Sampling with Kernelized Wasserstein Gradient Flows

Anna Korba
CREST, ENSAE, Institut Polytechnique de Paris

IMSI Applied Optimal Transport Workshop

Outline

Sampling

Problem: Sample (=generate new examples) from a target distribution π over $\mathbb{R}^d$, whose density w.r.t. Lebesgue measure is known up to an intractable normalisation constant Z :

$$\pi(\theta) = \frac{\tilde{\pi}(\theta)}{Z}, \quad \tilde{\pi} \text{ known, } Z \text{ unknown.}$$

Main application: Bayesian inference, where π is the posterior distribution over parameters of a model.

Bayesian inference

Let $\mathcal{D} = (w_i, y_i)_{i=1}^m$ a dataset of labelled examples $(w_i, y_i) \overset{i.i.d.}{\sim} P_{\text{data}}$.
Assume an underlying model parametrized by θ, e.g. :

$$y = g(w, \theta) + \epsilon, \quad \epsilon \sim \mathcal{N}(0, I)$$

Goal: learn the best distribution over θ to fit the data.

Bayesian inference

Let $\mathcal{D} = (w_i, y_i)_{i=1}^m$ a dataset of labelled examples $(w_i, y_i) \overset{i.i.d.}{\sim} P_{\text{data}}$.
Assume an underlying model parametrized by θ, e.g. :

$$y = g(w, \theta) + \epsilon, \quad \epsilon \sim \mathcal{N}(0, I)$$

Goal: learn the best distribution over θ to fit the data.

1. Compute the Likelihood:

$$p(\mathcal{D}|\theta) = \prod_{i=1}^{m} p(y_i|\theta, w_i) \propto \exp\left(-\frac{1}{2} \sum_{i=1}^{m} \|y_i - g(w_i, \theta)\|^2 \right).$$

Bayesian inference

Let $\mathcal{D} = (w_i, y_i)_{i=1}^m$ a dataset of labelled examples $(w_i, y_i) \overset{i.i.d.}{\sim} P_{\text{data}}$.
Assume an underlying model parametrized by θ, e.g. :

$$y = g(w, \theta) + \epsilon, \quad \epsilon \sim \mathcal{N}(0, I)$$

Goal: learn the best distribution over θ to fit the data.

1. Compute the Likelihood:

$$p(\mathcal{D}|\theta) = \prod_{i=1}^{m} p(y_i|\theta, w_i) \propto \exp\left(-\frac{1}{2}\sum_{i=1}^{m} \|y_i - g(w_i, \theta)\|^2\right).$$

2. Choose a prior distribution on the parameter:

$$\theta \sim p, \quad \text{e.g. } p(\theta) \propto \exp\left(-\frac{\|\theta\|^2}{2}\right).$$

Bayesian inference

Let $\mathcal{D} = (w_i, y_i)_{i=1}^m$ a dataset of labelled examples $(w_i, y_i) \overset{i.i.d.}{\sim} P_{\text{data}}$.
Assume an underlying model parametrized by θ, e.g. :

$$y = g(w, \theta) + \epsilon, \quad \epsilon \sim \mathcal{N}(0, I)$$

Goal: learn the best distribution over θ to fit the data.

1. Compute the Likelihood:

$$p(\mathcal{D}|\theta) = \prod_{i=1}^m p(y_i|\theta, w_i) \propto \exp\left(-\frac{1}{2}\sum_{i=1}^m \|y_i - g(w_i, \theta)\|^2\right).$$

2. Choose a prior distribution on the parameter:

$$\theta \sim p, \quad \text{e.g. } p(\theta) \propto \exp\left(-\frac{\|\theta\|^2}{2}\right).$$

3. Bayes' rule yields:

$$\pi(\theta) := p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{Z} \qquad Z = \int_{\mathbb{R}^d} p(\mathcal{D}|\theta)p(\theta)d\theta$$

$$\text{i.e. } \pi(\theta) \propto \exp\left(-V(\theta)\right), \quad V(\theta) = \frac{1}{2}\sum_{i=1}^m \|y_i - g(w_i, \theta)\|^2 + \frac{\|\theta\|^2}{2}.$$

π is needed both for

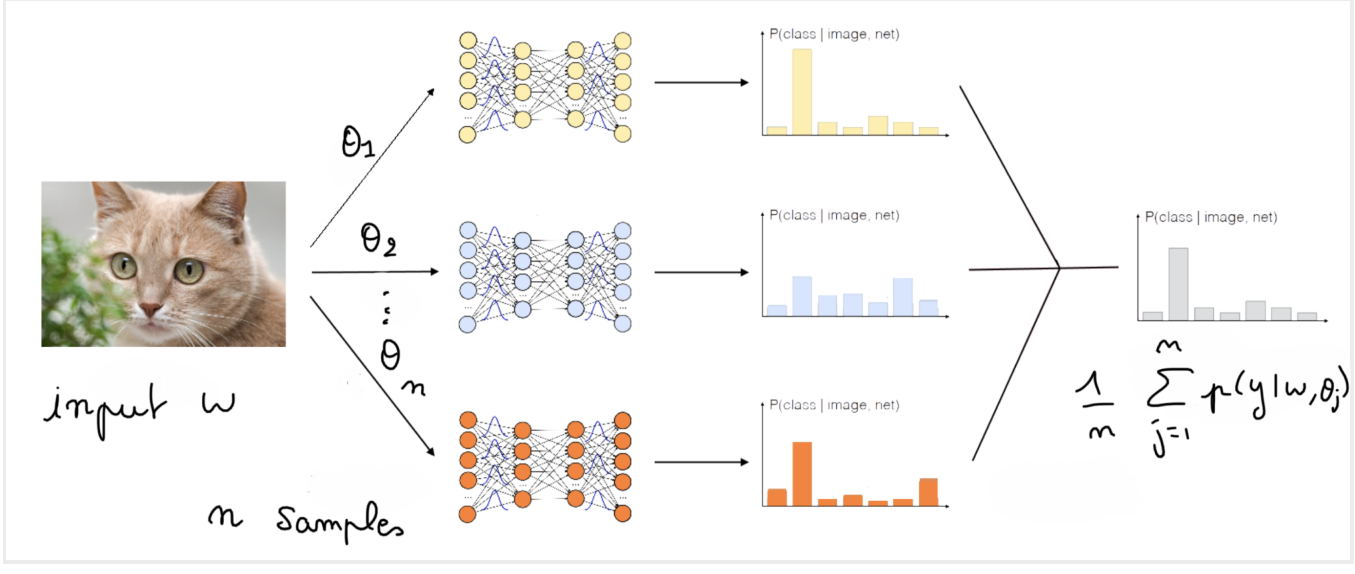
- prediction for a new input w: $y_{pred} = \int_{\mathbb{R}^d} g(w, \theta) d\pi(\theta)$
- measure uncertainty on the prediction.

π is needed both for

- prediction for a new input w: $y_{pred} = \int_{\mathbb{R}^d} g(w, \theta) d\pi(\theta)$
- measure uncertainty on the prediction.

Given a discrete approximation $\mu_n = \frac{1}{n} \sum_{j=1}^{n} \delta_{\theta_j}$ of π:

$$y_{pred} \approx \frac{1}{n} \sum_{j=1}^{n} g(w, \theta_j).$$

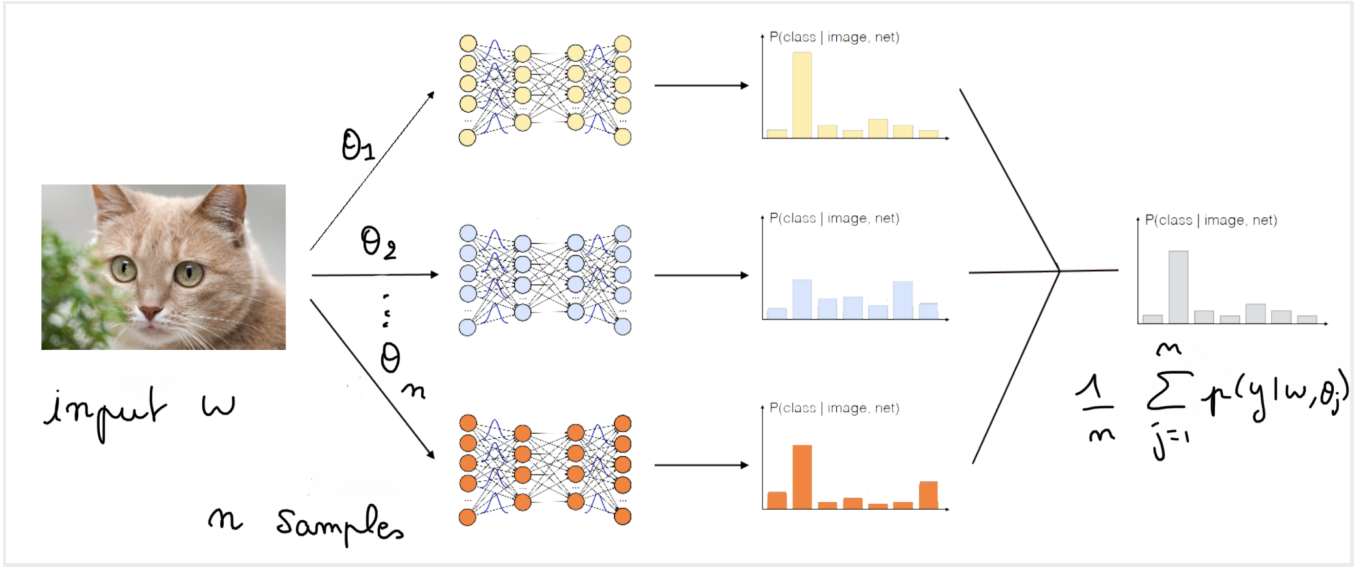


π is needed both for

- prediction for a new input w: $y_{pred} = \int_{\mathbb{R}^d} g(w, \theta) d\pi(\theta)$
- measure uncertainty on the prediction.

