaae Sønderby, and Ole Winther. "Auxiliary deep generative models". In: *International conference on machine learning*. PMLR. 2016, pp. 1445–1453.

[Mad+17] Chris J Maddison, Dieterich Lawson, George Tucker, Nicolas Heess, Mohammad Norouzi, Andriy Mnih, Arnaud Doucet, and Yee Whye Teh. "Filtering variational objectives". In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. 2017, pp. 6576–6586.

[Mad+18] Chris J Maddison, Daniel Paulin, Yee Whye Teh, Brendan O'Donoghue, and Arnaud Doucet. "Hamiltonian descent methods". In: *arXiv preprint arXiv:1809.05042* (2018).

[Mad+19] Wesley J Maddox, Pavel Izmailov, Timur Garipov, Dmitry P Vetrov, and Andrew Gordon Wilson. "A simple baseline for Bayesian uncertainty in deep learning". In: *Advances in Neural Information Processing Systems* 32 (2019).

[McA+17] Rowan McAllister, Yarin Gal, Alex Kendall, Mark Van Der Wilk, Amar Shah, Roberto Cipolla, and Adrian Weller. "Concrete problems for autonomous vehicle safety: Advantages of Bayesian deep learning". In: *International Joint Conferences on Artificial Intelligence, Inc.* IEEE. 2017.

[MCF15] Yi-An Ma, Tianqi Chen, and Emily Fox. "A complete recipe for stochastic gradient MCMC". In: *Advances in neural information processing systems* 28 (2015).

[MFR20] Gael M Martin, David T Frazier, and Christian P Robert. "Computing Bayes: Bayesian computation from 1763 to the 21st century". In: *arXiv preprint arXiv:2004.06425* (2020).

[MHB17] Stephan Mandt, Matthew D Hoffman, and David M Blei. "Stochastic gradient descent as approximate Bayesian inference". In: *arXiv preprint arXiv:1704.04289* (2017).

[Mic16] Manon Michel. "Irreversible Markov chains by the factorized Metropolis filter : algorithms and applications in particle systems and spin models". 2016PSLEE039. PhD thesis. 2016. URL: `http://www.theses.fr/2016PSLEE039/document`.

[Miy+18] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. "Spectral Normalization for Generative Adversarial Networks". In: *arXiv:1802.05957* (2018). eprint: `1802.05957` (cs.LG).

[MLY17] Seonwoo Min, Byunghan Lee, and Sungroh Yoon. "Deep learning in bioinformatics". In: *Briefings in bioinformatics* 18.5 (2017), pp. 851–869.

[MMT16] Chris J Maddison, Andriy Mnih, and Yee Whye Teh. "The concrete distribution: A continuous relaxation of discrete random variables". In: *arXiv preprint arXiv:1611.00712* (2016).

[Mon20] Pierre Monmarché. "High-dimensional MCMC with a standard splitting scheme for the underdamped Langevin". In: *arXiv preprint arXiv:2007.05455* (2020).

[Mon81] Gaspard Monge. "Mémoire sur la théorie des déblais et des remblais". In: *Histoire de l'Académie Royale des Sciences de Paris* (1781).

[MR16] Andriy Mnih and Danilo Rezende. "Variational inference for Monte Carlo objectives". In: *International Conference on Machine Learning*. PMLR. 2016, pp. 2188–2196.

[MT96] K. Mengersen and R. L. Tweedie. "Rates of Convergence of the Hastings and Metropolis Algorithms". In: *Ann. Statist.* 24 (1996), pp. 101–121.

[Mül+19] Thomas Müller, Brian McWilliams, Fabrice Rousselle, Markus Gross, and Jan Novák. "Neural importance sampling". In: *ACM Transactions on Graphics* 38.145 (2019).

[Nea01a] R. M. Neal. "Annealed importance sampling". In: *Statistics and Computing* 11 (2001), pp. 125–139.

[Nea01b] Radford M Neal. "Annealed importance sampling". In: *Statistics and Computing* 11.2 (2001), pp. 125–139.

[Nea03] R. M. Neal. "Slice sampling". In: *Ann. Statist.* 31.3 (June 2003), pp. 705–767. DOI: `10.1214/aos/1056562461`.

[Nea11] R. M. Neal. "MCMC Using Hamiltonian Dynamics". In: *Handbook of Markov Chain Monte Carlo* (2011), pp. 113–162.

[Nea12] Radford M Neal. *Bayesian learning for neural networks*. Vol. 118. Springer Science & Business Media, 2012.

[Nek+20] Kirill Neklyudov, Max Welling, Evgenii Egorov, and Dmitry Vetrov. "Involutive MCMC: a Unifying Framework". In: *arXiv preprint arXiv:2006.16653* (2020).

[Nij+20] Erik Nijkamp, Mitch Hill, Tian Han, Song-Chun Zhu, and Ying Nian Wu. "On the anatomy of MCMC-based maximum likelihood learning of energy-based models". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34. 04. 2020, pp. 5272–5280.

[NYC15] Anh Nguyen, Jason Yosinski, and Jeff Clune. "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 427–436.

[Oor+17] Aaron van den Oord et al. *Parallel WaveNet: Fast High-Fidelity Speech Synthesis*. 2017. arXiv: `1711.10433 [cs.LG]`.

[Ott16] Michela Ottobre. "Markov chain Monte Carlo and irreversibility". In: *Reports on Mathematical Physics* 77.3 (2016), pp. 267–292.

[OZ00a] Art Owen and Yi Zhou. "Safe and Effective Importance Sampling". In: *Journal of the American Statistical Association* 95.449 (2000), pp. 135–143.

[OZ00b] Art Owen and Yi Zhou. "Safe and effective importance sampling". In: *Journal of the American Statistical Association* 95.449 (2000), pp. 135–143.

[Pap+19] George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. "Normalizing flows for probabilistic modeling and inference". In: *arXiv preprint arXiv:1912.02762* (2019).

[Pap+21] George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. "Normalizing flows for probabilistic modeling and inference". In: *Journal of Machine Learning Research* 22.57 (2021), pp. 1–64.

