Adaptive Gradient Langevin Algorithms for Stochastic Optimization and Bayesian Inference

Pierre Bras

Under the supervision of Gilles Pages

Laboratoire de Probabilités, Statistique et Modélisation
Sorbonne Université, Paris, France

PhD thesis publicly defended on September 11th, 2023

Outline

Outline

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

Optimization problem

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

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

Optimization problem

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

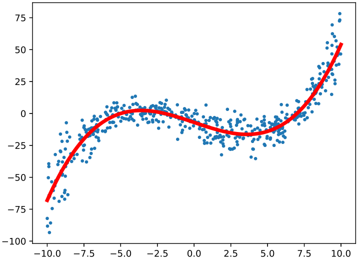
Objective: $\underset{x \in \mathbb{R}^d}{\text{Minimize}} \quad V(x)$.

Examples:

1. Determining optimal allocation of resources to minimize 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

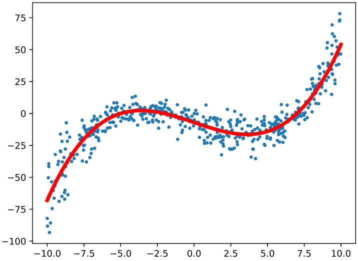
- Data $(u_i, v_i) \in \mathbb{R}^{d_{\text{in}}} \times \mathbb{R}^{d_{\text{out}}}$ (inputs and outputs) for $1 \leq i \leq N$ with $N \gg 1$.



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:

$$\text{Find } \Phi : \mathbb{R}^{d_{\text{in}}} \to \mathbb{R}^{d_{\text{out}}}$$
$$\text{such that: } \forall i,\ \Phi(u_i) \approx v_i.$$

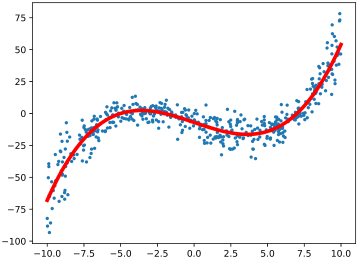


Example: Regression and Neural Networks

• 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:

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

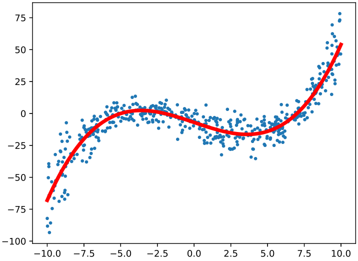
Example: Regression and Neural Networks

• 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:

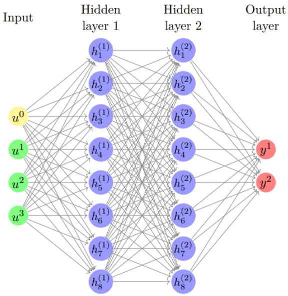
$$\Phi_{x_1,x_2}(u) = x_1 \cdot u + x_2, \quad x_1 \text{ matrix, } x_2 \text{ vector.}$$

• **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

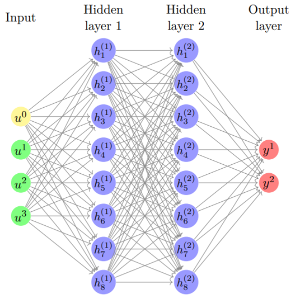
• **Neural Networks** are families
of parametrized functions that can approximate many functions in practice (Cybenko, 1989), (AlexNet 2012).

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

Input: $h_0 = u$,

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

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



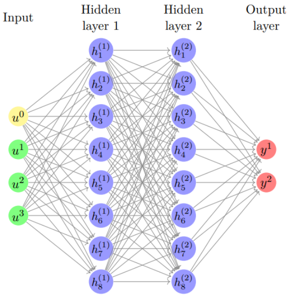
Example: Regression and Neural Networks

- **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**:

$$\text{Input: } h_0 = u,$$
$$h_k = \varphi(\alpha_k \cdot h_{k-1} + \beta_k) \in \mathbb{R}^{d_k},\ 1 \le k \le K-1,$$
$$\text{Output: } h_K = \alpha_K \cdot h_{K-1} + \beta_K =: \Phi_{(\alpha_k,\beta_k)_{0\le k\le K}}(u)$$

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.

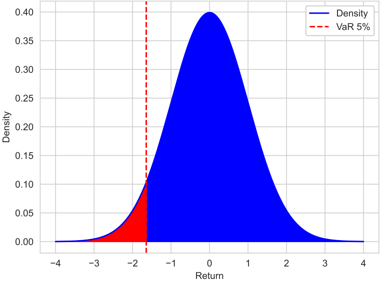


Example: Computation of quantiles and VaR

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

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

In finance, q_α is called **Value at Risk** and is widely used for risk management (Jorion, 1996).



