Stein Variational Gradient Descent: Convergence and New Algorithms

Aymane El Firdoussi[1]

[1]Télécom Paris, Institut Polytechnique de Paris

Abstract

Bayesian inference problems, Monte Carlo estimators and a lot of tasks in Machine learning require sampling from probability distributions. One of the methods that was recently discovered by Qiang Liu and Dilin Wang in 2016 is Stein Variational Gradient Descent (SVGD), which is a deterministic algorithm for sampling from a target density π known up to a multiplicative constant, i.e $\pi \propto \exp\{-F\}$. In this paper, we will recall the properties of this algorithm, then we will move on to prove the weak convergence of the generated process to the target distribution in the population limit under some non-restrictive conditions, and finally we will propose some new algorithms, we called one of them the **Stochastic SVGD**, that can increase the performances of ordinary SVGD and most importantly enables us to simulate correctly from complex distributions such as Gaussion mixtures, where most of known algorithms fail to sample from.

Acknowledgment

First of all, I would like to express my deep gratitude to Prof. **Pascal Bianchi** for his unwavering support throughout this project, and to all people who made this achievement possible.

1 Introduction

Sampling and Variational Inference are the most common paradigms for extracting information from a target distribution arising in many Machine learning tasks, including Bayesian Machine learning, where the distribution of interest is the posterior distribution of the parameters. In most of the time, the posterior distribution is generally difficult to compute due to the presence of an intractable integral, and often takes this form

$$\pi(x) \propto e^{-F(x)}$$

Where $F : \mathbb{R}^d \longrightarrow \mathbb{R}$ is a function called the potential function.

In this context, many algorithms were developed to solve this problem. Their applications are so vast and basically touch any current domain of research and industry. For example, biologists need such algorithms to simulate for example the propagation of an epidemic knowing its law, and researchers in finance do need sampling as well to study the behaviour of some processes or to apply Monte-Carlo estimations, and so on. The most common sampling algorithms are Markov Chain Monte Carlo (MCMC) methods, which generate a Markov Chain process that has the target probability π as a stationary distribution. And one of the most popular MCMC method is the Langevin algorithm, which performs a gradient descent algorithm on the Kullback-Leibler divergence in the Wasserstein space (see Otto et al. [1]). This algorithm is so powerful, but requires the potential F to be convex. If it is the case, then we are sure that this algorithm will converge to our target distribution since the Kullback-Leibler divergence will be geodesically convex, and therefore by performing a gradient descent on it, we are sure to converge to its global minimum. We recall that the Kullback-Leibler divergence has this nice property: $\forall \mu \in \mathcal{P}_2(\mathbb{R}^d)$

$$KL(\mu|\pi) \geq 0$$

And that :

$$KL(\mu|\pi) = 0 \Longleftrightarrow \mu = \pi$$

Therefore, the global minimum of the KL divergence is achieved in π.
Unfortunatly, it is not guaranteed that this algorithm converges to π when F is not convex ! In fact, if π is a Gaussian mixture, then minimizing the KL divergence will not lead to this distribution. This is illustrated in the following figure:

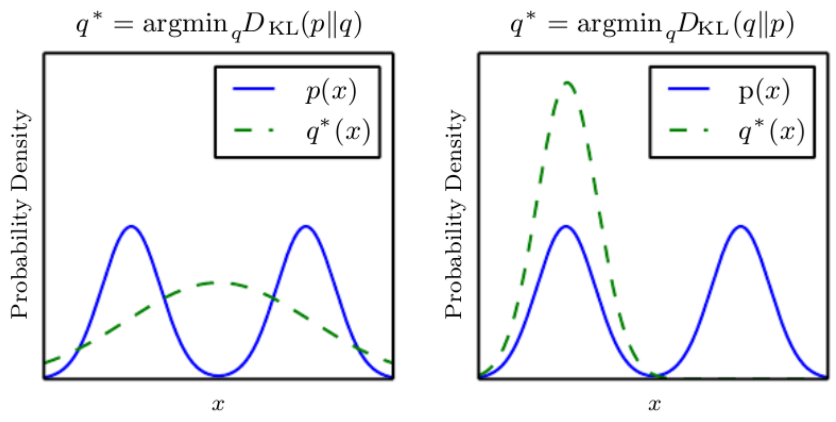


Figure 1: Minimizing the KL divergence (Deep Learning: Goodfellow et al.)

Recently in 2016, Qiang Liu and Dilin Wang have introduced a new particle based optimisation algorithm that they call Stein Variational Gradient Descent (SVGD) (see [10]). It uses a set of interacting particles to approximate the target distribution, and applies iteratively to these particles a form of gradient descent of the relative en-

tropy, where the descent direction is restricted to belong to a unit ball in a Reproducing Kernel Hilbert space (RKHS). Unlike typical Monte Carlo algorithms that rely on randomness for approximation, SVGD constructs a set of points (or particles) by iteratively applying deterministic updates that is constructed to optimally decrease the KL divergence to the target distribution at each iteration.
The empirical performance of this algorithm and its variants have been largely demonstrated in various tasks in machine learning such as Bayesian inference, learning deep probabilistic models, or reinforcement learning. In the population limit (limit of infinite particles), the algorithm is known to converge to the target distribution under appropriate growth assumptions on the potential.
The literature on theoretical properties of SVGD is scarce compared to that of Langevin algorithm, and limited to a few recent works. Many improvements on the SVGD algorithm have been recently discovered, but they are still poor, hard to implement, and sometimes limit the use of the algorithm to some really specific target densities.
In this article, we will prove the convergence of SVGD in the population limit using a property of the Kernelized Stein Discrepancy that we will detail later, and then we will propose some new algorithms that we invented and that seem to improve the performances of SVGD at least in terms of complexity.

Notations

- For any Hilbert space, we denote by $\langle .,. \rangle_H$ the inner product defined in H and by $\|.\|_H$ its related norm.

- $C_0(\mathcal{X})$ denotes the set of continuous functions from $\mathcal{X}$ to $\mathbb{R}$ vanishing at infinity.
- $C^1(\mathcal{X},\mathcal{Y})$ denotes the set of continuously differentiable functions from $\mathcal{X}$ to $\mathcal{Y}$.
- If ϕ $\in$C$^1(\mathcal{X},\mathbb{R})$, its gradient is denoted by $\nabla\phi$, and if ϕ $\in$C$^1(\mathcal{X},\mathcal{X})$, its Jacobian is denoted by Jϕ, which is a d x d matrix.
- For any matrix $A \in \mathcal{M}_{d,d}(\mathbb{R})$, we define its operator norm by:

$$|||A||| = \sup\{||Ax||, s.t| \, ||x|| = 1\}$$

- $\mathcal{P}_p(\mathcal{X})$ is the set of probability measures μ over $\mathcal{X}$ with finite p^{th} moment, which means: $\int ||x||^p d\mu(x) < \infty$
- The image (or pushforward) measure of μ is denoted as $T_{\#}\mu$
- Sometimes in this article, we will confound measure and density, for example μ can sometimes refer to the measure, and sometimes to its density.

2 Construction of SVGD

Before tackling the proof of the convergence of SVGD, let us recall some basics properties about this algorithm.
We recall here the definition of the Kullback-Leibler divergence, which measures how two probability distributions are far from each other. However, it cannot be considered as a distance between measures since it is not symmetric. This quantity will be very useful along our study.

Definition 2.1 (KL divergence). *The Kullback-Leibler divergence between two probability measures μ and π is :*

$$KL(\mu|\pi) = \int \log\left(\frac{\mu(x)}{\pi(x)}\right)\mu(x)\mathrm{d}x \tag{1}$$

This quantity is always well defined and takes values in [0,+∞], and if: $\mu << \pi$, then it is finite.

Now, let π be a probability measure of interest with a positive, (weakly) differentiable density on a open set $X \subset \mathbb{R}^d$. Our goal is to approximate π with a set of particles $(x_i)_{i=1}^n$ whose empirical measure $\widehat{\mu_n}(x) = \frac{1}{n}\sum_{i=1}^n \delta_{x_i}(x)dx$ weakly converges to π as $n \longrightarrow +\infty$.
To achieve this, we initialize the particles $(x_0^i)_{i=1}^N$ with some distribution μ_0, and update them via the map:

$$T(x) = x + \epsilon\phi(x)$$

where ϵ is a small step size, and $\phi(x)$ is a velocity field. Updating the particles with the map T means that we apply the induction relation:

$$\forall n \in \mathbb{N},\, \forall i \in \{1, ..., N\}\;\; x_{n+1}^i = T(x_n^i)$$

We want to chose ϕ in a way to maximally decrease the KL divergence between the particle distribution $\widehat{\mu_n}$ and the target measure. To do so, we will minimize the directional derivative of the KL divergence between $T_{\#}\mu$ and π in direction ϕ, where μ is an arbitrary measure in $\mathcal{P}_2(\mathbb{R}^d)$.