[Par81] Giorgio Parisi. "Correlation functions and computer simulations". In: *Nuclear Physics B* 180.3 (1981), pp. 378–384.

[Pas+17] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. "Automatic differentiation in PyTorch". In: (2017).

[Pas+19] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. "Pytorch: An imperative style, high-performance deep learning library". In: *arXiv preprint arXiv:1912.01703* (2019).

[Pes73] Peter H Peskun. "Optimum Monte Carlo sampling using Markov chains". In: *Biometrika* 60.3 (1973), pp. 607–612.

[Pra19a] Dennis Prangle. "Distilling importance sampling". In: *arXiv preprint arXiv:1910.03632* (2019).

[Pra19b] Dennis Prangle. "Distilling importance sampling". In: *arXiv preprint arXiv:1910.03632* (2019).

[RC13a] C. Robert and G. Casella. *Monte Carlo statistical methods*. Springer Science & Business Media, 2013.

[RC13b] Christian Robert and George Casella. *Monte Carlo statistical methods*. Springer Science & Business Media, 2013.

[RM15a] Danilo Rezende and Shakir Mohamed. "Variational Inference with Normalizing Flows". In: *Proceedings of the 32nd International Conference on Machine Learning*. Ed. by Francis Bach and David Blei. Vol. 37. Proceedings of Machine Learning Research. Lille, France: PMLR, July 2015, pp. 1530–1538.

[RM15b] Danilo Rezende and Shakir Mohamed. "Variational inference with normalizing flows". In: *International Conference on Machine Learning*. PMLR. 2015, pp. 1530–1538.

[RMC16] Alec Radford, Luke Metz, and Soumith Chintala. "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks". In: *arXiv:1511.06434* (2016). eprint: 1511.06434 (cs.LG).

[RMW14] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. "Stochastic backpropagation and approximate inference in deep generative models". In: *arXiv preprint arXiv:1401.4082* (2014).

[Rob07] C. Robert. *The Bayesian choice: from decision-theoretic foundations to computational implementation*. Springer Science & Business Media, 2007.

[RR04] Gareth O Roberts and Jeffrey S Rosenthal. "General state space Markov chains and MCMC algorithms". In: *Probability surveys* 1 (2004), pp. 20–71.

[RR98] Gareth O Roberts and Jeffrey S Rosenthal. "Optimal scaling of discrete approximations to Langevin diffusions". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 60.1 (1998), pp. 255–268.

[RT19] Francisco Ruiz and Michalis Titsias. "A Contrastive Divergence for Combining Variational Inference and MCMC". In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. Long Beach, California, USA: PMLR, Sept. 2019, pp. 5537–5545.

[RT96] G. O. Roberts and R. L. Tweedie. "Geometric convergence and central limit theorems for multidimensional Hastings and Metropolis algorithms". In: *Biometrika* 83.1 (Mar. 1996), pp. 95–110. ISSN: 0006-3444. DOI: 10.1093/biomet/83.1.95. eprint: https://academic.oup.com/biomet/article-pdf/83/1/95/709644/83-1-95.pdf. URL: https://doi.org/10.1093/biomet/83.1.95.

[RTB16] Rajesh Ranganath, Dustin Tran, and David Blei. "Hierarchical variational models". In: *International Conference on Machine Learning*. PMLR. 2016, pp. 324–333.

[Rub87] Donald B Rubin. "Comment: A noniterative Sampling/Importance Resampling alternative to the data augmentation algorithm for creating a few imputations when fractions of missing information are modest: The SIR algorithm". In: *Journal of the American Statistical Association* 82.398 (1987), pp. 542–543.

[Rui+21] Francisco JR Ruiz, Michalis K Titsias, Taylan Cemgil, and Arnaud Doucet. "Unbiased Gradient Estimation for Variational Auto-Encoders using Coupled Markov Chains". In: *Uncertainty in Artificial Intelligence*. 2021.

[RV19] G.M. Rotskoff and E. Vanden-Eijnden. "Dynamical Computation of the Density of States and Bayes Factors Using Nonequilibrium Importance Sampling". In: *Physical Review Letters* 122.15 (2019), p. 150602.

[SBH03] Øivind Skare, Erik Bølviken, and Lars Holden. "Improved sampling-importance resampling and reduced bias importance sampling". In: *Scandinavian Journal of Statistics* 30.4 (2003), pp. 719–737.

[SC18] Tobias Schwedes and Ben Calderhead. "Quasi Markov chain Monte Carlo methods". In: *arXiv preprint arXiv:1807.00070* (2018).

[SFM20] Span Spanbauer, Cameron Freer, and Vikash Mansinghka. "Deep involutive generative models for neural MCMC". In: *arXiv preprint arXiv:2006.15167* (2020).

[SG92] Adrian FM Smith and Alan E Gelfand. "Bayesian statistics without tears: a sampling–resampling perspective". In: *The American Statistician* 46.2 (1992), pp. 84–88.

[SK21] Yang Song and Diederik P Kingma. "How to train your energy-based models". In: *arXiv preprint arXiv:2101.03288* (2021).

[Ski04] John Skilling. "Nested sampling". In: *AIP Conference Proceedings*. Vol. 735. 1. American Institute of Physics. 2004, pp. 395–405.

[Ski06] John Skilling. "Nested sampling for general Bayesian computation". In: *Bayesian Analysis* 1.4 (2006), pp. 833–859.

[SKW15a] Tim Salimans, Diederik Kingma, and Max Welling. "Markov chain Monte Carlo and variational inference: Bridging the gap". In: *International Conference on Machine Learning*. 2015, pp. 1218–1226.

[SKW15b] Tim Salimans, Diederik Kingma, and Max Welling. "Markov chain Monte Carlo and variational inference: Bridging the gap". In: *International Conference on Machine Learning*. 2015, pp. 1218–1226.

[SM08a] R. Salakhutdinov and I. Murray. "On the quantitative analysis of deep belief networks". In: *Proceedings of the 25th international conference on Machine Learning*. 2008, pp. 872–879.