Given a discrete approximation $\mu_n = \frac{1}{n} \sum_{j=1}^{n} \delta_{\theta_j}$ of π:

$$y_{pred} \approx \frac{1}{n} \sum_{j=1}^{n} g(w, \theta_j).$$



Question: how can we build μ_n?

Sampling as optimisation

Notice that

$$\pi = \operatorname*{argmin}_{\mu \in \mathcal{P}(\mathbb{R}^d)} \mathrm{KL}(\mu|\pi), \quad \mathrm{KL}(\mu|\pi) = \begin{cases} \int_{\mathbb{R}^d} \log\left(\frac{\mu}{\pi}(\theta)\right) d\mu(\theta) & \text{if } \mu \ll \pi \\ +\infty & \text{else.} \end{cases}$$

(does not depend on the normalisation constant Z in $\pi(\theta) = \tilde{\pi}(\theta)/Z$!)

Sampling as optimisation

Notice that

$$\pi = \underset{\mu \in \mathcal{P}(\mathbb{R}^d)}{\operatorname{argmin}} \ \mathrm{KL}(\mu|\pi), \quad \mathrm{KL}(\mu|\pi) = \begin{cases} \int_{\mathbb{R}^d} \log\left(\frac{\mu}{\pi}(\theta)\right) d\mu(\theta) & \text{if } \mu \ll \pi \\ +\infty & \text{else.} \end{cases}$$

(does not depend on the normalisation constant Z in $\pi(\theta) = \tilde{\pi}(\theta)/Z$!)

Two (non parametric) ways to produce an approximation μ_n:

1. Markov Chain Monte Carlo (MCMC) methods: generate a Markov chain whose law converges to $\pi \propto \exp(-V)$

   Example: Langevin Monte Carlo (LMC), discretizes an overdamped Langevin diffusion

   $$d\theta_t = -\nabla V(\theta_t) + \sqrt{2} dB_t \Longrightarrow \theta_{l+1} = \theta_l - \gamma \nabla V(\theta_l) + \sqrt{2\gamma}\epsilon_l, \ \epsilon_l \sim \mathcal{N}(0, I_d)$$

   Its law corresponds to a Wasserstein gradient flow of the KL [Jordan et al., 1998].

Sampling as optimisation

Notice that

$$\pi = \underset{\mu \in \mathcal{P}(\mathbb{R}^d)}{\operatorname{argmin}} \ \mathrm{KL}(\mu|\pi), \quad \mathrm{KL}(\mu|\pi) = \begin{cases} \int_{\mathbb{R}^d} \log\left(\frac{\mu}{\pi}(\theta)\right) d\mu(\theta) & \text{if } \mu \ll \pi \\ +\infty & \text{else.} \end{cases}$$

(does not depend on the normalisation constant Z in $\pi(\theta) = \tilde{\pi}(\theta)/Z$!)

Two (non parametric) ways to produce an approximation μ_n:

1. Markov Chain Monte Carlo (MCMC) methods: generate a Markov chain whose law converges to $\pi \propto \exp(-V)$

   Example: Langevin Monte Carlo (LMC), discretizes an overdamped Langevin diffusion

   $$d\theta_t = -\nabla V(\theta_t) + \sqrt{2} dB_t \Longrightarrow \theta_{l+1} = \theta_l - \gamma \nabla V(\theta_l) + \sqrt{2\gamma}\epsilon_l, \ \epsilon_l \sim \mathcal{N}(0, I_d)$$

   Its law corresponds to a Wasserstein gradient flow of the KL [Jordan et al., 1998].

2. Interacting particle systems, e.g. by considering other metrics or functionals

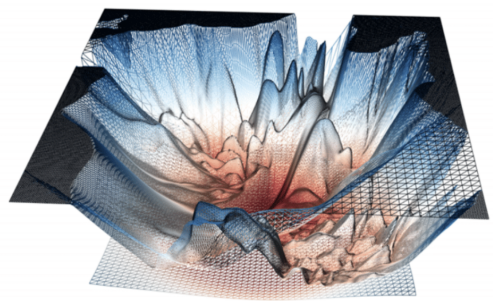
Difficult cases : non-convex potentials

Recall that

$$\pi(\theta) \propto \exp\left(-V(\theta)\right), \quad V(\theta) = \underbrace{\sum_{i=1}^{m} \|y_i - g(w_i, \theta)\|^2}_{\text{loss}} + \frac{\|\theta\|^2}{2}.$$

- if V is convex (e.g. $g(w, \theta) = \langle w, \theta \rangle$) many sampling methods are known to work quite well, including LMC
- but if its not (e.g. $g(w, \theta)$ is a neural network), the situation is much more delicate
- MCMC methods do not scale and require too many iterations

  [Izmailov et al., 2021]



A highly nonconvex loss surface, as is common in deep neural nets. From `https://www.telesens.co/2019/01/16/neural-network-loss-visualization`.

Sampling as optimization over distributions

Assume that $\pi \in \mathcal{P}_2(\mathbb{R}^d) = \left\{\mu \in \mathcal{P}(\mathbb{R}^d), \int \|x\|^2 d\mu(x) < \infty\right\}$.
We equip $\mathcal{P}_2(\mathbb{R}^d)$ with the Wasserstein-2 distance:

$$W_2^2(\nu, \mu) = \inf_{s \in \Gamma(\nu,\mu)} \int_{\mathbb{R}^d \times \mathbb{R}^d} \|x - y\|^2 \, ds(x, y) \qquad \forall \nu, \mu \in \mathcal{P}_2(\mathbb{R}^d)$$

where $\Gamma(\nu, \mu)$ is the set of possible couplings between ν and μ.

The sampling task can be recast as an optimization problem:

$$\pi = \operatorname*{argmin}_{\mu \in \mathcal{P}_2(\mathbb{R}^d)} \mathcal{F}(\mu), \quad \mathcal{F}(\mu) := D(\mu|\pi)$$

where D is a **dissimilarity functional** (f-div, IPM, OT distance...).

Starting from an initial distribution $\mu_0 \in \mathcal{P}_2(\mathbb{R}^d)$, one can then consider the **Wasserstein gradient flow** of $\mathcal{F}$ over $\mathcal{P}_2(\mathbb{R}^d)$ to transport μ_0 to π.

Outline

Euclidean gradient flow and continuity equation

Let $V : \mathbb{R}^d \to \mathbb{R}$. Consider the gradient flow

$$\frac{dX_t}{dt} = -\nabla V(x_t)$$

and assume x_0 random with density μ_0. What is the dynamics of the density μ_t of x_t ? Let $\phi : \mathbb{R}^d \to \mathbb{R}$ a test function.

$$\frac{d}{dt}\mathbb{E}(\phi(x_t)) = -\int \langle \nabla\phi, \nabla V\rangle \mu_t(x)dx = \int \phi(x)\boldsymbol{\nabla}\cdot(\mu_t \nabla V)(x)dx,$$

and

$$\frac{d}{dt}\mathbb{E}(\phi(x_t)) = \int \phi(x)\frac{\partial \mu_t}{\partial t}(x)dx.$$

Therefore,

$$\frac{\partial \mu_t}{\partial t} = \boldsymbol{\nabla}\cdot(\mu_t \nabla V).$$

Wasserstein gradient flows (WGF) [Ambrosio et al., 2008]

The first variation of $\mu \mapsto \mathcal{F}(\mu)$ evaluated at $\mu \in \mathcal{P}$, if it exists, is the unique function $\frac{\partial \mathcal{F}(\mu)}{\partial \mu} : \mathbb{R}^d \to \mathbb{R}$ s. t. for any $\mu, \mu' \in \mathcal{P}$:

$$\lim_{\epsilon \to 0} \frac{1}{\epsilon} \left[\mathcal{F}(\mu + \epsilon(\mu' - \mu)) - \mathcal{F}(\mu) \right] = \int_{\mathbb{R}^d} \frac{\partial \mathcal{F}(\mu)}{\partial \mu}(x)(d\mu' - d\mu)(x).$$

Wasserstein gradient flows (WGF) [Ambrosio et al., 2008]

The first variation of $\mu \mapsto \mathcal{F}(\mu)$ evaluated at $\mu \in \mathcal{P}$, if it exists, is the unique function $\frac{\partial \mathcal{F}(\mu)}{\partial \mu} : \mathbb{R}^d \to \mathbb{R}$ s. t. for any $\mu, \mu' \in \mathcal{P}$:

$$\lim_{\epsilon \to 0} \frac{1}{\epsilon} \left[\mathcal{F}(\mu + \epsilon(\mu' - \mu)) - \mathcal{F}(\mu)\right] = \int_{\mathbb{R}^d} \frac{\partial \mathcal{F}(\mu)}{\partial \mu}(x)(d\mu' - d\mu)(x).$$

The family $\mu : [0, \infty] \to \mathcal{P}, t \mapsto \mu_t$ satisfies a Wasserstein gradient flow of $\mathcal{F}$ if distributionally:

$$\frac{\partial \mu_t}{\partial t} = \boldsymbol{\nabla} \cdot (\mu_t \nabla_{W_2} \mathcal{F}(\mu_t)),$$

where $\nabla_{W_2} \mathcal{F}(\mu) := \nabla \frac{\partial \mathcal{F}(\mu)}{\partial \mu} \in L^2(\mu)$ denotes the Wasserstein gradient of $\mathcal{F}$.