Example: Computation of quantiles and VaR

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

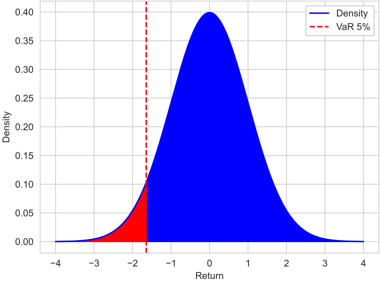
$$q_\alpha := \inf\{u \in \mathbb{R} : \ \mathbb{P}(Z \leq u) \geq \alpha\}.$$

In finance, q_α is called **Value at Risk** and is widely used for risk management (Jorion, 1996).

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

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

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

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

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

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

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

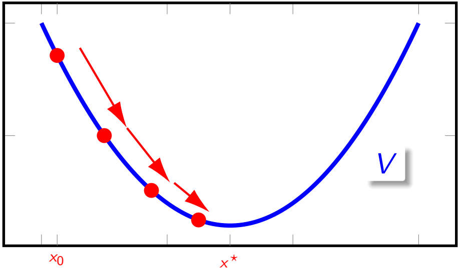
Gradient Descent Algorithm (GD)

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):

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



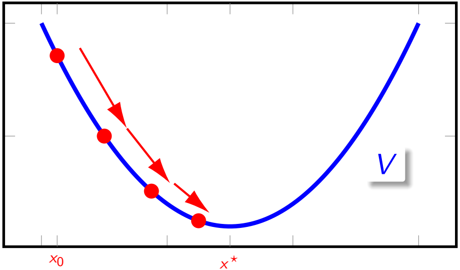
Gradient Descent Algorithm (GD)

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):

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



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

Example of Gradient Descent

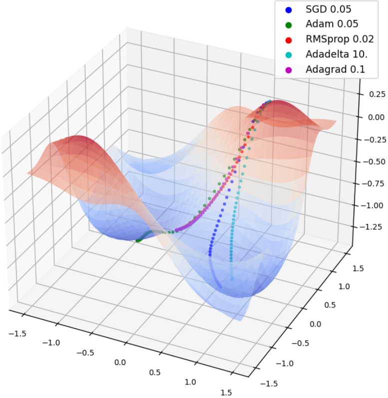


Figure: Example of different variants of gradient descent algorithms with $V(x, y) = -\sin(x^2)\cos(3y^2)e^{-x^2y^2} - e^{-(x+y)^2}$.

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 $N \gg 1$:

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

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 $N \gg 1$:

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

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 $N \gg 1$:

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

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):

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

$\Longrightarrow$ 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):

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

Stochastic Gradient Descent Algorithm, Mini-Batch version

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

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

$\Longrightarrow$ 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):

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

Stochastic Gradient Descent Algorithm, Mini-Batch version

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

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

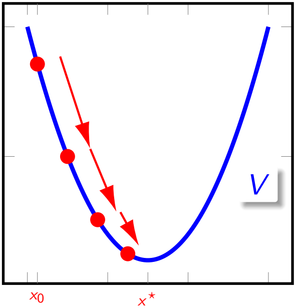
Introduced in (Robbins and Monro, 1951); Robbins-Siegmund Lemma of convergence (Robbins and Siegmund, 1971).

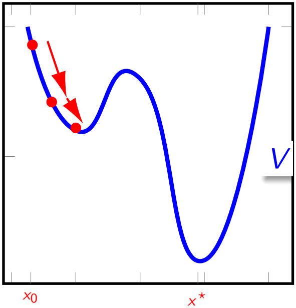
Outline

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

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 argmin(V):
Solve the Langevin equation $\implies$ approximation of ν_σ $\implies$ approximation of argmin(V).

