TELECOM Paris

IP PARIS

Parcours recherche

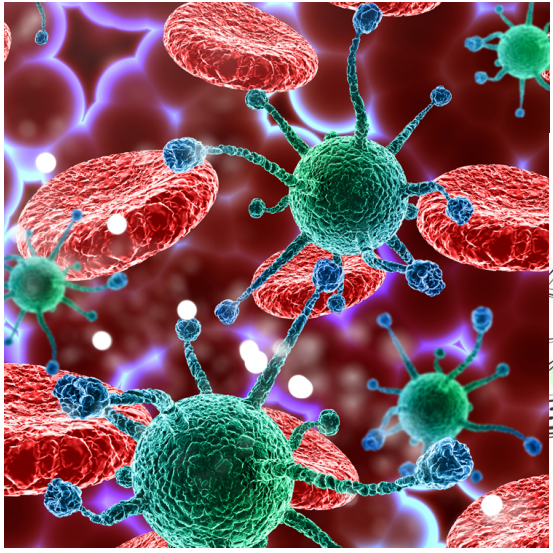
Sampling algorithms: SVGD

Aymane El Firdoussi
Pascal Bianchi

February 9, 2023

Introduction

Introduction

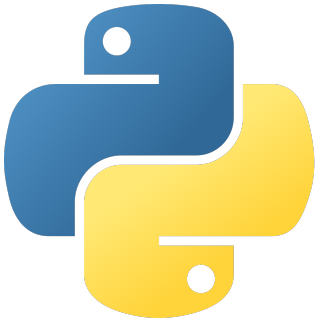


Python ?

With Python

We can use it only to generate usual laws such as :

Gaussian, exponential, Poisson, uniform,..



Sampling algorithms

Our task is the following:

Input : target density π

$\downarrow$ *algorithm*

Output : $x_1, x_2, ..., x_n$ realizations of $X \sim \pi$.

Langevin algorithm

We want to sample from

$$\pi(x) \propto \exp(-F(x))$$

where F is **convex !**. Then we do the following gradient descent algorithm:

$$X_{k+1} = X_k - h\nabla F(X_k) + \sqrt{2h}\xi_n \qquad \text{(LMC)}$$

Which is a discretization of the Fokker-Planck equation :

$$\frac{\partial \rho}{\partial t} = div(\nabla F(x)\rho) + \beta^{-1}\Delta\rho \qquad \text{(FP)}$$

Another interpretation of Langevin

Langevin can also be interpreted as a gradient flow on the KL divergence :

$$X_{n+1} = X_n - \gamma \nabla_W KL(\mu_n, \pi)(X_n) \qquad \text{(WGD)}$$

Overview

1. Background on optimal transport
2. Wasserstein gradient descent algorithm
3. Stein Variational Gradient descent
4. Improving SVGD

Background on optimal transport

Gradient flows

$$\begin{cases} x'(t) = v(x(t)) \quad \text{for } t > 0 \\ x(0) = x_0 \end{cases} \tag{1}$$

Where v is called the flow. Its Euler discretization is:

$$\begin{cases} x_{k+1}^{\tau} = x_k^{\tau} + \tau v(x_k^{\tau}) \quad \text{for } k \geq 0 \\ x_0^{\tau} = x_0 \end{cases} \tag{2}$$

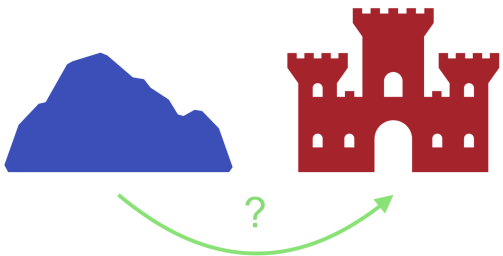
Case where

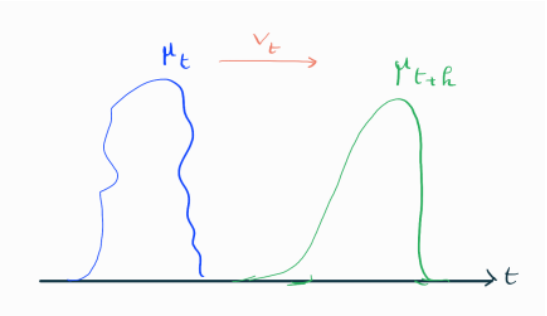
$$v = -\nabla F$$

Gradient flow in Wasserstein space

Now we know what are gradient flows in Euclidean spaces, what about Wasserstein spaces?

$$\dot{X}_t = v_t(\mu_t)(X_t) \qquad \text{(WGF)}$$





Properties

The continuity equation is:

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

Using the Chain rule:

$$\frac{d\mathcal{F}(\mu_t)}{dt} = \lim_{h \to 0} \frac{\mathcal{F}(\mu_{t+h}) - \mathcal{F}(\mu_t)}{h} = \langle \nabla_W \mathcal{F}(\mu_t), v_t \rangle_{L^2(\mu_t)} \tag{3}$$