Wasserstein gradient flows (WGF) [Ambrosio et al., 2008]

The first variation of $\mu \mapsto \mathcal{F}(\mu)$ evaluated at $\mu \in \mathcal{P}$, if it exists, is the unique function $\frac{\partial \mathcal{F}(\mu)}{\partial \mu} : \mathbb{R}^d \to \mathbb{R}$ s. t. for any $\mu, \mu' \in \mathcal{P}$:

$$\lim_{\epsilon \to 0} \frac{1}{\epsilon} \left[\mathcal{F}(\mu + \epsilon(\mu' - \mu)) - \mathcal{F}(\mu)\right] = \int_{\mathbb{R}^d} \frac{\partial \mathcal{F}(\mu)}{\partial \mu}(x)(d\mu' - d\mu)(x).$$

The family $\mu : [0, \infty] \to \mathcal{P}, t \mapsto \mu_t$ satisfies a Wasserstein gradient flow of $\mathcal{F}$ if distributionally:

$$\frac{\partial \mu_t}{\partial t} = \boldsymbol{\nabla} \cdot (\mu_t \nabla_{W_2} \mathcal{F}(\mu_t)),$$

where $\nabla_{W_2} \mathcal{F}(\mu) := \nabla \frac{\partial \mathcal{F}(\mu)}{\partial \mu} \in L^2(\mu)$ denotes the Wasserstein gradient of $\mathcal{F}$.

It can be implemented by the deterministic process:

$$\frac{dX_t}{dt} = -\nabla_{W_2} \mathcal{F}(\mu_t)(X_t)$$

Time and Space discretization - Particle system

Let $\gamma > 0$ be a step-size:

$$X_{l+1} = X_l - \gamma \nabla_{W_2} \mathcal{F}(\mu_l)(X_l)$$

Time and Space discretization - Particle system

Let $\gamma > 0$ be a step-size:

$$X_{l+1} = X_l - \gamma \nabla_{W_2} \mathcal{F}(\mu_l)(X_l)$$

Problem: the vector field depends on the unknown μ_l, the density of the particle at time l.

Time and Space discretization - Particle system

Let $\gamma > 0$ be a step-size:

$$X_{l+1} = X_l - \gamma \nabla_{W_2} \mathcal{F}(\mu_l)(X_l)$$

Problem: the vector field depends on the unknown μ_l, the density of the particle at time l.

Idea: replace it by the empirical measure of a system of n interacting particles:

$$X_0^1, \dots, X_0^n \sim \mu_0$$

and for $j = 1, \dots, n$:

$$X_{l+1}^j = X_l^j - \gamma \nabla_{W_2} \mathcal{F}(\hat{\mu}_l)(X_l^j)$$

where $\hat{\mu}_l = \frac{1}{n} \sum_{i=1}^n \delta_{X_l^j}$.

We recall that

$$\pi = \underset{\mu \in \mathcal{P}_2(\mathbb{R}^d)}{\text{argmin}} \ \mathrm{KL}(\mu|\pi), \quad \mathrm{KL}(\mu|\pi) = \int \log\Big(\frac{\mu}{\pi}\Big) d\mu \text{ if } \mu \ll \pi$$

and that we can consider the Forward time discretisation:

$$x_{l+1} = x_l - \gamma \nabla_{W_2} \mathrm{KL}(\mu_l|\pi)(x_l), \quad x_l \sim \mu_l,$$

where $\nabla_{W_2} \mathrm{KL}(\mu_l|\pi) = \nabla \frac{\partial \mathrm{KL}(\mu_l|\pi)}{\partial \mu} = \nabla \log\big(\frac{\mu_l}{\pi}(.)\big)$.

Problem: μ_l, hence $\nabla \log(\mu_l)$ is unknown and has to be estimated from a set of particles.

Outline

Background on kernels and RKHS [Steinwart and Christmann, 2008]

- Let $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ a positive, semi-definite kernel
  $((k(x_i, x_j)_{i=1}^n)$ is a p.s.d. matrix for all $x_1, \ldots, x_n \in \mathbb{R}^d$)

Background on kernels and RKHS [Steinwart and Christmann, 2008]

- Let $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ a positive, semi-definite kernel
  $((k(x_i, x_j)_{i=1}^n)$ is a p.s.d. matrix for all $x_1, \ldots, x_n \in \mathbb{R}^d)$
- examples:
  - the Gaussian kernel $k(x, y) = \exp\left(-\frac{\|x-y\|^2}{h}\right)$
  - the Laplace kernel $k(x, y) = \exp\left(-\frac{\|x-y\|}{h}\right)$
  - the inverse multiquadratic kernel
    $k(x, y) = (c + \|x - y\|)^{-\beta}$ with $\beta \in]0, 1[$

Background on kernels and RKHS [Steinwart and Christmann, 2008]

- Let $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ a positive, semi-definite kernel
  $((k(x_i, x_j)_{i=1}^n)$ is a p.s.d. matrix for all $x_1, \ldots, x_n \in \mathbb{R}^d)$
- examples:
  - the Gaussian kernel $k(x, y) = \exp\left(-\frac{\|x-y\|^2}{h}\right)$
  - the Laplace kernel $k(x, y) = \exp\left(-\frac{\|x-y\|}{h}\right)$
  - the inverse multiquadratic kernel
    $k(x, y) = (c + \|x - y\|)^{-\beta}$ with $\beta \in]0, 1[$
- $\mathcal{H}_k$ its corresponding RKHS (Reproducing Kernel Hilbert Space):

$$\mathcal{H}_k = \overline{\left\{ \sum_{i=1}^{m} \alpha_i k(\cdot, x_i);\ m \in \mathbb{N};\ \alpha_1, \ldots, \alpha_m \in \mathbb{R};\ x_1, \ldots, x_m \in \mathbb{R}^d \right\}}$$

Background on kernels and RKHS [Steinwart and Christmann, 2008]

- Let $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ a positive, semi-definite kernel
  $((k(x_i, x_j)_{i=1}^n)$ is a p.s.d. matrix for all $x_1, \ldots, x_n \in \mathbb{R}^d)$
- examples:
  - the Gaussian kernel $k(x, y) = \exp\left(-\frac{\|x-y\|^2}{h}\right)$
  - the Laplace kernel $k(x, y) = \exp\left(-\frac{\|x-y\|}{h}\right)$
  - the inverse multiquadratic kernel
    $k(x, y) = (c + \|x - y\|)^{-\beta}$ with $\beta \in]0, 1[$
- $\mathcal{H}_k$ its corresponding RKHS (Reproducing Kernel Hilbert Space):

$$\mathcal{H}_k = \overline{\left\{ \sum_{i=1}^{m} \alpha_i k(\cdot, x_i);\ m \in \mathbb{N};\ \alpha_1, \ldots, \alpha_m \in \mathbb{R};\ x_1, \ldots, x_m \in \mathbb{R}^d \right\}}$$

- $\mathcal{H}_k$ is a Hilbert space with inner product $\langle ., . \rangle_{\mathcal{H}_k}$ and norm $\|.\|_{\mathcal{H}_k}$.

Background on kernels and RKHS [Steinwart and Christmann, 2008]

- Let $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ a positive, semi-definite kernel
  $((k(x_i, x_j)_{i=1}^n)$ is a p.s.d. matrix for all $x_1, \ldots, x_n \in \mathbb{R}^d)$
- examples:
  - the Gaussian kernel $k(x, y) = \exp\left(-\frac{\|x-y\|^2}{h}\right)$
  - the Laplace kernel $k(x, y) = \exp\left(-\frac{\|x-y\|}{h}\right)$
  - the inverse multiquadratic kernel
    $k(x, y) = (c + \|x - y\|)^{-\beta}$ with $\beta \in]0, 1[$
- $\mathcal{H}_k$ its corresponding RKHS (Reproducing Kernel Hilbert Space):

$$\mathcal{H}_k = \overline{\left\{ \sum_{i=1}^{m} \alpha_i k(\cdot, x_i);\ m \in \mathbb{N};\ \alpha_1, \ldots, \alpha_m \in \mathbb{R};\ x_1, \ldots, x_m \in \mathbb{R}^d \right\}}$$

- $\mathcal{H}_k$ is a Hilbert space with inner product $\langle ., . \rangle_{\mathcal{H}_k}$ and norm $\|.\|_{\mathcal{H}_k}$.
- assume $\int_{\mathbb{R}^d \times \mathbb{R}^d} k(x, x) d\mu(x) < \infty$ for any $\mu \in \mathcal{P}(\mathbb{R}^d), \Longrightarrow \mathcal{H}_k \subset L^2(\mu)$.

