Algorithmes adaptatifs de gradient-Langevin pour l'optimisation stochastique et l'inférence Bayésienne

Pierre Bras
Laboratoire de Probabilités, Statistique et Modélisation, Sorbonne Université
Sous la direction de Gilles Pagès

Séminaire de Mathématiques Appliquées – Collège de France

8 Décembre 2023

Outline

Outline

Optimization problem

Let $V : \mathbb{R}^d \to \mathbb{R}$,

Objective: $\underset{x \in \mathbb{R}^d}{\text{Minimize}} \quad V(x).$

Optimization problem

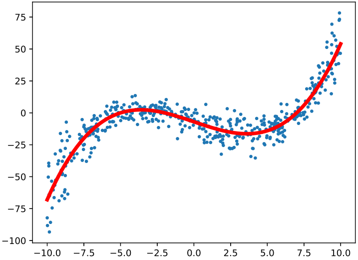
$$\text{Let} \quad V : \mathbb{R}^d \to \mathbb{R},$$
$$\textbf{Objective:} \quad \underset{x \in \mathbb{R}^d}{\text{Minimize}} \quad V(x).$$

Examples:

1. Determining optimal allocation of resources to maximize production output while minimizing costs.
2. Maximize the gains along a controlled time process with respect to the strategy.
3. Minimize the error of a model to the true data for prediction and regression tasks.

Example: Regression and Neural Networks

- Data $(u_i, v_i) \in \mathbb{R}^{d_{\text{in}}} \times \mathbb{R}^{d_{\text{out}}}$ (inputs and outputs) for $1 \leq i \leq N$ with $N \gg 1$.

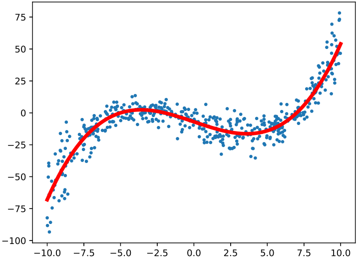


Example: Regression and Neural Networks

- Data $(u_i, v_i) \in \mathbb{R}^{d_{\text{in}}} \times \mathbb{R}^{d_{\text{out}}}$ (inputs and outputs) for $1 \leq i \leq N$ with $N \gg 1$.
- We want to find some formula relation between the inputs and the outputs:

$$\text{Find } \Phi : \mathbb{R}^{d_{\text{in}}} \to \mathbb{R}^{d_{\text{out}}}$$

$$\text{such that: } \forall i, \ \Phi(u_i) \approx v_i.$$



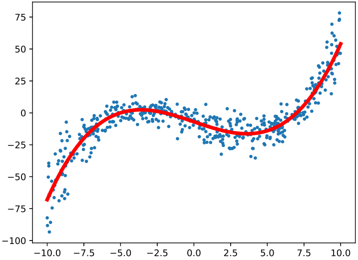
Example: Regression and Neural Networks

• Data $(u_i, v_i) \in \mathbb{R}^{d_{\text{in}}} \times \mathbb{R}^{d_{\text{out}}}$ (inputs and outputs) for $1 \leq i \leq N$ with $N \gg 1$.

• We want to find some formula relation between the inputs and the outputs:

$$\text{Find } \Phi : \mathbb{R}^{d_{\text{in}}} \to \mathbb{R}^{d_{\text{out}}}$$

$$\text{such that: } \forall i,\ \Phi(u_i) \approx v_i.$$



• We parametrize Φ with a finite number of parameters: $\{\Phi_x :\ x \in \mathbb{R}^d\}$.
For example, affine parametrization:

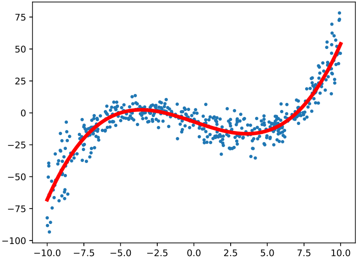
$$\Phi_{x_1,x_2}(u) = x_1 \cdot u + x_2, \quad x_1 \text{ matrix, } x_2 \text{ vector.}$$

Example: Regression and Neural Networks

- Data $(u_i, v_i) \in \mathbb{R}^{d_{\text{in}}} \times \mathbb{R}^{d_{\text{out}}}$ (inputs and outputs) for $1 \leq i \leq N$ with $N \gg 1$.
- We want to find some formula relation between the inputs and the outputs:

$$\text{Find } \Phi : \mathbb{R}^{d_{\text{in}}} \to \mathbb{R}^{d_{\text{out}}}$$

$$\text{such that: } \forall i, \ \Phi(u_i) \approx v_i.$$



- We parametrize Φ with a finite number of parameters: $\{\Phi_x : \ x \in \mathbb{R}^d\}$. For example, affine parametrization:

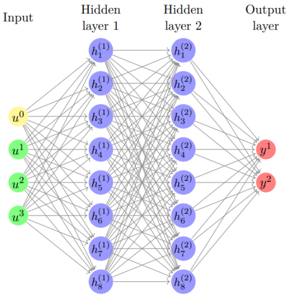
$\Phi_{x_1, x_2}(u) = x_1 \cdot u + x_2$, x_1 matrix, x_2 vector.

- **Objective as an optimization problem:** minimize the MSE:

$$\min_{x \in \mathbb{R}^d} \frac{1}{N} \sum_{i=1}^{N} |\Phi_x(u_i) - v_i|^2 =: \min_{x \in \mathbb{R}^d} V(x).$$

Example: Regression and Neural Networks

- **Neural Networks** are families of parametrized functions that can approximate many functions in practice (Cybenko, 1989), (AlexNet 2012).



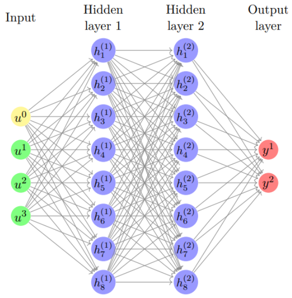
Example: Regression and Neural Networks

- **Neural Networks** are families of parametrized functions that can approximate many functions in practice (Cybenko, 1989), (AlexNet 2012).
- Written as **composition of linear and non-linear functions**:

Input: $h_0 = u$,

$h_k = \varphi(\alpha_k \cdot h_{k-1} + \beta_k) \in \mathbb{R}^{d_k},\ 1 \le k \le K-1,$

Output: $h_K = \alpha_K \cdot h_{K-1} + \beta_K =: \Phi_{(\alpha_k,\beta_k)_{0\le k\le K}}(u)$



Example: Regression and Neural Networks

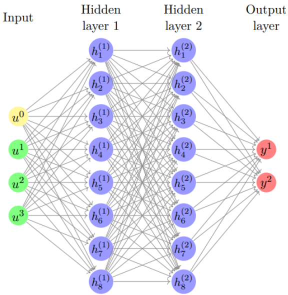
• **Neural Networks** are families of parametrized functions that can approximate many functions in practice (Cybenko, 1989), (AlexNet 2012).

• Written as **composition of linear and non-linear functions**:

Input: $h_0 = u$,

$$h_k = \varphi(\alpha_k \cdot h_{k-1} + \beta_k) \in \mathbb{R}^{d_k},\ 1 \le k \le K-1,$$

Output: $h_K = \alpha_K \cdot h_{K-1} + \beta_K =: \Phi_{(\alpha_k,\beta_k)_{0\le k\le K}}(u)$



where

- φ is a non-linear function applied coordinate-wise,
- (d_k) is a sequence of dimensions,
- $(\alpha_k)_k$ are matrices and $(\beta_k)_k$ are vectors parametrizing the neural network.

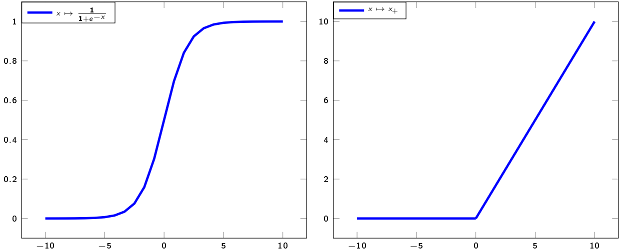
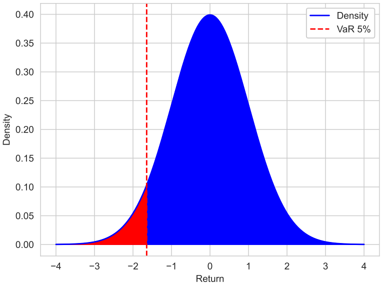


Figure: The Sigmoid and ReLU functions

Example: Computation of quantiles and VaR

For Z some random variable, the quantile of order $\alpha \in [0,1]$ is

$$q_\alpha := \inf\{u \in \mathbb{R} : \ \mathbb{P}(Z \le u) \ge \alpha\}.$$



Example: Computation of quantiles and VaR

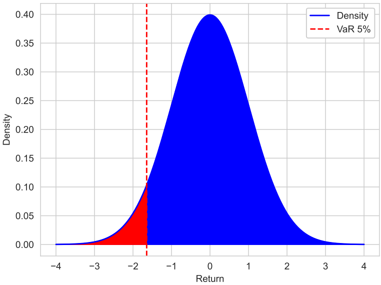
For Z some random variable, the quantile of order $\alpha \in [0,1]$ is

$$q_\alpha := \inf\{u \in \mathbb{R} : \mathbb{P}(Z \leq u) \geq \alpha\}.$$

Following (Uryasev and Rockafellar, 2001) we have the characterization:

$$q_\alpha = \operatorname*{argmin}_{x\in\mathbb{R}} \mathbb{E}\left[x + \frac{1}{1-\alpha}(Z-x)_+\right],$$

where $(\cdot)_+$ denotes the positive part.



Example: Stochastic control

- Dynamical stochastic system Y_t depending on some control u_t with Brownian motion W_t:

$$dY_t^u = b(Y_t^u, u_t)dt + \sigma(Y_t^u, u_t)dW_t, \ \ t \in [0, T].$$

Example: Stochastic control

- Dynamical stochastic system Y_t depending on some control u_t with Brownian motion W_t:
$$dY_t^u = b(Y_t^u, u_t)dt + \sigma(Y_t^u, u_t)dW_t, \ \ t \in [0, T].$$

- We choose the control to achieve
$$\min_u J(u) := \mathbb{E}\left[\int_0^T G(Y_t^u)dt + F(Y_T^u)\right]$$
where G and F are some scalar functions.

Example: Stochastic control

- Dynamical stochastic system Y_t depending on some control u_t with Brownian motion W_t:
$$dY_t^u = b(Y_t^u, u_t)dt + \sigma(Y_t^u, u_t)dW_t, \ \ t \in [0, T].$$
- We choose the control to achieve
$$\min_u J(u) := \mathbb{E}\left[\int_0^T G(Y_t^u)dt + F(Y_T^u)\right]$$
where G and F are some scalar functions.
- We can parametrize u by some neural network with parameter x:
$$u_t = u_x(t, Y_t)$$
and obtain an optimization problem on x with $V(x) = J(u_x)$.

Outline

1. Introduction
   - Optimization
   - Stochastic gradient descent algorithm
   - Langevin equation and algorithms
   - Objectives

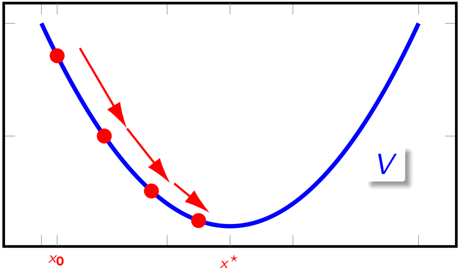
Gradient Descent Algorithm (GD)

Gradient descent algorithm: Assuming that $V \in \mathcal{C}^1$, for each iteration compute the gradient and "go down" the gradient with non-increasing positive step sequence (γ_k):

Gradient Descent Algorithm

With initialization $x_0 \in \mathbb{R}^d$ and step sequence (γ_k):

$$x_{n+1} = x_n - \gamma_{n+1} \nabla V(x_n).$$



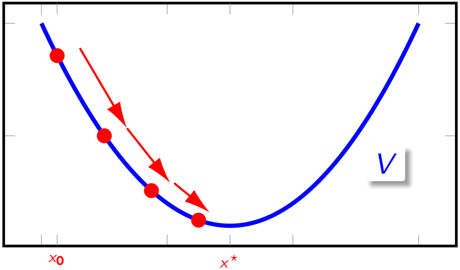
Gradient Descent Algorithm (GD)

Gradient descent algorithm: Assuming that $V \in \mathcal{C}^1$, for each iteration compute the gradient and "go down" the gradient with non-increasing positive step sequence (γ_k):

Gradient Descent Algorithm

With initialization $x_0 \in \mathbb{R}^d$ and step sequence (γ_k):

$$x_{n+1} = x_n - \gamma_{n+1} \nabla V(x_n).$$



$\implies$ Greedy algorithm: focus on local improvements around the current position at each iteration.