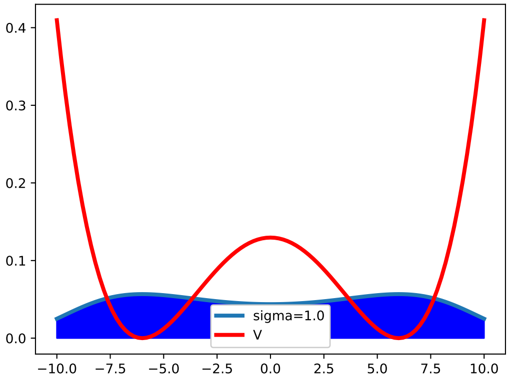


Figure: Concentration of Gibbs measure

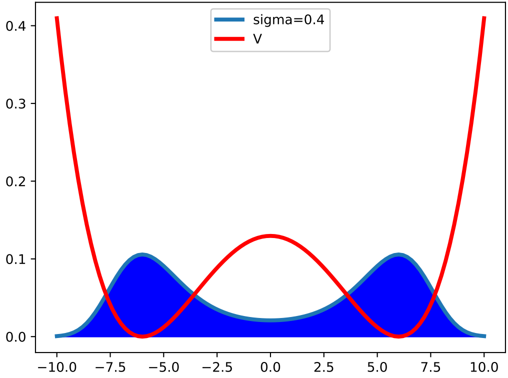


Figure: Concentration of Gibbs measure

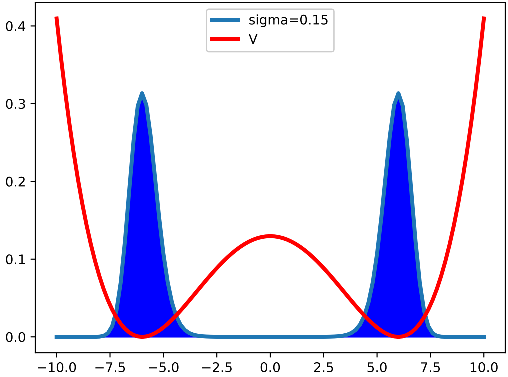


Figure: Concentration of Gibbs measure

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

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

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; Lemaire, 2005; Pagès and Panloup, 2023) 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 $u_1, \ldots, u_N$ 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, $p(x|u_1, \ldots, u_N)$, has density proportional to

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

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

- (Lamberton and Pagès, 2002, 2003; Lemaire, 2005; Pagès and Panloup, 2023) introduce and analyze sampling from a probability measure ν as invariant measure of $dX_t = b(X_t)dt + \sigma(X_t)dW_t$:

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

- (Durmus and Moulines, 2017, 2019; Brosse et al., 2019) focus on the Langevin equation.

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 $\mathrm{argmin}(V)$
- 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), (Monmarché, Fournier, and Tardif, 2021)

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), (Monmarché, Fournier, and Tardif, 2021)
- (Gelfand and Mitter, 1991): the convergence of the algorithm (x_n) .

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 and Contributions

Part I: Convergence of Langevin algorithms

- Convergence of the Langevin equation Y_t with multiplicative noise to $\text{argmin}(V)$ 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

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

$\implies$ Investigate the rate of $\mathcal{D}(\nu_a, \nu^\star)$ as $a \to 0$, including cases where $\nabla^2 V(x)$ is not positive definite for some $x \in \text{argmin}(V)$.

Part I: Convergence of Langevin algorithms

- Convergence of the Langevin equation Y_t with multiplicative noise to $\text{argmin}(V)$ 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

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

$\implies$ Investigate the rate of $\mathcal{D}(\nu_a, \nu^\star)$ as $a \to 0$, including cases where $\nabla^2 V(x)$ is not positive definite for some $x \in \text{argmin}(V)$.

Contributions:

1. Pierre Bras and Gilles Pagès. *Convergence of Langevin-Simulated Annealing algorithms with multiplicative noise*. *Mathematics of Computation*, 2023, and presented to the conference *MCM23*.
2. 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.
3. Pierre Bras. *Convergence rates of Gibbs measures with degenerate minimum*. *Bernoulli*, 28(4):2431 – 2458, 2022.

Part 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: image classification, deep stochastic control.
- (on-going) Introduce deep pathwise variance reduction for SDEs with reinforcement learning approaches.

Part II: Adaptive Langevin algorithms for deep neural networks

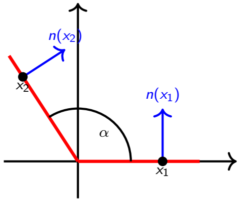
- Implement Langevin algorithms for different choices of σ (Adam, RMSprop, Adadelta etc) and compare with their corresponding non-Langevin counterpart.
- Investigate the benefits of Langevin algorithms on very deep learning: image classification, deep stochastic control.
- (on-going) Introduce deep pathwise variance reduction for SDEs with reinforcement learning approaches.

Contributions:

1. Pierre Bras. *Langevin algorithms for very deep Neural Networks with application to image classification*. *Procedia Computer Science*, 222:303 – 310, 2023, and presented at a workshop of IJCNN23.
2. Pierre Bras and Gilles Pagès. *Langevin algorithms for Markovian Neural Networks and Deep Stochastic control*. *IJCNN23 Proceedings*, 2023.
3. Pierre Bras and Gilles Pagès. *Policy Gradient Optimal Correlation Search for Variance Reduction in Monte Carlo simulation and Maximum Optimal Transport*. *arXiv e-prints*, page arXiv:2307.12703, 2023.

Part III: Numerical Simulation of Stochastic processes

• For Part I, we need bounds for d_{TV} between a stochastic process and its Euler scheme **in small time**. We use a multi-level Richardson-Romberg method.



