Hamiltonian Neural Flows[1]

Gaspard Beugnot

March 17th, 2020

[1] Hamiltonian Generative Networks, Thoth *et al.*

Introduction

- À partir d'une séquence vidéo, apprendre l'Hamiltonien d'un système
  - État = position + variable latente
  - Position = image
  - Permet déroulement temporel en avant, en arrière, accéléré
- Adapté au problème d'évaluation de densité:
  - État = échantillon + variable latente
  - Entraînement sur des échantillons tirés d'une loi
  - Permet de transformer une loi simple en une loi complexe (Variational Inference with Normalizing Flows)

Implémentation sur:
github.com/gaspardbe/HamiltonianGenerativeNetworks.

État de l'art

Inférence variationnelle

- Décomposition variables observées / variables latentes
- Approximer la postérieure des variables latentes par une fonction paramétrique
- Utiliser Jensen, optimiser l'approximation de la postérieure

$$
\begin{aligned}
\log p_\theta(\boldsymbol{x}) &= \log \int_{\mathcal{Z}} p_\theta(\boldsymbol{x}|z)p(z)\mathrm{d}z \\
&= \log \int \mathcal{Z} \frac{q_\phi(z|\boldsymbol{x})}{q_\phi(z|\boldsymbol{x})} p_\theta(\boldsymbol{x}|z)p(z)\mathrm{d}z \\
&\geq -\mathbb{D}_{KL}\left[q_\phi(z|\boldsymbol{x})|p(z)\right] + \mathbb{E}_q\left[\log p_\theta(\boldsymbol{x}|z)\right] \\
&= \mathcal{F}(\boldsymbol{x})
\end{aligned}
$$

(θ: paramètres du modèle; ϕ: paramètres du réseau)

Normalizing Flows (Rezende *et al.*)

- A priori: $\pi(\cdot)$

Normalizing Flows (Rezende *et al.*)

- A priori: $\pi(\cdot)$
- Fonction inversible $f(\cdot)$

Normalizing Flows (Rezende *et al.*)

- A priori: $\pi(\cdot)$
- Fonction inversible $f(\cdot)$
- Loi objectif: $p(\boldsymbol{x}) = \pi\left(f^{-1}(x)\right)\left|\det \frac{\partial f^{-1}}{\partial \boldsymbol{x}}\right|$

Normalizing Flows (Rezende *et al.*)

- A priori: $\pi(\cdot)$
- Fonction inversible $f(\cdot)$
- Loi objectif: $p(\boldsymbol{x}) = \pi\left(f^{-1}(x)\right)\left|\det \frac{\partial f^{-1}}{\partial \boldsymbol{x}}\right|$

Problème: comment calculer le Jacobien efficacement?

Variational inference with Normalizing Flows

- Planar Flows
  - Transformation de type:

$$f(\boldsymbol{z}) = \boldsymbol{z} + \boldsymbol{u}h(\boldsymbol{w}^\top \boldsymbol{z} + b)$$

  - Ensemble de contractions dilatations orthogonalement à $\boldsymbol{w}$.
  - Déterminant calculable en temps linéaire par rapport au nombre de dimensions.

Variational inference with Normalizing Flows

- Planar Flows
  - Transformation de type:

$$f(\boldsymbol{z}) = \boldsymbol{z} + \boldsymbol{u}h(\boldsymbol{w}^\top \boldsymbol{z} + b)$$