Example of Gradient Descent

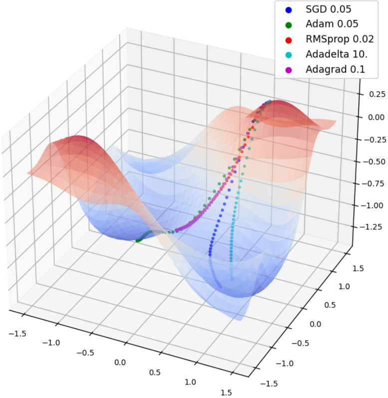


Figure: Example of different variants of gradient descent algorithms with $V(x,y) = -\sin(x^2)\cos(3y^2)e^{-x^2y^2} - e^{-(x+y)^2}$.

Stochastic Gradient Descent (SGD)

In many practical cases, it is not possible to compute the true gradient $\nabla V(x_n)$ at each iteration:

Stochastic Gradient Descent (SGD)

In many practical cases, it is not possible to compute the true gradient $\nabla V(x_n)$ at each iteration:

1. In big data applications with amount of data $N \gg 1$:

$$V(x) = \frac{1}{N}\sum_{i=1}^{N} V_i(x).$$

Stochastic Gradient Descent (SGD)

In many practical cases, it is not possible to compute the true gradient $\nabla V(x_n)$ at each iteration:

1. In big data applications with amount of data $N \gg 1$:

$$V(x) = \frac{1}{N} \sum_{i=1}^{N} V_i(x).$$

2. In applications with some continuous random variable Z:

$$V(x) = \mathbb{E}_Z[v(x, Z)]$$

with no close form expression of the expectation.

Stochastic Gradient Descent (SGD)

In many practical cases, it is not possible to compute the true gradient $\nabla V(x_n)$ at each iteration:

1. In big data applications with amount of data $N \gg 1$:

$$V(x) = \frac{1}{N}\sum_{i=1}^{N} V_i(x).$$

2. In applications with some continuous random variable Z:

$$V(x) = \mathbb{E}_Z[v(x,Z)]$$

   with no close form expression of the expectation.

3. Remark: in the 1st case we can also write:

$$V(x) = \mathbb{E}_Z[v(x,Z)], \quad Z \in \{1,\dots,N\}, \quad v(x,Z) = V_Z(x).$$

$\Longrightarrow$ In both cases we write

$$V(x) = \mathbb{E}_Z[v(x, Z)].$$

$\Longrightarrow$ In both cases we write

$$V(x) = \mathbb{E}_Z[v(x, Z)].$$

Stochastic Gradient Descent Algorithm

With initialization $x_0 \in \mathbb{R}^d$ and step sequence (γ_k):

$$x_{n+1} = x_n - \gamma_{n+1} \nabla v(x_n, Z_{n+1}),$$
$$Z_n \sim Z \quad \text{i.i.d.}.$$

$\Longrightarrow$ In both cases we write

$$V(x) = \mathbb{E}_Z[v(x, Z)].$$

Stochastic Gradient Descent Algorithm

With initialization $x_0 \in \mathbb{R}^d$ and step sequence (γ_k):

$$x_{n+1} = x_n - \gamma_{n+1} \nabla v(x_n, Z_{n+1}),$$
$$Z_n \sim Z \ \text{ i.i.d.}.$$

Stochastic Gradient Descent Algorithm, Mini-Batch version

With initialization $x_0 \in \mathbb{R}^d$ and step sequence (γ_k):

$$x_{n+1} = x_n - \gamma_{n+1} \frac{1}{M} \sum_{i=1}^{M} \nabla v(x_n, Z_{n+1}^i),$$
$$Z_n^i \sim Z \ \text{ i.i.d.}$$

$\Longrightarrow$ In both cases we write

$$V(x) = \mathbb{E}_Z[v(x, Z)].$$

Stochastic Gradient Descent Algorithm

With initialization $x_0 \in \mathbb{R}^d$ and step sequence (γ_k):

$$x_{n+1} = x_n - \gamma_{n+1} \nabla v(x_n, Z_{n+1}),$$
$$Z_n \sim Z \quad \text{i.i.d..}$$

Stochastic Gradient Descent Algorithm, Mini-Batch version

With initialization $x_0 \in \mathbb{R}^d$ and step sequence (γ_k):

$$x_{n+1} = x_n - \gamma_{n+1} \frac{1}{M} \sum_{i=1}^{M} \nabla v(x_n, Z_{n+1}^i),$$
$$Z_n^i \sim Z \quad \text{i.i.d.}$$

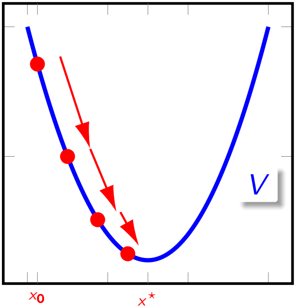
Introduced in (Robbins and Monro, 1951); Robbins-Siegmund Lemma of convergence (Robbins and Siegmund, 1971).

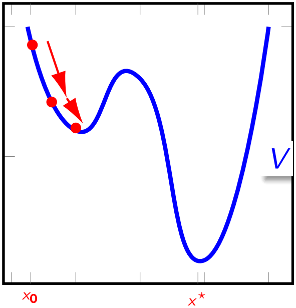
Outline

1. Introduction
   - Optimization
   - Stochastic gradient descent algorithm
   - Langevin equation and algorithms
   - Objectives

Problem: traps for gradient descent

The gradient descent x_n can be trapped in a local (but not global) minimum (e.g. if V is not convex):





Langevin Equation

- We add a white noise to x_n, hoping to escape traps and explore:

Langevin Equation

- We add a white noise to x_n, hoping to escape traps and explore:

Stochastic Gradient Langevin Dynamics (SGLD), (Welling and Teh, 2011)

$$x_{n+1} = x_n - \gamma_{n+1}\nabla\tilde{V}(x_n) + \sqrt{\gamma_{n+1}}\sigma\xi_{n+1},$$
$$\xi_{n+1} \sim \mathcal{N}(0, I_d),\ \sigma > 0.$$

Langevin Equation

- We add a white noise to x_n, hoping to escape traps and explore:

Stochastic Gradient Langevin Dynamics (SGLD), (Welling and Teh, 2011)

$$x_{n+1} = x_n - \gamma_{n+1}\nabla\tilde{V}(x_n) + \sqrt{\gamma_{n+1}}\sigma\xi_{n+1},$$
$$\xi_{n+1} \sim \mathcal{N}(0, I_d), \ \sigma > 0.$$

The noise is **exogenous** and scales as $\sqrt{\gamma_{n+1}}$.

Langevin Equation

- We add a white noise to x_n, hoping to escape traps and explore:

Stochastic Gradient Langevin Dynamics (SGLD), (Welling and Teh, 2011)

$$x_{n+1} = x_n - \gamma_{n+1}\nabla\tilde{V}(x_n) + \sqrt{\gamma_{n+1}}\sigma\xi_{n+1},$$
$$\xi_{n+1} \sim \mathcal{N}(0, I_d), \ \ \sigma > 0.$$

The noise is **exogenous** and scales as $\sqrt{\gamma_{n+1}}$.

- The continuous version becomes:

Langevin equation

$$dX_s = -\nabla V(X_s)ds + \sigma dW_s$$

where (W_s) is a Brownian motion.

- Its invariant measure is the **Gibbs measure**

$$\nu_\sigma(x)dx \propto e^{-2V(x)/\sigma^2}dx.$$

Langevin Equation

- We add a white noise to x_n, hoping to escape traps and explore:

Stochastic Gradient Langevin Dynamics (SGLD), (Welling and Teh, 2011)

$$x_{n+1} = x_n - \gamma_{n+1} \nabla \tilde{V}(x_n) + \sqrt{\gamma_{n+1}} \sigma \xi_{n+1},$$
$$\xi_{n+1} \sim \mathcal{N}(0, I_d), \ \sigma > 0.$$

The noise is **exogenous** and scales as $\sqrt{\gamma_{n+1}}$.

- The continuous version becomes:

Langevin equation

$$dX_s = -\nabla V(X_s)ds + \sigma dW_s$$

where (W_s) is a Brownian motion.

- Its invariant measure is the **Gibbs measure**

$$\nu_\sigma(x)dx \propto e^{-2V(x)/\sigma^2} dx.$$

- For small σ, ν_σ is concentrated around $\mathrm{argmin}(V)$:
Solve the Langevin equation $\implies$ approximation of ν_σ $\implies$ approximation of $\mathrm{argmin}(V)$.

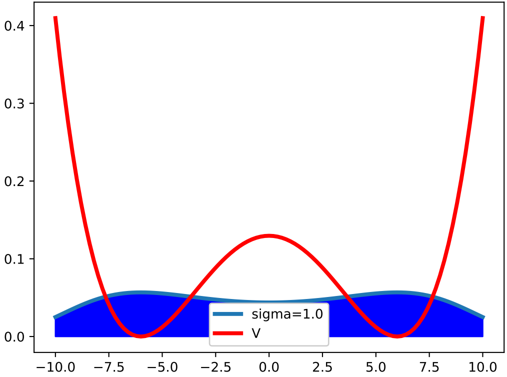


Figure: Concentration of Gibbs measure

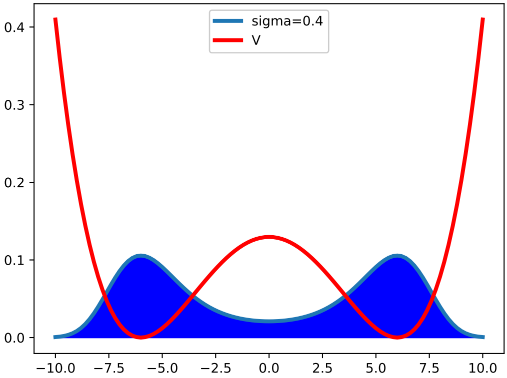


Figure: Concentration of Gibbs measure

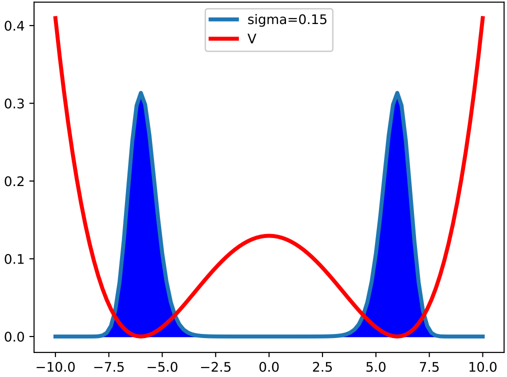


Figure: Concentration of Gibbs measure

Bayesian inference and sampling from distribution

Stochastic algorithms are also used for sampling from a probability measure.

- Given data $u_1, \dots, u_N$ with $N \gg 1$, we consider a family of probability distributions $\{p(u|x)du : \ x \in \mathbb{R}^d\}$ and a prior densities $p_0(x)dx$. Then the posterior distribution on x, $p(x|u_1, \dots, u_N)$, has density proportional to

$$p_0(x)p(u_1|x)\dots p(u_N|x) =: e^{-V(x)},$$
$$V(x) := -\log(p_0(x)) - \log(p(u_1|x)) - \dots - \log(p(u_N|x)),$$

which is invariant measure of $dX_s = -\nabla V(x)ds + \sqrt{2}dW_s$ (Welling and Teh, 2011).

Bayesian inference and sampling from distribution

Stochastic algorithms are also used for sampling from a probability measure.

- Given data $u_1, \dots, u_N$ with $N \gg 1$, we consider a family of probability distributions $\{p(u|x)du : \ x \in \mathbb{R}^d\}$ and a prior densities $p_0(x)dx$. Then the posterior distribution on x, $p(x|u_1, \dots, u_N)$, has density proportional to

$$p_0(x)p(u_1|x)\dots p(u_N|x) =: e^{-V(x)},$$
$$V(x) := -\log(p_0(x)) - \log(p(u_1|x)) - \dots - \log(p(u_N|x)),$$

  which is invariant measure of $dX_s = -\nabla V(x)ds + \sqrt{2}dW_s$ (Welling and Teh, 2011).
- (Lamberton and Pagès, 2002, 2003) introduce and analyze sampling from a probability measure ν as invariant measure of $dX_t = b(X_t)dt + \sigma(X_t)dW_t$:

Bayesian inference and sampling from distribution

Stochastic algorithms are also used for sampling from a probability measure.