• We study weak error rates of numerical schemes for (regular) stochastic Volterra equations, having in mind the rough case:

$$X_t = X_0 + \int_0^t K_1(t,s)b(X_s)ds + \int_0^t K_2(t,s)\sigma(X_s)dW_s.$$

• We study the Brownian motion in $\mathbb{R}^2$ which is stopped or reflected on a wedge; we give density formulas as well as simulation algorithms.

Contributions:

1. 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.
2. Pierre Bras and Masaaki Fukasawa. *Weak error rates for numerical schemes of non-singular Stochastic Volterra equations with application to option pricing under path-dependent volatility*. *In review*.
3. Pierre Bras and Arturo Kohatsu-Higa. *Simulation of reflected Brownian motion on two dimensional wedges. Stochastic Process. Appl.*, 156:349–378, 2023.

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

Objectives and assumptions

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

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

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

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}:

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

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

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}:

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

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

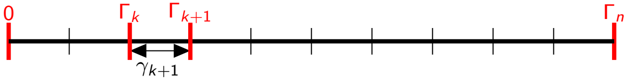
Important assumptions (Pagès and Panloup, 2023):

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

- *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:

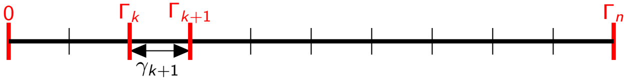
$$\begin{aligned}
\mathcal{W}_1(Y_{\Gamma_n}, X_{\Gamma_n}) &\leq |\mathbb{E}f(Y_{\Gamma_n}) - \mathbb{E}f(X_{\Gamma_n})| \\
&= |P^2_{\gamma_1} \circ \cdots \circ P^2_{\gamma_n} f(x) - P^2_{\Gamma_n} f(x)| \\
&= \left| \sum_{k=1}^{n} P^2_{\gamma_1} \circ \cdots \circ P^2_{\gamma_{k-1}} \circ (P^2_{\gamma_k}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



Domino strategy

- *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:

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



In the sum we bound two types of terms:

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

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

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

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:

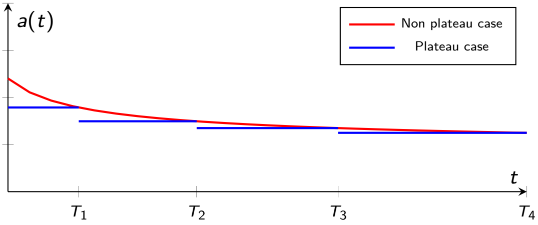
$$dX_t = -\sigma\sigma^\top \nabla V(X_t)dt + a_{n+1}\sigma(X_t)dW_t + a_{n+1}^2 \Upsilon(X_t)dt, \quad t \in [T_n, T_{n+1}),$$
$$a_n = A \log^{-1/2}(T_n),$$

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 $\mathcal{W}_1(X_{T_n}, Y_{T_n})$:

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


a(t)
Non plateau case
Plateau case
t
T1
T2
T3
T4

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).

$\Longrightarrow$ 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:

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

- 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:

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

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

For $\quad dX_t = b_1(X_t)dt + \sigma_1(X_t)dW_t, \quad dY_t = b_2(Y_t)dt + \sigma_2(Y_t)dW_t,$
$\quad X_0 = Y_0, \quad \sigma_1(X_0) = \sigma_2(Y_0),$

then

$$\mathrm{d}_{\mathrm{TV}}(X_t, Y_t) \le Ct^{1/2}e^{c\sqrt{\log(1/t)}}.$$

These results uses:

Theorem

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

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

These results uses:

Theorem

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

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

We use a **weighted multi-level Richardson-Romberg extrapolation** (Giles, 2008; Lemaire and Pagès, 2017):

- $f_\varepsilon(x) := \mathbb{E}[f(x + \varepsilon^{1/2}\zeta)]$ is Lipschitz, $\quad \zeta \sim \mathcal{N}(0, I_d)$,
- Taylor expansion of $y \mapsto \mathbb{E}[f(Z_1 + y)]$,
- $$|\mathbb{E}f(Z_1) - \mathbb{E}f(Z_2)| \le \left| \mathbb{E}f(Z_1) - \sum_{i=1}^r w_i \mathbb{E}f_{\varepsilon/n_i}(Z_1) \right| + \left| \sum_{i=1}^r w_i \mathbb{E}f_{\varepsilon/n_i}(Z_1) - \sum_{i=1}^r w_i \mathbb{E}f_{\varepsilon/n_i}(Z_2) \right| + \left| \sum_{i=1}^r w_i \mathbb{E}f_{\varepsilon/n_i}(Z_2) - \mathbb{E}f(Z_2) \right|,$$
- We choose w_i and n_i to cancel all the terms up to $i = r - 1$.