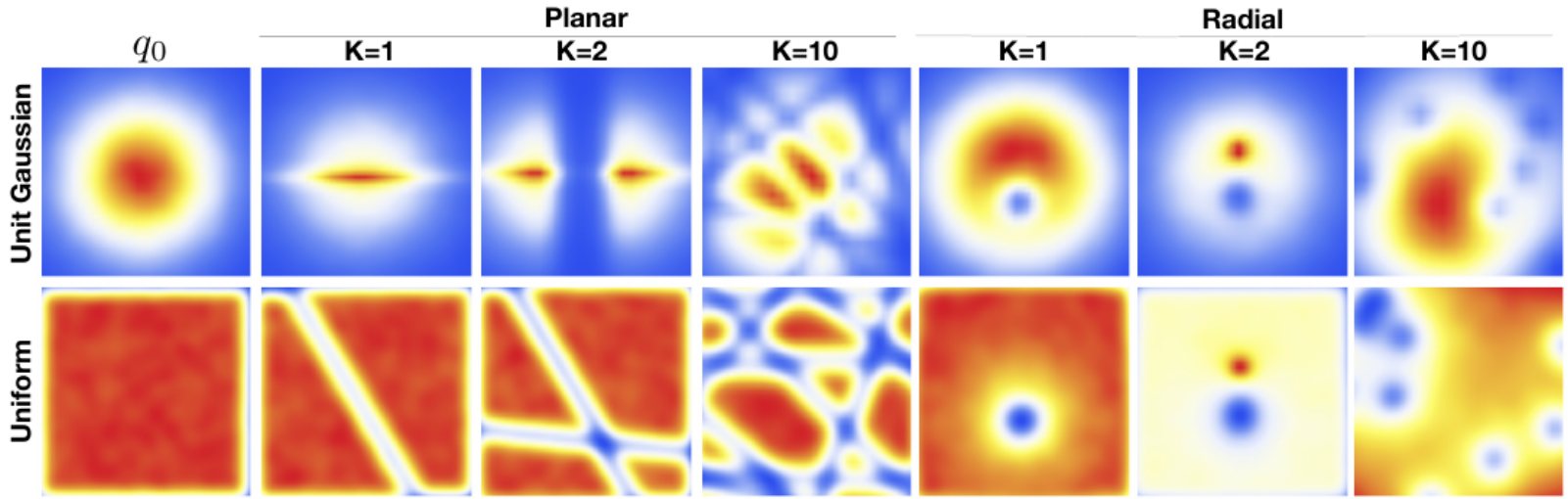
  - Ensemble de contractions dilatations orthogonalement à $\boldsymbol{w}$.
  - Déterminant calculable en temps linéaire par rapport au nombre de dimensions.
- Radial flows
  - Transformation de type:

$$f(\boldsymbol{z}) = \boldsymbol{z} + \beta \frac{\boldsymbol{z} - \boldsymbol{z}_0}{\alpha + \|\boldsymbol{z} - \boldsymbol{z}_0\|}$$

  - Contraction radiale par rapport à un point de référence $\boldsymbol{z}_0$.
  - Déterminant calculable en temps linéaire à nouveau.

Expressivité

Des flots qui restent expressifs (gaussienne en distribution initiale).



Hamiltonian Monte Carlo

Principe (Neal)

- MCMC = générer une chaîne de Markov avec loi a posteriori pour loi invariante
- Énergie :

$$H(q,p) = \frac{1}{Z}\exp\left(-U(q)\right)\exp\left(-K(p)\right)$$

avec $U(q) = -\log\left(\pi(q)p(q|\theta)\right)$.

- Si $K(p) \propto p^\top Mp$, $\iff$ p variable latente tirée suivant une gaussienne.

Hamiltonian Monte Carlo

Principe (Neal)

- MCMC = générer une chaîne de Markov avec loi a posteriori pour loi invariante
- Énergie :

$$H(q,p) = \frac{1}{Z}\exp\left(-U(q)\right)\exp\left(-K(p)\right)$$

  avec $U(q) = -\log\left(\pi(q)p(q|\theta)\right)$.
- Si $K(p) \propto p^\top Mp$, $\iff$ p variable latente tirée suivant une gaussienne.