- Given data $u_1, \dots, u_N$ with $N \gg 1$, we consider a family of probability distributions $\{p(u|x)du : \ x \in \mathbb{R}^d\}$ and a prior densities $p_0(x)dx$. Then the posterior distribution on x, $p(x|u_1, \dots, u_N)$, has density proportional to

$$p_0(x)p(u_1|x)\dots p(u_N|x) =: e^{-V(x)},$$
$$V(x) := -\log(p_0(x)) - \log(p(u_1|x)) - \dots - \log(p(u_N|x)),$$

  which is invariant measure of $dX_s = -\nabla V(x)ds + \sqrt{2}dW_s$ (Welling and Teh, 2011).
- (Lamberton and Pagès, 2002, 2003) introduce and analyze sampling from a probability measure ν as invariant measure of $dX_t = b(X_t)dt + \sigma(X_t)dW_t$:

$$\bar{X}_{n+1} = \bar{X}_n + \gamma_{n+1} b(\bar{X}_n) + \sqrt{\gamma_{n+1}}\sigma(\bar{X}_n)U_{n+1}, \quad U_n \sim \mathcal{N}(0, I_d) \text{ i.i.d.},$$
$$\nu_n := \frac{1}{\Gamma_n}\sum_{k=1}^{n} \gamma_k \delta_{\bar{X}_k}, \quad \Gamma_n = \gamma_1 + \dots + \gamma_n.$$

Langevin Simulated Annealing

- Another possibility : make $\sigma \to 0$ while iterating the algorithm:

Langevin Simulated Annealing

- Another possibility : make $\sigma \to 0$ while iterating the algorithm:

$$x_{n+1} = x_n - \gamma_{n+1}\nabla V(x_n) + a(\gamma_1 + \cdots + \gamma_{n+1})\sigma\sqrt{\gamma_{n+1}}\xi_{n+1}, \quad \xi_{n+1} \sim \mathcal{N}(0, I_d),$$

where $a(t)$ is decreasing and $a(t) \underset{t\to\infty}{\longrightarrow} 0$.

Langevin Simulated Annealing

- Another possibility : make $\sigma \to 0$ while iterating the algorithm:

$$x_{n+1} = x_n - \gamma_{n+1}\nabla V(x_n) + a(\gamma_1 + \cdots + \gamma_{n+1})\sigma\sqrt{\gamma_{n+1}}\xi_{n+1}, \quad \xi_{n+1} \sim \mathcal{N}(0, I_d),$$

where $a(t)$ is decreasing and $a(t) \underset{t\to\infty}{\longrightarrow} 0$.
The continuous version becomes :

Langevin-Simulated Annealing Equation

$$dX_t = -\nabla V(X_t)dt + a(t)\sigma dW_t,$$

Langevin Simulated Annealing

- Another possibility : make $\sigma \to 0$ while iterating the algorithm:

$$x_{n+1} = x_n - \gamma_{n+1}\nabla V(x_n) + a(\gamma_1 + \cdots + \gamma_{n+1})\sigma\sqrt{\gamma_{n+1}}\xi_{n+1}, \quad \xi_{n+1} \sim \mathcal{N}(0, I_d),$$

where $a(t)$ is decreasing and $a(t) \underset{t\to\infty}{\longrightarrow} 0$.
The continuous version becomes :

Langevin-Simulated Annealing Equation

$$dX_t = -\nabla V(X_t)dt + a(t)\sigma dW_t,$$

- The 'instantaneous' invariant measure $\nu_{a(t)\sigma}(dx) \propto \exp\left(-2V(x)/(a^2(t)\sigma^2)\right)$ converges itself to argmin(V)

Langevin Simulated Annealing

- Another possibility : make $\sigma \to 0$ while iterating the algorithm:

$$x_{n+1} = x_n - \gamma_{n+1}\nabla V(x_n) + a(\gamma_1 + \cdots + \gamma_{n+1})\sigma\sqrt{\gamma_{n+1}}\xi_{n+1}, \quad \xi_{n+1} \sim \mathcal{N}(0, I_d),$$

where $a(t)$ is decreasing and $a(t) \underset{t\to\infty}{\longrightarrow} 0$.
The continuous version becomes :

Langevin-Simulated Annealing Equation

$$dX_t = -\nabla V(X_t)dt + a(t)\sigma dW_t,$$

- The 'instantaneous' invariant measure $\nu_{a(t)\sigma}(dx) \propto \exp\left(-2V(x)/(a^2(t)\sigma^2)\right)$ converges itself to $\text{argmin}(V)$
- Schedule $a(t) = A\log^{-1/2}(t)$ then $X_t \underset{t\to\infty}{\longrightarrow} \text{argmin}(V)$ in law (Chiang et al., 1987; Miclo, 1992)

Langevin Simulated Annealing

- Another possibility : make $\sigma \to 0$ while iterating the algorithm:

$$x_{n+1} = x_n - \gamma_{n+1}\nabla V(x_n) + a(\gamma_1 + \cdots + \gamma_{n+1})\sigma\sqrt{\gamma_{n+1}}\xi_{n+1}, \quad \xi_{n+1} \sim \mathcal{N}(0, I_d),$$

where $a(t)$ is decreasing and $a(t) \underset{t\to\infty}{\longrightarrow} 0$.
The continuous version becomes :

Langevin-Simulated Annealing Equation

$$dX_t = -\nabla V(X_t)dt + a(t)\sigma dW_t,$$

- The 'instantaneous' invariant measure $\nu_{a(t)\sigma}(dx) \propto \exp\left(-2V(x)/(a^2(t)\sigma^2)\right)$ converges itself to $\mathrm{argmin}(V)$
- Schedule $a(t) = A\log^{-1/2}(t)$ then $X_t \underset{t\to\infty}{\longrightarrow} \mathrm{argmin}(V)$ in law (Chiang et al., 1987; Miclo, 1992)
- (Gelfand and Mitter, 1991): the convergence of the algorithm (x_n) .

Convergence of Langevin Simulated Annealing with Kullback-Liebler divergence

$$dX_t = -\nabla V(X_t)dt + a(t)\sigma dW_t,$$

Idea of proof in (Miclo, 1992):

Convergence of Langevin Simulated Annealing with Kullback-Liebler divergence

$$dX_t = -\nabla V(X_t)dt + a(t)\sigma dW_t,$$

Idea of proof in (Miclo, 1992):

- We consider the KL-divergence:

$$\mathcal{J}_t := \mathrm{d}_{\mathrm{KL}}\left(X_t \| \nu_{a(t)}\right) = \int_{\mathbb{R}^d} \log\left(\frac{p(t,x)}{\nu_{a(t)}(x)}\right) p(t,x)dx,$$

with $p(x,t)$ the density of X_t.

Convergence of Langevin Simulated Annealing with Kullback-Liebler divergence

$$dX_t = -\nabla V(X_t)dt + a(t)\sigma dW_t,$$

Idea of proof in (Miclo, 1992):

- We consider the KL-divergence:

$$\mathcal{J}_t := \mathrm{d}_{\mathrm{KL}}\left(X_t \| \nu_{a(t)}\right) = \int_{\mathbb{R}^d} \log\left(\frac{p(t,x)}{\nu_{a(t)}(x)}\right) p(t,x)dx,$$

  with $p(x,t)$ the density of X_t.
- Fokker-Planck equation:

$$\partial_t p(t,x) = \nabla \cdot (\nabla V(x) p(t,x)) + \frac{1}{2}a^2(t)\Delta p(t,x)$$

  .
- Log-Sobolev inequality:

$$\int_{\mathbb{R}^d} f^2 \log(f^2) d\nu_{a(t)} \le C \int_{\mathbb{R}^d} |\nabla f|^2 d\nu_{a(t)} + \left(\int_{\mathbb{R}^d} f^2 d\nu_{a(t)}\right) \log\left(\int_{\mathbb{R}^d} f^2 d\nu_{a(t)}\right)$$

- Using integration by parts on $d\mathcal{J}/dt$, we obtain a bound and the convergence of $\mathcal{J}_t$ to 0.

Multiplicative noise and Adaptive algorithms

- Noise $\sigma > 0 \implies$ isotropic, homogeneous noise $\implies$ not adapted to V.

Multiplicative noise and Adaptive algorithms

- Noise $\sigma > 0 \implies$ isotropic, homogeneous noise $\implies$ not adapted to V.
- Instead : $\sigma(X_t)$ is a matrix depending on the position.

Multiplicative noise and Adaptive algorithms

- Noise $\sigma > 0 \implies$ isotropic, homogeneous noise $\implies$ not adapted to V.
- Instead : $\sigma(X_t)$ is a matrix depending on the position.
- Extensively used in Machine Learning without theoretical guarantee.

Multiplicative noise and Adaptive algorithms

- Noise $\sigma > 0 \implies$ isotropic, homogeneous noise $\implies$ not adapted to V.
- Instead : $\sigma(X_t)$ is a matrix depending on the position.
- Extensively used in Machine Learning without theoretical guarantee.

$$dY_t = -(\sigma\sigma^\top \nabla V)(Y_t)dt + a(t)\sigma(Y_t)dW_t + \underbrace{\left(a^2(t)\left[\sum_{j=1}^{d} \partial_i(\sigma\sigma^\top)(Y_t)_{ij}\right]_{1\le i\le d}\right)}_{\text{correction term}\quad \Upsilon(Y_t)} dt$$

$a(t) = A/\sqrt{\log(t)}$.

Multiplicative noise and Adaptive algorithms

- Noise $\sigma > 0 \implies$ isotropic, homogeneous noise $\implies$ not adapted to V.
- Instead : $\sigma(X_t)$ is a matrix depending on the position.
- Extensively used in Machine Learning without theoretical guarantee.

$$dY_t = -(\sigma\sigma^\top \nabla V)(Y_t)dt + a(t)\sigma(Y_t)dW_t + \underbrace{\left(a^2(t) \left[\sum_{j=1}^{d} \partial_i(\sigma\sigma^\top)(Y_t)_{ij} \right]_{1\le i\le d} \right)}_{\text{correction term} \quad \Upsilon(Y_t)} dt$$

$a(t) = A/\sqrt{\log(t)}$.

• Correction term so that $\nu_{a(t)} \propto \exp\left(-2V(x)/a^2(t)\right)$ is still the "instantaneous" invariant measure (Li et al., 2016; Pagès and Panloup, 2023).

Outline

1. Introduction
   - Optimization
   - Stochastic gradient descent algorithm
   - Langevin equation and algorithms
   - Objectives

Objectives

I: Convergence of adaptive Langevin algorithms

- Convergence of the Langevin equation Y_t with multiplicative noise to argmin(V) as well as the discretized scheme $\bar{Y}_t$.
- Weak convergence for Wasserstein-1 and Total Variation.
- For $\mathcal{D} = \mathcal{W}_1$ or d_{TV} and $\nu^\star$ being the target measure, we have

$$\mathcal{D}(Y_t, \nu^\star) \le \mathcal{D}(Y_t, \nu_{a(t)}) + \mathcal{D}(\nu_{a(t)}, \nu^\star).$$

II: Adaptive Langevin algorithms for deep neural networks

- Implement Langevin algorithms for different choices of σ (Adam, RMSprop, Adadelta etc) and compare with their corresponding non-Langevin counterpart.
- Investigate the benefits of Langevin algorithms on very deep learning.

Outline

2. Convergence of adaptive Langevin-Simulated Annealing algorithms
   - Convergence of Langevin-Simulated Annealing algorithms for $\mathcal{W}_1$ and d_{TV}
   - Convergence rates of Gibbs measures with degenerate minimum

Outline

2. Convergence of adaptive Langevin-Simulated Annealing algorithms
   - Convergence of Langevin-Simulated Annealing algorithms for $\mathcal{W}_1$ and d_{TV}
   - Convergence rates of Gibbs measures with degenerate minimum

Convergence of Langevin-Simulated Annealing algorithms for $\mathcal{W}_1$ and d_{TV}

- Pierre Bras and Gilles Pagès. *Convergence of Langevin-Simulated Annealing algorithms with multiplicative noise*. *Mathematics of Computation*, 2023.
- Pierre Bras and Gilles Pagès. *Convergence of Langevin-Simulated Annealing algorithms with multiplicative noise II: Total Variation*. *Monte Carlo Methods and Applications*, 29(3):203–219, 2023.

Objectives and assumptions

$$dY_t = -(\sigma\sigma^\top \nabla V)(Y_t)dt + a(t)\sigma(Y_t)dW_t + \underbrace{\left(a^2(t) \left[\sum_{j=1}^{d} \partial_i(\sigma\sigma^\top)(Y_t)_{ij} \right]_{1\le i\le d} \right)}_{\text{correction term} \quad \Upsilon(Y_t)} dt$$

$a(t) = A/\sqrt{\log(t)}$,

$\nu_{a(t)} \propto \exp\left(-2V(x)/a^2(t)\right)$ instantaneous invariant measure, $\nu^\star = \lim_{a\to 0} \nu_a$.

Objectives and assumptions

$$dY_t = -(\sigma\sigma^\top \nabla V)(Y_t)dt + a(t)\sigma(Y_t)dW_t + \underbrace{\left(a^2(t) \left[\sum_{j=1}^{d} \partial_i(\sigma\sigma^\top)(Y_t)_{ij} \right]_{1\le i\le d} \right)}_{\text{correction term } \Upsilon(Y_t)} dt$$

$a(t) = A/\sqrt{\log(t)}$,

$\nu_{a(t)} \propto \exp\left(-2V(x)/a^2(t)\right)$ instantaneous invariant measure, $\quad \nu^\star = \lim_{a\to 0} \nu_a$.