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 $\mathrm{argmin}(V)$ is finite and $\nabla^2 V(x_i^\star) > 0$ for all $x_i^\star \in \mathrm{argmin}(V)$ (Hwang, 1980, 1981). Then

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

- 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 $\text{argmin}(V)$ is finite and $\nabla^2 V(x_i^\star) > 0$ for all $x_i^\star \in \text{argmin}(V)$ (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 $\text{argmin}(V)$ is finite with **degenerate minimum**.
- This happens in practice when training over-parametrized neural networks (Sagun, Bottou, and LeCun, 2016):

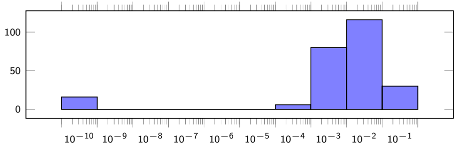


Figure: Distribution of the eigenvalues of the Hessian matrix at the end of training of a neural network on the MNIST dataset.

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

Theorem

Assume that $\text{argmin}(V) = \{x^\star\}$. 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 $X \sim \nu_a$:

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

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

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

(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
   - Policy Gradient Optimal Correlation Search for Variance Reduction in Monte Carlo simulation and Maximum Optimal Transport

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
- Policy Gradient Optimal Correlation Search for Variance Reduction in Monte Carlo simulation and Maximum Optimal Transport

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 $\sigma > 0$:

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

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.

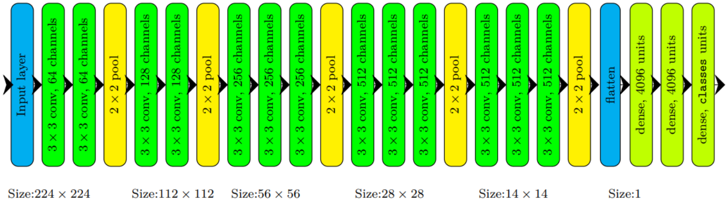


Figure: Architecture of the VGG-16 network for an input image of size 224 × 224.

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.

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.

We implement the package `langevin_optimizers` for ready-to-use `TensorFlow` optimizers using the method `_resource_apply_dense` from the base class `tf.keras.optimizers.Optimizer`.

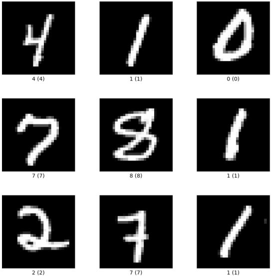


Figure: MNIST image dataset

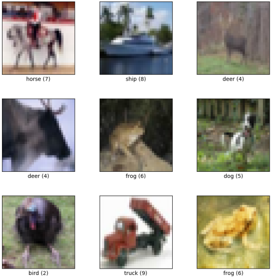


Figure: CIFAR-10 image dataset

Results for dense (fully connected) networks

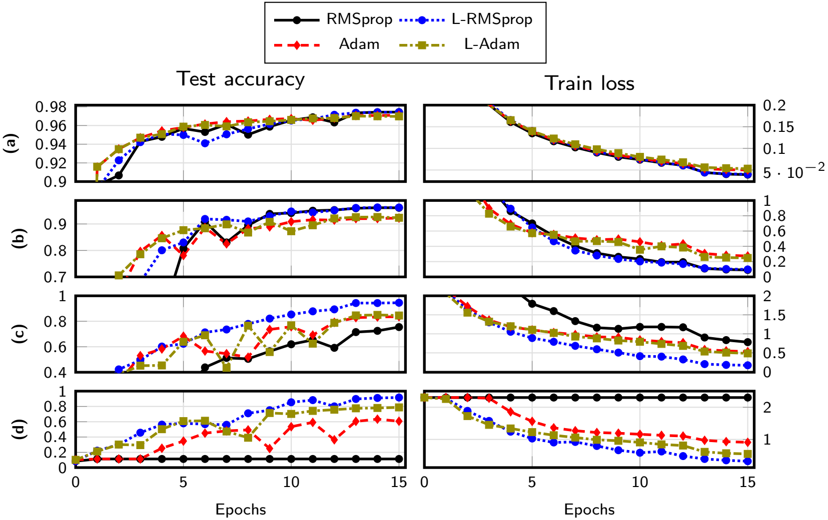


Figure: Training of neural networks of various depths on the MNIST dataset using Langevin algorithms compared with their non-langevin counterparts. (a): 3 hidden layers, (b): 20 hidden layers, (c): 30 hidden layers, (d): 40 hidden layers.

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.