Résultats:

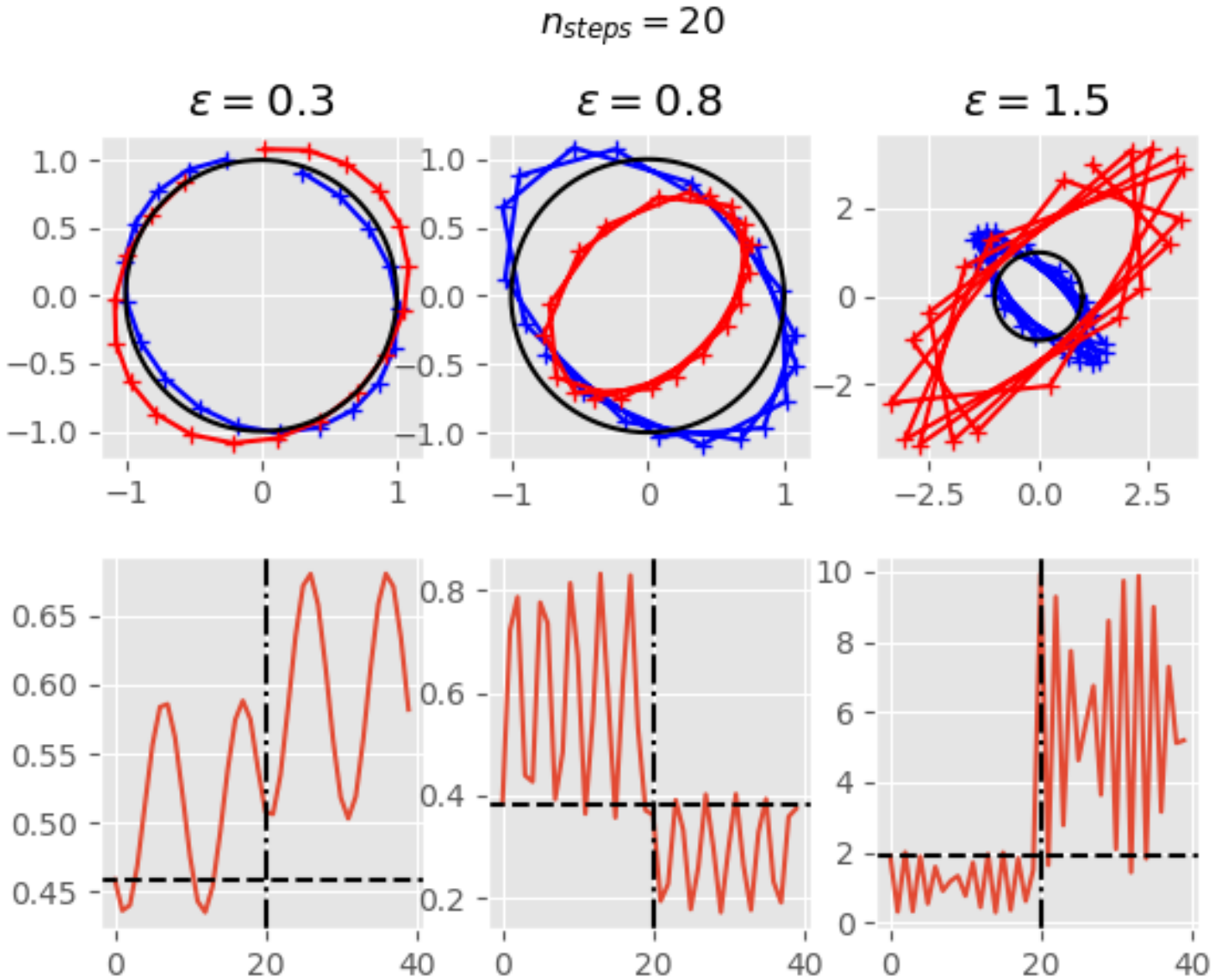
- déplacements sur des isodensités pour H, peut explorer différents modes pour q.
- Réversibilité du Hamiltonien donne automatiquement la *detailed balance equation*.
- Nécessite de choisir judicieusement les paramètres d'intégration.