Background on kernels and RKHS [Steinwart and Christmann, 2008]

- Let $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ a positive, semi-definite kernel
  $((k(x_i, x_j)_{i=1}^n)$ is a p.s.d. matrix for all $x_1, \ldots, x_n \in \mathbb{R}^d)$
- examples:
  - the Gaussian kernel $k(x, y) = \exp\left(-\frac{\|x-y\|^2}{h}\right)$
  - the Laplace kernel $k(x, y) = \exp\left(-\frac{\|x-y\|}{h}\right)$
  - the inverse multiquadratic kernel
    $k(x, y) = (c + \|x - y\|)^{-\beta}$ with $\beta \in]0, 1[$
- $\mathcal{H}_k$ its corresponding RKHS (Reproducing Kernel Hilbert Space):

$$\mathcal{H}_k = \overline{\left\{ \sum_{i=1}^{m} \alpha_i k(\cdot, x_i);\ m \in \mathbb{N};\ \alpha_1, \ldots, \alpha_m \in \mathbb{R};\ x_1, \ldots, x_m \in \mathbb{R}^d \right\}}$$

- $\mathcal{H}_k$ is a Hilbert space with inner product $\langle ., . \rangle_{\mathcal{H}_k}$ and norm $\|.\|_{\mathcal{H}_k}$.
- assume $\int_{\mathbb{R}^d \times \mathbb{R}^d} k(x, x) d\mu(x) < \infty$ for any $\mu \in \mathcal{P}(\mathbb{R}^d), \Longrightarrow \mathcal{H}_k \subset L^2(\mu)$.
- It satisfies the reproducing property:

$$\forall \quad f \in \mathcal{H}_k,\ x \in \mathbb{R}^d, \quad f(x) = \langle f, k(x, .) \rangle_{\mathcal{H}_k}.$$

Stein Variational Gradient Descent [Liu and Wang, 2016]

Consider the following metric depending on k [1]

$$W_k^2(\mu_0, \mu_1) = \inf_{(\mu_t, v_t)} \left\{ \int_0^1 \|v_t\|_{\mathcal{H}_k^d}^2 dt : \frac{\partial \mu_t}{\partial t} = \nabla \cdot (\mu_t v_t) \right\}.$$

Then, the W_k gradient flow of the KL writes as the PDE [Liu, 2017], [Duncan et al., 2019]:

$$\frac{\partial \mu_t}{\partial t} + \nabla \cdot \left(\mu_t P_{\mu_t} \nabla \log \left(\frac{\mu_t}{\pi} \right) \right) = 0, \quad P_\mu : f \mapsto \int k(x, .) f(x) d\mu(x).$$

It converges to $\pi \propto \exp(-V)$ under mild conditions on k and if V grows at most polynomially [Lu et al., 2019].

[1] $W_2^2(\mu_0, \mu_1) = \inf_{(\mu_t, v_t)_{t \in [0,1]}} \left\{ \int_0^1 \|v_t\|_{L^2(\mu_t)}^2 dt : \frac{\partial \mu_t}{\partial t} = \nabla \cdot (\mu_t v_t) \right\}$.

SVGD algorithm

SVGD trick: applying the kernel integral operator to the W_2 gradient of $\mathrm{KL}(\cdot|\pi)$ leads to

$$
\begin{aligned}
P_\mu \nabla \log\left(\frac{\mu}{\pi}\right)(\cdot) &= \int \nabla \log\left(\frac{\mu}{\pi}\right)(x) k(x,.) d\mu(x) \\
&= \int -\nabla \log(\pi(x)) k(x,.) d\mu(x) + \int \nabla(\mu(x)) k(x,.) dx \\
&\overset{I.P.P.}{=} -\int [\nabla \log \pi(x) k(x,\cdot) + \nabla_x k(x,\cdot)] d\mu(x),
\end{aligned}
$$

under appropriate boundary conditions on k and π, e.g. $\lim_{\|x\|\to\infty} k(x,\cdot)\pi(x) \to 0$.

SVGD algorithm

SVGD trick: applying the kernel integral operator to the W_2 gradient of $\mathrm{KL}(\cdot|\pi)$ leads to

$$
\begin{aligned}
P_\mu \nabla \log\left(\frac{\mu}{\pi}\right)(\cdot) &= \int \nabla \log\left(\frac{\mu}{\pi}\right)(x) k(x,.) d\mu(x) \\
&= \int -\nabla \log(\pi(x)) k(x,.) d\mu(x) + \int \nabla(\mu(x)) k(x,.) dx \\
&\overset{I.P.P.}{=} -\int [\nabla \log \pi(x) k(x,\cdot) + \nabla_x k(x,\cdot)] d\mu(x),
\end{aligned}
$$

under appropriate boundary conditions on k and π, e.g. $\lim_{\|x\|\to\infty} k(x,\cdot)\pi(x) \to 0$.

Algorithm : Starting from n i.i.d. samples $(X_0^i)_{i=1,\dots,n} \sim \mu_0$, SVGD algorithm updates the n particles as follows :

$$
\begin{aligned}
X_{l+1}^i &= X_l^i - \gamma \left[\frac{1}{n}\sum_{j=1}^n \nabla_{X_l^j} \log \pi(X_l^j) k(X_l^i, X_l^j) + \nabla_{X_l^j} k(X_l^j, X_l^i)\right] \\
&= X_l^i - \gamma P_{\mu_l^n} \nabla \log\left(\frac{\mu_l^n}{\pi}\right)(X_l^i), \quad \text{with } \mu_l^n = \frac{1}{n}\sum_{j=1}^n \delta_{X_l^j}
\end{aligned}
$$

SVGD in practice

- more than 600 citations for [Liu and Wang, 2016]
- Relative empirical success in Bayesian inference and more recently for deep networks
- It can suffer for multimodal distributions [Wenliang and Kanagawa, 2020], underestimate the target variance [Ba et al., 2021], but still can be very efficient on difficult sampling problems.

		AUROC(H)	AUROC(MD)	Accuracy	H_o/H_t	MD_o/MD_t	ECE	NLL
FashionMNIST	**Deep ensemble [38]**	0.958±0.001	0.975±0.001	91.122±0.013	6.257±0.005	6.394±0.001	**0.012±0.001**	0.129±0.001
	SVGD [46]	0.960±0.001	0.973±0.001	91.134±0.024	6.315±0.019	6.395±0.018	0.014±0.001	0.127±0.001
	f-SVGD [67]	0.956±0.001	0.975±0.001	89.884±0.015	5.652±0.009	6.531±0.005	0.013±0.001	0.150±0.001
	kde-WGD (ours)	0.960±0.001	0.970±0.001	91.238±0.019	6.587±0.019	6.379±0.018	0.014±0.001	0.128±0.001
	sge-WGD (ours)	0.960±0.001	0.970±0.001	**91.312±0.016**	6.562±0.007	6.363±0.009	**0.012±0.001**	0.128±0.001
	ssge-WGD (ours)	0.968±0.001	0.979±0.001	91.198±0.024	6.522±0.009	6.610±0.012	**0.012±0.001**	0.130±0.001
	kde-fWGD (ours)	**0.971±0.001**	**0.980±0.001**	91.260±0.011	7.079±0.016	6.887±0.015	0.015±0.001	**0.125±0.001**
	sge-fWGD (ours)	0.969±0.001	0.978±0.001	91.192±0.013	7.076±0.004	6.900±0.005	0.015±0.001	**0.125±0.001**
	ssge-fWGD (ours)	**0.971±0.001**	**0.980±0.001**	91.240±0.022	**7.129±0.006**	**6.951±0.005**	0.016±0.001	**0.124±0.001**
CIFAR10	**Deep ensemble [38]**	**0.843±0.004**	0.736±0.005	85.552±0.076	**2.244±0.006**	1.667±0.008	0.049±0.001	0.277±0.001
	SVGD [46]	0.825±0.001	0.710±0.002	85.142±0.017	2.106±0.003	1.567±0.004	0.052±0.001	0.287±0.001
	fSVGD [67]	0.783±0.001	0.712±0.001	84.510±0.031	1.968±0.004	1.624±0.003	0.049±0.001	0.292±0.001
	kde-WGD (ours)	0.838±0.001	0.735±0.004	**85.904±0.030**	2.205±0.003	1.661±0.008	0.053±0.001	**0.276±0.001**
	sge-WGD (ours)	0.837±0.003	0.725±0.004	85.792±0.035	2.214±0.010	1.634±0.004	0.051±0.001	**0.275±0.001**
	ssge-WGD (ours)	0.832±0.003	0.731±0.005	85.638±0.038	2.182±0.015	1.655±0.001	0.049±0.001	**0.276±0.001**
	kde-fWGD (ours)	0.791±0.002	**0.758±0.002**	84.888±0.030	1.970±0.004	**1.749±0.005**	**0.044±0.001**	0.282±0.001
	sge-fWGD (ours)	0.795±0.001	0.754±0.002	84.766±0.060	1.984±0.003	1.729±0.002	0.047±0.001	0.288±0.001
	ssge-fWGD (ours)	0.792±0.002	0.752±0.002	84.762±0.034	1.970±0.006	1.723±0.005	0.046±0.001	0.286±0.001

From *Repulsive Deep Ensembles are Bayesian. F. D'angelo, V. Fortuin. Conference on Neural Information Processing Systems (NeurIPS 2021).*

Continuous-time dynamics of SVGD

$$\frac{\partial \mu_t}{\partial t} + \boldsymbol{\nabla} \cdot \left(\mu_t P_{\mu_t} \nabla \log \left(\frac{\mu_t}{\pi} \right) \right) = 0, \quad P_\mu : f \mapsto \int k(x, .) f(x) d\mu(x).$$

[2] $P_\mu = S_\mu^* \circ S_\mu$, where $S_\mu : L^2(\mu) \to \mathcal{H}_k, f \mapsto \int k(x, .) f(x) d\mu(x)$ and $S_\mu^* = \iota_{\mathcal{H}_k \to L^2(\mu)}$ the injection from $\mathcal{H}_k$ to $L^2(\mu)$. We sometimes abuse notation here between P_μ, S_μ for ease of presentation.