[SM08b] Ruslan Salakhutdinov and Iain Murray. "On the quantitative analysis of deep belief networks". In: *Proceedings of the 25th international conference on Machine learning*. 2008, pp. 872–879.

[SMD14] Jascha Sohl-Dickstein, Mayur Mudigonda, and Michael R DeWeese. "Hamiltonian Monte Carlo without detailed balance". In: *arXiv preprint arXiv:1409.5191* (2014).

[Smi17] Leslie N Smith. "Cyclical learning rates for training neural networks". In: *2017 IEEE winter conference on applications of computer vision (WACV)*. IEEE. 2017, pp. 464–472.

[SN+18] Alexander Y Shestopaloff, Radford M Neal, et al. "Sampling latent states for high-dimensional non-linear state space models with the embedded HMM method". In: *Bayesian Analysis* 13.3 (2018), pp. 797–822.

[SN18] Alexander Y Shestopaloff and Radford M Neal. "Sampling latent states for high-dimensional non-linear state space models with the embedded HMM method". In: *Bayesian Analysis* 13.3 (2018), pp. 797–822.

[So06] Mike KP So. "Bayesian analysis of nonlinear and non-Gaussian state space models via multiple-try sampling methods". In: *Statistics and Computing* 16.2 (2006), pp. 125–141.

[Sri+14] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. "Dropout: a simple way to prevent neural networks from overfitting". In: *The journal of machine learning research* 15.1 (2014), pp. 1929–1958.

[ST19] Chris Sherlock and Alexandre H. Thiery. "A Discrete Bouncy Particle Sampler". In: *arXiv preprint 1707.05200* (2019).

[SZE17] Jiaming Song, Shengjia Zhao, and Stefano Ermon. "A-NICE-MC: Adversarial training for MCMC". In: *Advances in Neural Information Processing Systems*. 2017, pp. 5140–5150.

[Tan19] Akinori Tanaka. "Discriminator optimal transport". In: *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. 2019, pp. 6816–6826.

[TCV11] Konstantin S. Turitsyn, Michael Chertkov, and Marija Vucelja. "Irreversible Monte Carlo algorithms for efficient sampling". In: *Physica D: Nonlinear Phenomena* 240.4 (2011), pp. 410–414. ISSN: 0167-2789. DOI: https://doi.org/10.1016/j.physd.2010.10.003. URL: http://www.sciencedirect.com/science/article/pii/S0167278910002782.

[Thi+20a] Achille Thin, Nikita Kotelevskii, Christophe Andrieu, Alain Durmus, Eric Moulines, and Maxim Panov. "Nonreversible MCMC from conditional invertible transforms: a complete recipe with convergence guarantees". In: *arXiv preprint arXiv:2012.15550* (2020).

[Thi+20b] Achille Thin, Nikita Kotelevskii, Jean-Stanislas Denain, Leo Grinsztajn, Alain Durmus, Maxim Panov, and Eric Moulines. "MetFlow: A New Efficient Method for Bridging the Gap between Markov Chain Monte Carlo and Variational Inference". In: *arXiv preprint arXiv:2002.12253* (2020).

[Thi+21a] Achille Thin, Yazid Janati El Idrissi, Sylvain Le Corff, Charles Ollion, Eric Moulines, Arnaud Doucet, Alain Durmus, and Christian Robert. "NEO: Non Equilibrium Sampling on the Orbits of a Deterministic Transform". In: *Advances in Neural Information Processing Systems* 34 (2021).

[Thi+21b] Achille Thin, Nikita Kotelevskii, Arnaud Doucet, Alain Durmus, Eric Moulines, and Maxim Panov. "Monte Carlo variational auto-encoders". In: *International Conference on Machine Learning*. PMLR. 2021, pp. 10247–10257.

[Tie94] Luke Tierney. "Markov Chains for Exploring Posterior Distributions". In: *The Annals of Statistics* 22.4 (1994), pp. 1701–1728.

[Tie98] Luke Tierney. "A note on Metropolis-Hastings kernels for general state spaces". In: *Ann. Appl. Probab.* 8.1 (Feb. 1998), pp. 1–9. DOI: 10.1214/aoap/1027961031. URL: https://doi.org/10.1214/aoap/1027961031.

[Tje04] Hakon Tjelmeland. *Using all Metropolis–Hastings proposals to estimate mean values*. Tech. rep. 2004.

[TK10] Surya T Tokdar and Robert E Kass. "Importance sampling: a review". In: *Wiley Interdisciplinary Reviews: Computational Statistics* 2.1 (2010), pp. 54–60.

[TT13] Esteban G Tabak and Cristina V Turner. "A family of nonparametric density estimation algorithms". In: *Communications on Pure and Applied Mathematics* 66.2 (2013), pp. 145–164.

[Tur+19a] Ryan Turner, Jane Hung, Eric Frank, Yunus Saatchi, and Jason Yosinski. "Metropolis–Hastings generative adversarial networks". In: *International Conference on Machine Learning*. PMLR. 2019, pp. 6345–6353.

[Tur+19b] Ryan Turner, Jane Hung, Eric Frank, Yunus Saatchi, and Jason Yosinski. "Metropolis-Hastings generative adversarial networks". In: *International Conference on Machine Learning*. PMLR. 2019, pp. 6345–6353.

[TV10] Esteban G Tabak and Eric Vanden-Eijnden. "Density estimation by dual ascent of the log-likelihood". In: *Communications in Mathematical Sciences* 8.1 (2010), pp. 217–233.

[Vou+18] Athanasios Voulodimos, Nikolaos Doulamis, Anastasios Doulamis, and Eftychios Protopapadakis. "Deep learning for computer vision: A brief review". In: *Computational intelligence and neuroscience* 2018 (2018).

[VVN03] Jakob Verbeek, Nikos Vlassis, and Jan Nunnink. "A variational EM algorithm for large-scale mixture modeling". In: *9th Annual Conference of the Advanced School for Computing and Imaging (ASCI '03)*. Ed. by S. Vassiliades, L.M.J. Florack, J.W.J. Heijnsdijk, and A. van der Steen. Heijen, Netherlands, June 2003, pp. 136–143.