Prove the convergence of Y_t to $\nu^\star$ for $\mathcal{W}_1$ and d_{TV}:

$$\mathcal{W}_1(X,Y) = \sup\left\{ \left|\mathbb{E}[f(X)] - \mathbb{E}[f(Y)]\right| : \ [f]_{\mathrm{Lip}} = 1 \right\},$$

$$\mathrm{d}_{\mathrm{TV}}(X,Y) = \sup\left\{ \left|\mathbb{E}[f(X)] - \mathbb{E}[f(Y)]\right| : \ \sup_{\mathbb{R}^d} f = 1 \right\}.$$

Objectives and assumptions

$$dY_t = -(\sigma\sigma^\top \nabla V)(Y_t)dt + a(t)\sigma(Y_t)dW_t + \underbrace{\left(a^2(t) \left[\sum_{j=1}^d \partial_i(\sigma\sigma^\top)(Y_t)_{ij} \right]_{1\le i\le d} \right)}_{\text{correction term} \quad \Upsilon(Y_t)} dt$$

$a(t) = A/\sqrt{\log(t)}$,

$\nu_{a(t)} \propto \exp\left(-2V(x)/a^2(t)\right)$ instantaneous invariant measure, $\quad \nu^\star = \lim_{a\to 0} \nu_a$.

Prove the convergence of Y_t to $\nu^\star$ for $\mathcal{W}_1$ and d_{TV}:

$$\mathcal{W}_1(X,Y) = \sup\left\{ \left|\mathbb{E}[f(X)] - \mathbb{E}[f(Y)]\right| : \ [f]_{\mathrm{Lip}} = 1 \right\},$$

$$\mathrm{d}_{\mathrm{TV}}(X,Y) = \sup\left\{ \left|\mathbb{E}[f(X)] - \mathbb{E}[f(Y)]\right| : \ \sup_{\mathbb{R}^d} f = 1 \right\}.$$

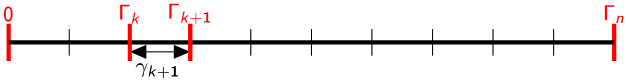
Important assumptions (Pagès and Panloup, 2023):

1. V is strongly convex outside some compact set, ∇V is Lipschitz.
2. σ is bounded and elliptic: $\sigma\sigma^\top \ge \sigma_0 I_d$, $\sigma_0 > 0$.

Domino strategy

- (Pagès and Panloup, 2023): convergence of the Euler scheme of a general SDE $dX_t = b(X_t)dt + \sigma(X_t)dW_t$ to the invariant measure $\nu^\star$.
- *Domino strategy:* (Pagès and Panloup, 2023) for f 1-Lipschitz, P^1, P^2 kernels of processes X, Y, (γ_n) step sequence and $\Gamma_n := \gamma_1 + \cdots + \gamma_n$, we have:

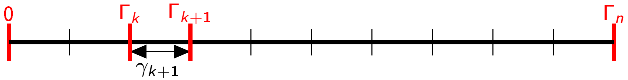
$$\begin{aligned}
\mathcal{W}_1(Y_{\Gamma_n}, X_{\Gamma_n}) &\le |\mathbb{E}f(Y_{\Gamma_n}) - \mathbb{E}f(X_{\Gamma_n})| \\
&= |P^2_{\gamma_1} \circ \cdots \circ P^2_{\gamma_n} f(x) - P^1_{\Gamma_n} f(x)| \\
&= \left| \sum_{k=1}^{n} P^2_{\gamma_1} \circ \cdots \circ P^2_{\gamma_{k-1}} \circ (P^2_{\gamma_k} - P^1_{\gamma_k}) \circ P^1_{\Gamma_n - \Gamma_k} f(x) \right| \\
&\le \sum_{k=1}^{n} \left| P^2_{\gamma_1} \circ \cdots \circ P^2_{\gamma_{k-1}} \circ (P^2_{\gamma_k} - P^1_{\gamma_k}) \circ P^1_{\Gamma_n - \Gamma_k} f(x) \right|,
\end{aligned}$$



Domino strategy

- (Pagès and Panloup, 2023): convergence of the Euler scheme of a general SDE $dX_t = b(X_t)dt + \sigma(X_t)dW_t$ to the invariant measure $\nu^\star$.
- *Domino strategy:* (Pagès and Panloup, 2023) for f 1-Lipschitz, P^1, P^2 kernels of processes X, Y, (γ_n) step sequence and $\Gamma_n := \gamma_1 + \cdots + \gamma_n$, we have:

$$\begin{aligned}
\mathcal{W}_1(Y_{\Gamma_n}, X_{\Gamma_n}) &\le |\mathbb{E}f(Y_{\Gamma_n}) - \mathbb{E}f(X_{\Gamma_n})| \\
&= |P^2_{\gamma_1} \circ \cdots \circ P^2_{\gamma_n} f(x) - P^1_{\Gamma_n} f(x)| \\
&= \left| \sum_{k=1}^{n} P^2_{\gamma_1} \circ \cdots \circ P^2_{\gamma_{k-1}} \circ (P^2_{\gamma_k} - P^1_{\gamma_k}) \circ P^1_{\Gamma_n - \Gamma_k} f(x) \right| \\
&\le \sum_{k=1}^{n} \left| P^2_{\gamma_1} \circ \cdots \circ P^2_{\gamma_{k-1}} \circ (P^2_{\gamma_k} - P^1_{\gamma_k}) \circ P^1_{\Gamma_n - \Gamma_k} f(x) \right|,
\end{aligned}$$



In the sum we bound two types of terms:

1. For large $k \implies$ Error in small time $\implies$ use bounds for $\|X_t^x - Y_t^x\|_p$
2. For small $k \implies$ Ergodic properties (Eberle, 2016; Wang, 2020).

Contraction property with ellipticity parameter a

• Problem: non-homogeneous Markov chain + the ellipticity parameter fades away in $a(t)$.
$\Longrightarrow$ What is the dependency of the constants C and ρ in the ellipticity ?

Contraction property with ellipticity parameter a

• Problem: non-homogeneous Markov chain + the ellipticity parameter fades away in $a(t)$.
$\Longrightarrow$ What is the dependency of the constants C and ρ in the ellipticity ?

Consider $dX_t = b(X_t)dt + a\sigma(X_t)dW_t$, $a > 0$ with invariant measure ν_a and with

$$\forall x, y \in \mathcal{B}(0,R)^c,\ \langle b(x) - b(y), x - y\rangle + \frac{a^2}{2}\|\sigma(x) - \sigma(y)\|^2 \leq -\alpha|x-y|^2.$$

Then

$$\mathcal{W}_1(X_t^x, X_t^y) \leq Ce^{C_1/a^2}|x-y|e^{-\rho_a t}, \quad \rho_a := e^{-C_2/a^2}$$
$$\mathcal{W}_1(X_t^x, \nu_a) \leq Ce^{C_1/a^2}e^{-\rho_a t}\mathbb{E}|\nu_a - x|.$$

"By plateaux" process

We first consider the plateau SDE:

$$dX_t = -\sigma\sigma^\top \nabla V(X_t)dt + a_{n+1}\sigma(X_t)dW_t + a_{n+1}^2 \Upsilon(X_t)dt, \quad t \in [T_n, T_{n+1}),$$
$$a_n = A \log^{-1/2}(T_n)$$

"By plateaux" process

We first consider the plateau SDE:

$$dX_t = -\sigma\sigma^\top \nabla V(X_t)dt + a_{n+1}\sigma(X_t)dW_t + a_{n+1}^2 \Upsilon(X_t)dt, \quad t \in [T_n, T_{n+1}),$$
$$a_n = A \log^{-1/2}(T_n)$$

We apply the contraction property on every plateau:

$$\mathcal{W}_1(X_{T_{n+1}}, \nu_{a_{n+1}} \,|X_{T_n}) \leq C e^{C_1/a_{n+1}^2} e^{-\rho_{a_{n+1}}(T_{n+1}-T_n)} \mathbb{E}\left[|\nu_{a_{n+1}} - X_{T_n}| \; |X_{T_n}\right].$$

"By plateaux" process

We first consider the plateau SDE:

$$dX_t = -\sigma\sigma^\top \nabla V(X_t)dt + a_{n+1}\sigma(X_t)dW_t + a_{n+1}^2 \Upsilon(X_t)dt, \quad t \in [T_n, T_{n+1}),$$
$$a_n = A \log^{-1/2}(T_n)$$

We apply the contraction property on every plateau:

$$\mathcal{W}_1(X_{T_{n+1}}, \nu_{a_{n+1}} \; | X_{T_n}) \leq C e^{C_1/a_{n+1}^2} e^{-\rho_{a_{n+1}}(T_{n+1}-T_n)} \mathbb{E}\left[|\nu_{a_{n+1}} - X_{T_n}| \; | X_{T_n} \right].$$

We integrate over the law of X_{T_n}, giving

$$\begin{aligned}\mathcal{W}_1(X^{x_0}_{T_{n+1}}, \nu_{a_{n+1}}) &\leq C e^{C_1/a_{n+1}^2} e^{-\rho_{a_{n+1}}(T_{n+1}-T_n)} \mathcal{W}_1(X^{x_0}_{T_n}, \nu_{a_{n+1}}) \\ &\leq C e^{C_1/a_{n+1}^2} e^{-\rho_{a_{n+1}}(T_{n+1}-T_n)} \left(\mathcal{W}_1(X^{x_0}_{T_n}, \nu_{a_n}) + \mathcal{W}_1(\nu_{a_n}, \nu_{a_{n+1}}) \right).\end{aligned}$$

"By plateaux" process

We first consider the plateau SDE:

$$dX_t = -\sigma\sigma^\top \nabla V(X_t)dt + a_{n+1}\sigma(X_t)dW_t + a_{n+1}^2\Upsilon(X_t)dt, \quad t \in [T_n, T_{n+1}),$$
$$a_n = A\log^{-1/2}(T_n)$$

We apply the contraction property on every plateau:

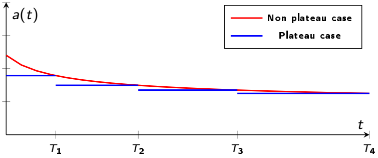
$$\mathcal{W}_1(X_{T_{n+1}}, \nu_{a_{n+1}} \ |X_{T_n}) \leq Ce^{C_1/a_{n+1}^2}e^{-\rho_{a_{n+1}}(T_{n+1}-T_n)}\mathbb{E}\left[|\nu_{a_{n+1}} - X_{T_n}| \ |X_{T_n}\right].$$

We integrate over the law of X_{T_n}, giving

$$\begin{aligned}\mathcal{W}_1(X_{T_{n+1}}^{x_0}, \nu_{a_{n+1}}) &\leq Ce^{C_1/a_{n+1}^2}e^{-\rho_{a_{n+1}}(T_{n+1}-T_n)}\mathcal{W}_1(X_{T_n}^{x_0}, \nu_{a_{n+1}}) \\ &\leq Ce^{C_1/a_{n+1}^2}e^{-\rho_{a_{n+1}}(T_{n+1}-T_n)}\left(\mathcal{W}_1(X_{T_n}^{x_0}, \nu_{a_n}) + \mathcal{W}_1(\nu_{a_n}, \nu_{a_{n+1}})\right).\end{aligned}$$

And we iterate:

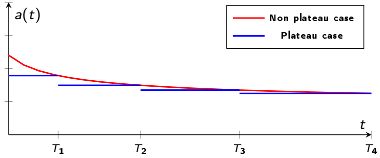
$$\begin{aligned}\mathcal{W}_1(X_{T_{n+1}}^{x_0}, \nu_{a_{n+1}}) &\leq \mu_{n+1}\mathcal{W}_1(\nu_{a_n}, \nu_{a_{n+1}}) + \mu_{n+1}\mu_n\mathcal{W}_1(\nu_{a_{n-1}}, \nu_{a_n}) + \cdots \\ &\quad + \mu_{n+1}\cdots\mu_1\mathcal{W}_1(\nu_{a_0}, \nu_{a_1}) + \mu_{n+1}\cdots\mu_1\mathcal{W}_1(\delta_{x_0}, \nu_{a_0}), \\ \mu_n &:= Ce^{C_1/a_n^2}e^{-\rho_{a_n}(T_n-T_{n-1})}.\end{aligned}$$



$$\mathcal{W}_1(X^{x_0}_{T_{n+1}}, \nu_{a_{n+1}}) \le \mu_{n+1}\mathcal{W}_1(\nu_{a_n}, \nu_{a_{n+1}}) + \mu_{n+1}\mu_n\mathcal{W}_1(\nu_{a_{n-1}}, \nu_{a_n}) + \cdots$$
$$+ \mu_{n+1}\cdots\mu_1\mathcal{W}_1(\nu_{a_0}, \nu_{a_1}) + \mu_{n+1}\cdots\mu_1\mathcal{W}_1(\delta_{x_0}, \nu_{a_0}),$$
$$\mu_n = Ce^{C_1/a_n^2}e^{-\rho_{a_n}(T_n - T_{n-1})}, \quad \rho_{a_n} = e^{-C_2/a_n^2}$$