Continuous-time dynamics of SVGD

$$\frac{\partial \mu_t}{\partial t} + \nabla \cdot \left(\mu_t P_{\mu_t} \nabla \log \left(\frac{\mu_t}{\pi} \right) \right) = 0, \quad P_\mu : f \mapsto \int k(x, .) f(x) d\mu(x).$$

How fast the KL decreases along SVGD dynamics? Apply the chain rule in the Wasserstein space[2]:

$$\frac{d \, \mathrm{KL}(\mu_t | \pi)}{dt} = \left\langle V_t, \nabla \log \left(\frac{\mu_t}{\pi} \right) \right\rangle_{L^2(\mu_t)} = - \underbrace{\left\| P_{\mu_t} \nabla \log \left(\frac{\mu_t}{\pi} \right) \right\|^2_{\mathcal{H}_k}}_{\mathrm{KSD}^2(\mu_t | \pi)} \leq 0.$$

[2] $P_\mu = S_\mu^* \circ S_\mu$, where $S_\mu : L^2(\mu) \to \mathcal{H}_k, f \mapsto \int k(x, .) f(x) d\mu(x)$ and $S_\mu^* = \iota_{\mathcal{H}_k \to L^2(\mu)}$ the injection from $\mathcal{H}_k$ to $L^2(\mu)$. We sometimes abuse notation here between P_μ, S_μ for ease of presentation.

Continuous-time dynamics of SVGD

$$\frac{\partial \mu_t}{\partial t} + \nabla \cdot \left(\mu_t P_{\mu_t} \nabla \log \left(\frac{\mu_t}{\pi} \right) \right) = 0, \quad P_\mu : f \mapsto \int k(x, .) f(x) d\mu(x).$$

How fast the KL decreases along SVGD dynamics? Apply the chain rule in the Wasserstein space[2]:

$$\frac{d \, \mathrm{KL}(\mu_t | \pi)}{dt} = \left\langle V_t, \nabla \log \left(\frac{\mu_t}{\pi} \right) \right\rangle_{L^2(\mu_t)} = - \underbrace{\left\| P_{\mu_t} \nabla \log \left(\frac{\mu_t}{\pi} \right) \right\|^2_{\mathcal{H}_k}}_{\mathrm{KSD}^2(\mu_t | \pi)} \leq 0.$$

On the r.h.s. we have the **Kernel Stein discrepancy (KSD)** [Chwialkowski et al., 2016] or **Stein Fisher information** of μ_t relative to π:

$$\begin{aligned} \left\| P_{\mu,k} \nabla \log \left(\frac{\mu}{\pi} \right) \right\|^2_{\mathcal{H}_k} &= \langle P_{\mu,k} \nabla \log \left(\frac{\mu}{\pi} \right), P_{\mu,k} \nabla \log \left(\frac{\mu}{\pi} \right) \rangle_{\mathcal{H}_k} \\ &= \iint \nabla \log \left(\frac{\mu}{\pi}(x) \right) \nabla \log \left(\frac{\mu}{\pi}(y) \right) k(x, y) d\mu(x) d\mu(y). \end{aligned}$$

Recall that the Fisher divergence is defined as $\| \nabla \log \left(\frac{\mu}{\pi} \right) \|^2_{L^2(\mu)}$.

[2] $P_\mu = S^*_\mu \circ S_\mu$, where $S_\mu : L^2(\mu) \to \mathcal{H}_k, f \mapsto \int k(x, .) f(x) d\mu(x)$ and $S^*_\mu = \iota_{\mathcal{H}_k \to L^2(\mu)}$ the injection from $\mathcal{H}_k$ to $L^2(\mu)$. We sometimes abuse notation here between P_μ, S_μ for ease of presentation.

A descent lemma in discrete time for SVGD [Korba et al., 2020]

Idea: in optimisation, descent lemmas can be shown if the objective function has a bounded Hessian.

A descent lemma in discrete time for SVGD [Korba et al., 2020]

Idea: in optimisation, descent lemmas can be shown if the objective function has a bounded Hessian.

Assume that $\pi \propto \exp(-V)$ where $\|H_V(x)\| \leq M$.
The Hessian of the KL at μ is an operator on $L^2(\mu)$:

$$\langle f, Hess_{\mathrm{KL}(.|\pi)}(\mu) f \rangle_{L^2(\mu)} = \mathbb{E}_{X\sim\mu}\left[\langle f(X), H_V(X) f(X)\rangle + \|Jf(X)\|^2_{HS}\right]$$

and yet, this operator **is not bounded** due to the Jacobian term.

A descent lemma in discrete time for SVGD [Korba et al., 2020]

Idea: in optimisation, descent lemmas can be shown if the objective function has a bounded Hessian.

Assume that $\pi \propto \exp(-V)$ where $\|H_V(x)\| \leq M$.
The Hessian of the KL at μ is an operator on $L^2(\mu)$:

$$\langle f, Hess_{\mathrm{KL}(.|\pi)}(\mu) f \rangle_{L^2(\mu)} = \mathbb{E}_{X\sim\mu}\left[\langle f(X), H_V(X) f(X)\rangle + \|Jf(X)\|_{HS}^2\right]$$

and yet, this operator **is not bounded** due to the Jacobian term.

However: In the case of SVGD, the descent directions f are restricted to $\mathcal{H}_k$ (bounded functions, bounded derivatives for bounded $k, \nabla k$).

Proposition: Assume (boundedness of k and ∇k, H_V and moments on the trajectory), then for γ small enough:

$$\mathrm{KL}(\mu_{l+1}|\pi) - \mathrm{KL}(\mu_l|\pi) \leq -c_\gamma \underbrace{\left\| P_{\mu_l} \nabla \log\left(\frac{\mu_l}{\pi}\right)\right\|_{\mathcal{H}_k}^2}_{\mathrm{KSD}^2(\mu_l|\pi)}.$$

Rates in KSD

Consequence of the descent lemma: for γ small enough,

$$\min_{l=1,\dots,L} \mathrm{KSD}^2(\mu_l|\pi) \leq \frac{1}{L}\sum_{l=1}^{L} \mathrm{KSD}^2(\mu_l|\pi) \leq \frac{\mathrm{KL}(\mu_0|\pi)}{c_\gamma L}.$$

This result only relies on the smoothness of V, not on any kind of convexity, in contrast with many convergence results on LMC.

The KSD metrizes convergence for instance when [Gorham and Mackey, 2017]:

- π is distantly dissipative (log concave at infinity, e.g. mixture of Gaussians)
- k is the IMQ kernel defined by $k(x, y) = (c^2 + \|x - y\|_2^2)^\beta$ for $c > 0$ and $\beta \in (-1, 0)$.

Open question 1: Rates in terms of the KL objective?

To obtain rates, one may combine a descent lemma (1) of the form

$$\mathrm{KL}(\mu_{l+1}|\pi) - \mathrm{KL}(\mu_l|\pi) \leq -c_\gamma \left\| P_{\mu_n} \nabla \log\left(\frac{\mu_l}{\pi}\right) \right\|^2_{\mathcal{H}_k}$$

and the Stein log-Sobolev inequality (2) with constant λ:

$$\mathrm{KL}(\mu|\pi) \leq \frac{1}{2\lambda} \mathrm{KSD}^2(\mu|\pi) \text{ for all } \mu.$$

Open question 1: Rates in terms of the KL objective?

To obtain rates, one may combine a descent lemma (1) of the form

$$\mathrm{KL}(\mu_{l+1}|\pi) - \mathrm{KL}(\mu_l|\pi) \leq -c_\gamma \left\| P_{\mu_n} \nabla \log\left(\frac{\mu_l}{\pi}\right) \right\|_{\mathcal{H}_k}^2$$

and the Stein log-Sobolev inequality (2) with constant λ:

$$\mathrm{KL}(\mu|\pi) \leq \frac{1}{2\lambda} \mathrm{KSD}^2(\mu|\pi) \text{ for all } \mu.$$

Then:

$$\mathrm{KL}(\mu_{l+1}|\pi) - \mathrm{KL}(\mu_l|\pi) \underbrace{\leq}_{(1)} -c_\gamma \left\| P_{\mu_l} \nabla \log\left(\frac{\mu_n}{\pi}\right) \right\|_{\mathcal{H}_k}^2 \underbrace{\leq}_{(2)} -c_\gamma 2\lambda \, \mathrm{KL}(\mu_n|\pi).$$

Iterating this inequality yields $\mathrm{KL}(\mu_l|\pi) \leq (1 - 2c_\gamma\lambda)^l \, \mathrm{KL}(\mu_0|\pi)$.

Problem: not possible to combine (1) and (2). (2) fails to hold if k is too regular with respect to π (e.g. k bounded, π Gaussian) [Duncan et al., 2019] . Some working examples in dimension 1, open question in greater dimensions...