[Wai19] Martin J. Wainwright. *High-Dimensional Statistics: A Non-Asymptotic Viewpoint*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 2019. DOI: 10.1017/9781108627771.

[Wen+20] Florian Wenzel, Kevin Roth, Bastiaan S Veeling, Jakub Świątkowski, Linh Tran, Stephan Mandt, Jasper Snoek, Tim Salimans, Rodolphe Jenatton, and Sebastian Nowozin. "How good is the bayes posterior in deep neural networks really?" In: *arXiv preprint arXiv:2002.02405* (2020).

[WI20] Andrew G Wilson and Pavel Izmailov. "Bayesian deep learning and a probabilistic perspective of generalization". In: *Advances in neural information processing systems* 33 (2020), pp. 4697–4708.

[Wil92] Ronald J Williams. "Simple statistical gradient-following algorithms for connectionist reinforcement learning". In: *Machine Learning* 8.3-4 (1992), pp. 229–256.

[Wir+20] Peter Wirnsberger, Andrew J Ballard, George Papamakarios, Stuart Abercrombie, Sébastien Racanière, Alexander Pritzel, Danilo Jimenez Rezende, and Charles Blundell. "Targeted free energy estimation via learned mappings". In: *The Journal of Chemical Physics* 153.14 (2020), p. 144112.

[WJ+08] Martin J Wainwright, Michael I Jordan, et al. "Graphical models, exponential families, and variational inference". In: *Foundations and Trends® in Machine Learning* 1.1–2 (2008), pp. 1–305.

[WKN20a] Hao Wu, Jonas Köhler, and Frank Noe. "Stochastic Normalizing Flows". In: *Advances in Neural Information Processing Systems*. Vol. 33. 2020.

[WKN20b] Hao Wu, Jonas Köhler, and Frank Noé. "Stochastic Normalizing Flows". In: *Advances in Neural Information Processing Systems* (2020).

[WKN20c] Hao Wu, Jonas Köhler, and Frank Noé. "Stochastic normalizing flows". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 5933–5944.

[WKS16] Christopher Wolf, Maximilian Karl, and Patrick van der Smagt. "Variational inference with Hamiltonian Monte Carlo". In: *arXiv preprint arXiv:1609.08203* (2016).

[WL19] Antoine Wehenkel and Gilles Louppe. "Unconstrained monotonic neural networks". In: *Advances in Neural Information Processing Systems*. 2019, pp. 1543–1553.

[WT11] Max Welling and Yee W Teh. "Bayesian learning via stochastic gradient Langevin dynamics". In: *Proceedings of the 28th international conference on machine learning (ICML-11)*. Citeseer. 2011, pp. 681–688.

[Wu+16] Yuhuai Wu, Yuri Burda, Ruslan Salakhutdinov, and Roger Grosse. "On the quantitative analysis of decoder-based generative models". In: *arXiv preprint arXiv:1611.04273* (2016).

[Xie+18] Jianwen Xie, Yang Lu, Ruiqi Gao, and Ying Nian Wu. "Cooperative learning of energy-based model and latent variable model via MCMC teaching". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 32. 1. 2018.

[YSN21] Jiancheng Yang, Rui Shi, and Bingbing Ni. "Medmnist classification decathlon: A lightweight automl benchmark for medical image analysis". In: *2021 IEEE 18th International Symposium on Biomedical Imaging (ISBI)*. IEEE. 2021, pp. 191–195.

[Zha+19] Ruqi Zhang, Chunyuan Li, Jianyi Zhang, Changyou Chen, and Andrew Gordon Wilson. "Cyclical stochastic gradient MCMC for Bayesian deep learning". In: *arXiv preprint arXiv:1902.03932* (2019).

List of Figures

1.1 Schéma de la différence entre une réseau de neurones classique et un réseau de neurones bayésien. Source : `https://towardsdatascience.com/why-you-should-use-bayesian-neural-network`

1.2 Exemples d'observations des datasets MNIST (à gauche) et Fashion MNIST (à droite). 16

1.3 Schéma de l'inférence variationnelle amortie des Auto Encodeurs Variationnels, [KW19]. 18

1.4 Effet de la non réversibilité sur un algorithme MCMC. 23

1.5 À gauche: Variance de NEO en fonction de la longueur de la trajectoire K sur l'orbite comparé à la variance (en rouge) de l'échantillonneur préférentiel avec $(K+1)$ échantillons en échelle logarithmique (la variance la plus basse est à privilégier). De gauche à droite après: Quatre exemples de trajectoires permettant de calculer l'estimateur de la constante de normalisation selon le réglage du terme de friction γ (de gauche à droite, $\gamma = 0.1, 1, 2$), sur une distribution cible mélange de 4 Gaussiennes. 23

1.6 Sortie consécutive des différentes étapes d'un flot normalisant (en haut) par rapport aux étapes de MetFlow construit avec la même architecture de flot normalisant. A gauche : prior Gaussienne standard, puis l'effet successifs des cinq transformations qui constituent le flot. 24

2.1 MNIST (left) and Fashion MNIST (right) datasets. 32

2.2 Amortized inference scheme, [KW19]. 33

2.3 Effect of the irreversibility of an MCMC sampler. 38

2.4 Left: Variance of NEO as a function of the lengtt K of the trajectory vs Variance of IS estimator with K samples (red) in $\log_{10}$-scale (the lower the better). Second left to right: Four examples of the corresponding trajectories with the same random seed for different values of the friction term of the conformal Hamiltonian γ (from left to right, $\gamma = 0.1, 1, 2$). 39

2.5 Consecutive outputs of normalizing flows (top) or metropolized flows (MetFlow, bottom). Left: prior normal distribution, then successive effect of the 5 transformations. 40

2.6 Comparison of Metropolized flow against the same normalizing flow learnt classically on difficult target distributions in $\mathbb{R}^2$. 41

3.1 Consecutive outputs of normalizing flows. Left: prior normal distribution, then successive effect of the 5 transformations. 50