HMC with NN

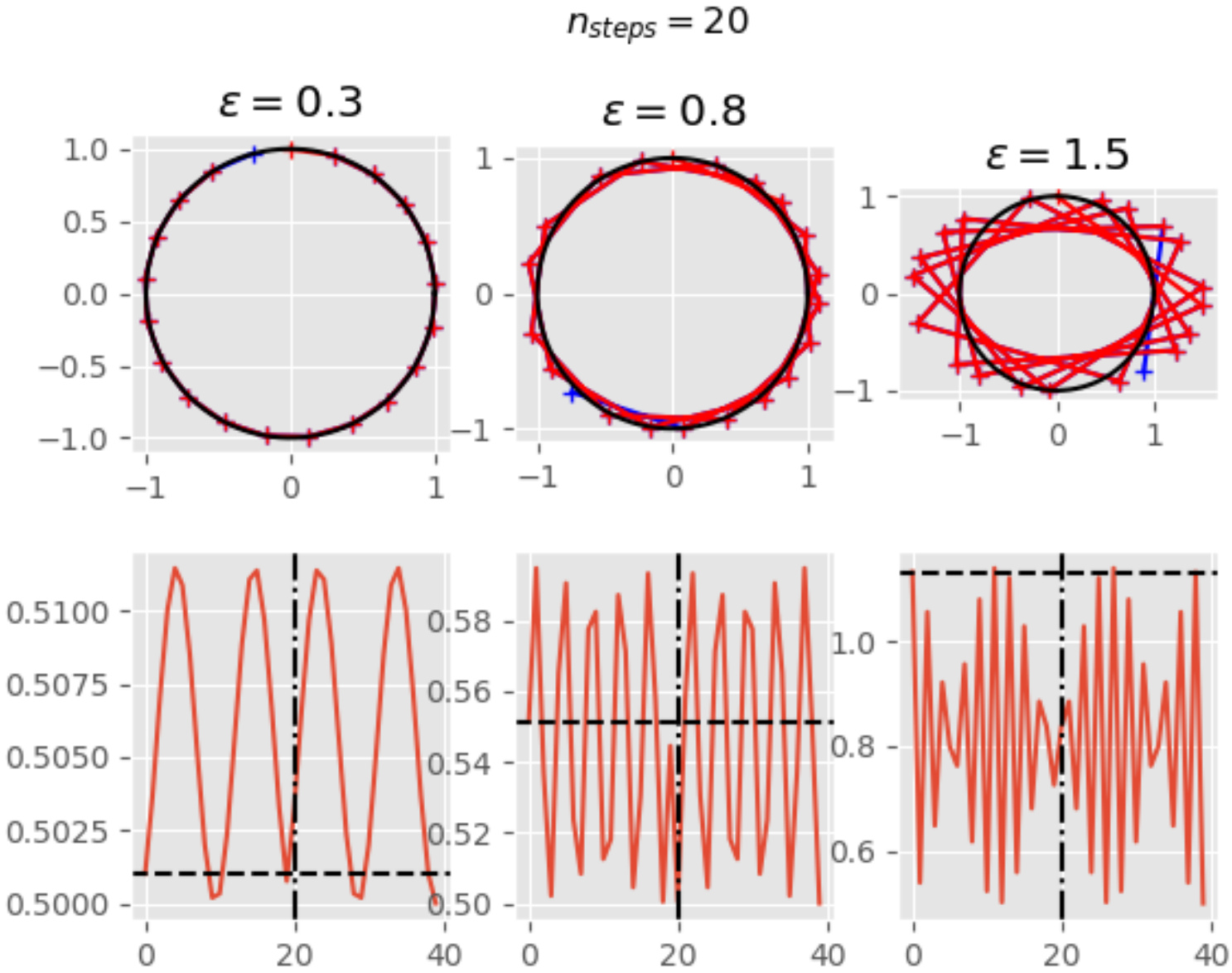
Principe (Levy):

- Utiliser des opérateurs d'échelles sur la mise à jour des états (translations, dilatation sur l'état et sur le gradient).
- Coût doit:
  - maximiser le mixing (maximise la taille des sauts, ie. distance entre les états)
  - encourage un burn-in rapide (atteindre l'état stationnaire rapidement)
- Convergence vers la loi invariante garantie par la réversibilité

Schémas numériques pour l'intégration - Euler



Schémas numériques pour l'intégration - Leapfrog



Neural Hamiltonian Flow

Principe:

- Utiliser T hamiltoniens pour modéliser un flot qui emmène d'un a priori simple à une fonction complexe.
  - **Préserve le volume** : pas de déterminant à calculer
  - **Réflexif** : inverse naturel et facile à calculer
- État $= \boldsymbol{s}_0, ..., \boldsymbol{s}_T$ avec $\boldsymbol{s}_i = \boldsymbol{q}_i, \boldsymbol{p}_i$. Moment $=$ **variable latente**:

$$
\begin{aligned}
p(\boldsymbol{q}_T) &= \int p(\boldsymbol{q}_T, \boldsymbol{p}_T) \mathrm{d}\boldsymbol{p}_T \\
&= \int \pi(\mathcal{H}_1^{-\mathrm{d}t} \circ \cdots \circ \mathcal{H}_T^{-\mathrm{d}t}(\boldsymbol{q}_T, \boldsymbol{p}_T)) \mathrm{d}\boldsymbol{p}_T
\end{aligned}
$$

Impossible d'explorer tout l'espace des $\boldsymbol{p}$: $\rightarrow$ encoder le moment depuis la position.

ELBO

On approche la probabilité par une fonction paramétrique:

$$
\begin{aligned}
\log p(\boldsymbol{q}_T) &= \log \int p(\boldsymbol{q}_T, \boldsymbol{p}_T)\, \mathrm{d}\boldsymbol{p}_T \\
&= \log \int \frac{p(\boldsymbol{q}_T, \boldsymbol{p}_T)}{f_\psi(\boldsymbol{p}_T|\boldsymbol{q}_T)} f_\psi(\boldsymbol{p}_T|\boldsymbol{q}_T)\, d\boldsymbol{p}_T \\
&\geq \mathbb{E}_{f_\psi(\boldsymbol{p}_T|\boldsymbol{q}_T)}[\log\ \pi(\mathcal{H}_1^{-dt} \circ \ldots \circ \mathcal{H}_T^{-dt}(\boldsymbol{q}_T, \boldsymbol{p}_T)) \\
&- \log\ f_\psi(\boldsymbol{p}_T|\boldsymbol{q}_T)\ ] \\
&= \mathrm{ELBO}(\boldsymbol{q}_T)
\end{aligned}
$$