First Experiments (d=1)

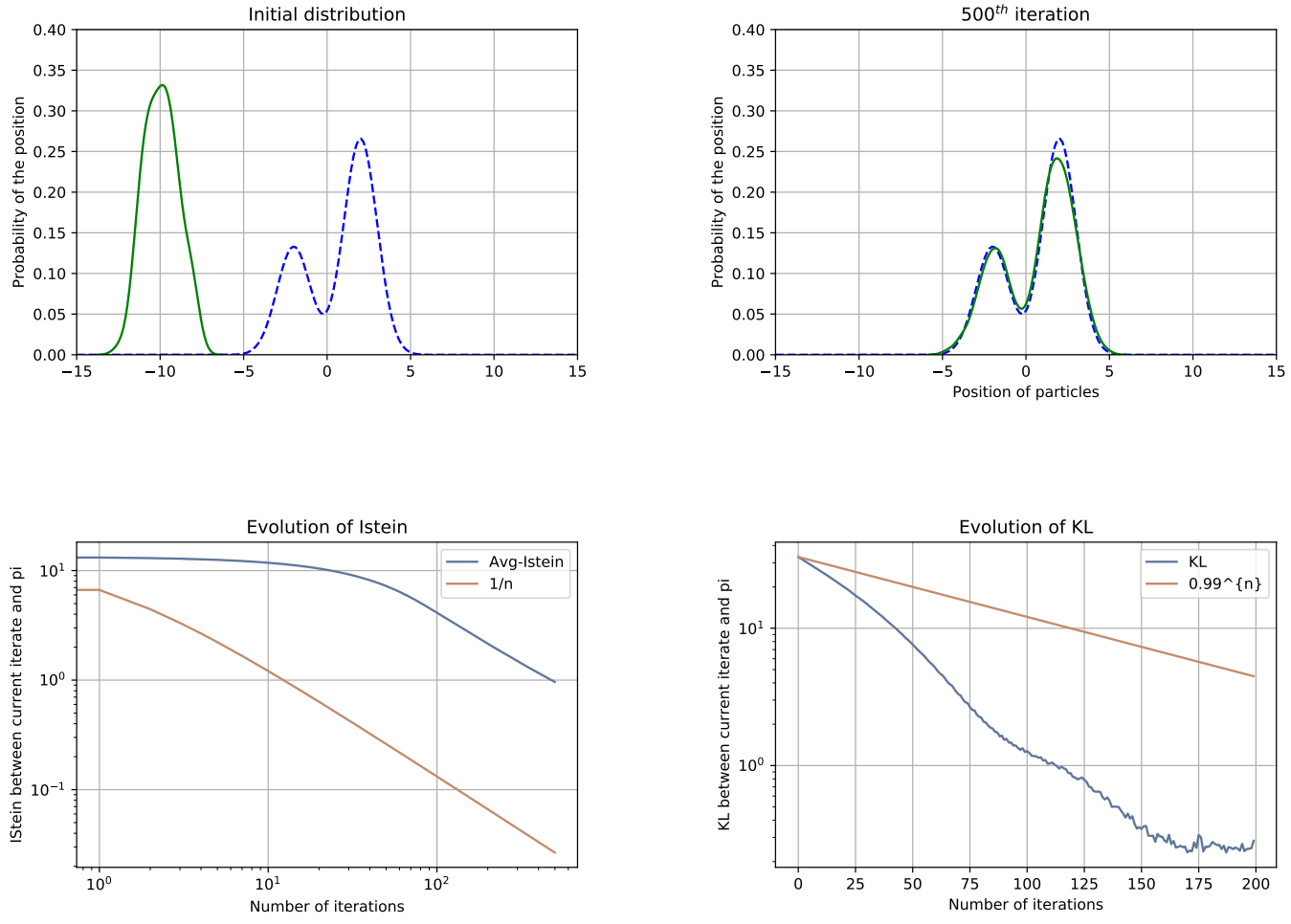
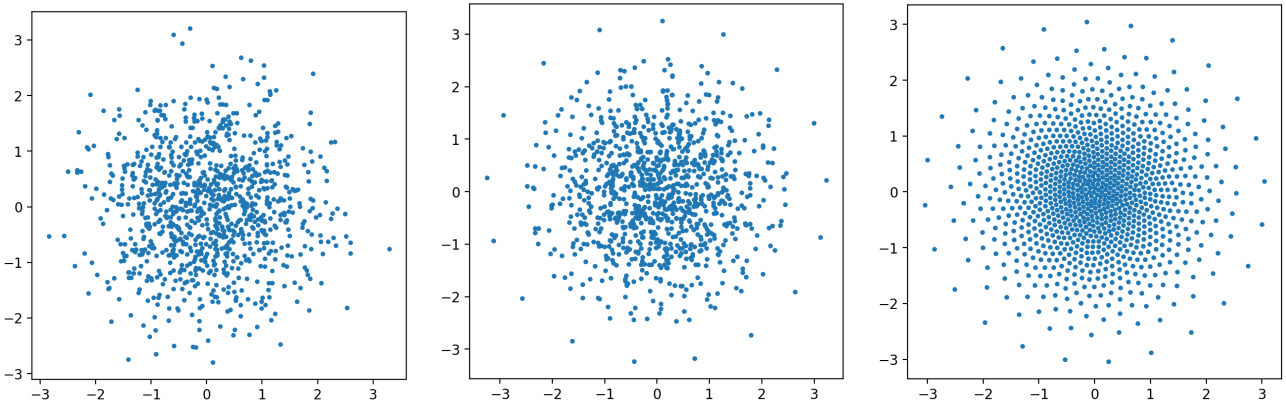


Figure: The particle implementation of the SVGD algorithm illustrates the convergence of $\mathrm{KSD}^2(\mu_l^n|\pi)$ and $\mathrm{KL}(k \star \mu_l^n|\pi)$ to 0.

Open question 2: SVGD quantisation

The quality of a set of points $(x^1, \ldots, x^n)$ can be measured by the integral approximation error:

$$E(x_1, \ldots, x_n) = \left| \frac{1}{n} \sum_{i=1}^{n} f(x^i) - \int_{\mathbb{R}^d} f(x) d\pi(x) \right|. \tag{1}$$



(a) i.i.d. (b) SVGD Gaussian k (c) SVGD Laplace k

For i.i.d. points or MCMC iterates, (1) is of order $n^{-\frac{1}{2}}$. Can we bound (1) for SVGD final states?

Accurate quantization of measures via interacting particle-based optimization. Xu, L., Korba, A., Slepčev, D. ICML 2022.

Outline

A lot of problems previously came from the fact that the KL is not defined for discrete measures μ_n. Can we consider functionals that are well-defined for μ_n?

A lot of problems previously came from the fact that the KL is not defined for discrete measures μ_n. Can we consider functionals that are well-defined for μ_n?

Remember the Kernel Stein discrepancy of μ relative to π:

$$\mathrm{KSD}^2(\mu|\pi) = \left\| P_{\mu,k} \nabla \log\left(\frac{\mu}{\pi}\right) \right\|^2_{\mathcal{H}_k}, \; P_{\mu,k} : f \mapsto \int f(x) k(x,.) d\mu(x).$$

With several integration by parts we have:

$$\begin{aligned}
\mathrm{KSD}^2(\mu|\pi) &= \left\| P_{\mu,k} \nabla \log\left(\frac{\mu}{\pi}\right) \right\|^2_{\mathcal{H}_k} \\
&= \int\int \nabla \log\left(\frac{\mu}{\pi}(x)\right) \nabla \log\left(\frac{\mu}{\pi}(y)\right) k(x,y) d\mu(x) d\mu(y) \\
&= \iint \nabla \log \pi(x)^T \nabla \log \pi(y) k(x,y) + \nabla \log \pi(x)^T \nabla_2 k(x,y) \\
&\quad + \nabla_1 k(x,y)^T \nabla \log \pi(y) + \boldsymbol{\nabla} \cdot_1 \nabla_2 k(x,y) d\mu(x) d\mu(y) \\
&:= \iint k_\pi(x,y) d\mu(x) d\mu(y).
\end{aligned}$$

can be written in closed-form for discrete measures μ.

KSD Descent - algorithms [Korba et al., 2021]

We propose two ways to implement KSD Descent:

Algorithm 1 KSD Descent GD

Input: initial particles $(x_0^i)_{i=1}^N \sim \mu_0$, number of iterations M, step-size γ

for $n = 1$ **to** M **do**

$$[x_{n+1}^i]_{i=1}^N = [x_n^i]_{i=1}^N - \frac{2\gamma}{N^2} \sum_{j=1}^N [\nabla_2 k_\pi(x_n^j, x_n^i)]_{i=1}^N,$$

end for

Return: $[x_M^i]_{i=1}^N$.

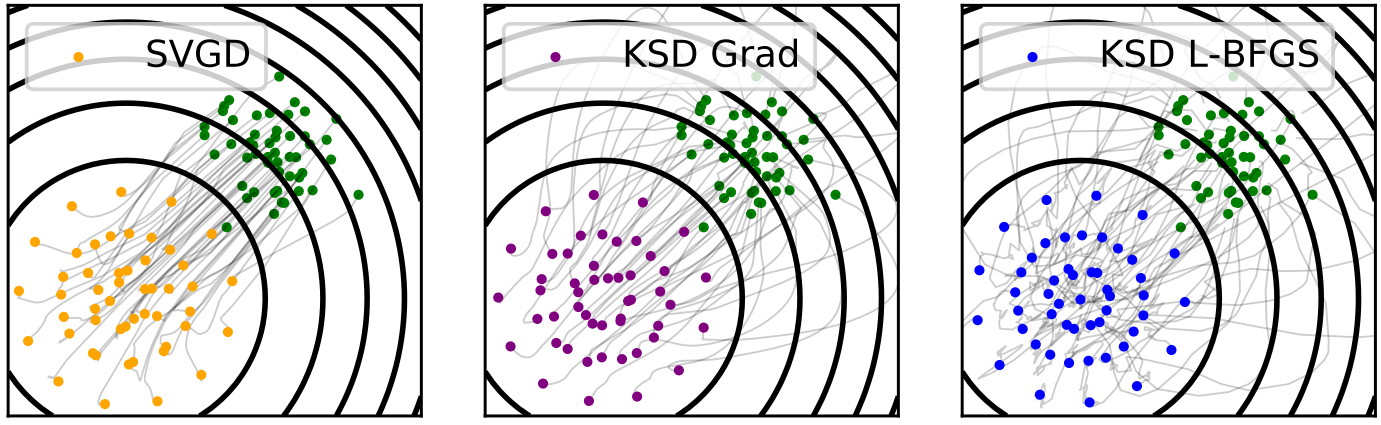
Algorithm 2 KSD Descent L-BFGS

Input: initial particles $(x_0^i)_{i=1}^N \sim \mu_0$, tolerance tol

Return: $[x_*^i]_{i=1}^N = \text{L-BFGS}(L, \nabla L, [x_0^i]_{i=1}^N, \text{tol})$.