4.1 From left to right: L2HMC, Lifted irreversible kernel with cIT and MALA transitions, MALA algorithm. L2HMC performs high jump with less cover of the modes, while lifted kernels with cIT and MALA transitions cover effectively the mixture. A classical MALA algorithm struggles to mix. 100

4.2 From left to right: Autocorrelation plot for different implementations of NICE, for a Mixture of Gaussian with diagonal variances of respectively 0.1 and 0.25, and 0.3 and 0.3 101

4.3 From left to right: Autocorrelation plots for different implementations of NICE and L2HMC algorithms on a Strongly Correlated Gaussian. 101

5.1 Left: $E^{\mathbb{1}_{[K]}}_{\mathrm{T}_h}(K) - 1$ vs $E^{\mathrm{IS}}(K) - 1$ (red) in $\log_{10}$-scale as a function of the length of trajectories K (the lower the better). Second left to right: Four examples of orbits with the same random seed for different values of γ (from left to right, $\gamma = 0.1, 1, 2$). 107
5.2 Boxplots of 500 independent estimations of the normalizing constant in dimension $d = \{10, 20, 45\}$ (from left to right) for `MG25` (top) and `Fun` (bottom). The true value is given by the red line. The figure displays the median (solid lines), the interquartile range, and the mean (dashed lines) over the 500 runs. 111
5.3 Empirical 2-D histogram of the samples of different algorithms targeting `MG25` (top) and `Fun` (bottom). Left to right: samples from the target distribution, correlated i-SIR, NUTS, NEO-MCMC. 112
5.4 Two examples for the Gibbs inpainting task for CelebA dataset. From top to bottom (twice) : i-SIR, HMC and NEO-MCMC: From left to right, original image, blurred image to reconstruct, and output every 5 iterations of the Markov chain. Last line: a forward orbit used in NEO-MCMC for the second example. 113
5.5 Boxplots of 500 independent estimations of the normalizing constant of the Cauchy mixture in dimension $d = 10, 15$ (top, bottom). The true value is given by the red line. The figure displays the median (solid lines), the interquartile range, and the mean (dashed lines) over the 500 runs 129
5.6 NEO v. NEIS. 25 GM with $\sigma^2 = 0.005$, $d = 5$. 500 runs each. 129
5.7 Forward orbits of NEO-MCMC. 130
5.8 Additional examples for the Gibbs inpainting task for CelebA dataset. From top to bottom: i-SIR, HMC and NEO-MCMC: From left to right, original image, blurred image to reconstruct, and output every 5 iterations of the Markov chain. 131

6.1 Sampling from $\mathcal{N}(0, \mathrm{Id}_d)$ with the proposal $\mathcal{N}(0, 2\,\mathrm{Id}_d)$. The rightmost plot illustrates the number of rejections rapidly growing for vanilla i-SIR algorithm (see Section 6.11.1 for the definition of ESS). The correlated proposals in Ex^{2}MCMC help to achieve efficient sampling even in high dimensions. We display confidence intervals for i-SIR and Ex^{2}MCMC obtained from 20 independent runs as blue and red regions, respectively. . 138
6.2 Graphical model for the proposals $X^{1:N}$. 140
6.3 Sampling results for the asymmetric banana-shaped distribution. The Sliced TV, ESS and EMD metrics are reported as functions of the dimension of the space. 142
6.4 Asymmetric banana-shaped distribution in dimension 50: projections on first two coordinates.Left: MALA, right: Ex^{2}MCMC. 143
6.5 Allen-Cahn equation. From left to right,trajectories sampled by Augmented MALA, by FlEx^{2}MCMC and autocorrelation plot. 143
6.6 Bayesian logistic regression: average $\hat{p}(y|x, \mathcal{D})$ for (left to right) Covertype, EEG and Digits datasets. 144
6.7 CIFAR-10 dataset with DC-GAN architecture. From left to right, average energy values, FID and discriminator scores for 600 sampling iterations. 144
6.8 CIFAR-10 dataset with SN-GAN architecture: From left to right, average energy values, FID and discriminator scores for first 600 sampling iterations. 148
6.9 Sampling from mixture of two Gaussian distributions $\mathcal{N}(-\mathbf{1.5}, \mathrm{Id})$, $\mathcal{N}(\mathbf{1.5}, \mathrm{Id})$ in high dimensions. 165
6.10 Funnel distribution, dim 15: projection of resulted samples on first two coordinates. From left to right: MALA, Ex^{2}MCMC, FlEx^{2}MCMC 166
6.11 Sampling from Funnel distribution. 166
6.12 Symmetric banana-shaped distribution, dimension $d = 50$: projection of resulted samples on first two coordinates. From left to right: MALA, Ex^{2}MCMC, FlEx^{2}MCMC 166
6.13 Sampling from symmetric banana-shaped distribution. 166
6.14 Ill-conditioned Gaussian: Autocorrelations vs. sampling iteration. 168