An example of Layer Langevin optimization

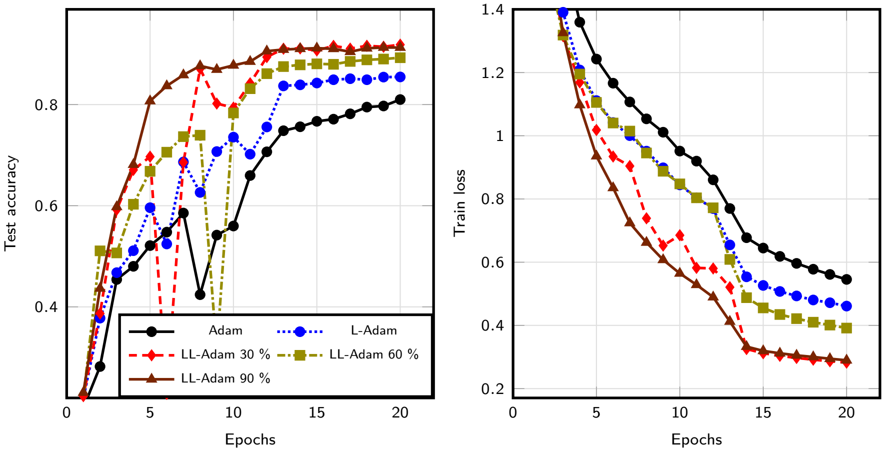


Figure: Layer Langevin comparison on a dense neural network with 30 hidden layers on the MNIST dataset.

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).

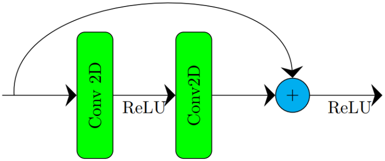
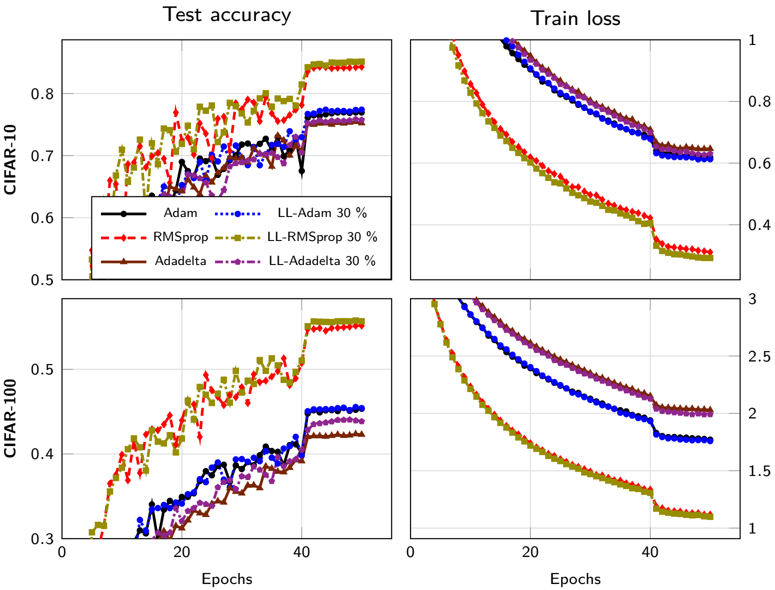


Figure: ResNet elementary block

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
- Policy Gradient Optimal Correlation Search for Variance Reduction in Monte Carlo simulation and Maximum Optimal Transport

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

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

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

$\xi_k \sim \mathcal{N}(0, I_{d_2})$ 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 $u_{t_k} = \bar{u}_{\theta^k}(Y_{t_k})$
- 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).

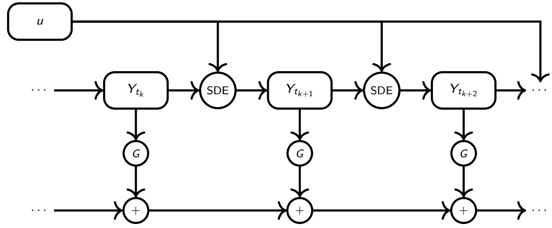


Figure: Markovian Neural Network with one control.

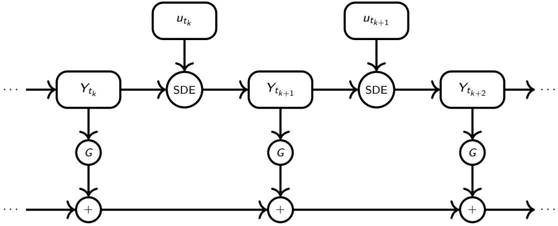


Figure: Markovian neural network with one control for every time step. Layer Langevin algorithms can be used in this case.

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

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

where ν is a convex risk measure. We consider the assets S_t to be follow a Heston model and are tradable along with variance swap options.