We use (technical)

$$\mathcal{W}_1(\nu_{a_n}, \nu_{a_{n+1}}) \le C(a_n - a_{n+1}).$$



$$\mathcal{W}_1(X^{x_0}_{T_{n+1}}, \nu_{a_{n+1}}) \leq \mu_{n+1}\mathcal{W}_1(\nu_{a_n}, \nu_{a_{n+1}}) + \mu_{n+1}\mu_n\mathcal{W}_1(\nu_{a_{n-1}}, \nu_{a_n}) + \cdots$$
$$+ \mu_{n+1}\cdots\mu_1\mathcal{W}_1(\nu_{a_0}, \nu_{a_1}) + \mu_{n+1}\cdots\mu_1\mathcal{W}_1(\delta_{x_0}, \nu_{a_0}),$$
$$\mu_n = Ce^{C_1/a_n^2}e^{-\rho_{a_n}(T_n - T_{n-1})}, \quad \rho_{a_n} = e^{-C_2/a_n^2}$$

We use (technical)

$$\mathcal{W}_1(\nu_{a_n}, \nu_{a_{n+1}}) \leq C(a_n - a_{n+1}).$$

We now choose

$$T_{n+1} - T_n = Cn^\beta, \beta > 0, \quad a_n = \frac{A}{\sqrt{\log(T_n)}}, \quad A > 0 \text{ large enough}$$

yielding

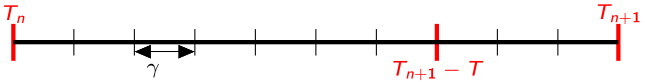
$$\mathcal{W}_1(X^{x_0}_{T_{n+1}}, \nu_{a_{n+1}}) \leq C(1 + |x_0|)\mu_n a_n,$$

where $\mu_n = O\left(\exp(-Cn^\eta)\right)$.

- This gives the convergence of X_t to its instantaneous invariant measure.
- To get the convergence of X_t to $\nu^\star$, we rely to classical bounds on $\mathcal{W}_1(\nu_{a_n}, \nu^\star)$ (see later).

Convergence of Y_t with continuously decreasing $(a(t))$

- We apply *domino strategy* to bound $\mathcal{W}_1(X_t, Y_t)$:



- For f Lipschitz-continuous and fixed $T > 0$:

$$\left|\mathbb{E}f(X^{x,n}_{T_{n+1}-T_n}) - \mathbb{E}f(Y^x_{T_{n+1}-T_n,T_n})\right|$$
$$\leq \sum_{k=1}^{\lfloor (T_{n+1}-T_n-T)/\gamma \rfloor} \left| P^Y_{(k-1)\gamma,T_n} \circ (P^Y_{\gamma,T_n+(k-1)\gamma} - P^{X,n}_\gamma) \circ P^{X,n}_{T_{n+1}-T_n-k\gamma} f(x)\right|$$
$$+ \sum_{k=\lfloor (T_{n+1}-T_n-T)/\gamma \rfloor+1}^{\lfloor (T_{n+1}-T_n)/\gamma \rfloor} \left| P^Y_{(k-1)\gamma,T_n} \circ (P^Y_{\gamma,T_n+(k-1)\gamma} - P^{X,n}_\gamma) \circ P^{X,n}_{T_{n+1}-T_n-k\gamma} f(x)\right|$$

- For $k = 1, \ldots, (T_{n+1} - T_n - T)/\gamma$, the kernel $P^{X,n}_{T_{n+1}-T_n-k\gamma}$ has an exponential contraction effect on time $> T$:

$$|(P^Y_{\gamma,T_n+(k-1)\gamma} - P^{X,n}_\gamma) \circ P^{X,n}_{T_{n+1}-T_n-k\gamma} f(x)| \leq Ce^{C_1 a_{n+1}^{-2}} e^{-\rho_{n+1}(T_{n+1}-T_n-k\gamma)}[f]_{\text{Lip}}\sqrt{\gamma}(a_n - a_{n+1}).$$

- Bounds for the error on time intervals no longer than T:

$$|(P^Y_{\gamma,T_n+(k-1)\gamma} - P^{X,n}_\gamma) \circ P^{X,n}_{T_{n+1}-T_n-k\gamma} f(x)| \leq Ca_{n+1}^{-2}(a_n - a_{n+1})[f]_{\text{Lip}} \frac{\gamma V(x)}{\sqrt{T_{n+1} - T_n - k\gamma}}$$

- We apply on the time interval $[T_n, T_{n+1}]$ and obtain the recursive inequality

$$\mathcal{W}_1(X^{x,n}_{T_{n+1}-T_n}, Y^{x}_{T_{n+1}-T_n,T_n}) \leq \underbrace{Ce^{C_1 a_{n+1}^{-2}}(a_n - a_{n+1})\rho_{n+1}^{-1}}_{=:\lambda_{n+1}} V(x).$$

With $x_n := X^{x_0}_{T_n}$, $y_n = Y^{x_0}_{T_n}$:

$$\begin{aligned}\mathcal{W}_1(X^{x_0}_{T_{n+1}}, Y^{x_0}_{T_{n+1}}) &= \mathcal{W}_1(X^{x_n,n}_{T_{n+1}-T_n}, Y^{y_n}_{T_{n+1}-T_n,T_n}) \\ &\leq \mathcal{W}_1(X^{x_n,n}_{T_{n+1}-T_n}, X^{y_n,n}_{T_{n+1}-T_n}) + \mathcal{W}_1(X^{y_n,n}_{T_{n+1}-T_n}, Y^{y_n}_{T_{n+1}-T_n,T_n}) \\ &\leq \underbrace{Ce^{C_1 a_{n+1}^{-2}} e^{-\rho_{n+1}(T_{n+1}-T_n)}}_{\mu_{n+1}} \mathcal{W}_1(X^{x_0}_{T_n}, Y^{x_0}_{T_n}) + \underbrace{Ce^{C_1 a_{n+1}^{-2}}(a_n - a_{n+1})\rho_{n+1}^{-1}}_{\lambda_{n+1}} \mathbb{E}V(Y^{x_0}_{T_n}),\end{aligned}$$

The convergence is controlled by

$$\lambda_{n+1} := Ce^{C_1 a_{n+1}^{-2}}(a_n - a_{n+1})\rho_{n+1}^{-1}$$

with

$$a_n \simeq \frac{A}{\sqrt{\log(T_n)}}$$

$$T_{n+1} \simeq Cn^{\beta+1}$$

$$a_n - a_{n+1} \asymp \frac{1}{n\log^{3/2}(n)}$$

$$e^{C_1 a_{n+1}^{-2}} \simeq n^{(\beta+1)C_1/A^2}$$

$$\rho_n^{-1} = e^{C_2 a_{n+1}^{-2}} \simeq n^{(\beta+1)C_2/A^2}$$

$$\implies \lambda_n \asymp n^{-(1-(\beta+1)(C_1+C_2)/A^2)}$$

and we choose A large enough such that

$$1 - (\beta+1)(C_1 + C_2)/A^2 > 0.$$

Then:

$$\begin{aligned}\mathcal{W}_1(Y_{T_{n+1}}^{x_0}, \nu_{a_{n+1}}) &\leq \mathcal{W}_1(Y_{T_{n+1}}^{x_0}, X_{T_{n+1}}^{x_0}) + \mathcal{W}_1(X_{T_{n+1}}^{x_0}, \nu_{a_{n+1}}) \\ &\lessapprox CV(x_0)n^{-(1-(\beta+1)(C_1+C_2)/A^2)}\end{aligned}$$

$$\mathcal{W}_1(Y_{T_{n+1}}^{x_0}, \nu^\star) \leq \mathcal{W}_1(Y_{T_{n+1}}^{x_0}, X_{T_{n+1}}^{x_0}) + \mathcal{W}_1(X_{T_{n+1}}^{x_0}, \nu^\star) \lessapprox CV(x_0)a_n$$

Plan of the proof

- Ellipticity parameter $a(t) \to 0 \implies$ we rework the dependency of the ergodic bound in the ellipticity for a general SDE.

Plan of the proof

- Ellipticity parameter $a(t) \to 0 \implies$ we rework the dependency of the ergodic bound in the ellipticity for a general SDE.
- We then prove the convergence for the auxiliary "by plateau" process:

$$dX_t = -\sigma\sigma^\top \nabla V(X_t)dt + a_{n+1}\sigma(X_t)dW_t + a_{n+1}^2 \Upsilon(X_t)dt, \quad t \in [T_n, T_{n+1}),$$
$$a_n = A\log^{-1/2}(T_n),$$

and obtain ergodic bounds for $\mathcal{W}_1(X_{T_{n+1}}, \nu_{a_{n+1}})$; then

$$\mathcal{W}_1(X_{T_n}, \nu^\star) \le \mathcal{W}_1(X_{T_n}, \nu_{a_n}) + \mathcal{W}_1(\nu_{a_n}, \nu^\star) \to 0.$$

Plan of the proof

- Ellipticity parameter $a(t) \to 0 \implies$ we rework the dependency of the ergodic bound in the ellipticity for a general SDE.
- We then prove the convergence for the auxiliary "by plateau" process:

$$dX_t = -\sigma\sigma^\top \nabla V(X_t)dt + a_{n+1}\sigma(X_t)dW_t + a_{n+1}^2 \Upsilon(X_t)dt, \quad t \in [T_n, T_{n+1}),$$
$$a_n = A \log^{-1/2}(T_n),$$

and obtain ergodic bounds for $\mathcal{W}_1(X_{T_{n+1}}, \nu_{a_{n+1}})$; then

$$\mathcal{W}_1(X_{T_n}, \nu^\star) \leq \mathcal{W}_1(X_{T_n}, \nu_{a_n}) + \mathcal{W}_1(\nu_{a_n}, \nu^\star) \to 0.$$

- We then use the *domino strategy* to give bounds on $\mathcal{W}_1(X_{T_n}, Y_{T_n})$:

$$\mathcal{W}_1(Y_{T_n}, \nu^\star) \leq \mathcal{W}_1(Y_{T_n}, X_{T_n}) + \mathcal{W}_1(X_{T_n}, \nu^\star) \to 0.$$

Convergence of the Euler scheme with decreasing steps

Euler-Maruyama scheme

$$\bar{Y}^{x_0}_{\Gamma_{n+1}} = \bar{Y}_{\Gamma_n} + \gamma_{n+1}\left(b_{a(\Gamma_n)}(\bar{Y}^{x_0}_{\Gamma_n}) + \zeta_{n+1}(\bar{Y}^{x_0}_{\Gamma_n})\right) + a(\Gamma_n)\sigma(\bar{Y}^{x_0}_{\Gamma_n})(W_{\Gamma_{n+1}} - W_{\Gamma_n})$$

γ_{n+1} decreasing to 0, $\sum_n \gamma_n = \infty$, $\sum_n \gamma_n^2 < \infty$, $\Gamma_n = \gamma_1 + \cdots + \gamma_n$,

$\forall x,\ \mathbb{E}[\zeta_n(x)] = 0$ (mini-batch noise).

$\implies$ Same strategy of proof.

Total variation case

- Proofs are similar with $\mathcal{W}_1$ distance.

Total variation case

- Proofs are similar with $\mathcal{W}_1$ distance.
- Main difficulty: error bounds in short time. Indeed:

$$|f(X_t) - f(Y_t)| \leq [f]_{\mathrm{Lip}}|X_t - Y_t| \qquad \text{if } f \text{ is Lipschitz.}$$
$$|f(X_t) - f(Y_t)| \leq ??? \qquad \text{if } f \text{ is bounded.}$$

- We investigate d_{TV} bounds in short time for general SDEs:

Total variation case

- Proofs are similar with $\mathcal{W}_1$ distance.
- Main difficulty: error bounds in short time. Indeed:

$$|f(X_t) - f(Y_t)| \leq [f]_{\text{Lip}}|X_t - Y_t| \qquad \text{if } f \text{ is Lipschitz.}$$
$$|f(X_t) - f(Y_t)| \leq ??? \qquad \text{if } f \text{ is bounded.}$$

- We investigate d_{TV} bounds in short time for general SDEs:

For
$$dX_t = b_1(X_t)dt + \sigma_1(X_t)dW_t, \quad dY_t = b_2(Y_t)dt + \sigma_2(Y_t)dW_t,$$
$$X_0 = Y_0, \quad \sigma_1(X_0) = \sigma_2(Y_0),$$
then
$$\mathrm{d}_{\mathrm{TV}}(X_t, Y_t) \leq Ct^{1/2}e^{c\sqrt{\log(1/t)}}.$$

- Pierre Bras, Gilles Pagès, and Fabien Panloup. *Total variation distance between two diffusions in small time with unbounded drift: application to the Euler-Maruyama scheme.*. *Electron. J. Probab.*, 27:1–19, 2022.