Wasserstein gradient descent algorithm

Approach

We want to go from an initial distribution μ_0 to a final one π.
To do so, we can use a gradient flow, but what is the flow in that case ?

$$KL(\mu, \pi) = \int \log\left(\frac{\mu(x)}{\pi(x)}\right)\mu(x)dx \tag{KL}$$

This quantity is positive, and is finite iff $\mu << \pi$.

$$KL(\mu, \pi) = 0 \iff \mu = \pi$$

WGD

Objective : Minimize $\mathcal{F}(\mu) = KL(\mu, \pi)$ (π is the target density).

$$\dot{X}_t = -\nabla_W \mathcal{F}(\mu_t)(X_t) \qquad \text{(WGF)}$$

$$X_{n+1} = X_n - \gamma \nabla_W \mathcal{F}(\mu_n)(X_n) \qquad \text{(WGD)}$$

Non-implementable in practice !

Fokker-Planck :

$$v_t = -\nabla_W \mathcal{F}(\mu_t)$$

Stein Variational Gradient descent

SVGD

To simplify the last algorithm, we kernelize the gradient of Wasserstein with an operator P_μ, which gives us SVGD :

$$X_{n+1} = X_n - \gamma P_{\mu_n} \nabla log(\frac{\mu_n}{\pi})(X_n) \qquad \text{(SVGD)}$$

Where :

$$P_\mu \nabla \log\left(\frac{d\mu}{d\pi}\right)(y) = -\int [\nabla \log(\pi(x))k(x,y) + \nabla_x k(x,y)] d\mu(x) \qquad (4)$$

Algorithm : 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, and a step size $\gamma > 0$

converged $\leftarrow$ *False*

while not converged **do**

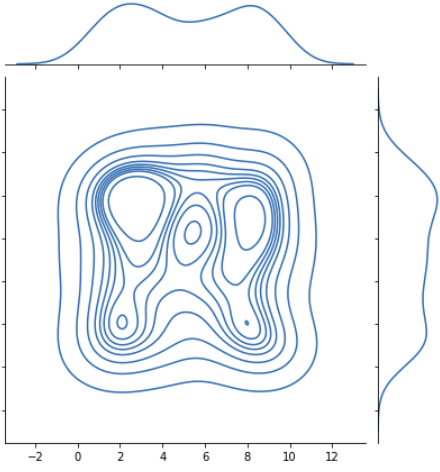
  for i = 1,2,..,N **do**

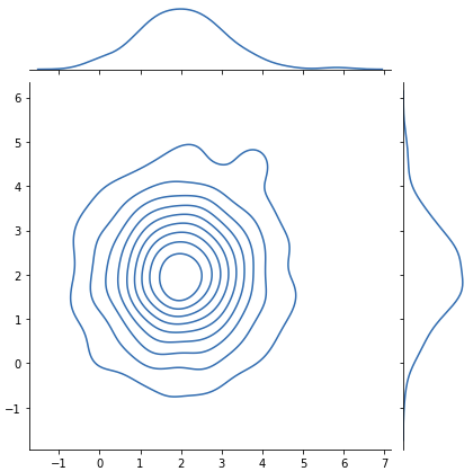
    $x_{n+1}^i = x_n^i - \frac{\gamma}{N} \sum_{i=1}^{i=N} k(x_n^i, x_n^j) \nabla F(x_n^j) - \nabla_2 k(x_n^i, x_n^j)$

  converged $\leftarrow$ *criterion*

Simulation : normal distribution

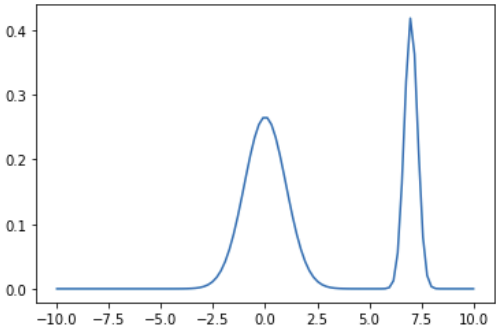
We want to simulate : $\pi = \mathcal{N}(2, 1)$

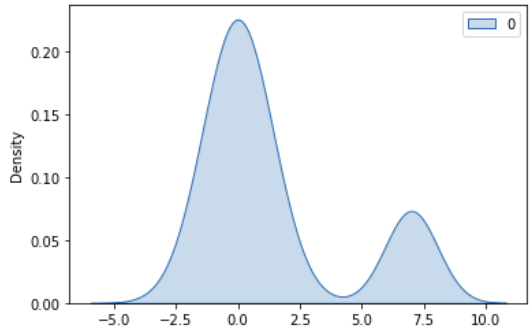




Simulation : Gaussian mixture

We try it with the following Guassian mixture: $\frac{2}{3}\mathcal{N}(0,1) + \frac{1}{3}\mathcal{N}(7,\sqrt{0.1})$





Improving SVGD

Some improvements

- Importance weights (acceleration term)
- Laplacian Adjusted Wasserstein Gradient descent
- Regularized SVGD

Perspectives

- Consider a new divergence ?
- Improve Langevin in some way ?
- Get inspired by improving techniques proposed