Results for Deep Hedging

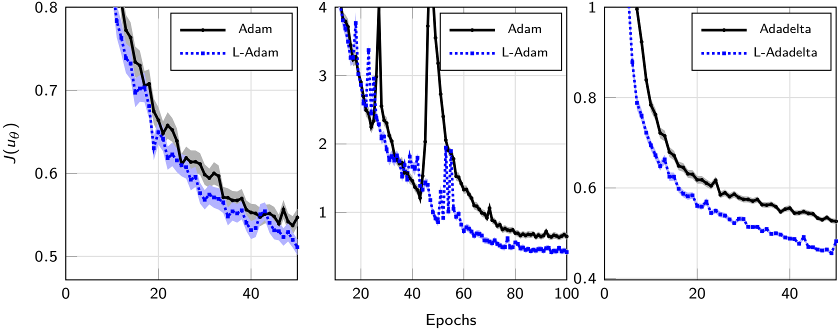


Figure: Comparison of algorithms with $N = 30, 50, 50$ respectively

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

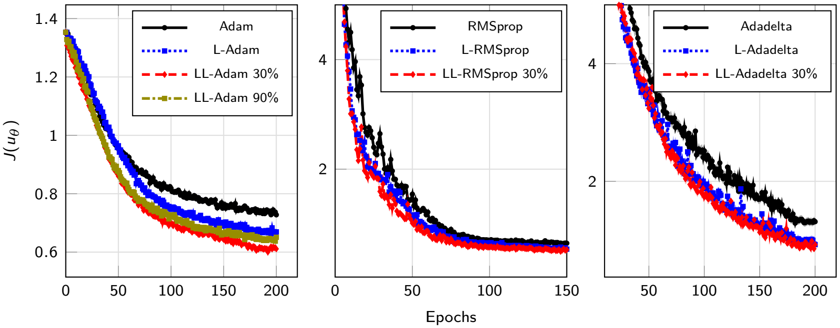


Figure: Training of the deep hedging problem with multiple controls with $N = 10$

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

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
- Policy Gradient Optimal Correlation Search for Variance Reduction in Monte Carlo simulation and Maximum Optimal Transport

For the process:

$$Y_0 \in \mathbb{R}^d, \quad dY_t = b(Y_t)dt + \sigma(Y_t)dW_t,$$

and $f : \mathbb{R}^d \to \mathbb{R}$, we are looking for an **estimator of** $\mathbb{E}[f(Y_T)]$ with reduced variance.
We propose the **unbiased** estimator:

For the process:

$$Y_0 \in \mathbb{R}^d, \quad dY_t = b(Y_t)dt + \sigma(Y_t)dW_t,$$

and $f : \mathbb{R}^d \to \mathbb{R}$, we are looking for an **estimator of** $\mathbb{E}[f(Y_T)]$ with reduced variance.
We propose the **unbiased** estimator:

Correlation estimator

$$\text{Estimator: } \left(f(Y_T^1) + f(Y_T^2)\right)/2,$$
$$dY_t^1 = b(Y_t^1)dt + \sigma(Y_t^1)dW_t^1,$$
$$dY_t^2 = b(Y_t^2)dt + \sigma(Y_t^2)dW_t^2,$$
$$d\langle W_t^1, W_t^2\rangle = \rho_t dt.$$

For the process:

$$Y_0 \in \mathbb{R}^d, \quad dY_t = b(Y_t)dt + \sigma(Y_t)dW_t,$$

and $f : \mathbb{R}^d \to \mathbb{R}$, we are looking for an **estimator of** $\mathbb{E}[f(Y_T)]$ with reduced variance.
We propose the **unbiased** estimator:

Correlation estimator

$$\text{Estimator: } \left(f(Y_T^1) + f(Y_T^2)\right)/2,$$
$$dY_t^1 = b(Y_t^1)dt + \sigma(Y_t^1)dW_t^1,$$
$$dY_t^2 = b(Y_t^2)dt + \sigma(Y_t^2)dW_t^2,$$
$$d\langle W_t^1, W_t^2\rangle = \rho_t dt.$$

- For $\rho_t \equiv -I_d$, this gives the **antithetic method** (Hammersley and Morton, 1956).

Optimal correlation search

- We look to minimize the variance of the estimator through the choice of $\rho_t = \rho(t, Y_t^1, Y_t^2)$ as a neural network.
- We write the correlation search as stochastic control:

$$\text{Minimize } \ \mathbb{E}\big[f(Y_T^1)f(Y_T^2)\big].$$

Optimal correlation search

- We look to minimize the variance of the estimator through the choice of $\rho_t = \rho(t, Y_t^1, Y_t^2)$ as a neural network.
- We write the correlation search as stochastic control:

$$\text{Minimize} \quad \mathbb{E}\left[f(Y_T^1)f(Y_T^2)\right].$$

- We solve using gradient descent and **Reinforcement Learning** approaches.
- Package `relocor`: *REinforcement Learning Optimal CORrelation search*, using `OpenAI Gym`.