These result uses:

Theorem

Let Z_1 and Z_2 be two random vectors admitting densities p_1 and p_2. Then

$$\mathrm{d}_{\mathrm{TV}}(Z_1, Z_2) \leq C_{d,r}\mathcal{W}_1(Z_1, Z_2)^{2r/(2r+1)} \left(\int_{\mathbb{R}^d} \left(\|\nabla^{2r} p_1(\xi)\| + \|\nabla^{2r} p_2(\xi)\| \right) d\xi \right)^{1/(2r+1)}.$$

Outline

2. Convergence of adaptive Langevin-Simulated Annealing algorithms
   - Convergence of Langevin-Simulated Annealing algorithms for $\mathcal{W}_1$ and d_{TV}
   - Convergence rates of Gibbs measures with degenerate minimum

- To get the convergence of Langevin algorithms, we need the convergence of

$$\mathcal{D}(\nu_a, \nu^\star), \quad a \to 0,$$
$$\nu_a(x) \propto \exp\left(-2V(x)/a^2\right),$$
$$\nu^\star = \lim_{a\to 0} \nu_a.$$

- It is known to be of order a if $\operatorname{argmin}(V)$ is finite and $\nabla^2 V(x_i^\star) > 0$ for all $x_i^\star \in \operatorname{argmin}(V)$ (Hwang, 1980, 1981). Then

$$\nu^\star = \left(\sum_i \det(\nabla^2 V(x_i^\star))^{-1/2}\right)^{-1} \sum_i \det(\nabla^2 V(x_i^\star))^{-1/2} \delta_{x_i^\star}.$$

- To get the convergence of Langevin algorithms, we need the convergence of

$$\begin{aligned}
&\mathcal{D}(\nu_a, \nu^\star), \quad a \to 0, \\
&\nu_a(x) \propto \exp\left(-2V(x)/a^2\right), \\
&\nu^\star = \lim_{a \to 0} \nu_a.
\end{aligned}$$

- It is known to be of order a if $\mathrm{argmin}(V)$ is finite and $\nabla^2 V(x_i^\star) > 0$ for all $x_i^\star \in \mathrm{argmin}(V)$ (Hwang, 1980, 1981). Then

$$\nu^\star = \left(\sum_i \det(\nabla^2 V(x_i^\star))^{-1/2} \right)^{-1} \sum_i \det(\nabla^2 V(x_i^\star))^{-1/2} \delta_{x_i^\star}.$$

- We investigate the case where $\mathrm{argmin}(V)$ is finite with **degenerate minimum**.
- This happens in practice when training over-parametrized neural networks (Sagun, Bottou, and LeCun, 2016):

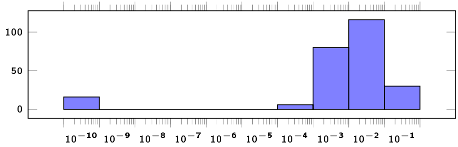


Figure: Distribution of the eigenvalues of the Hessian matrix at the end of training of a neural network on the MNIST dataset.

Considering recursively the spaces of cancellation of $\nabla^{2k}V$, we obtain:

Theorem

Assume that $\mathrm{argmin}(V) = \{x^\star\}$. Define (F_k) recursively as

$$F_0 = \mathbb{R}^d, \; F_k = \{h \in F_{k-1} : \; \forall h' \in F_{k-1}, \; \nabla^{2k}V(x^\star) \cdot h \otimes h'^{\otimes 2k-1} = 0\}$$

and E_k the orthogonal complement of F_k in F_{k-1}. Let B a basis adapted to $\mathbb{R}^d = E_1 \oplus \cdots \oplus E_p$ and $\alpha_i = 1/(2j)$ on the subspace E_j, then if $X \sim \nu_a$:

$$\left(\frac{1}{a^{2\alpha_1}}, \ldots, \frac{1}{a^{2\alpha_d}}\right) * (B^{-1} \cdot (X_{a^2} - x^\star)) \to X \quad \text{as } a \to 0, \; \text{in law},$$

where X has a density proportional to $e^{-g(x)}$ with

$$g(x) = \sum_{k=2}^{2p} \frac{1}{k!} \sum_{\substack{i_1,\ldots,i_p \in \{0,\ldots,k\} \\ i_1+\cdots+i_p=k \\ \frac{i_1}{2}+\cdots+\frac{i_p}{2p}=1}} \binom{k}{i_1,\ldots,i_p} \nabla^k V(x^\star) \cdot p_{E_1}(B\cdot x)^{\otimes i_1} \otimes \cdots \otimes p_{E_p}(B\cdot x)^{\otimes i_p}.$$

Considering recursively the spaces of cancellation of $\nabla^{2k}V$, we obtain:

Theorem

Assume that $\operatorname{argmin}(V) = \{x^\star\}$. Define (F_k) recursively as

$$F_0 = \mathbb{R}^d,\ F_k = \{h \in F_{k-1} :\ \forall h' \in F_{k-1},\ \nabla^{2k}V(x^\star)\cdot h \otimes h'^{\otimes 2k-1} = 0\}$$

and E_k the orthogonal complement of F_k in F_{k-1}. Let B a basis adapted to $\mathbb{R}^d = E_1 \oplus \cdots \oplus E_p$ and $\alpha_i = 1/(2j)$ on the subspace E_j, then if $X \sim \nu_a$:

$$\left(\frac{1}{a^{2\alpha_1}},\ldots,\frac{1}{a^{2\alpha_d}}\right) * (B^{-1}\cdot(X_{a^2} - x^\star)) \to X \quad \text{as } a \to 0, \text{ in law},$$

where X has a density proportional to $e^{-g(x)}$ with

$$g(x) = \sum_{k=2}^{2p}\frac{1}{k!}\sum_{\substack{i_1,\ldots,i_p\in\{0,\ldots,k\}\\ i_1+\cdots+i_p=k\\ \frac{i_1}{2}+\cdots+\frac{i_p}{2p}=1}}\binom{k}{i_1,\ldots,i_p}\nabla^k V(x^\star)\cdot p_{E_1}(B\cdot x)^{\otimes i_1}\otimes\cdots\otimes p_{E_p}(B\cdot x)^{\otimes i_p}.$$

$\implies$ For j such that $2j$ is the maximum order of degeneracy of $\nabla^{2j}V(x^\star)$, then $\mathcal{D}(\nu_a, \nu^\star)$ is of order $a^{1/j}$.

Pierre Bras. *Convergence rates of Gibbs measures with degenerate minimum.* *Bernoulli*, 28(4):2431 – 2458, 2022, (extension of (Athreya and Hwang, 2010)).

Outline

3. Adaptive Langevin algorithms for Neural Networks
   - Langevin versus non-Langevin for very deep learning
   - Langevin algorithms for Markovian Neural Networks and Deep Stochastic control

Outline

3. Adaptive Langevin algorithms for Neural Networks
   - Langevin versus non-Langevin for very deep learning
   - Langevin algorithms for Markovian Neural Networks and Deep Stochastic control

Preconditioned Langevin Gradient Descent

Preconditioned Langevin Gradient Descent (Li et al., 2016)

For some preconditioner rule P_{n+1} depending on the previous updates of the gradient ($g_n \simeq \nabla V(\theta_n)$) and $\sigma > 0$:

$$\text{Preconditioned Gradient Descent:} \quad \theta_{n+1} = \theta_n - \gamma_{n+1} P_{n+1} \cdot g_{n+1},$$

$$\text{Preconditioned Langevin:} \quad \theta_{n+1} = \theta_n - \gamma_{n+1} P_{n+1} \cdot g_{n+1} + \sigma\sqrt{\gamma_{n+1}}\mathcal{N}(0, \boldsymbol{P_{n+1}})$$

Preconditioned Langevin Gradient Descent

Preconditioned Langevin Gradient Descent (Li et al., 2016)

For some preconditioner rule P_{n+1} depending on the previous updates of the gradient ($g_n \simeq \nabla V(\theta_n)$) and $\sigma > 0$:

$$\text{Preconditioned Gradient Descent:} \quad \theta_{n+1} = \theta_n - \gamma_{n+1} P_{n+1} \cdot g_{n+1},$$
$$\text{Preconditioned Langevin:} \quad \theta_{n+1} = \theta_n - \gamma_{n+1} P_{n+1} \cdot g_{n+1} + \sigma\sqrt{\gamma_{n+1}}\mathcal{N}(0, \boldsymbol{P_{n+1}})$$

- Per-dimension adaptive step size.
- Adding noise is known to improve the learning in some cases. (Neelakantan et al., 2015; Anirudh Bhardwaj, 2019; Gulcehre et al., 2016)

Examples of gradient algorithms

Algorithm Adam (Kingma and Ba, 2015)

Parameters: $\beta_1, \beta_2, \lambda > 0$

$$M_{n+1} = \beta_1 M_n + (1-\beta_1) g_{n+1}$$
$$\mathrm{MS}_{n+1} = \beta_2 \, \mathrm{MS}_n + (1-\beta_2) g_{n+1} \odot g_{n+1}$$
$$\widehat{M}_{n+1} = M_{n+1} / (1-\beta_1^{n+1})$$
$$\widehat{\mathrm{MS}}_{n+1} = \mathrm{MS}_{n+1} / (1-\beta_2^{n+1})$$
$$P_{n+1} = \mathrm{diag}\left(\mathbb{1} \oslash \left(\lambda \mathbb{1} + \sqrt{\widehat{\mathrm{MS}}_{n+1}}\right)\right)$$
$$\theta_{n+1} = \theta_n - \gamma_{n+1} P_{n+1} \cdot \widehat{M}_{n+1}.$$

Algorithm RMSprop (Tieleman and Hinton, 2012)

Parameters: $\alpha, \lambda > 0$

$$\mathrm{MS}_{n+1} = \alpha \, \mathrm{MS}_n + (1-\alpha) g_{n+1} \odot g_{n+1}$$
$$P_{n+1} = \mathrm{diag}\left(\mathbb{1} \oslash \left(\lambda \mathbb{1} + \sqrt{\mathrm{MS}_{n+1}}\right)\right)$$
$$\theta_{n+1} = \theta_n - \gamma_{n+1} P_{n+1} \cdot g_{n+1}$$

Algorithm Adadelta (Zeiler, 2012)

Parameters: $\beta_1, \beta_2, \lambda > 0$

$$\mathrm{MS}_{n+1} = \beta_1 \, \mathrm{MS}_n + (1-\beta_1) g_{n+1} \odot g_{n+1}$$
$$P_{n+1} = \mathrm{diag}\left((\lambda \mathbb{1} + \widehat{\mathrm{MS}}_n) \oslash \left(\lambda \mathbb{1} + \sqrt{\widehat{\mathrm{MS}}_n}\right)\right)$$
$$\theta_{n+1} = \theta_n - \gamma_{n+1} P_{n+1} \cdot g_{n+1}.$$
$$\widehat{\mathrm{MS}}_{n+1} = \beta_2 \, \mathrm{MS}_n + (1-\beta_2)(\theta_{n+1} - \theta_n) \odot (\theta_{n+1} - \theta_n).$$

Training very deep Neural Networks

- Very deep neural networks are crucial, in particular in image classification (He et al., 2016).
- However much more difficult to train: much more "non-linear", local traps, vanishing gradients (Dauphin et al., 2014).
- (Neelakantan et al., 2015): hints that noisy optimizers bring more improvements.

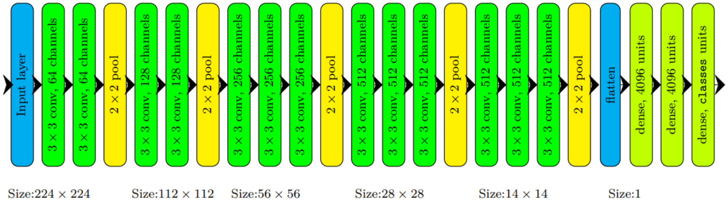


Figure: Architecture of the VGG-16 network for an input image of size 224×224.

We compare Preconditioned Langevin optimizers with their non-Langevin counterparts while increasing the depth of the networkon the MNIST, CIFAR-10 and CIFAR-100 datasets.

Pierre Bras. *Langevin algorithms for very deep Neural Networks with application to image classification*. *Procedia Computer Science*, 222:303 – 310, 2023.