8.1 Sampling a mixture of 8 Gaussian distributions. Top row from left to right: Target distribution, MetFlow, MetFlow with 145 resampled innovation noise. Bottom row from left to right: Prior distribution, First run of RNVP, Second run of RNVP. MetFlow finds all the modes and improves with more iterations, while RNVP depend on a good initialization to find all the modes and fails to separate them correctly. 182
8.2 Density matching example [RM15a] and comparison between RNVP and MetFlow. . . 183
8.3 Mixture of '3' digits. Top: Fixed digits, Middle: NAF samples, Bottom: MetFlow samples. Compared to NAF, MetFlow is capable to mix better between these modes, while NAF seems to collapse. 183
8.4 Top line: Mean-Field approximation and MetFlow, Middle line: Mean-Field approximation, Bottom line: Mean-Field Approximation and NAF. Orange samples on the left represent the initialization image. We observe that MetFlow easily mixes between the modes while other methods are stuck in one mode. 184
8.5 Consecutive outputs of each MetFlow kernel. Left: prior normal distribution, then successive effect of the 5 trained MetFlow kernels. 193
8.6 Consecutive outputs of each block of R-NVP. Left: prior normal distribution, then successive effect of the 5 trained R-NVP blocks - 6 transforms each. 193
8.7 Changing the prior to a mixture of two separated Gaussians, having trained the method on a standard normal prior. Top row, from left to right: Subsituted prior, 5 trained MetFlow kernels, re-iteration of 100 MetFlow kernels, 200 MetFlow kernels. Bottom row: Substituted prior, R-NVP flow. 194
8.8 Number of modes retrieved by different methods. The target distribution is a mixture of 8 isotropic Gaussian distributions of variance 1 located at the corners of a d-dimensional hypercube. The methods were trained in a way to use the same computational budget, and mode retrieval is computed by counting the number of samples in a ball of radius $2\sqrt{d}$ around the center of a mode. Error bars represent the standard deviation of the mean number of modes retrieved for different runs of the method (different initialization and random seed). 195
8.9 The figure demonstrates, that all these methods used approximately the same computational budget. 196
8.10 Density matching for funnel. Top row: Target distribution, MetFlow with 5 trained kernels, MetFlow with 5 trained kernel iterated 100 times. Bottom row: Prior distribution, First run of 5 R-NVP, second run of 5 R-NVP 197
8.11 Fixed digits for mixture experiment. 198
8.12 Mixture of 3, MetFlow approximation. 199
8.13 Mixture of 3, NAF approximation. 199
8.14 Gibbs inpainting experiments starting from digit 0. 199
8.15 Gibbs inpainting experiments starting from digit 3. 200
8.16 Gibbs inpainting experiments starting from digit 9. 200
8.17 Gibbs inpainting experiments starting from digit 6. 200
8.18 Gibbs inpainting experiments starting from digit 4. 200
8.19 Comparison of the different settings described for a mixture of digits experiment. From left to right, deterministic setting, pseudo-random setting, fully random setting. 201

9.1 Visualization of the posterior approximation given after optimization of different bounds for toy generation process. Top row, from left to right: True posterior, VAE posterior, IWAE posterior. Bottom row, from left to right: VI with RealNVP posterior, A-MCVAE posterior, L-MCVAE posterior. 212
9.2 Representation of the different estimators (top) and their gradient (bottom) of the true log likelihood. From left to right, a/ L-MCVAE, $K = 5$, b/ L-MCVAE, $K = 10$, c/ A-MCVAE, $K = 5$, d/ A-MCVAE, $K = 10$, e/ A-MCVAE, $K = 5$ with control variates. 213

9.3 Log-likelihood of L-MCVAE depending on the number of Langevin steps K. Increasing K improves performance, however at the expense of the computational complexity. . . 214
9.4 Evolution of the held-out loglikelihood during training for A-MCVAE, L-MCVAE, IWAE and VAE on MNIST. 215
S5 Squared error for parameter's estimates, obtained using different models. 219
S6 Representation of the different estimators (left) and their gradient (right) of the true log likelihood. From left to right, a/ L-MCVAE, $K = 5$, b/ L-MCVAE, $K = 10$, c/ L-MCVAE, $K = 1$, learnable reverse, d/ L-MCVAE, $K = 2$ learnable reverse, e/ A-MCVAE, $K = 5$, f/ A-MCVAE, $K = 10$, g/ A-MCVAE, $K = 5$ with control variates. 219

List of Tables

4.1 Mixture of Gaussians with diagonal covariances 0.1 and 0.25 in dimension 10 100
4.2 Mixture of Gaussians with diagonal covariances 0.3 and 0.3 in dimension 10 100
4.3 Strongly Correlated Gaussian in dimension 2 101

5.1 Evaluation of the log-likelihood (normalizing constant) of different Variational Auto Encoders. 131
5.2 Negative Log Likelihood estimates for VAE models for different latent space dimensions. 133

6.1 GAN sampling from 243 Gaussians 146
6.2 Results for Swiss Roll dataset 146
6.3 CIFAR-10 hyperparameters for DC-GAN architecture. 147
6.4 CIFAR-10 hyperparameters for SN-GAN architecture. 148
6.5 FID for CIFAR-10 GAN-based models 149
6.6 Hyperparameters used in experiments. 162
6.7 GAN sampling from mixture of Gaussian distributions 168

9.1 Results of the different models on MNIST. A more detailed version of this table is included in the supplementary material. 214
9.2 Results of the different models on CelebA. A more detailed version of this table is included in the supplementary material. 11400 must be added to all scores in this table. 215
9.3 Results of the different models on CIFAR. A more detailed version of this table is included in the supplementary material. 2800 must be added to all scores in this table. 216
9.4 Results of the different models on MNIST with different annealing schemes. 220
9.5 Full results of the different models on CelebA. All scores must be added 11400 in this table. 221
9.6 Results of the different models on CIFAR-10 with different annealing schemes. All scores must be added 2800 in this table. 222

Appendix A

Appendix and supplementary material

A.1 Notations, definitions and general Markov chain theory

In this section, we recall some basic facts and notations in a form that is useful for establishing properties of Markov chains. Let $(\mathbb{Z}, \mathcal{Z})$ be a measurable space where $\mathcal{Z}$ is a countably generated σ-algebra.

Definition 73 (Kernel)**.** *A* kernel *on $\mathbb{Z} \times \mathcal{Z}$ is a map $P\colon \mathbb{Z} \times \mathcal{Z} \to \mathbb{R}_+$ such that*

(i) for any $A \in \mathcal{Z}$, $z \mapsto P(z, A)$ is measurable;

(ii) for any $z \in \mathbb{Z}$, the function $A \mapsto P(z, A)$ is a finite measure on $\mathcal{Z}$.

Definition 74 (Markov and sub-Markovian kernel)**.** *A kernel P is Markovian (or P is a Markov kernel) if $P(z, \mathbb{Z}) = 1$ for all $z \in \mathbb{Z}$. A kernel P is submarkovian (or P is a sub-Markov kernel) if $P(z, \mathbb{Z}) \leqslant 1$ for all $z \in \mathbb{Z}$.*

For $f\colon \mathbb{Z} \to \mathbb{R}$ a measurable function, ν a probability distribution, and P a kernel on $\mathbb{Z} \times \mathcal{Z}$, we let $\nu(f) \stackrel{:=}{\int} f(z)\nu(\mathrm{d}z)$ and denote for $(z, A) \in \mathbb{Z} \times \mathcal{Z}$,

$$\nu P(A) = \int \nu(\mathrm{d}z) P(z, A) \;, \quad Pf(z) = \int P(z, \mathrm{d}z') f(z') \;.$$

Further, for $(z, A) \in \mathbb{Z} \times \mathcal{Z}$ define recursively for $n \geqslant 2$: $P^n(z, A) = \int P^{n-1}(z, \mathrm{d}z') P(z', A)$.