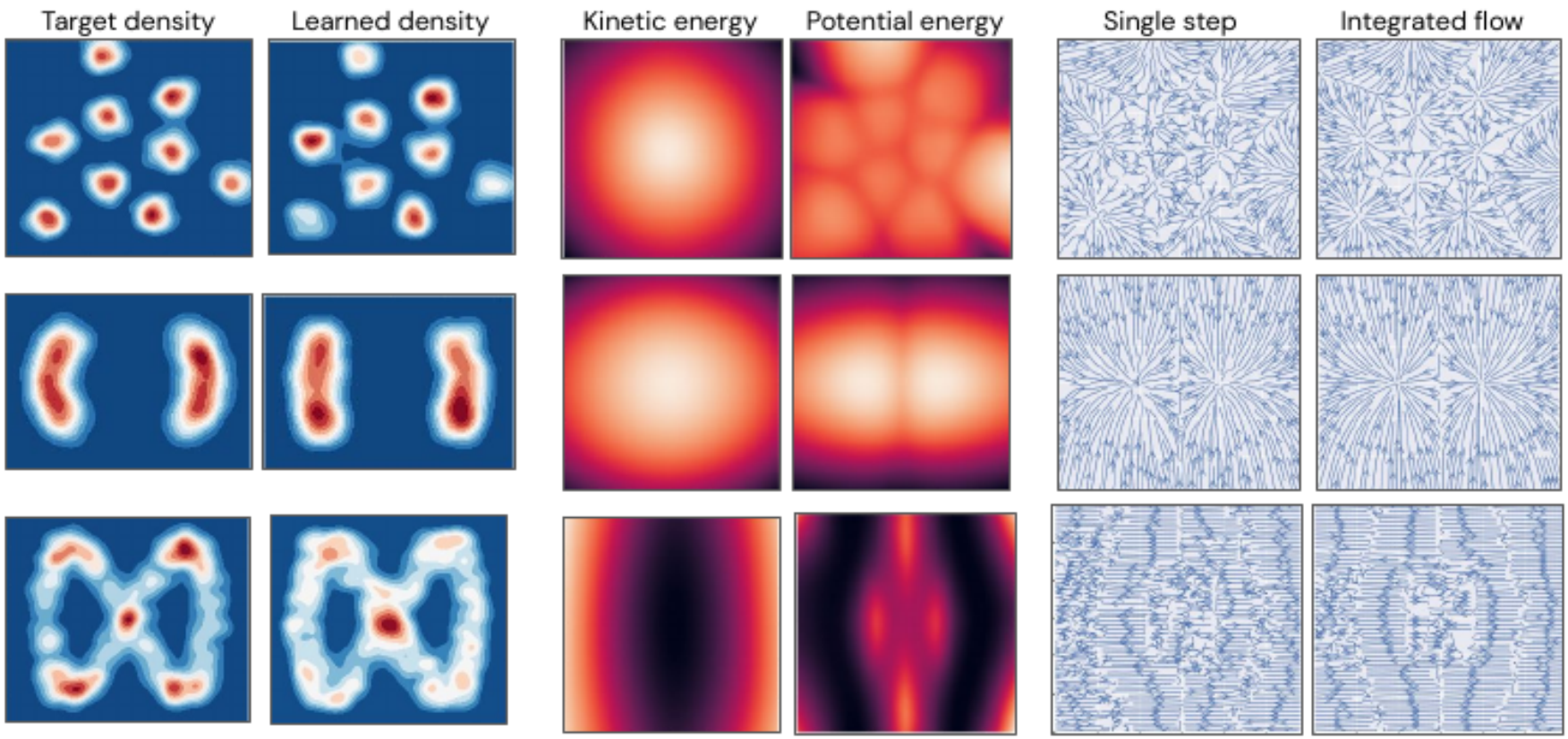
On optimise cette borne inf par rapport à ψ et les T hamiltoniens.

Détails Techniques

Plusieurs modules:

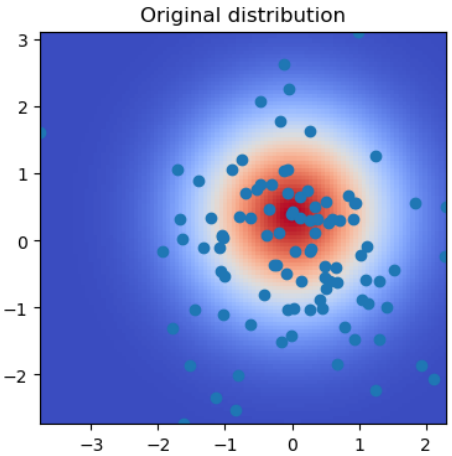
- Hamiltonien:
  - 2 réseaux de neurones (MLP): $U(\boldsymbol{q})$ (potentiel) et $K(\boldsymbol{p})$ cinétique
  - Une fonction d'intégration: $\boldsymbol{q}, \boldsymbol{p} \mapsto \boldsymbol{q}', \boldsymbol{p}'$.
- Un encodeur:
  - 2 réseaux de neurones (MLP): $\boldsymbol{\mu}, \boldsymbol{\sigma} : \mathbb{R}^d \to \mathbb{R}^d$.
  - Échantillonne selon une gaussienne de loi $\boldsymbol{\mu}, \boldsymbol{\sigma}$.
- Le flot:
  - Définit une a priori
  - Implémente les fonctions de coûts
  - Optimise les paramètres par rétropropagation.