Definition 2.2 (Directional derivative). *The directional derivative of a function f : $\mathbb{R}^d \longrightarrow \mathbb{R}$ in direction u (a unit vector) is the slope of the function in this direction. In other words, it is the derivative of the function $f(x + \epsilon u)$ with respect to ϵ, evaluated at $\epsilon = 0$. Using the chain rule, we can easily find that:*

$$\frac{\partial}{\partial\epsilon} f(x + \epsilon u)|_{\epsilon=0} = \langle u, \nabla_x f(x)\rangle$$

To minimize f, we would like to find the direction in which f decreases the fastest. We can do this using the directional derivative, and then solving the following optimization problem:

$$\min_{u,\|u\|^2=1} \langle u, \nabla_x f(x) \rangle \tag{2}$$

If the norm is the L^2 norm, then we have a closed form solution. To exhibit it, let us examine the case of $u \in \mathbb{R}^2$. It is well known that the problem (2) is equivalent to:

$$\min_{u,\|u\|_2^2=1} \langle u, \nabla_x f(x) \rangle = \min_{u,\|u\|_2^2=1} \|u\|_2 \|\nabla_x f(x)\|_2 \cos(\theta)$$

where θ is the angle between u and the gradient. Using the fact that $\|u\|_2 = 1$ and ignoring the factors that do not depend on u, the problem simplifies to

$$\min_{u,\|u\|_2=1} \cos(\theta)$$

And this is minimized when u points in the opposite direction of the gradient (to have $\cos(\theta) = -1$). Hence the solution u is proportional to $-\nabla_x f(x)$. And this is how we got the gradient descent algorithms !
One can generalize this result into the space L^2, and prove that the solution to the optimization problem (2) is:

$$u = -\frac{\nabla_x f(x)}{\|\nabla_x f(x)\|_{L^2}}$$

And to make this solution easier and practical, we just take:

$$u = -\nabla_x f(x) \tag{3}$$

This is known as the method of steepest descent or gradient descent, which proposes to do the updates:

$$T(x) = x - \epsilon \nabla_x f(x)$$

We will use this directional derivative to construct our main optimization problem. Now we will introduce another notion that will be useful in our analysis.

Definition 2.3 (The pushforward measure)**.** *Let Y be a real random vector following a distribution μ. Let T be a mapping $T : \mathbb{R}^d \longrightarrow \mathbb{R}^d$, and $Z = T(Y)$. Then Z has distribution $T_{\#}\mu$ called the pushforward distribution of μ, or the the image measure of μ by T, which is defined by:*

$$\forall A \in \mathcal{B}(\mathbb{R}^d) \;\; T_{\#}\mu(A) = \mu(T^{-1}(A))$$

Now let us formalize our problem. Let $X \sim \mu$, our goal is to apply a transformation $T(x) = x + \epsilon\phi(x)$ on X in order to "move" the distribution μ towards the target π. Then, one way to do that is by determining the velocity field ϕ that maximizes the decrease of the Kullback-Leibler (KL) divergence between $T_{\#}\mu$ and π. This is equivalent to finding the direction ϕ in which the KL decreases the fastest. Hence, we can use the directional derivative, and this leads us to the following optimization problem (see (2)):

$$\max_{\phi\in\mathcal{H}}\left\{-\frac{\mathrm{d}}{\mathrm{d}\epsilon}KL(T_{\#}\mu|\pi)|_{\epsilon=0},\ s.t\ \|\phi\|_{\mathcal{H}}\leq 1\right\} \tag{4}$$

Using the chain rule, and that the gradient of the KL divergence in the Wasserstein space is: $\nabla_W KL(\mu|\pi) = \nabla\log\left(\frac{\mu}{\pi}\right)$, we obtain that (see Appendix for the proof):

$$\frac{\mathrm{d}}{\mathrm{d}\epsilon}KL(T_{\#}\mu|\pi)|_{\epsilon=0} = \int\langle log(\pi(x)),\phi(x)\rangle + div(\phi(x))\mathrm{d}\mu(x) = \mathbb{E}_{\mu}[S_{\pi}\phi] \tag{5}$$

Where

$$S_{\pi}\phi(x) = \langle log(\pi(x)),\phi(x)\rangle + div(\phi(x)) \tag{6}$$

S_{π} is a linear operator, called the Stein operator, that maps a vector-valued function ϕ to a scalar-valued function $S_{\pi}\phi$. Therefore, the optimization (4) becomes:

$$SD(\mu|\pi) = \max_{\phi\in\mathcal{H}}\{\mathbb{E}_{\mu}[S_{\pi}\phi]\ s.t\ \|\phi\|_{\mathcal{H}}\leq 1\} \tag{7}$$

This quantity is called the Stein discrepancy, which provides a discrepancy measure between the measures μ and π since:

$$SD(\mu|\pi) > 0\ if\ \mu\neq\pi$$

And

$$SD(\mu|\pi) = 0 \iff \mu = \pi$$

Remark that we want to find a maximum ϕ in a certain space $\mathcal{H}$. We did not precise yet the nature of this space. We will see that the choice of this space is what makes the difference between Langevin algorithm and SVGD.

When $\mathcal{H} = L^2_{\mu}(\mathbb{R}^d)$, then the optimal velocity field ϕ^* that satisfies the optimization problem (7) is, as we have shown in (3) $\phi^* = -\nabla\log\left(\frac{\mu}{\pi}\right)$, which is the gradient of the KL divergence in the Wasserstein space $\mathbb{W}_2(\mathbb{R}^d)$. Hence, the best map T to apply on μ (or equivalently the random vector Y) in order to maximize the decrease of the KL is:

$$T(x) = x - \epsilon\nabla\log\left(\frac{\mu(x)}{\pi(x)}\right)$$

And this is exactly the gradient descent mapping on the Kullback-Leibler divergence. Applying this mapping many times generates a Markov Chain $(X_n)_n$ where $X_0\sim\mu_0$, and

$$X_{n+1} = T(X_n) = X_n - \epsilon\nabla\log\left(\frac{\mu_n(X_n)}{\pi(X_n)}\right) \tag{WGD}$$

This algorithm is called Wasserstein Gradient descent (WGD). This algorithm is not implementable in practice, since it requires knowing the distributions $(\mu_n)_{n\geq 1}$, which is a very hard task. However, Otto et al.[1] showed that the Langevin algorithm does a gradient descent on the KL divergence, hence Langevin is actually WGD. We recall that Langevin algorithm is the following:

$$X_{n+1} = X_n - \epsilon \nabla F(X_n) + \sqrt{2\epsilon}\xi_n \tag{LA}$$

where $\epsilon > 0$ is a step-size (small), and ξ_n is a standard normal variable, i.e $\xi_n \sim \mathcal{N}(0,1)$.

However, we said in the introduction that this algorithm is not so efficient unless F is convex.
Now let us take an interesting case of the space $\mathcal{H}$ which will lead us to know the origin of SVGD.
Let K be a reproducing kernel, and $\mathcal{H}$ its associated Reproducing Kernel Hilbert Space (RKHS), which means that if $f \in \mathcal{H}$ then $\exists \alpha_1, ..., \alpha_n \in \mathbb{R}$ and $\exists x_1, ..., x_n \in \mathbb{R}^d$ such that:

$$\forall x \in \mathbb{R}^d \;\; f(x) = \sum_{i=1}^{n} \alpha_i K(x_i, x)$$

Let us go back now to our optimization problem (7). If we take now $\mathcal{H}$ a RKHS, then the optimization leads to a closed form solution. In fact, it can be shown that the optimal solution of (7) is :

$$\phi^*_{\mu,\pi}(y) \propto \mathbb{E}_{X\sim\mu}[S_\pi \otimes K(X,y)] \tag{8}$$

Where

$$S_\pi \otimes K(X,y) := -\nabla F(X)K(X,y) + \nabla_1 K(X,y)$$

and ∇_1 means the gradient with respect to the first argument. Hence, we obtained another velocity field by restricting $\mathcal{H}$ to a RKHS, and in that case, the Stein Discrepancy (SD) becomes a Kernelized Stein Discrepancy (KSD):

$$KSD(\mu|\pi) = \sqrt{\mathbb{E}_{X\sim\mu}[S_\pi\phi *_{\mu,\pi}(X)]} = \|\phi^*_{\mu,\pi}\|_{\mathcal{H}} \tag{KSD}$$

And we have that:

$$\frac{\mathrm{d}}{\mathrm{d}\epsilon} KL(T_\# \mu|\pi)|_{\epsilon=0} = -KSD(\mu|\pi)^2 \tag{9}$$

This last inequality will be useful in the next section where we will deal with convergence. We now have a new velocity field $\phi^*_{\mu,\pi}$ which we will adopt in our mapping

$$T(x) = x + \epsilon\phi^*_{\mu,\pi}(x)$$

to construct finally an algorithm doing the updates: $X_{n+1} = T(X_n)$ which will be called SVGD.