Definition 75 (Total variation distance)**.** *For μ, ν two probability distributions on $(\mathbb{Z}, \mathcal{Z})$ we define the total variation distance between μ and ν by $\|\mu - \nu\|_{\mathrm{TV}} := \sup_{|f| \leqslant 1} |\mu(f) - \nu(f)|$, where the supremum is taken over the measurable function $f\colon \mathbb{Z} \to \mathbb{R}$.*

Definition 76 (Harmonic function)**.** *Let P be a kernel on $(\mathbb{Z}, \mathcal{Z})$. Then a non-negative measurable function $h : \mathbb{Z} \to \mathbb{R}$ is said to be* harmonic *if $Ph = h$.*

Definition 77 (Irreducibility)**.** *Let ν be a non trivial σ-finite measure on $\big(\mathbb{Z}, \mathcal{Z}\big)$. A kernel P is said to be ν-irreducible if for all $(z, A) \in \mathbb{Z} \times \mathcal{Z}$ such that $\nu(A) > 0$ there exists $n = n(z, A) \in \mathbb{N}$ such that $P^n(z, A) > 0$.*

Definition 78 (Periodicity and Aperiodicity)**.** *P is periodic if there exists $n \in \mathbb{N}$, $n \geqslant 2$, and $A_i \in \mathcal{Z}$ for $i \in 1, \ldots, n$, non-empty and disjoint, such that for $z \in A_i$, $P(z, A_{i+1}) = 1$ with the convention $A_{n+1} = A_1$. Aperiodicity is the negation of periodicity.*

General Markov chain theory provides us with powerful tools to establish validity and convergence of MCMC algorithms, leading to basic convergence theorems such as those found in [Tie94, Theorem 1 and 3] and distilled below. We informally comment on the result below.

Theorem S79 ([Tie94])**.** *Suppose P is such that $\pi P = P$ and is $\pi-$irreducible. Then π is the unique invariant probability distribution of P and for any $f\colon \mathbb{Z} \to \mathbb{R}$ such that $\pi(|f|) < \infty$*

$$\lim_{n\to\infty} n^{-1} \sum_{i=1}^{n} f(Z_i) = \pi(f) \;, \tag{S1}$$

almost surely for $\pi-$almost all $z \in \mathbb{Z}$. If in addition P is aperiodic then for $\pi-$almost all $z \in \mathbb{Z}$

$$\lim_{n\to\infty} \|P^n(z,\cdot) - \pi(\cdot)\|_{\mathrm{TV}} = 0 \;. \tag{S2}$$

The result is fairly intuitive. Invariance of π is a fixed point property ensuring that if $Z_i \sim \pi$ then $Z_{i+1} \sim \pi$. $\pi-$irreducibility simply says that the Markov chain should be able to reach any set of $\pi-$positive probability from any $z \in \mathbb{Z}$ in a finite number of iterations. Periodicity would clearly prevent (S2) since the Markov chain would then periodically avoid visiting sets of positive $\pi-$probability. Averaging in (S1) removes the need for this property. We note that establishing these properties is often overlooked and a necessary prerequisite to any more refined analysis characterising their performance, such as quantitative finite time convergence bounds as found for example in [Dal17; DK19; DM17].

A.1.1 Proof for the AIS estimator

Note that if we suppose that γ_t has a density on $\mathbb{R}^d$ also denoted γ_t for $0 \leqslant t \leqslant K+1$, and that for $x \in \mathbb{R}^d$, $\gamma_t(x) > 0 \implies \gamma_{t+1}(x) > 0$, then we can define the Radon-Nikodym derivative $\mathrm{d}\gamma_{t+1}/\mathrm{d}\gamma_t = \gamma_{t+1}/\gamma_t$. However, we only require the condition $\gamma_t \ll \gamma_{t+1}$ in the following.

Define the extended proposal distribution

$$\mathrm{Q}(\mathrm{d}x_{0:K}) = q(\mathrm{d}x_0) \prod_{t=1}^{K} M_t(x_{t-1}, \mathrm{d}x_t) \;, \tag{S3}$$

and similarly, the extended target unnormalized distribution (with normalizing constant Z)

$$\mathrm{P}(\mathrm{d}x_{K:0}) = \tilde{\pi}(\mathrm{d}x_K) \prod_{t=K}^{1} M_t(x_t, \mathrm{d}x_{t-1}) \tag{S4}$$

Lemma S80. *By [And+18], we have,*

$$\prod_{t=0}^{K} \frac{\mathrm{d}\gamma_{t+1}}{\mathrm{d}\gamma_t}(x_t) \mathrm{Q}(\mathrm{d}x_{0:K}) = \mathrm{P}(\mathrm{d}x_{K:0}) \;. \tag{S5}$$

We can thus write the estimator, sampling for $1 \leqslant i \leqslant n$, $x^{(i)} \sim \mathrm{Q}$,

$$\hat{\mathrm{Z}}_{\mathrm{AIS}}(x_{1:n}) = \frac{1}{n} \sum_{i=1}^{n} \left[\prod_{t=0}^{K} \frac{\mathrm{d}\gamma_{t+1}}{\mathrm{d}\gamma_t}(x_t^{(i)}) \right] \;. \tag{S6}$$

Proof. The proof is given in [And+18] and goes as follows. Note, for $1 \leqslant t \leqslant K$, $\mathrm{Z}_t = \int \gamma_t$ the normalizing constant of the distribution γ_t, and note $\mu_t = \gamma_t / \mathrm{Z}_t$ and $\mu = \mu_{K+1} = \gamma_{K+1} / \mathrm{Z}_{K+1} = \gamma / \mathrm{Z}$. By convention, $\prod_{t=j}^{K} = I$ for $j > K$.