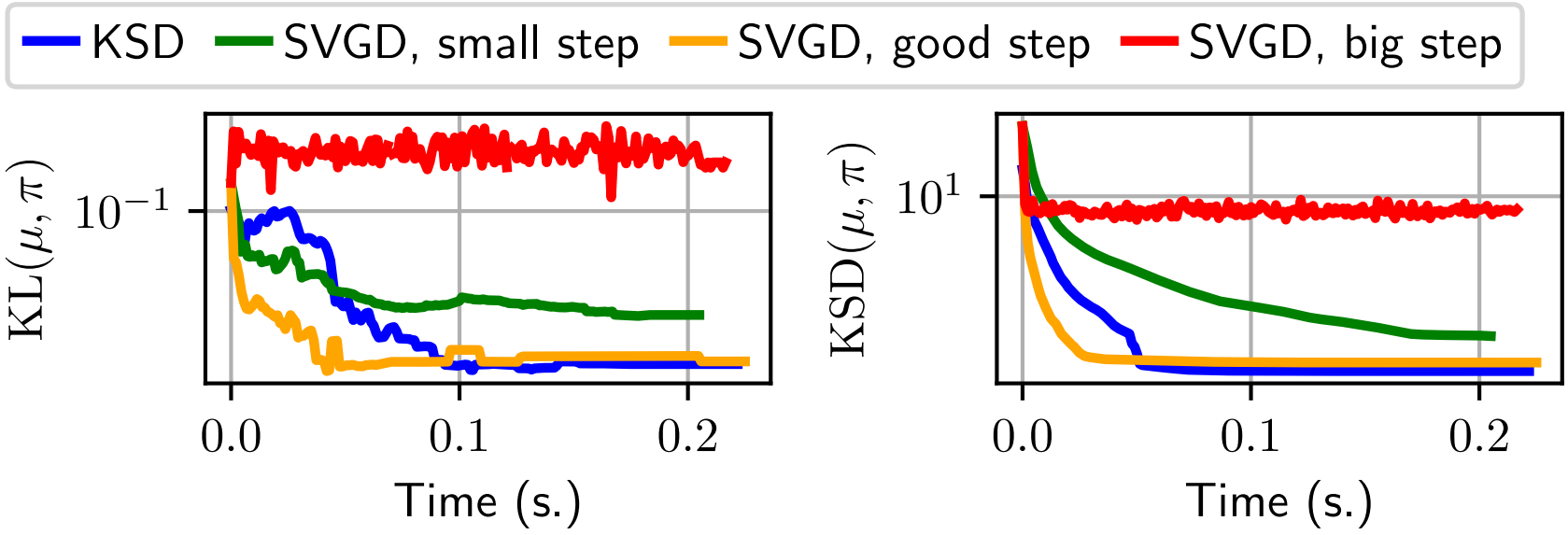
L-BFGS [Liu and Nocedal, 1989] is a quasi Newton algorithm that is faster and more robust than Gradient Descent, and **does not require the choice of step-size!**

Toy experiments - 2D standard gaussian



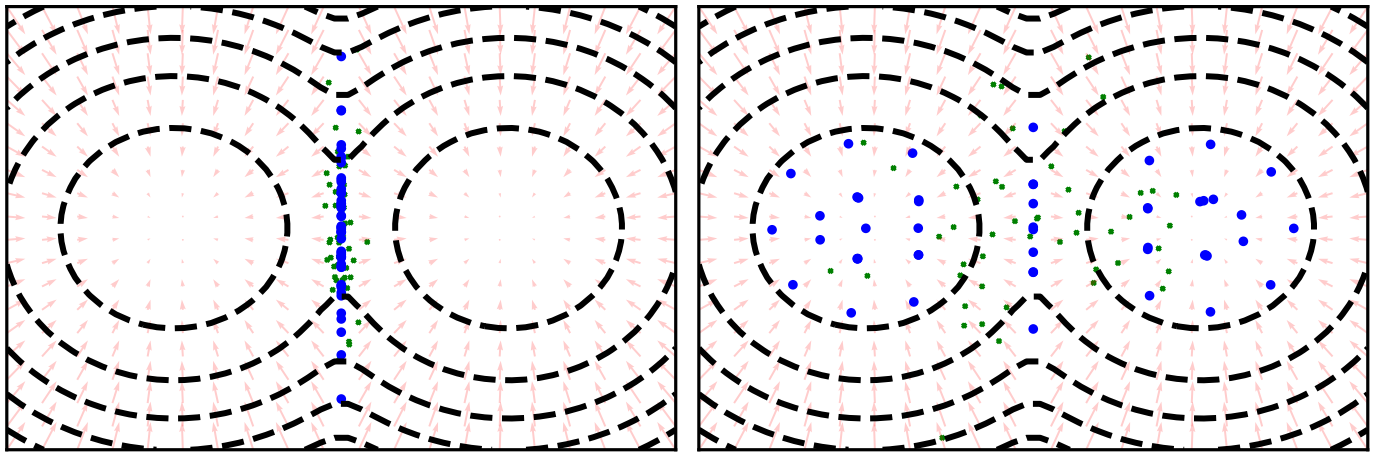
The green points represent the initial positions of the particles.
The light grey curves correspond to their trajectories.

SVGD vs KSD Descent - importance of the step-size



Convergence speed of KSD and SVGD on a Gaussian problem in 1D, with 30 particles.

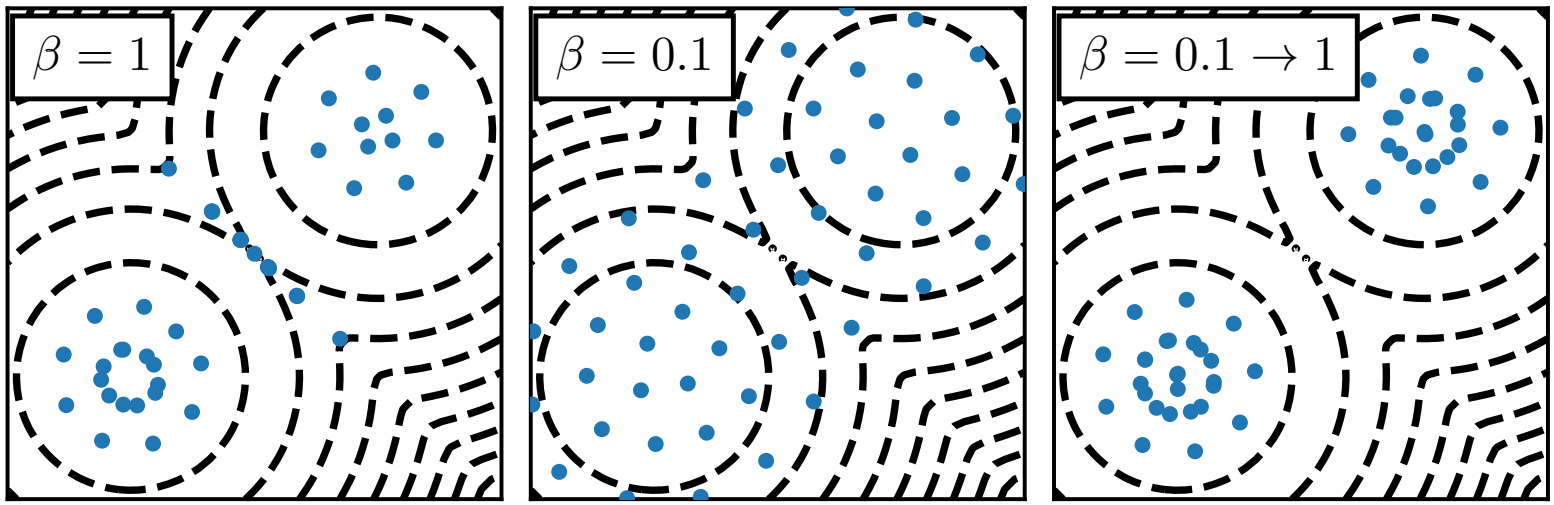
2D mixture of (isolated) Gaussians - failure cases



The green crosses indicate the initial particle positions
the blue ones are the final positions
The light red arrows correspond to the score directions.

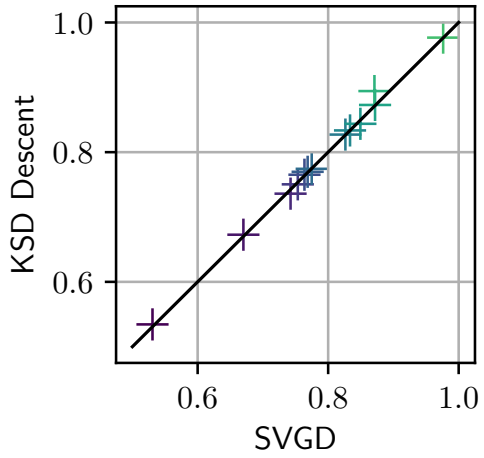
Isolated Gaussian mixture - annealing

Add an inverse temperature variable β : $\pi^{\beta}(x) \propto \exp(-\beta V(x))$, with $0 < \beta \leq 1$ (i.e. multiply the score by β.)