Algorithm 1 Stein Variational Gradient descent (Liu and Wang, 2016)

```
Require: a set $X_0^1, ..., X_0^N \in \mathcal{X}$ of N particles, a kernel K, the number of iterations M,
    and a step size $\epsilon > 0$
1: for n = 1, 2, ...., M do
2:     for i = 1,2,..,N do
3:         $X_{n+1}^i = X_n^i - \frac{\epsilon}{N}\sum_{j=1}^{j=N} K(X_n^j, X_n^i)\nabla F(X_n^j) - \nabla_1 K(X_n^j, X_n^i)$
```

Remark that to get this algorithm, we approximated the expectation $\mathbb{E}_\mu$ with its Monte-Carlo estimator.

3 Convergence of SVGD in the population limit

Now that we recalled the construction of SVGD, let us now study its convergence, and provide a clear proof of it.
Denote by $\widehat{\mu}_l^n$ the empirical measure of the SVGD with n particles in time step l. We will study both convergences when $l \to +\infty$ and when $n \to +\infty$. In fact we will show that : $\lim_{l\to+\infty}\lim_{n\to+\infty}\widehat{\mu}_l^n = \pi$ under some conditions. We want to precise as well that the order of the limits cannot be inverted (according to our current knowledge), we will talk about this specific point at the end of this section.
First we will assume that we have the weak convergence of the empirical measures $\hat{\mu}_l^n$ to μ_l the continuous ones when $n \to +\infty$ (it was also shown in Liu 2017 [3]). This is called the population limit regime, which is the limit of infinite particles. So let us prove that μ_l converges weakly to π as $l \longrightarrow +\infty$.
Let us define the optimal transform $T_{\mu,\pi}(x) = x + \epsilon\phi^*_{\mu,\pi}(x)$ with $\phi^*_{\mu,\pi}$ defined in (8). We know that SVGD performs the following updates:

$$X_{n+1} = X_n + \epsilon\phi^*_{\mu_n,\pi}(X_n)$$

This is the Euler discretization of the following gradient flow in the continuous time limit ($\epsilon \longrightarrow 0$)

$$\frac{\mathrm{d}X_t}{\mathrm{d}t} = \phi^*_{\mu_t,\pi}(X_t) \tag{SVGF}$$

If we denote $(\mu_t)_t$ the continuous time measures obtained by applying SVGD, i.e $X_t \sim \mu_t$, this means that they satisfy the following **continuity equation**:

$$\frac{\partial\mu_t}{\partial t} + div(\phi^*_{\mu_t,\pi}\mu_t) = 0 \tag{10}$$

Using the equation (9), we have that:

$$\frac{\mathrm{d}}{\mathrm{d}t}KL(\mu_t|\pi) = -KSD(\mu_t|\pi)^2 \tag{11}$$

let us recall the definition of the weak (or narrow) convergence.

Definition 3.1 (Weak convergence)**.** *We say that a sequence of measures $(\mu_n)_n$ in $\mathcal{P}_2(\mathbb{R}^d)$ converges weakly to a probability measure μ if and only if for all bounded continuous functions f, we have when $n \to +\infty$:*

$$\int f(x)\mu_n(x)\mathrm{d}x \to \int f(x)\mu(x)\mathrm{d}x$$

In this case, we write that: $\mu_n \Longrightarrow \mu$

Korba et al.[2] showed that under an inequality called the Stein Log-Sobolev inequality, we have the convergence of SVGD in the population limit with an exponential rate. However, the problem with this inequality is that it is almost never satisfied by a distribution π. Hence, we cannot use this proof on ordinary examples.

Convergence in population limit

As we said, the Stein LSI is not satisfied by most of the distributions, so it is not fair to prove the convergence of SVGD just by using it. This is why we will now propose another proof of convergence in the infinite particle regime (population limit) that does not require strong assumptions.

Definition 3.2 (Tightness)**.** *A family of probability measures $(\mu_t)_{t\in\mathcal{X}}$ is* ***tight*** *if and only if for all $\epsilon > 0$, there exists a compact set $K_\epsilon \subset \mathcal{B}(\mathbb{R}^d)$ such that:*

$$\sup_{t\in\mathcal{X}} \mu_t(K_\epsilon) \geq 1 - \epsilon$$

which means that for an arbitrary small ϵ, we can find a compact set that contains almost all the mass associated with the family $(\mu_t)_t$.
In discrete time, we say that the sequence of measures $(\mu_n)_n$ is tight if and only if for all $\epsilon > 0$, there exists a compact set $K_\epsilon \subset \mathcal{B}(\mathbb{R}^d)$ such that:

$$\sup_{n\in\mathbb{N}} \mu_n(K_\epsilon) \geq 1 - \epsilon$$

Now we will state our **main theorem** in this article.

Theorem 1 (Convergence in population limit)**.** *Let $(\mu_n)_n$ the sequence of measures associated to the particles generated by SVGD at each time step $n \in \mathbb{N}$. If F is* ***coercive***

and $KL(\mu_0|\pi) < +\infty$, then μ_n converges weakly to π, i.e

$$\mu_n \Longrightarrow \pi \tag{12}$$

Before going through the proof, let us recall some basic properties in asymptotic statistics.

Lemma 2. *Let $(\mu_n)_n$ be a sequence of measures, and $F : \mathbb{R}^d \longrightarrow \mathbb{R}$ be a coercive function, i.e*

$$\lim_{\|x\| \to +\infty} F(x) = +\infty$$

Assume that

$$\sup_n \int F(x)\mu_n(x)\mathrm{d}x < +\infty \tag{13}$$

Then the family $(\mu_n)_n$ is tight, i.e $\forall \epsilon > 0$ $\exists K_\epsilon$ compact such that:

$$\sup_n \mu_n(K_\epsilon) \geq 1 - \epsilon \tag{14}$$

Proof of lemma. If we take $F(x) = \|x\|$, then F is clearly coercive. Let $\forall n \in \mathbb{N}$ $X_n \sim \mu_n$ (it is always easier to work with random vectors rather than their measures). Our main hypothesis states that $\sup_n \mathbb{E}(\|X_n\|) < +\infty$.
We have by Markov inequality:

$$\forall R \in [0, +\infty[,\ \ \mathbb{P}[\|X_n\|] > R] \leq \frac{\mathbb{E}[\|X_n\|]}{R} \leq \sup_n \frac{\mathbb{E}[\|X_n\|]}{R}$$

Hence if we take $R_\epsilon = \sup_n \frac{\mathbb{E}[\|X_n\|]}{\epsilon}$ we then have $K_\epsilon = \mathcal{B}(0_d, R_\epsilon) = \{x \ s.t \ \|x\| \leq R_\epsilon\}$ is a compact set (the closed ball of center 0_d and radius R_ϵ), and

$$\mu_n(K_\epsilon) = \mathbb{P}[\|X_n\| \leq R_\epsilon] \geq 1 - \epsilon$$

And now we can easily generalize this result to any coercive function F by taking the compact $K_\epsilon = \{x \ s.t \ F(x) \leq R_\epsilon\}$ where $R_\epsilon = \sup_n \frac{\mathbb{E}[F(X_n)]}{\epsilon}$

□

And then we need other lemmas to finally be able to prove the convergence theorem.

Lemma 3 (Prokhorov). *A sequence of probability measures is tight if and only if every subsequence of $(\mu_n)_n$ admits a further subsequence which converges weakly.*

Lemma 4 (Narrow convergence). *Let $(\mu_n)_n$ be a sequence of probability measures in $\mathcal{P}_2(\mathbb{R}^d)$. Let $\mu \in \mathcal{P}_2(\mathbb{R}^d)$. Assume that every narrowly convergent subsequence $(\mu_{\phi(n)})_n$ of $(\mu_n)_n$ has μ as a limit. Then:*

$$\mu_n \Longrightarrow \mu$$

Lemma 5 (Weak continuity of KSD). *Let $(\mu_n)_n$ be a sequence of probability measures on $\mathcal{P}_2(\mathbb{R}^d)$. We have:*

$$\lim_{n\to+\infty} KSD(\mu_n|\pi) = 0 \Longleftrightarrow \mu_n \Longrightarrow \pi \tag{15}$$

The proof of this last lemma is provided in Gorham et al. [7]. Now let us prove our convergence theorem

Proof of theorem. Let $(\mu_t)_t$ a family of probability measures associated with the (SVGF). We have shown that (see (11)):

$$\frac{\mathrm{d}}{\mathrm{d}t} KL(\mu_t|\pi) = -KSD(\mu_t|\pi)^2$$

then the function $t \longmapsto KL(\mu_t|\pi)$ is non-increasing, then

$$\forall t \in \mathbb{R}^+, \ \ KL(\mu_t|\pi) \leq KL(\mu_0|\pi) < +\infty$$

Let us consider a time discretization $(t_n)_n$ which is simply an increasing sequence in $\mathbb{R}_+$ ($t_{n+1} > t_n$ and $\lim_{n\to+\infty} t_n = +\infty$).
We have:

$$KL(\mu_{t_n}|\pi) = \int \log\left(\frac{\mu_{t_n}(x)}{\pi(x)}\right)\mu_{t_n}(x)\mathrm{d}x = \int F(x)\mu_{t_n}(x)\mathrm{d}x + \int \log(\mu_{t_n}(x))\mu_{t_n}(x)\mathrm{d}x$$