The proof is direct if $K = 0$. Suppose now $K \geqslant 1$. We show by induction for $1 \leqslant j \leqslant K$ that

$$\frac{1}{\mathrm{Z}} \mathrm{P}(\mathrm{d}x_{K:0}) \tag{S7}$$

$$= \left[\prod_{t=1}^{j} \frac{\mathrm{d}\mu_{K-t+2}}{\mathrm{d}\mu_{K-t+1}}(\mathrm{d}x_{K-t+1}) M_{K-t+1}(x_{K-t}, \mathrm{d}x_{K-t+1}) \right] \mu_{K-j+1}(\mathrm{d}x_{K-j}) \prod_{t=j+1}^{K} M_{K-t+1}(x_{K-t+1}, \mathrm{d}x_{K-t}) \;. \tag{S8}$$

For $j = 1$, we have

$$\frac{1}{\mathrm{Z}}\mathrm{P}(\mathrm{d}x_{K:0}) = \mu(\mathrm{d}x_K) \prod_{t=K}^{1} M_t(x_t, \mathrm{d}x_{t-1}) \tag{S9}$$

$$= \frac{\mathrm{d}\mu_{K+1}}{\mathrm{d}\mu_K}(\mathrm{d}x_K)\mu_K(\mathrm{d}x_{K-1})M_K(x_{K-1}, \mathrm{d}x_K) \prod_{t=K-1}^{1} M_t(x_t, \mathrm{d}x_{t-1}) \tag{S10}$$

as M_K is γ_K-reversible. Assume now the result holds for some $1 \leqslant j \leqslant K$. Then we write

$$\mu_{K-j+1}(\mathrm{d}x_{K-j})M_{K-j}(x_{K-j}, \mathrm{d}x_{K-j-1}) = \frac{\mathrm{d}\mu_{K-j+1}}{\mu_{K-j}}(\mathrm{d}x_{K-j})\mu_{K-j}(\mathrm{d}x_{K-j})M_{K-j}(x_{K-j}, \mathrm{d}x_{K-j-1}) \tag{S11}$$

$$= \frac{\mathrm{d}\mu_{K-j+1}}{\mu_{K-j}}(\mathrm{d}x_{K-j})\mu_{K-j}(\mathrm{d}x_{K-j-1})M_{K-j}(x_{K-j-1}, \mathrm{d}x_{K-j}) \,. \tag{S12}$$

This identity concludes the recursion. Thus, for $j = K$, we write

$$\frac{1}{\mathrm{Z}}\mathrm{P}(\mathrm{d}x_{K:0}) = \prod_{t=0}^{K} \frac{\mathrm{d}\mu_{t+1}}{\mathrm{d}\mu_t}(x_t)\mathrm{Q}(\mathrm{d}x_{0:K}) \tag{S13}$$

$$= \prod_{t=0}^{K} \frac{\mathrm{Z}_t}{\mathrm{Z}_{t+1}} \frac{\mathrm{d}\gamma_{t+1}}{\mathrm{d}\gamma_t}(x_t)\mathrm{Q}(\mathrm{d}x_{0:K}) \tag{S14}$$

$$= \frac{1}{\mathrm{Z}} \prod_{t=0}^{K} \frac{\mathrm{d}\gamma_{t+1}}{\mathrm{d}\gamma_t}(x_t)\mathrm{Q}(\mathrm{d}x_{0:K}) \,, \tag{S15}$$

as $\mathrm{Z}_0 = 1$ as we assume that m is normalized. Finally, the estimator is the Monte Carlo estimator of $\int \mathrm{P}(\mathrm{d}x_{K:0})$. □

ECOLE
DOCTORALE
DE MATHEMATIQUES
HADAMARD

Titre : Nouvelles approches variationnelles pour l'inférence et l'apprentissage

Mots clés : Variationnel, Inférence, Apprentissage, Monte Carlo, chaînes de Markov

Résumé : Cette thèse porte sur le problème de l'inférence en grande dimension. Nous proposons différentes méthodes pour l'estimation de constantes de normalisation et l'échantiollonnage de distributions complexes. Dans une première partie, nous développons plusieurs méthodes de Monte Carlo par chaînes de Markov. D'une part, nous développons une nouvelle approche pour des noyaux non-réversibles. D'autre part, nous proposons deux méthodes massivement parallélisables combinant des propriétés locales et globales des méthodes de Monte Carlo par chaînes de Markov, en particulier en se basant sur un nouvel estimateur de constante de normalisation. Nous appliquons ces méthodes à une tâche d'inférence approchée de distribution *a posteriori* de réseaux de neurones bayésiens profonds, dans un cas où l'espace d'état est à très haute dimension. Dans une deuxième partie, nous proposons deux modèles génératifs, basés sur une nouvelle forme de flots normalisants combinés à des chaînes de Markov, ou à de nouvelles méthodes d'inférence variationnelles, en construisant en particulier un nouvel auto encodeur variationnel. Ces méthodes permettent en particulier de combiner inférence variationnelle et Monte Carlo par chaînes de Markov.

Title : Novel Variational Approaches to Inference and Learning

Keywords : Variational, Inference, Learning, Monte Carlo, Markov chain

Abstract : This thesis addresses the problem of high dimensional inference. We propose different methods for estimating normalizing constants and sampling complex distributions. In a first part, we develop several Markov chain Monte Carlo methods. On the one hand, we develop a new approach for non-reversible kernels. On the other hand, we propose two massively parallelizable methods combining local and global properties of Markov chain Monte Carlo methods, in particular based on a new normalization constant estimator. We apply these methods to the approximate inference of the posterior distribution of deep Bayesian neural networks, in a case where the state space is very high dimensional. In a second part, we propose two generative models, based on a new form of normalising flows combined with Markov chains, or new variational inference methods, by building in particular a new variational autoencoder. These methods allow in particular to combine variational inference and Monte Carlo by Markov chains.

Institut Polytechnique de Paris
91120 Palaiseau, France