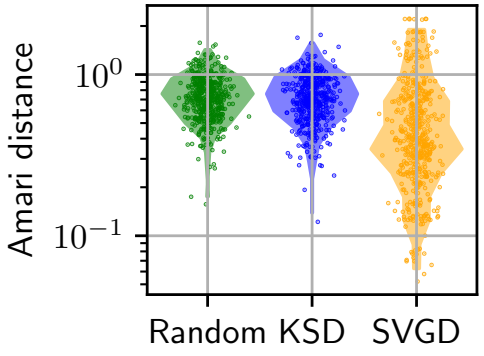


This is a hard problem, even for Langevin diffusions, where tempering strategies also have been proposed [Lee et al., 2018].

Real world experiments (10 particles)

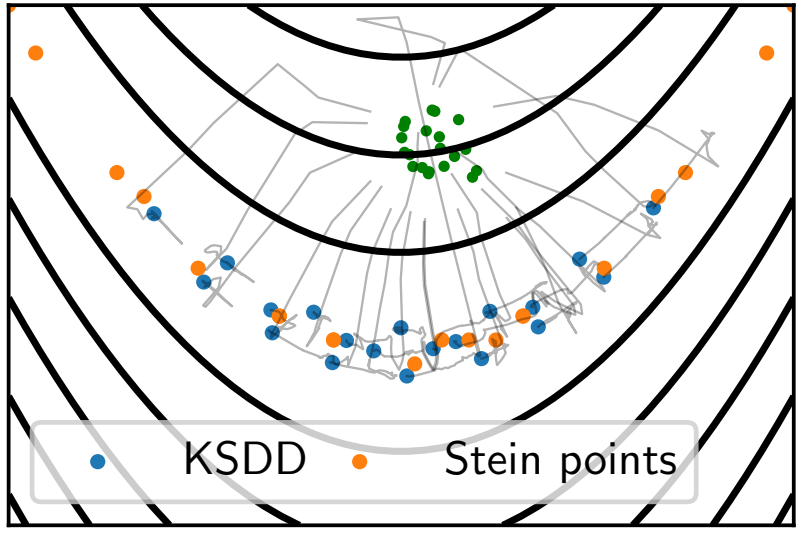


Bayesian logistic regression.
Accuracy of the KSD descent and SVGD for 13 datasets ($d \approx 50$).
Both methods yield similar results. KSD is better by 2% on one dataset.
Hint: convex likelihood.



Bayesian ICA.
Each dot is the Amari distance between an estimated matrix and the true unmixing matrix ($d \leq 8$).
KSD is not better than random.
Hint: highly non-convex likelihood.

So.. when does it work?



Comparison of KSD Descent and Stein points on a “banana” distribution. Green points are the initial points for KSD Descent. Both methods work successfully here, **even though it is not a log-concave distribution.**

We posit that KSD Descent succeeds because **there is no saddle point in the potential.**

Theoretical properties of KSD flow

Stationary measures:

- we show that if a stationary measure μ_∞ is full support, then $\mathcal{F}(\mu_\infty) = 0$.
- however, we also show that if *supp*$(\mu_0) \subset \mathcal{M}$, where $\mathcal{M}$ is a plane of symmetry of π, then for any time t it remains true for μ_t: *supp*$(\mu_t) \subset \mathcal{M}$.

Theoretical properties of KSD flow

Stationary measures:

- we show that if a stationary measure μ_∞ is full support, then $\mathcal{F}(\mu_\infty) = 0$.
- however, we also show that if *supp*$(\mu_0) \subset \mathcal{M}$, where $\mathcal{M}$ is a plane of symmetry of π, then for any time t it remains true for μ_t: *supp*$(\mu_t) \subset \mathcal{M}$.

Explain convergence in the log-concave case? again an open question:

- the KSD is not geodesically convex
- it is not strongly geo convex near the global optimum π
- convergence of the continuous dynamics can be shown with a functional inequality, but which does not hold for discrete measures

KSD quantization

Theorem (Xu, K., Slečev): Assume that k is a Gaussian kernel and that $\pi \propto \exp(-U)$ where $U \in C^\infty(\mathbb{R}^d)$ is such that $U(x) > c_1|x|$ for large enough x, there exists polynomial f with degree m such that $\|\partial^\alpha U(x)\| \leq f(x)$ for all $1 \leq |\alpha| \leq d$. Then there exist points $x_1, ..., x_n$ such that $\mu_n = \sum_{i=1}^n \delta_{x_i}$ satisfies:

$$\mathrm{KSD}(\mu_n|\pi) \leq C_d \frac{(\log n)^{\frac{6d+2m+1}{2}}}{n}.$$

Note that for Gaussian mixtures π satisfies the conditions of the theorem.

Conclusion

- Mixing kernels and Wasserstein gradient flows enable to design deterministic interacting particle systems

Conclusion

- Mixing kernels and Wasserstein gradient flows enable to design deterministic interacting particle systems
- They can provide a better approximation of the target for a finite number of particles

Conclusion

- Mixing kernels and Wasserstein gradient flows enable to design deterministic interacting particle systems
- They can provide a better approximation of the target for a finite number of particles
- Theory does not match practice yet

Conclusion

- Mixing kernels and Wasserstein gradient flows enable to design deterministic interacting particle systems
- They can provide a better approximation of the target for a finite number of particles
- Theory does not match practice yet
- Numerics can be improved, via perturbed dynamics, change of geometry...

Python package to try KSD descent and SVGD:
pip install ksddescent
website: `pierreablin.github.io/ksddescent/`

```
>>> import torch
>>> from ksddescent import ksdd_lbfgs
>>> n, p = 50, 2
>>> x0 = torch.rand(n, p)  # start from uniform distribution
>>> score = lambda x: x  # simple score function
>>> x = ksdd_lbfgs(x0, score)  # run the algorithm
```

Thank you!

References:

- *A non-asymptotic Analysis of Stein Variational Gradient Descent.* Korba A., Salim A., Arbel, M., Luise. G, Gretton, A. Neurips 2020.
- *Kernel Stein Discrepancy Descent.* Korba, A., Aubin-Frankowski, P-C., Majewski, S., Ablin, P. ICML 2021.
- *Accurate quantization of measures via interacting particle-based optimization.* Xu, L., Korba, A., Slepčev, D. ICML 2022.

References I


Ambrosio, L., Gigli, N., and Savaré, G. (2008).
Gradient flows: in metric spaces and in the space of probability measures.
Springer Science & Business Media.


Ba, J., Erdogdu, M. A., Ghassemi, M., Sun, S., Suzuki, T., Wu, D., and Zhang, T. (2021).
Understanding the variance collapse of svgd in high dimensions.
In *International Conference on Learning Representations*.


Carrillo, J. A., Craig, K., and Patacchini, F. S. (2019).
A blob method for diffusion.
Calculus of Variations and Partial Differential Equations, 58(2):1–53.

References II


Chwialkowski, K., Strathmann, H., and Gretton, A. (2016).
A kernel test of goodness of fit.
In *International conference on machine learning*.


Duncan, A., Nüsken, N., and Szpruch, L. (2019).
On the geometry of stein variational gradient descent.
arXiv preprint arXiv:1912.00894.


Gorham, J. and Mackey, L. (2017).
Measuring sample quality with kernels.
In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1292–1301. JMLR.org.

References III


Izmailov, P., Vikram, S., Hoffman, M. D., and Wilson, A. G. G. (2021).
What are bayesian neural network posteriors really like?
In *International Conference on Machine Learning*, pages 4629–4640. PMLR.


Jordan, R., Kinderlehrer, D., and Otto, F. (1998).
The variational formulation of the fokker–planck equation.
SIAM journal on mathematical analysis, 29(1):1–17.


Korba, A., Aubin-Frankowski, P.-C., Majewski, S., and Ablin, P. (2021).
Kernel stein discrepancy descent.
In *International Conference on Machine Learning*, pages 5719–5730. PMLR.

References IV


Korba, A., Salim, A., Arbel, M., Luise, G., and Gretton, A. (2020).
A non-asymptotic analysis for stein variational gradient descent.
arXiv preprint arXiv:2006.09797.


Lee, H., Risteski, A., and Ge, R. (2018).
Beyond log-concavity: Provable guarantees for sampling multi-modal distributions using simulated tempering langevin monte carlo.
Advances in neural information processing systems, 31.


Liu, D. C. and Nocedal, J. (1989).
On the limited memory BFGS method for large scale optimization.
Mathematical programming, 45(1-3):503–528.

References V


Liu, Q. (2017).
Stein variational gradient descent as gradient flow.
In *Advances in neural information processing systems*, pages 3115–3123.


Liu, Q. and Wang, D. (2016).
Stein variational gradient descent: A general purpose bayesian inference algorithm.
In *Advances in neural information processing systems*, pages 2378–2386.


Lu, J., Lu, Y., and Nolen, J. (2019).
Scaling limit of the stein variational gradient descent: The mean field regime.
SIAM Journal on Mathematical Analysis, 51(2):648–671.

References VI


Steinwart, I. and Christmann, A. (2008).
Support vector machines.
Springer Science & Business Media.


Wenliang, L. K. and Kanagawa, H. (2020).
Blindness of score-based methods to isolated components and mixing proportions.
arXiv preprint arXiv:2008.10087.