Simulations

- We test variance reduction for option pricing with multi-basket Black-Scholes and Heston models.
- We give examples where we achieve better than trivial correlations (antithetic, minus-plus etc).

Simulations

• We test variance reduction for option pricing with multi-basket Black-Scholes and Heston models.
• We give examples where we achieve better than trivial correlations (antithetic, minus-plus etc).

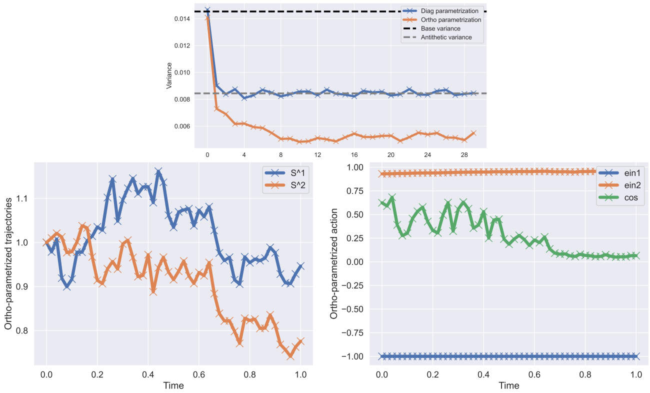


Figure: Black-Scholes variance reduction with $d = 2$. Left: trajectories of the assets; Right: coefficients of the diagonal matrix and cosine of the rotation matrix.

Link with maximal optimal transport

- We look to correlate two random variables Y_T^1 and Y_T^2 with fixed marginal law and the variance reduction can be written as:

$$\max_{\rho} \mathbb{E}\left|f(Y_T^1) - f(Y_T^2)\right|^2.$$

- Any coupling between Y_T^1 and Y_T^2 can be written as coupling W^1 and W^2 with ρ.
- $\Longrightarrow$ Our reduction variance is in fact a L^2-optimal **maximal** transport.

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.

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.
- Langevin algorithms have returned to the forefront of research: latent diffusion generative models (Rombach et al., 2021).

Figure: Théâtre d'Opéra Spatial

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.

N. Brosse, A. Durmus, É. Moulines, and S. Sabanis. The tamed unadjusted langevin algorithm. *Stochastic Processes and their Applications*, 129(10):3638–3663, 2019. ISSN 0304-4149. doi: https://doi.org/10.1016/j.spa.2018.10.002.

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 $\mathbf{R}^n$. *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 29332941, Cambridge, MA, USA, 2014. MIT Press.

A. Durmus and É. Moulines. Nonasymptotic convergence analysis for the unadjusted Langevin algorithm. *The Annals of Applied Probability*, 27(3):1551 – 1587, 2017. doi: 10.1214/16-AAP1238. URL https://doi.org/10.1214/16-AAP1238.

A. Durmus and E. Moulines. High-dimensional Bayesian inference via the unadjusted Langevin algorithm. *Bernoulli*, 25(4A):2854–2882, 2019. ISSN 1350-7265. doi: 10.3150/18-BEJ1073. URL https://doi.org/10.3150/18-BEJ1073.

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.

Citations II

S. B. Gelfand and S. K. Mitter. Recursive stochastic algorithms for global optimization in $\mathbf{R}^d$. *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.

M. B. Giles. Multilevel Monte Carlo path simulation. *Oper. Res.*, 56(3):607–617, 2008. ISSN 0030-364X. doi: 10.1287/opre.1070.0496. URL https://doi.org/10.1287/opre.1070.0496.

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 30593068. JMLR.org, 2016.

J. M. Hammersley and K. W. Morton. A new monte carlo technique: antithetic variates. *Mathematical Proceedings of the Cambridge Philosophical Society*, 52(3):449475, 1956. doi: 10.1017/S0305004100031455.

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.

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.

Citations III

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.

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.

V. Lemaire and G. Pagès. Multilevel Richardson-Romberg extrapolation. *Bernoulli*, 23(4A):2643–2692, 2017. ISSN 1350-7265. doi: 10.3150/16-BEJ822. URL https://doi.org/10.3150/16-BEJ822.

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 17881794. AAAI Press, 2016.

Citations IV

L. Miclo. Recuit simulé sur $\mathbf{R}^n$. 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.

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.

R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-Resolution Image Synthesis with Latent Diffusion Models. *arXiv e-prints*, art. arXiv:2112.10752, Dec. 2021. doi: 10.48550/arXiv.2112.10752.

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.

Citations V

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 681688. Omnipress, 2011. ISBN 9781450306195.

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