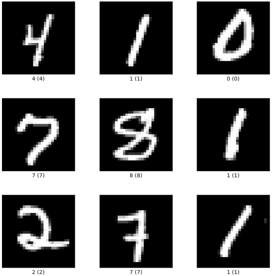


Figure: MNIST image dataset

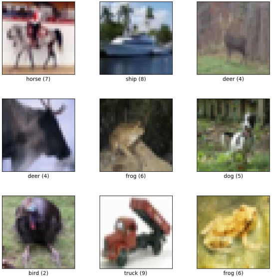


Figure: CIFAR-10 image dataset

Results for dense (fully connected) networks

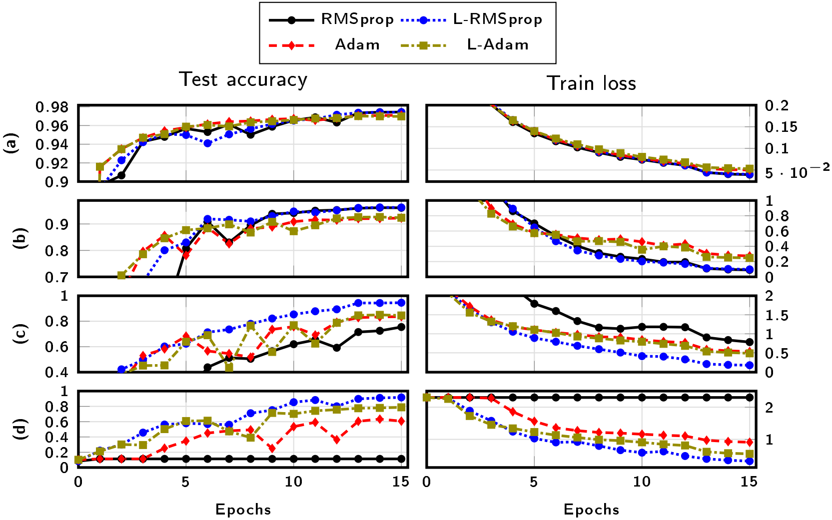


Figure: Training of neural networks of various depths on the MNIST dataset using Langevin algorithms compared with their non-langevin counterparts. (a): 3 hidden layers, (b): 20 hidden layers, (c): 30 hidden layers, (d): 40 hidden layers.

Layer Langevin Algorithm

Idea: The deepest layers of the network bear the most non-linearities $\implies$ are more subject to Langevin optimization

Layer Langevin Algorithm

$$\theta_{n+1}^{(i)} = \theta_n^{(i)} - \gamma_{n+1}[P_{n+1} \cdot g_{n+1}]^{(i)} + \mathbb{1}_{i \in \mathcal{J}} \sigma \sqrt{\gamma_{n+1}} \left[\mathcal{N}(0, P_{n+1})\right]^{(i)},$$

where $\mathcal{J}$: subset of weight indices; P_n: preconditioner.

We choose $\mathcal{J}$ to be the first k layers.

- Hypoelliptic Langevin diffusion (Hu and Spiliopoulos, 2017)

An example of Layer Langevin optimization

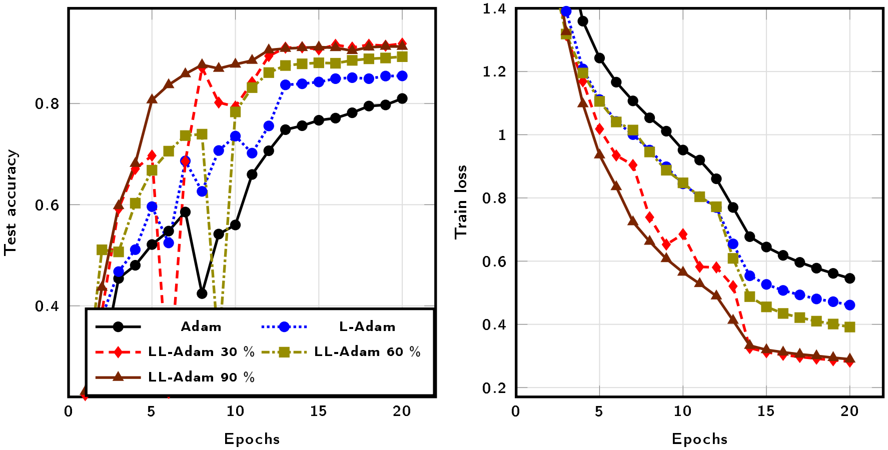


Figure: Layer Langevin comparison on a dense neural network with 30 hidden layers on the MNIST dataset.

Application to deep architectures for image classification

- Typical architecture in image recognition: Succession of convolutional layers with non-linearities (ReLU) (Simonyan and Zisserman, 2015)
- Residual connections: each layer behaves in part like the identity layer to pass the information through the successive layers (He et al., 2016; Huang et al., 2017).

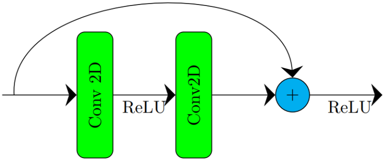
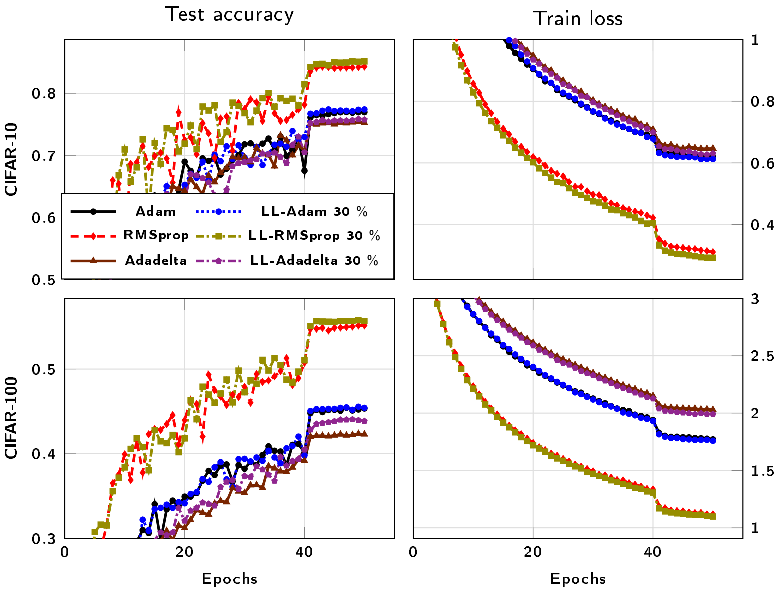


Figure: ResNet elementary block

Layer Langevin for training of ResNet-20



Outline

3. Adaptive Langevin algorithms for Neural Networks
   - Langevin versus non-Langevin for very deep learning
   - Langevin algorithms for Markovian Neural Networks and Deep Stochastic control

Stochastic control

$$\min_u J(u) := \mathbb{E}\left[\int_0^T G(Y_t)dt + F(Y_T)\right],$$
$$dY_t = b(Y_t, u_t)dt + \sigma(Y_t, u_t)dW_t,\ \ t \in [0, T],$$

G: path-dependent return, F: final return, u_t: control, Y_t: trajectory.

Discretization and numerical scheme

Euler-Maruyama scheme

$$\min_{\theta} \bar{J}(\bar{u}_\theta) := \mathbb{E}\Big[\sum_{k=0}^{N-1}(t_{k+1}-t_k)G(\bar{Y}^\theta_{t_{k+1}}) + F(\bar{Y}^\theta_{t_N})\Big],$$

$$\bar{Y}^\theta_{t_{k+1}} = \bar{Y}^\theta_{t_k} + (t_{k+1}-t_k)b\big(\bar{Y}^\theta_{t_k}, \bar{u}_{k,\theta}(\bar{Y}^\theta_{t_k})\big) + \sqrt{t_{k+1}-t_k}\,\sigma\big(\bar{Y}^\theta_{t_k}, \bar{u}_{k,\theta}(\bar{Y}^\theta_{t_k})\big)\xi_{k+1},$$

$\xi_k \sim \mathcal{N}(0, I_{d_2})$ i.i.d.

- **Time discretization** of $[0,T]$: $t_k := kT/N,\ k \in \{0,\dots,N\}, \quad h := T/N.$
- **Control** u with **parameter** θ using either one time-dependant neural network either N distinct neural networks: $u_{t_k} = \bar{u}_\theta(t_k, Y_{t_k})$ or $u_{t_k} = \bar{u}_{\theta^k}(Y_{t_k})$

Discretization and numerical scheme

Euler-Maruyama scheme

$$\min_\theta \bar{J}(\bar{u}_\theta) := \mathbb{E}\Big[\sum_{k=0}^{N-1}(t_{k+1}-t_k)G(\bar{Y}^\theta_{t_{k+1}}) + F(\bar{Y}^\theta_{t_N})\Big],$$

$$\bar{Y}^\theta_{t_{k+1}} = \bar{Y}^\theta_{t_k} + (t_{k+1}-t_k)b(\bar{Y}^\theta_{t_k}, \bar{u}_{k,\theta}(\bar{Y}^\theta_{t_k})) + \sqrt{t_{k+1}-t_k}\sigma(\bar{Y}^\theta_{t_k}, \bar{u}_{k,\theta}(\bar{Y}^\theta_{t_k}))\xi_{k+1},$$

$\xi_k \sim \mathcal{N}(0, I_{d_2})$ i.i.d.

- **Time discretization** of $[0,T]$: $t_k := kT/N$, $k \in \{0,\dots,N\}$, $\quad h := T/N$.
- **Control** u with **parameter** θ using either one time-dependant neural network either N distinct neural networks: $u_{t_k} = \bar{u}_\theta(t_k, Y_{t_k}) \quad$ or $\quad u_{t_k} = \bar{u}_{\theta^k}(Y_{t_k})$
- We refer to (Gobet and Munos, 2005; Han and E, 2016).
- The gradient is computed by recursively tracking the dependency of along the trajectory (Giles and Glasserman, 2005; Giles, 2007).
- Pierre Bras and Gilles Pagès. *Langevin algorithms for Markovian Neural Networks and Deep Stochastic control.* *IJCNN23 Proceedings*, 2023.

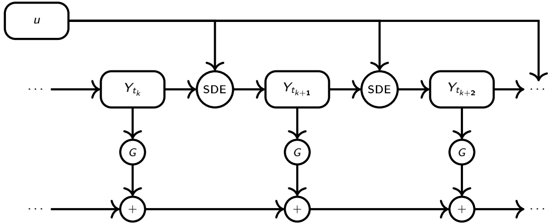


Figure: Markovian Neural Network with one control.

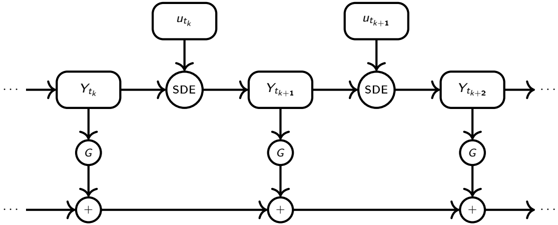


Figure: Markovian neural network with one control for every time step. Layer Langevin algorithms can be used in this case.

Fishing quotas, (Laurière, Pagès, and Pironneau, 2023)

Fish biomass $Y_t \in \mathbb{R}^{d_1}$ with:

- Inter-species interaction κY_t
- Fishing following imposed quotas u_t
- Objective: keep Y_t close to an ideal state $\mathcal{Y}_t$.

Figure: Source: Unsplash

$$dY_t = Y_t * ((r - u_t - \kappa Y_t)dt + \eta dW_t)$$

$$J(u) = \mathbb{E}\left[\int_0^T (|Y_t - \mathcal{Y}_t|^2 - \langle \alpha, u_t \rangle)dt + \beta[u]^{0,T}\right]$$

Deep Financial Hedging, (Buehler, Gonon, Teichmann, and Wood, 2019)

We aim to replicate some payoff Z defined on some portfolio S_t by trading some of the assets with transaction costs; the control u_t is the amount of held assets. The objective is

Figure: Source: Unsplash

$$J(u) = \nu\left(-Z + \underbrace{\sum_{k=0}^{N-1}\langle u_{t_k}, S_{t_{k+1}} - S_{t_k}\rangle}_{\text{trading}} - \underbrace{\sum_{k=0}^{N}\langle c_{tr}, S_{t_k} * |u_{t_k} - u_{t_{k-1}}|\rangle}_{\text{transaction costs}}\right)$$

where ν is a convex risk measure.