Then $\forall n \in \mathbb{N}, \ \int F(x)\mu_{t_n}(x)\mathrm{d}x < +\infty$, then $\sup_n \int F(x)\mu_{t_n}(x)\mathrm{d}x < +\infty$, then by the Lemma 2, we conclude that $(\mu_{t_n})_n$ is tight. Let $\phi : \mathbb{N} \longrightarrow \mathbb{N}$ an increasing function such that $(\mu_{t_{\phi(n)}})_n$ converges narrowly (weakly) to a measure μ^*. This measure exists by the Prokhorov's theorem (see Lemma 3). Then $\mu_{t_{\phi(n)}} \Longrightarrow \mu^*$.
In another hand, we have that

$$KL(\mu_t|\pi) - KL(\mu_0|\pi) = -\int_0^t KSD(\mu_x|\pi)\mathrm{d}x$$

Hence, since the $t \longmapsto KL(\mu_t|\pi)$ is non-increasing, and that $KL(\mu_t|\pi) \geq 0$, then the difference $KL(\mu_t|\pi) - KL(\mu_0|\pi)$ is finite for all $t \in \mathbb{R}^+$.
Then the integral $\int_0^t KSD(\mu_x|\pi)\mathrm{d}x$ is also finite. This stays true when $t \to +\infty$, hence $\liminf_t KSD(\mu_t|\pi)^2 = 0$, then by definition of the limit inferior, which is the smallest accumulation point of any sequence of $(\mu_n)_n$, there exists a sequence of time discretization $(s_n)_n$ such that $\lim_{n\to+\infty} KSD(\mu_{s_n}|\pi) = 0$
Then by Lemma 5, we get that

$$\mu_{s_n} \Longrightarrow \pi$$

Then, for every increasing function $\psi : \mathbb{N} \longrightarrow \mathbb{N}$, we have that $\mu_{s_{\psi(n)}} \Longrightarrow \pi$.

Now we can construct a function ψ such that: $\forall n \in \mathbb{N}$

$$s_{\psi(n)} \leq t_{\phi(n)} \leq s_{\psi(n+1)}$$

And then we get that:

$$0 \leq KL(\mu_{t_{\phi(n)}}|\pi) \leq KL(\mu_{s_{\psi(n)}}|\pi)$$

When n goes to $+\infty$, we have that $KL(\mu_{s_{\psi(n)}}|\pi) \to 0$ because $\mu_{s_{\psi(n)}} \Longrightarrow \pi$, and then we get that:

$$KL(\mu_{t_{\phi(n)}}|\pi) \to 0$$

And then equivalently,

$$\mu_{t_{\phi(n)}} \Longrightarrow \pi$$

And since $(\mu_{t_{\phi(n)}})$ converges to μ^*, we get then $\mu^* = \pi$
Therefore, we have shown that every convergent subsequence $(\mu_{t_{\phi(n)}})_n$ of $(\mu_{t_n})_n$ converges to π. Therefore, we finally conclude by Lemma 4 that $\forall (t_n)_n \in \mathbb{R}^{\mathbb{N}}$

$$\mu_{t_n} \Longrightarrow \pi$$

If we consider $(t_n)_n$ the discretization of the SVGD algorithm, then we have shown the convergence of this algorithm in the population limit (limit of infinite number of particles).

□

Comment on the convergence result

To the best of our knowledge, this proof of convergence in the population limit is the one with the weaker assumptions. In fact, other proofs often require strong assumptions like some inequalities that are not always satisfied.
Hence, we have shown that:

$$\lim_{l \to +\infty} \lim_{n \to +\infty} \widehat{\mu}_l^n = \pi \tag{16}$$

Which means that, in the case of the infinite number of particles, we have the convergence of their resulting distribution. However, it was not proved until now that we will always have the convergence of SVGD if we change the order of the limits in (16), which means that we don't have a theoretical proof of convergence of SVGD in the finite number of particles. It was shown by experience that SVGD gives very good approximations of the target distribution π in the finite number of particles, so in the near future, we should be able to find such a proof. This was one of the missions that we would like to have tackled in this project, but we did not have enough time for it.

4 New algorithms

It was proved in Liu 2017 [3] that SVGD performs a gradient flow on the Kullback-Leibler divergence, and of course not in the Wasserstein space $\mathbb{W}_2(\mathbb{R}^d)$ (this is Langevin), but in a new space with a specific structure called $\mathcal{H}-$Wasserstein space, where $\mathcal{H}$ is the RKHS associated with the chosen reproducing kernel K.
So, if we write $grad_{\mathcal{H}}$ the gradient in this space, then it was proved that the optimal velocity field put in this term $div(\phi^*_{\mu,\pi}.\mu)$ defined in (8) is actually the gradient of the Kullback-Leibler divergence in this $\mathcal{H}-$Wasserstein space, which means that is satisfies the following identity:

$$div(\phi^*_{\mu,\pi}.\mu_t) = grad_{\mathcal{H}} KL(\mu_t|\pi)$$

And therefore, the continuity equation can be written as follows:

$$\frac{\partial \mu_t}{\partial t} = -grad_{\mathcal{H}} KL(\mu_t|\pi) \tag{17}$$

This result is interesting because it tells us that SVGD is actually a gradient flow on a functional (the KL divergence). So SVGD behaves like Langevin but in another space, an unusual space, which properties are introduced in Liu [3].
We will use the above equation (17) to analyze the behaviour of some new algorithms that we will propose in the following paragraphs. Indeed, the continuity equation helps us see the evolution of the intermediate measures $(\mu_t)_t$ generated throughout the execution of an algorithm.

4.1 Adding noise to SVGD

We know that SVGD is a fully deterministic algorithm and does not use any randomness (apart from generating the very first samples from μ_0). So what if we add some noise to this algorithm ? Which means, (if we get inspired by Langevin) we can introduce this new algorithm that does noisy updates:

Algorithm 2 Noisy SVGD

```
Require: a set $X_0^1, ..., X_0^N \in \mathcal{X}$ of N particles, a kernel K, the number of iterations M,
    and a step size $\epsilon > 0$
1: for n = 1, 2, ...., M do
2:     for i = 1,2,..,N do
3:         $X_{n+1}^i = X_n^i - \frac{\epsilon}{N}\sum_{j=1}^{j=N} K(X_n^j, X_n^i)\nabla F(X_n^j) - \nabla_1 K(X_n^j, X_n^i) + B_n^i$
```

Where B_n^i is some noise at step n. Remark that if we take $B_n^i = \sqrt{2\epsilon}\xi_n$ (the same noise as of Langevin), where $\xi_n \sim \mathcal{N}(0,1)$ then this will lead us to the following continuity equation:

$$\frac{\partial \mu_t}{\partial t} = -grad_{\mathcal{H}} KL(\mu_t|\pi) + \Delta\mu_t \tag{18}$$

This means that we do not perform a gradient descent on the KL divergence in the $\mathcal{H}$-Wasserstein space anymore (because we have the term $\Delta\mu_t$ in (18)). However, it remains of great interest to study the theoretical properties of this algorithm. Indeed, by varying the type of noise B_n^i that we choose at each iteration, we can find more robust algorithms that can simulate many hard distributions. For example, let us test this last algorithm to generate $N = 100$ random variables of distribution $\mathcal{N}(2, 1)$ and visualize the results: We start by generating 100 random variable of distribution $\mathcal{U}([-5, 5])$, and then we apply each algorithm on these variables:

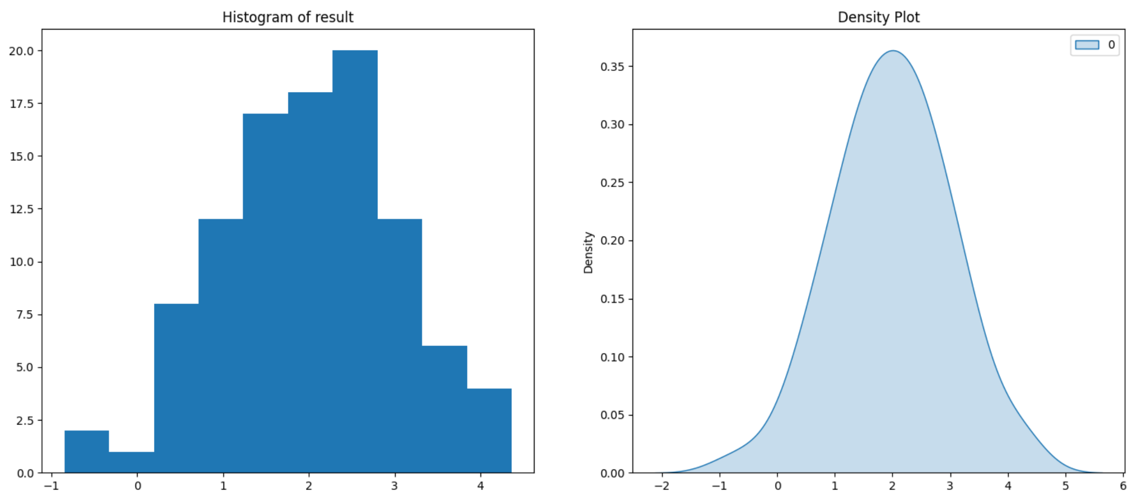


Figure 2: Ordinary SVGD

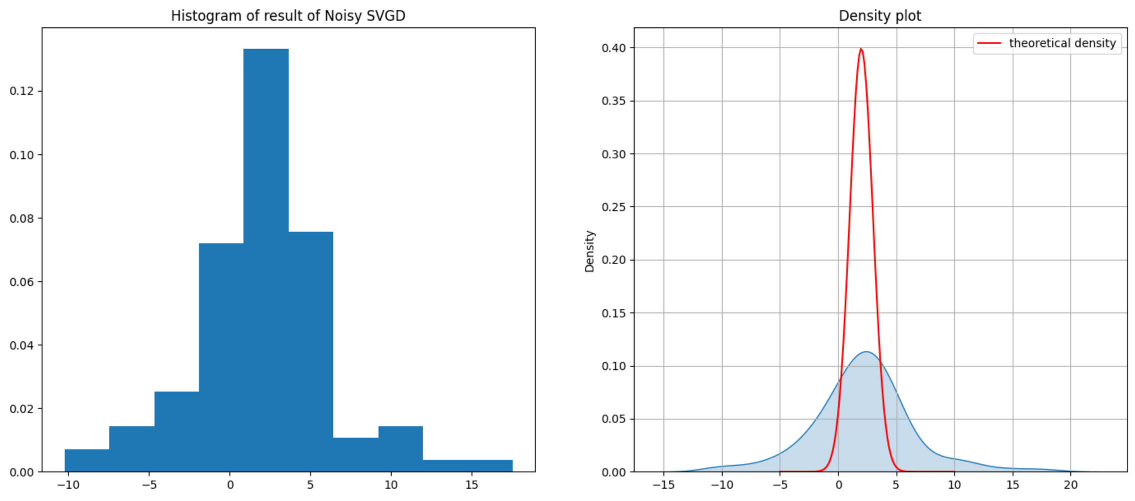


Figure 3: Noisy SVGD

Observing the last figures, it is cristal clear that the ordinary SVGD performs much better than the noisy one for this distribution. And this was actually predictable, in fact, we know that the continuity equation of the ordinary SVGD is given by:(as we said in (17))

$$\frac{\partial \mu_t}{\partial t} = -grad_{\mathcal{H}} KL(\mu_t|\pi)$$

Then, it performs a gradient descent on the KL divergence, and since the potential F for the Gaussian $\mathcal{N}(2,1)$ is convex (it is equal to: $F(x) = \frac{(x-2)^2}{2}$), then doing a gradient descent of the KL divergence will lead us to its global minimum !
Hence, in this case, the use of ordinary SVGD or even Langevin leads to very good results, but the use of noisy SVGD changes the limit of the empirical measures, and then it is not equal to π anymore.
To verify that, and since normal distributions are totally determined by their mean and variance, then we can use their estimators to compute an approximation of these quantities. Let $(X_i)_{i=1}^N$ be the output of SVGD, we recall that the estimator of the mean, denoted $\hat{m}$, is simply the Monte-Carlo estimator of an expectation (the empirical mean), and the unbiased estimator of the variance, denoted by $\hat{\sigma}^2$, are given by:

$$\hat{m} = \frac{1}{N}\sum_{i=1}^{N} X_i \ , \quad \hat{\sigma}^2 = \frac{1}{N-1}\sum_{i=1}^{N}(X_i - \hat{m})^2$$

For these two last examples, we obtain the following results:

- **Ordinary SVGD**: $\hat{m} = 1,99$ and $\hat{\sigma}^2 = 1,02$
- **Noisy SVGD**: $\hat{m} = 2,12$ and $\hat{\sigma}^2 = 18,2$

However, if we choose another noise, then the results will definitely change (we can see that by the continuity equation). In the next section, we will use a specific type of noise that will help us improve the ordinary SVGD.

4.2 Stochastic SVGD

The following algorithm is one of the major contributions to SVGD that we provide in this article. It is inspired from the Stochastic Gradient Descent (SGD) algorithm.
Let us recall some Statistical Learning basics in order to get to this algorithm. Given a labelled data set $(x_i, y_i)_{i=1}^N$ where y_i is called the label of x_i (if $y_i \in \mathbb{R}$ then this is called a regression problem, and if y_i belongs to a finite space, then this is called a classification problem), our main goal is to learn from this data a certain function $\hat{f}$, called the predictor, that predicts the label y given an input x. The criterion that is used to measure the performance of a predictor is the Risk, which is simply defined by the expectation of a certain loss function L. This loss function can be seen as a penalty that we give to our predictor $\hat{f}$ when it predicts a label that is different from the real one. A typical example of a loss function is the squared-loss : $L(\hat{f}(x), y) = \|\hat{f}(x) - y\|^2$, this loss is often used for regression problems.
Hence, any Machine Learning problem can formalized as a Risk Minimization problem:

$$\hat{f} = \arg\min_{f \in \mathcal{H}} \mathbb{E}_{x,y}[L(f(x), y)]$$

The space $\mathcal{H}$ is called the representation space, which contains the models that we want to use in our problem. For example, in classification problems, this space can be the set of hyperplanes of normal vector θ.
If we consider the parametric case, where the functions f_θ belonging to the space $\mathcal{H}$ are fully determined by a parameter θ, then finding the function $\hat{f}$ in our optimization problem becomes finding its parameter $\hat{\theta}$, i.e

$$\hat{\theta} = \arg\min_{\theta \in \mathbb{R}^d} \mathbb{E}_{x,y}[L(f_\theta(x), y)]$$

In most of the cases, the joint distribution of (x, y) is unknown, then we cannot compute explicitly the expectation $\mathbb{E}_{x,y}[L(f_\theta(x), y)]$, so we will estimate it with its empirical version (Monte-Carlo), and then the Risk Minimization turns out to an Empirical Risk Minimization problem:

$$\hat{\theta} = \arg\min_{\theta \in \mathbb{R}^d} \frac{1}{N} \sum_{i=1}^{N} L(f_\theta(x_i), y_i)$$

A classical algorithm to solve this last problem is the Gradient Descent algorithm, which compute the updates:

$$\theta_{n+1} = \theta_n - \frac{\epsilon}{N} \sum_{i=1}^{N} \nabla L(f_{\theta_n}(x_i), y_i)$$

Where $\epsilon > 0$ is called the learning rate. This algorithm converges to a local minimum of the loss function, and if L is convex, then it converges to its global minimum.
Sometimes, the computation of the gradient $\nabla L(f_\theta(x_i), y_i)$ can be costly, so to reduce the complexity of this algorithm, we will update θ is the Gradient Descent algorithm using only one couple (x_k, y_k) chosen randomly from our dataset, which means that we obtain the following algorithm:

$$\theta_{n+1} = \theta_n - \epsilon \nabla L(f_\theta(x_k), y_k) \tag{19}$$

which means that instead of computing the gradient of the loss function at each couple $(x_i, y_i)_{i=1}^{N}$, we only do it for a randomly selected couple (x_k, y_k).
Doing so, we improve so much the complexity of our algorithm, and now many Machine Learning algorithms use (19) instead of the ordinary gradient descent (see [5]).
Using the same idea, we can construct Stochastic SVGD from the ordinary SVGD. Indeed, we know that the updates that are done by SVGD are of the form:

$$X_{n+1}^i = X_n^i - \frac{\epsilon}{N} \sum_{j=1}^{N} K(X_n^j, X_n^i) \nabla F(X_n^j) - \nabla_1 K(X_n^j, X_n^i)$$

Thus the **Stochastic SVGD** is the following:

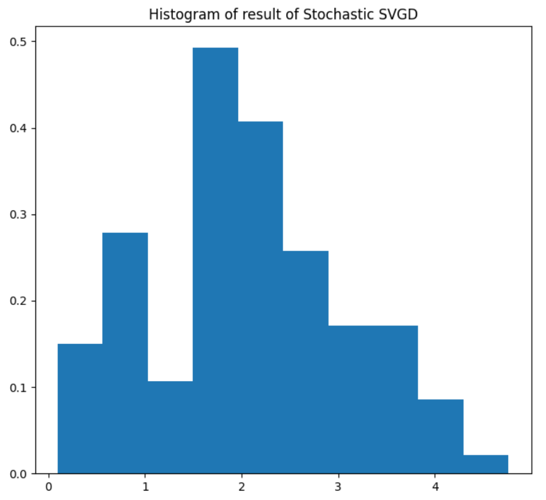
Algorithm 3 Stochastic SVGD

```
Require: a set $X_0^1, ..., X_0^N \in \mathcal{X}$ of N particles, a kernel K, the number of iterations M,
    and a step size $\epsilon > 0$
1: for n = 1, 2, ...., M do
2:     for i = 1,2,..,N  do
3:         take a random $l \in \{1, ..., N\}$
4:         $X_{n+1}^i = X_n^i - \epsilon(K(X_n^l, X_n^i)\nabla F(X_n^l) - \nabla_1 K(X_n^l, X_n^i))$
```

This algorithm is better than SVGD in terms of complexity, and the computation of the gradient in multi-dimension using the finite differences method is costly, in fact, the complexities of both algorithms are:

- SVGD: $\mathcal{O}(N^2 M)$
- Stochastic SVGD: $\mathcal{O}(NM)$

In addition, this algorithm has proved his efficiency in simulations, since we obtain for $\pi = \mathcal{N}(2,1)$ the following result:



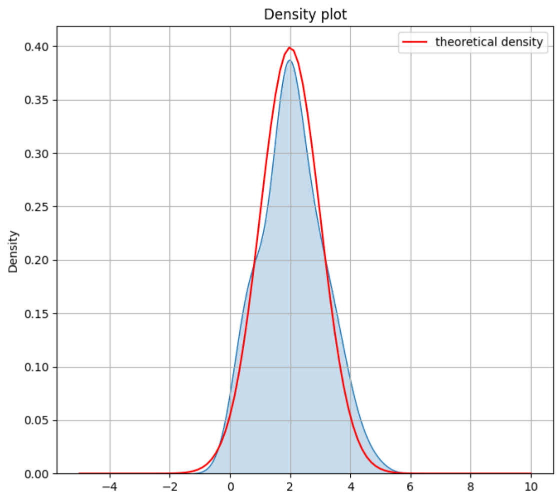


Figure 4: Stochastic SVGD

And the estimators of the means and the variance give us:

- **Estimators** : $\hat{m} = 2,06$ and $\hat{\sigma}^2 = 1,05$

And most importantly, the algorithm converges very fast (in the order of seconds), whereas the ordianry SVGD take much longer time to generate the same number of samples. Therefore, people can keep using Stochastic SVGD instead of the ordinary SVGD to improve the complexity of their simulations and get almost perfect results.

A very good result

We know that simulating from a Gaussian mixture is a very hard task, since only minimizing the KL divergence will not lead us to the desired distribution (see Figure 1) because we will be trapped in one node. However, experiments have shown that Stochastic SVGD succeeds, as well as SVGD, in simulating Gaussian mixtures. For example, if we set the target measure $\pi = \frac{2}{3}\mathcal{N}(0,1) + \frac{1}{3}\mathcal{N}(4,1)$, we obtain the following results:

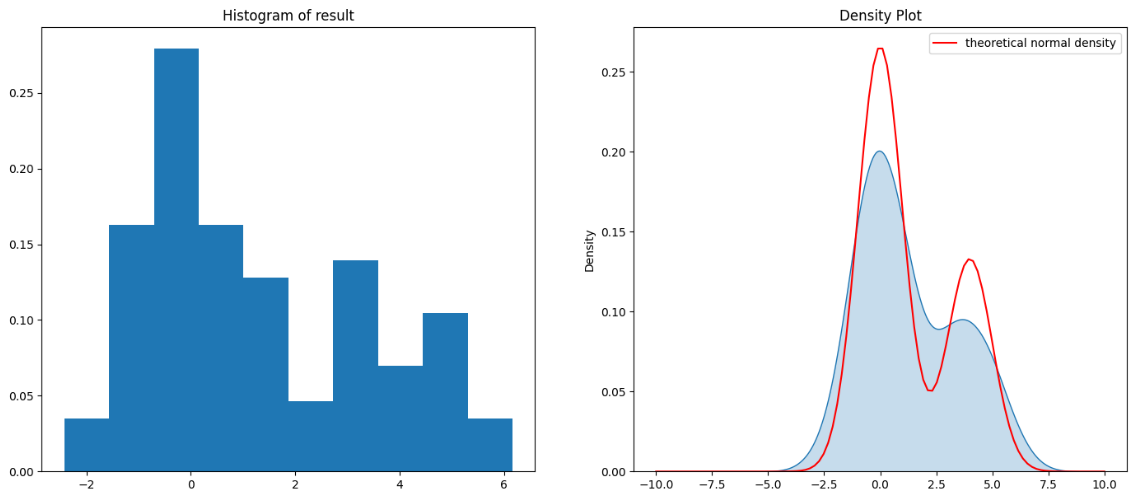


Figure 5: Gaussian mixture $\frac{2}{3}\mathcal{N}(0,1) + \frac{1}{3}\mathcal{N}(4,1)$ with Stochastic SVGD

Although both algorithms give approximately the same result at the end, stochastic SVGD remains much faster and practical.

We can also regard Stochastic SVGD as a Noisy SVGD with noise:

$$B_n^i = \frac{\epsilon}{N}\sum_{j=1}^{N} K(X_n^j, X_n^i)\nabla F(X_n^j) - \nabla_1 K(X_n^j, X_n^i) - \epsilon(K(X_n^l, X_n^i)\nabla F(X_n^l) - \nabla_1 K(X_n^l, X_n^i))$$

This can be the first step towards the study of the theoretical properties of Stochastic SVGD.

4.3 Mixing Langevin and SVGD

Both Langevin and SVGD are interesting in practice, so what if we mix them in one single algorithm called **Langevin SVGD**. To do so, we know that the updates of Langevin are the following:

$$X_{n+1}^i = X_n^i - \epsilon\nabla F(X_n^i) + \sqrt{2\epsilon}\xi_n^i$$

And those of SVGD are:

$$X_{n+1}^i = X_n^i - \frac{\epsilon}{N}\sum_{j=1}^{N} K(X_n^j, X_n^i)\nabla F(X_n^j) - \nabla_1 K(X_n^j, X_n^i)$$

So mixing them gives us the following updates (the average of the two updates):

$$X_{n+1}^i = X_n^i - \frac{\epsilon}{2}\nabla F(X_n^i) + \sqrt{2\epsilon}\xi_n^i - \frac{\epsilon}{2N}\left(\sum_{j=1}^{N}\nabla F(X_n^j)K(X_n^j, X_n^j) - \nabla K(X_n^j, X_n^i)\right)$$

And thus we obtain the following algorithm:

Algorithm 4 Langevin SVGD

```
Require: a set $X_0^1, ..., X_0^N \in \mathcal{X}$ of N particles, a kernel K, the number of iterations M,
    and a step size $\epsilon > 0$
1: for n = 1, 2, ...., M do
2:     for i = 1,2,..,N do
3:         $X_{n+1}^i = X_n^i - \frac{\epsilon}{2}\nabla F(X_n^i) + \sqrt{2\epsilon}\xi_n^i - \frac{\epsilon}{2N}(\sum_{j=1}^{N}\nabla F(X_n^j)K(X_n^j, X_n^j) - \nabla_1 K(X_n^j, X_n^i))$
```

This algorithm gives us a sequence of probability measures $(\mu_n)_n$ that satisfy the following continuity equation:

$$\frac{\partial \mu_t}{\partial t} = \frac{1}{2}div(\nabla F.\mu_t) - \frac{1}{2}grad_{\mathcal{H}}KL(\mu_t|\pi) + \Delta\mu_t \tag{20}$$

This can be seen as a weighted descent algorithm, where we try to minimize F (Langevin part) and interact with the other particles (SVGD) at the same time.
This algorithm performs very well in practice, since it can also simulate well some complex distributions like Gaussian mixtures. We can see that by testing it with the following Gaussian mixture: $\pi = \frac{1}{2}\mathcal{N}(0,1) + \frac{1}{2}\mathcal{N}(4,1)$ and we obtain the following plots:

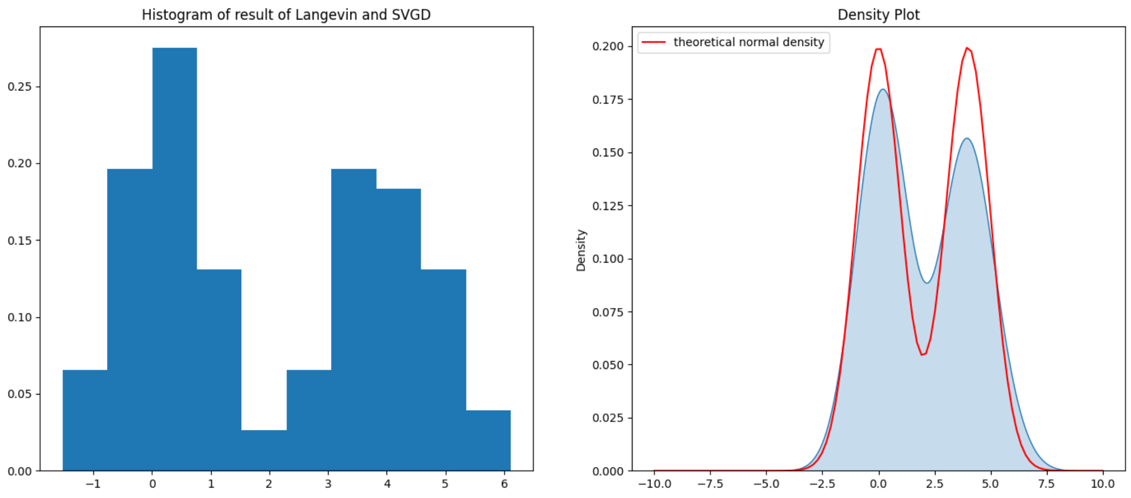


Figure 6: Langevin SVGD

We can also mix Langevin and Stochastic SVGD to obtain a much faster algorithm:

Algorithm 5 Stochastic Langevin SVGD

```
Require: a set $X_0^1, ..., X_0^N \in \mathcal{X}$ of N particles, a kernel K, the number of iterations M,
    and a step size $\epsilon > 0$
1: for n = 1, 2, ...., M do
2:     for i = 1,2,..,N  do
3:         take randomly $l \in \{1, ..., N\}$
4:         $X_{n+1}^i = X_n^i - \frac{\epsilon}{2}\nabla F(X_n^i) + \sqrt{2\epsilon}\xi_n^i - \frac{\epsilon}{2}(\nabla F(X_n^l)K(X_n^l, X_n^j) - \nabla_1 K(X_n^l, X_n^i))$
```

And as expected, this algorithm works also very well in practice, since it can also simulate Gaussian mixtures. We can see that too by testing it on with the same Gaussian mixture as for Langevin SVGD: $\pi = \frac{1}{2}\mathcal{N}(0,1) + \frac{1}{2}\mathcal{N}(4,1)$ and we obtain the following plots:

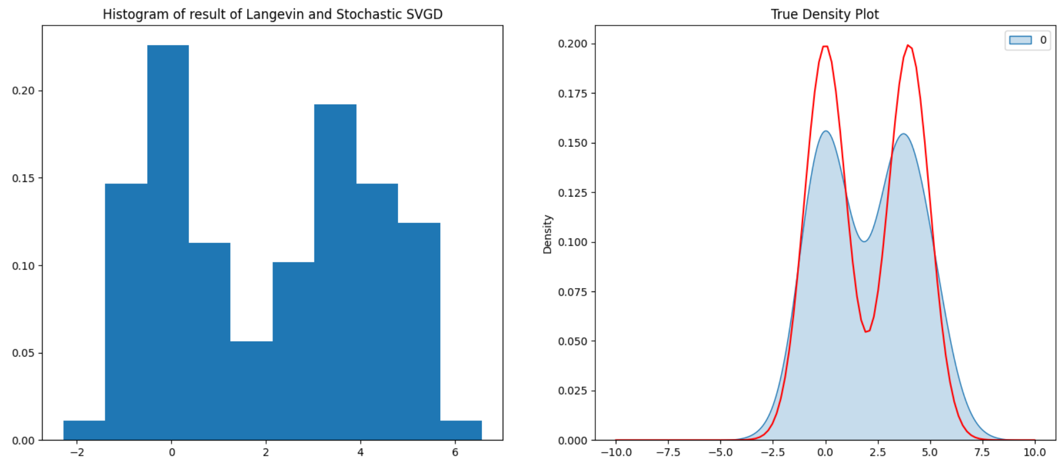


Figure 7: Mixture of Langevin and Stochastic SVGD

Therefore, mixing SVGD and Langevin seems to be a very good idea to simulate complex distributions. Moreover, we can also see that Langevin coupled with Stochastic SVGD is better than coupling Langevin and the Ordianry SVGD since we have that the proportion of each Gaussian ($\frac{1}{2}$ and $\frac{1}{2}$ in front of each normal distribution) are more respected in the mixture of Langevin and Stochastic SVGD.
And of course, this algorithm can also be seen as a noisy SVGD, with the following noises:

- **Langevin SVGD**: $B_n^i = -\frac{\epsilon}{2}\nabla F(X_n^i) + \sqrt{2\epsilon}\xi_n^i - \frac{\epsilon}{2N}(\sum_{j=1}^N \nabla F(X_n^j)K(X_n^j, X_n^j) - \nabla_1 K(X_n^j, X_n^i))$
- **Langevin and Stochastic SVGD**: $B_n^i = -\frac{\epsilon}{2}\nabla F(X_n^i) + \sqrt{2\epsilon}\xi_n^i + \frac{\epsilon}{2N}\sum_{j=1}^N K(X_n^j, X_n^i)\nabla F(X_n^j) - \nabla_1 K(X_n^j, X_n^i) - \frac{\epsilon}{2}(K(X_n^l, X_n^i)\nabla F(X_n^l) - \nabla_1 K(X_n^l, X_n^i))$

Therefore, the study of the theoretical properties of noisy SVGD can help us prove the theoretical efficiency of these proposed algorithms.

4.4 Bonus: Implementation of Wasserstein Gradient Descent

The Wasserstein Gradient Descent is the Euler discretization of the Wasserstein Gradient flow, which performs a gradient descent on the Kullback-Leibler divergence (see WGD). We recall here the WGD updates:

$$X_{n+1} = X_n - \epsilon\nabla\log\left(\frac{\mu_n(X_n)}{\pi(X_n)}\right)$$

Hence

$$X_{n+1} = X_n - \epsilon\left(\nabla F(X_n) + \frac{\nabla\mu_n(X_n)}{\mu_n(X_n)}\right)$$

Its direct implementation is hard because it requires knowing the intermediate distributions $(\mu_n)_n$ obtained throughout the algorithm, which is a very hard task.
However, in this section, we will develop an algorithm that enables us to perform the Wasserstein Gradient descent using finite number of particles. The main idea here is to approximate the densities μ_n with their Kernel estimators.

Definition 4.1 (Kernel estimator of a density). *Let $K : \mathbb{R}^d \times \mathbb{R}^d \longrightarrow \mathbb{R}_+$ be a symmetric positive definite kernel, which means that: $\forall \alpha_1, ..., \alpha_n \in \mathbb{R}$ and $\forall x_1, ..., x_n \in \mathbb{R}^d$ we have that:*

$$\sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j K(x_i, x_j) \geq 0$$

We can also define a kernel in $\mathbb{R}^d$ by: $K(x, y) = K(x - y)$.
Let now $X_1, ..., X_N$ be random vectors with the same density f. The kernel estimator of f is defined by:

$$\hat{f}(t, X_1, ..., X_N) = \frac{1}{Nh}\sum_{i=1}^N K\left(\frac{x-y}{h}\right) \tag{21}$$

where $h > 0$.

Therefore, we will consider a positive definite kernel, like the Gaussian kernel:

$$K(x-y) = \exp\{-\gamma \|x-y\|^2\}$$

And we will approximate the probability densities $(\mu_n)_n$ by their kernel estimators. So the updates will be the following:

$$X_{n+1}^i = X_n^i - \epsilon \left(\nabla F(X_n^i) + \frac{\frac{1}{h}\sum_{j=1}^N \nabla K\left(\frac{X_n^i - X_n^j}{h}\right)}{\sum_{j=1}^N K\left(\frac{X_n^i - X_n^j}{h}\right)} \right) \tag{22}$$

To my best knowledge, this algorithm was discussed in [4], but no one gave a method to implement it as we do in this paper.