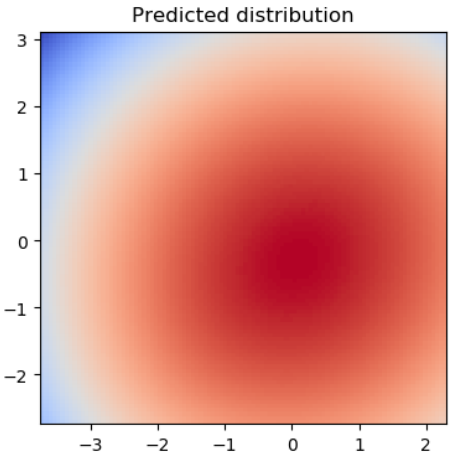
Objectif

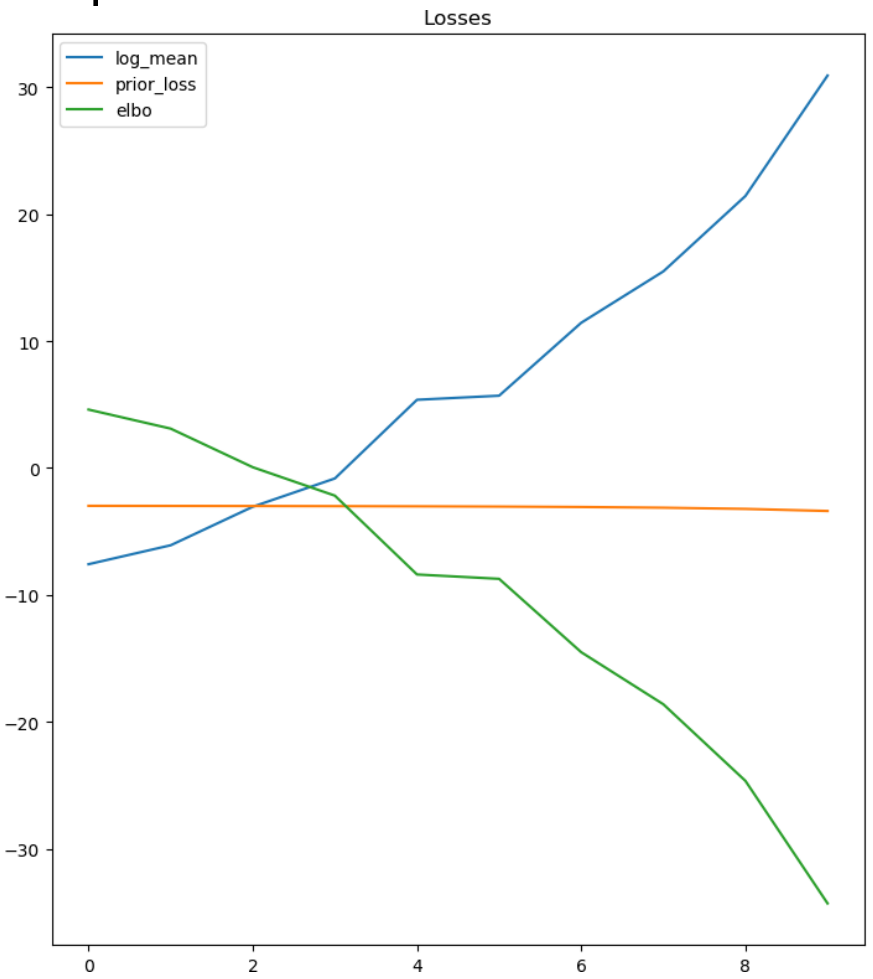

Target density
Learned density
Kinetic energy
Potential energy
Single step
Integrated flow

Résultats numériques

Instabilités numériques...







Prolongements

Beaucoup de questions en suspend:

- Quelle est l'utilité de plusieurs hamiltoniens vs. 1 hamiltonien complexe?
- Comment procéder à l'échantillonnage ? *First, we assume that the initial state* s_0 *is a sample from a simple prior* $s_0 \sim \pi_0(\cdot)$.
  - On ne peut pas explorer tout l'espace des moments
  - Entraîner un **nouvel** encodeur ?
- Notion de dynamique:
  - Peu de pas d'intégration...
  - Interprétation simple $\implies$ overkill? Échantillons = bassin d'attraction?