Model on (S_t)

$$J(u) = \nu \left(-Z + \sum_{k=0}^{N-1} \langle u_{t_k}, S_{t_{k+1}} - S_{t_k} \rangle - \sum_{k=0}^{N} \langle c_{tr}, S_{t_k} * |u_{t_k} - u_{t_{k-1}}| \rangle \right)$$

- We consider a multi-dimensional Heston model ($1 \le i \le d_1'$):

$$dS_t^{1,i} = \sqrt{V_t^i} S_t^{1,i} dB_t^i,$$
$$dV_t^i = a^i(b^i - V_t^i)dt + \eta^i \sqrt{V_t^i} dW_t^i, \quad \langle W^i, B^i \rangle_t = \rho t.$$

- V is not tradable directly but through swap options:

$$S_t^{2,i} := \mathbb{E}\left[\int_0^T V_s^i ds \Big| \mathcal{F}_t \right] = \int_0^t V_s^i ds + L^i(t, V_t^i),$$
$$L^i(t, v) := \frac{v - b^i}{a^i} \left(1 - e^{a^i(T-t)} \right) + b^i(T - t).$$

- Call payoff:

$$Z = \sum_{i=1}^{d_1'} \left(S_T^{1,i} - K^i \right)_+$$

Results for Deep Hedging

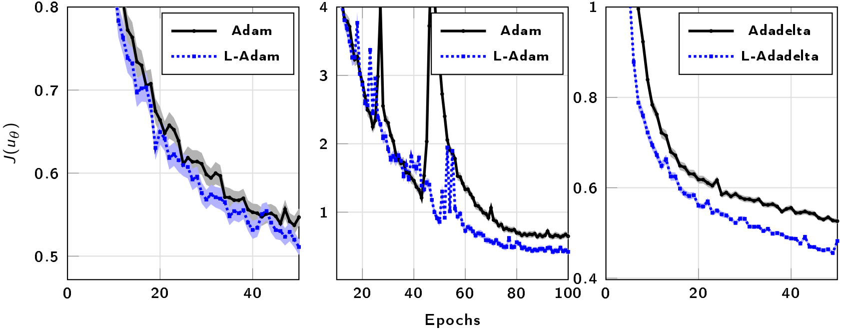


Figure: Comparison of algorithms with $N = 30, 50, 50$ respectively

Table: Best performance

	Adam, $N = 30$	Adam, $N = 50$	Adadelta, $N = 50$
Vanilla	0.4448	0.6355	0.4671
Langevin	0.4306	0.4182	0.3773

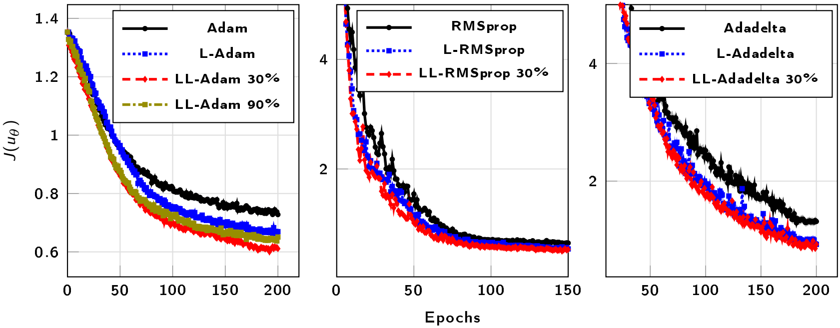


Figure: Training of the deep hedging problem with multiple controls with $N = 10$

Table: Best performance

	Adam	RMSprop	Adadelta
Vanilla	0.7278	0.5618	1.2900
Langevin	0.6626	0.4441	0.9250
Layer Langevin 30%	0.6004	0.4102	0.8554
Layer Langevin 90%	0.6377	–	–

Outline

4 Conclusion and perspectives

Conclusion

- A wide range of problems can be tackled with optimization methods and gradient descent, while neural networks help to approximate the solution function.

Conclusion

- A wide range of problems can be tackled with optimization methods and gradient descent, while neural networks help to approximate the solution function.
- We prove the convergence of Langevin algorithms with multiplicative noise and give theoretical guarantees, whereas these algorithms has been used by practitioners without theory.

Conclusion

- A wide range of problems can be tackled with optimization methods and gradient descent, while neural networks help to approximate the solution function.
- We prove the convergence of Langevin algorithms with multiplicative noise and give theoretical guarantees, whereas these algorithms has been used by practitioners without theory.
- We give theoretical founding including degenerate minimum cases.

Conclusion

- A wide range of problems can be tackled with optimization methods and gradient descent, while neural networks help to approximate the solution function.
- We prove the convergence of Langevin algorithms with multiplicative noise and give theoretical guarantees, whereas these algorithms has been used by practitioners without theory.
- We give theoretical founding including degenerate minimum cases.
- We prove the interest of Langevin or Layer Langevin algorithms for various problems, involving very deep learning.

Thank you for your attention !

Citations I

C. Anirudh Bhardwaj. Adaptively Preconditioned Stochastic Gradient Langevin Dynamics. *arXiv e-prints*, art. arXiv:1906.04324, June 2019.

K. B. Athreya and C.-R. Hwang. Gibbs measures asymptotics. *Sankhya A*, 72(1):191–207, 2010. ISSN 0976-836X. doi: 10.1007/s13171-010-0006-5. URL https://doi.org/10.1007/s13171-010-0006-5.

H. Buehler, L. Gonon, J. Teichmann, and B. Wood. Deep hedging. *Quant. Finance*, 19(8):1271–1291, 2019. ISSN 1469-7688. doi: 10.1080/14697688.2019.1571683. URL https://doi.org/10.1080/14697688.2019.1571683.

T.-S. Chiang, C.-R. Hwang, and S. J. Sheu. Diffusion for global optimization in R^n. *SIAM J. Control Optim.*, 25(3):737–753, 1987. ISSN 0363-0129. doi: 10.1137/0325042. URL https://doi.org/10.1137/0325042.

G. Cybenko. Approximation by superpositions of a sigmoidal function. *Math. Control Signals Systems*, 2 (4):303–314, 1989. ISSN 0932-4194. doi: 10.1007/BF02551274. URL https://doi.org/10.1007/BF02551274.

Y. N. Dauphin, R. Pascanu, C. Gulcehre, K. Cho, S. Ganguli, and Y. Bengio. Identifying and Attacking the Saddle Point Problem in High-Dimensional Non-Convex Optimization. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, page 2933–2941, Cambridge, MA, USA, 2014. MIT Press.

A. Eberle. Reflection couplings and contraction rates for diffusions. *Probab. Theory Related Fields*, 166 (3-4):851–886, 2016. ISSN 0178-8051. doi: 10.1007/s00440-015-0673-1. URL https://doi.org/10.1007/s00440-015-0673-1.

S. B. Gelfand and S. K. Mitter. Recursive stochastic algorithms for global optimization in R^d. *SIAM J. Control Optim.*, 29(5):999–1018, 1991. ISSN 0363-0129. doi: 10.1137/0329055. URL https://doi.org/10.1137/0329055.

M. B. Giles. Monte Carlo evaluation of sensitivities in computational finance. Technical Report NA07/12, Oxford University Computing Laboratory, 2007.

Citations II

M. B. Giles and P. Glasserman. Smoking adjoints: fast evaluation of Greeks in Monte Carlo calculations. Technical Report NA05/15, Oxford University Computing Laboratory, 2005.

E. Gobet and R. Munos. Sensitivity analysis using Itô-Malliavin calculus and martingales, and application to stochastic optimal control. *SIAM J. Control Optim.*, 43(5):1676–1713, 2005. ISSN 0363-0129. doi: 10.1137/S0363012902419059. URL https://doi.org/10.1137/S0363012902419059.

C. Gulcehre, M. Moczulski, M. Denil, and Y. Bengio. Noisy activation functions. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, page 3059–3068. JMLR.org, 2016.

J. Han and W. E. Deep Learning Approximation for Stochastic Control Problems. *Deep Reinforcement Learning Workshop, NIPS (2016)*, Nov. 2016.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. doi: 10.1109/CVPR.2016.90.

W. Hu and K. Spiliopoulos. Hypoelliptic multiscale Langevin diffusions: large deviations, invariant measures and small mass asymptotics. *Electronic Journal of Probability*, 22(none):1 – 38, 2017. doi: 10.1214/17-EJP72. URL https://doi.org/10.1214/17-EJP72.

G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger. Densely connected convolutional networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2261–2269, 2017. doi: 10.1109/CVPR.2017.243.

C.-R. Hwang. Laplace's method revisited: weak convergence of probability measures. *Ann. Probab.*, 8 (6):1177–1182, 1980. ISSN 0091-1798. URL http://links.jstor.org/sici?sici=0091-1798(198012)8:6<1177:LMRWCO>2.0.CO;2-1&origin=MSN.

C. R. Hwang. A generalization of Laplace's method. *Proc. Amer. Math. Soc.*, 82(3):446–451, 1981. ISSN 0002-9939. doi: 10.2307/2043958. URL https://doi.org/10.2307/2043958.

P. Jorion. *Value at Risk: A New Benchmark for Measuring Derivative Risk*. Irwin Professional Publishing, 1996.

Citations III

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In Y. Bengio and Y. LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

D. Lamberton and G. Pagès. Recursive computation of the invariant distribution of a diffusion. *Bernoulli*, 8(3):367–405, 2002. ISSN 1350-7265. doi: 10.1142/S0219493703000838. URL https://doi.org/10.1142/S0219493703000838.

D. Lamberton and G. Pagès. Recursive computation of the invariant distribution of a diffusion: the case of a weakly mean reverting drift. *Stoch. Dyn.*, 3(4):435–451, 2003. ISSN 0219-4937. doi: 10.1142/S0219493703000838. URL https://doi.org/10.1142/S0219493703000838.

M. Laurière, G. Pagès, and O. Pironneau. Performance of a Markovian Neural Network versus dynamic programming on a fishing control problem. *Probability, Uncertainty and Quantitative Risk*, pages –, 2023. ISSN 2095-9672. doi: 10.3934/puqr.2023006. URL /article/id/63c741a4b5351f4889aff727.

V. Lemaire. *Estimation récursive de la mesure invariante d'un processus de diffusion.* Theses, Université de Marne la Vallée, Dec. 2005. URL https://tel.archives-ouvertes.fr/tel-00011281.

C. Li, C. Chen, D. Carlson, and L. Carin. Preconditioned stochastic gradient langevin dynamics for deep neural networks. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, page 1788–1794. AAAI Press, 2016.

L. Miclo. Recuit simulé sur R^n. Étude de l'évolution de l'énergie libre. *Ann. Inst. H. Poincaré Probab. Statist.*, 28(2):235–266, 1992. ISSN 0246-0203. URL http://www.numdam.org/item?id=AIHPB_1992__28_2_235_0.

P. Monmarché, N. Fournier, and C. Tardif. Simulated annealing in $\mathbb{R}^d$ with slowly growing potentials. *Stochastic Process. Appl.*, 131:276–291, 2021. ISSN 0304-4149. doi: 10.1016/j.spa.2020.09.014. URL https://doi.org/10.1016/j.spa.2020.09.014.

A. Neelakantan, L. Vilnis, Q. V. Le, I. Sutskever, L. Kaiser, K. Kurach, and J. Martens. Adding Gradient Noise Improves Learning for Very Deep Networks. *arXiv e-prints*, art. arXiv:1511.06807, Nov. 2015.

Citations IV

G. Pagès and F. Panloup. Unadjusted Langevin algorithm with multiplicative noise: Total variation and Wasserstein bounds. *The Annals of Applied Probability*, 33(1):726 – 779, 2023. doi: 10.1214/22-AAP1828. URL https://doi.org/10.1214/22-AAP1828.

H. Robbins and S. Monro. A stochastic approximation method. *Ann. Math. Statistics*, 22:400–407, 1951. ISSN 0003-4851. doi: 10.1214/aoms/1177729586. URL https://doi.org/10.1214/aoms/1177729586.

H. Robbins and D. Siegmund. A convergence theorem for non negative almost supermartingales and some applications. In *Optimizing methods in statistics (Proc. Sympos., Ohio State Univ., Columbus, Ohio, 1971)*, pages 233–257. Academic Press, New York, 1971.

L. Sagun, L. Bottou, and Y. LeCun. Eigenvalues of the Hessian in Deep Learning: Singularity and Beyond. *arXiv e-prints*, art. arXiv:1611.07476, 2016.

K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In Y. Bengio and Y. LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

T. Tieleman and G. E. Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. Coursera: Neural Networks for Machine Learning, 2012.

S. Uryasev and R. T. Rockafellar. Conditional value-at-risk: optimization approach. In *Stochastic optimization: algorithms and applications (Gainesville, FL, 2000)*, volume 54 of *Appl. Optim.*, pages 411–435. Kluwer Acad. Publ., Dordrecht, 2001. doi: 10.1007/978-1-4757-6594-6_17. URL https://doi.org/10.1007/978-1-4757-6594-6_17.

F.-Y. Wang. Exponential contraction in Wasserstein distances for diffusion semigroups with negative curvature. *Potential Anal.*, 53(3):1123–1144, 2020. ISSN 0926-2601. doi: 10.1007/s11118-019-09800-z. URL https://doi.org/10.1007/s11118-019-09800-z.

M. Welling and Y. W. Teh. Bayesian Learning via Stochastic Gradient Langevin Dynamics. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, page 681–688. Omnipress, 2011. ISBN 9781450306195.

M. D. Zeiler. ADADELTA: An Adaptive Learning Rate Method. *arXiv e-prints*, art. arXiv:1212.5701, Dec. 2012.