Algorithm 6 Kernelized Wasserstein Gradient Descent

```
Require: a set $X_0^1, ..., X_0^N \in \mathcal{X}$ of N particles, a kernel K, the number of iterations M,
    and a step size $\epsilon > 0$
1: for n = 1, 2, ...., M do
2:     for i = 1,2,..,N do
3:         $X_{n+1}^i = X_n^i - \epsilon \left( \nabla F(X_n^i) + \frac{\frac{1}{h}\sum_{j=1}^N \nabla K\left(\frac{X_n^i - X_n^j}{h}\right)}{\sum_{j=1}^N K\left(\frac{X_n^i - X_n^j}{h}\right)} \right)$
```

We test this algorithm on $\pi = \mathcal{N}(2,1)$, and by choosing the best window h of the Gaussian kernel K, we obtain a very good approximation of the target density as shown in these plots:

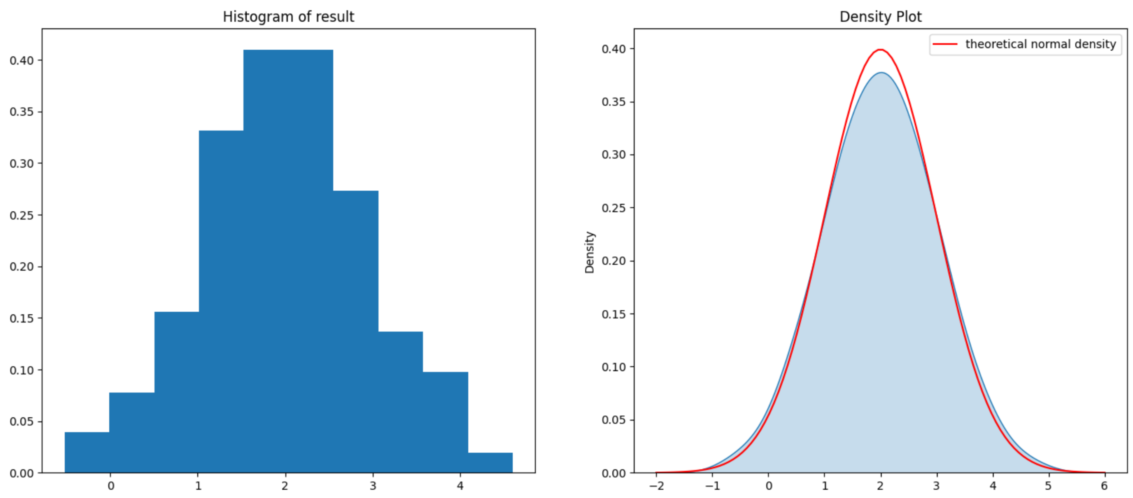


Figure 8: Kernelized WGD

Hence, now we got a clear way (rather than Langevin) to implement the Wasserstein Gradient Descent algorithm, and using deterministic updates without introducing randomness as Langevin does.

5 Conclusion

In this paper, we have proved the convergence of SVGD in the population limit under weak conditions compared to the proofs that were given in other articles. In fact, we proved that the coercivity of the potential and the finite divergence between the initial distribution and the target one are enough to have the convergence. This result is very interesting, since it shows us that if we choose well the parameters of SVGD (initial distribution, kernel K, ...), then we are guaranteed to converge to our target π in the limit of infinite particles (if we have a great number of particles). However, it remains also of great interest to study the convergence of SVGD in the finite particle regime, since we cannot work with infinitely large number of particles, because this increase heavily the complexity of the algorithm, and therefore it will take an enormous time to converge.

We have also invented new algorithms that mainly adds noise to SVGD. These algorithms improve the complexity of SVGD, since by adding some noise, we make the algorithm explore more the space, and to avoid that he gets stuck in some local minimum of the Kullback-Leibler divergence, which we want to avoid as much as possible. The choice of this noise is the hardest task for performing these algorithms, since it disturbs the limit measure, and we can clearly see that by writing the continuity equation. And for a specific noise, we invented the Stochastic SVGD, which improves the ordinary SVGD in complexity and the intuition behind creating this algorithm came from the well-known Stochastic Gradient Descent algorithm (see [5]), which is widely used in Deep Learning to train Neural networks.

We examined also the case of mixing two different algorithms: Langevin and SVGD, which gave us good results for sampling from a Gaussian mixture. Therefore, this algorithm should also be an improvement of the ordinary SVGD and of Langevin, since it enables at the same time the minimization of the KL divergence, as well as the interaction with other particles.

Finally, we proposed a method to directly implement the Wasserstein Gradient Descent (WGD) thanks to the kernel estimation of a density, and we got a deterministic algorithm that uses a set of interacting particles (just like SVGD), unlike Langevin that introduces randomness in generating the Markov Chain, and that generates each particle individually.

References

[1] Richard Jordan, David Kinderlehrer and Felix Otto, *The Variational Formulation of the Fokker-Planck Equation*, Society of industrial and Applied Mathematics, 1999 https://epubs.siam.org/doi/abs/10.1137/S0036141096303359

[2] Anna Korba, Adil Salim, Michael Arbel, Giulia Luise and Arthur Gretton, *A Non-Asymptotic Analysis for Stein Variational Gradient Descent*, Advances in Neural Information Processing Systems 33 (NeurIPS 2020).

[3] Qiang Liu, *Stein Variational Gradient Descent as Gradient Flow*, NeurIPS Proceedings,
https://proceedings.neurips.cc/paper/2017/hash/17ed8abedc255908be746d245e50263a-Abstract.html

[4] Yifei Wang, Peng Chen, Wuchen Li, *Projected Wasserstein Gradient Descent for High-Dimensional Bayesian Inference*, Society for Industrial for Applied Mathematics, https://epubs.siam.org/doi/abs/10.1137/21M1454018

[5] Léon Bottou, *Stochastic Gradient Descent tricks*, Springer,
https://link.springer.com/chapter/10.1007/978-3-642-35289-$8_2$5

[6] Ian Goodfellow, Yoshua Bengio and Aaron Courville, *Deep Learning*, The MIT Press,https://www.deeplearningbook.org/

[7] J. Gorham and L. Mackey. Measuring sample quality with kernels. In International Conference on Machine Learning (ICML), 2017.

[8] Filippo Santambrogio, Optimal Transport for applied mathematicians, calculus of variations, PDEs and Modelling, Birkhäuser, June 2008

[9] Cédric Villani, Optimal transport, old and new, Springer, June 2008.

[10] Qiang Liu, Dilin Wang, *Stein Variational Gradient Descent: A General Purpose Bayesian Inference Algorithm*, NeurIPS Proceedings.
https://proceedings.neurips.cc/paper/2016/hash/b3ba8f1bee1238a2f37603d90b58898d-Abstract.html

[11] Adil Salim, Lukang Sun, Peter Richtarik, A Convergence Theory for SVGD in the Population Limit under Talagrand's Inequality T1, Proceedings of Machine Learning Research.

Appendix

Derivative of the KL divergence

Denote by

$$\mathcal{F}(\mu_t) = KL(\mu_t|\pi)$$

Let the family $(\mu_t)_t$ satisfy the following continuity equation:

$$\frac{\partial \mu_t}{\partial t} + div(v_t\mu_t) = 0$$

Then using the chain rule, we have:

$$\frac{d\mathcal{F}(\mu_t)}{dt} = \frac{d}{dt}\int \log\Big(\frac{\mu_t}{\pi}\Big)\mu_t(x)dx = \int \frac{d\mu_t(x)}{dt}\log\left(\frac{\mu_t(x)}{\pi(x)}\right)+\frac{d\mu_t(x)}{dt}\mathrm{d}x = \int \frac{d\mu_t(x)}{dt}(\log\left(\frac{\mu_t(x)}{\pi(x)}\right)+1)\mathrm{d}x$$

And using the continuity equation, we have that: $\frac{d\mu_t(x)}{dt} = -div(\mu_t(x)v_t(x))$ then we replace in the last equation and we do an integration by parts:

$$\frac{d\mathcal{F}(\mu_t)}{dt} = \int -div(\mu_t(x)v_t(x))(\log\left(\frac{\mu_t(x)}{\pi(x)}\right) + 1)dx = \int \langle \nabla \log\left(\frac{\mu_t(x)}{\pi(x)}\right), v_t(x)\rangle \mu_t(x)dx$$

Then :

$$\frac{d\mathcal{F}(\mu_t)}{dt} = \langle \nabla \log\left(\frac{\mu_t(x)}{\pi(x)}\right), v_t\rangle_{L^2(\mu_t)}$$

Hence, we have proved that:

$$\boxed{\nabla_W \mathcal{F}(\mu) = \nabla_W KL(\mu|\pi) = \nabla \log\Big(\frac{\mu}{\pi}\Big)} \tag{23}$$

Directional derivative of KL

Now we will prove the expression in (5). We have that:

$$\frac{d}{d\epsilon}Kl(T_{\#}\mu|\pi)|_{\epsilon=0} = \langle \phi, \nabla \log\Big(\frac{\mu_t}{\pi}\Big)\rangle_{L^2(\mu)} = -\int \langle \phi(x), \nabla \log(\pi(x))\rangle \mu(x)\mathrm{d}x + \int \langle \phi(x), \nabla\mu(x)\rangle \mathrm{d}x$$

By integrating by part:

$$-\int \langle \phi(x), \nabla \log(\pi(x))\rangle \mu(x)\mathrm{d}x + \int \langle \phi(x), \nabla\mu(x)\rangle \mathrm{d}x = -\int \langle \phi(x), \nabla \log(\pi(x))\rangle \mu(x)\mathrm{d}x - \int div(\phi(x))\mu(x)\mathrm{d}x$$

Finally :

$$-\frac{d}{d\epsilon}Kl(T_{\#}\mu|\pi)|_{\epsilon=0} = \int (\langle \phi(x), \nabla \log(\pi(x))\rangle - div(\phi(x)))\mu(x)\mathrm{d}x$$