Optimization by Gradient Descent

Mike PEREIRA[1]

[1]Geostatistics team, MINES ParisTech, PSL Research University

Machine Learning Group 17-18
Session 2
January 24th, 2018

Outline

Context
Optimization and supervised ML
Gradient descent
Convergence rates
Generalization risk optimization
Random walks and SGD
Annex

1 Context

2 Optimization and supervised ML

3 Gradient descent

4 Convergence rates

5 Generalization risk optimization

6 Random walks and SGD

7 Annex

Supervised Machine Learning

Context
Optimization and supervised ML
Gradient descent
Convergence rates
Generalization risk optimization
Random walks and SGD
Annex

- Supervised ML : predict the value of a partially observed variable using an explanatory variable
- Probabilistic model for supervised ML:
  - Input variable (= explanatory variable) : $X \in \mathcal{X}$
  - Output variable (= variable to predict) : $Y \in \mathcal{Y} \subset \mathbb{R}$
    $\rightarrow$ Observations = realizations of these variables
  - *Unknown* joint probability distribution :
    $$\rho : (x, y) \mapsto \mathbb{P}(X = x, Y = y)$$
- Goal : Find a function $\theta : \mathcal{X} \rightarrow \mathbb{R}$ such that $\theta(X)$ is a good predictor for Y
- Tool : Loss function $L : \mathcal{Y} \times \mathbb{R} \rightarrow \mathbb{R}_+$ such that $L(y, \theta(x))$ is the "loss"/"cost" associated with estimating y using $\theta(x)$

Regression

Context
Optimization and supervised ML
Gradient descent
Convergence rates
Generalization risk optimization
Random walks and SGD
Annex

θ is a linear function. Define

- a vector of parameters : $\boldsymbol{\theta} = (\theta_1, ..., \theta^d)^T \in \mathbb{R}^d$
- a transformation Φ of the input variable(s) :

$$X \in \mathcal{X} \mapsto \Phi(X) = (\Phi_1(X), ..., \Phi_d(X))^T \in \mathbb{R}^d$$

Then

$$\theta(X) = \langle \boldsymbol{\theta}, \Phi(X) \rangle = \sum_{i=1}^{d} \theta_i \Phi_i(X)$$

2 applications

- Multivariate regression : Quadratic loss function

$$L(y, \theta(x)) = (y - \theta(x))^2$$

- Logistic regression (classifier) : Sigmoid loss function

$$L(y, \theta(x)) = \frac{1}{1 + \exp(-y\theta(x))}$$

Outline

Context
Optimization and supervised ML
Gradient descent
Convergence rates
Generalization risk optimization
Random walks and SGD
Annex

1 Context

2 Optimization and supervised ML

3 Gradient descent

4 Convergence rates

5 Generalization risk optimization

6 Random walks and SGD

7 Annex

Risks in supervised ML I

Context
Optimization and supervised ML
Gradient descent
Convergence rates
Generalization risk optimization
Random walks and SGD
Annex

- The "quality" of $\theta(X)$ as a predictor of Y can be measured by the **Generalization risk** (or **true risk**):

$$\mathcal{R}(\theta) := \mathbb{E}_\rho[L(Y, \theta(X))] = \int_{\mathcal{X}\times\mathcal{Y}} L(y, \theta(x))\rho(x, y)dxdy$$

i.e. the "mean" loss from predicting an output y with an observed input $\theta(x)$

$\rightarrow$ The smaller, the better!

$\Rightarrow$ Find θ that minimizes the risk

Problem : ρ is unknown $\Rightarrow$ $\mathcal{R}$ is unknown... You can't minimize a function that you don't know...

Risks in supervised ML II

Context
Optimization and supervised ML
Gradient descent
Convergence rates
Generalization risk optimization
Random walks and SGD
Annex

- In supervised ML setting, we have a set of n observations:

$$(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}, i \in [\![1, n]\!]$$

Working hypothesis: the observations are i.i.d. samples from ρ

- The Generalization risk is in practise estimated by the **Empirical risk**

$$\widehat{\mathcal{R}}(\theta) := \frac{1}{n} \sum_{i=1}^{n} L(y_i, \theta(x_i))$$

- A Regularisation function $\Omega(\theta)$ must be introduced to avoid over-fitting (due to the fact that the Generalized risk is approximated)

A classical optimization problem

Supervised ML searches the best prediction law θ^* "risk-wise":

$$\theta^* = \underset{\theta}{\operatorname{argmin}} \left[\widehat{\mathcal{R}}(\theta) + \mu\Omega(\theta) \right] = \underset{\theta}{\operatorname{argmin}} \left[\frac{1}{n} \sum_{i=1}^{n} L(y_i, \theta(x_i)) + \mu\Omega(\theta) \right]$$

Outline

Context
Optimization and supervised ML
Gradient descent
Convergence rates
Generalization risk optimization
Random walks and SGD
Annex

1 Context

2 Optimization and supervised ML

3 Gradient descent

4 Convergence rates

5 Generalization risk optimization

6 Random walks and SGD

7 Annex

Gradient descent

Context
Optimization and supervised ML
Gradient descent
Convergence rates
Generalization risk optimization
Random walks and SGD
Annex

- Widely used first order optimization algorithm
- Based on the computation of the gradient of the function

Algorithm 1: Vanilla Gradient Descent

Number of iterations N_{iter}, initial state $\boldsymbol{\theta}^{(0)} \in \mathbb{R}^d$

for $k = 1$ **to** N_{iter} **do**

$$g^{(k-1)} = \nabla \widehat{\mathcal{R}}\left(\boldsymbol{\theta}^{(k-1)}\right) = \frac{1}{n}\sum_{i=1}^{n} \nabla L(y_i, \Phi(x_i)^T \boldsymbol{\theta}^{(k-1)})$$

$$\boldsymbol{\theta}^{(k)} \leftarrow \boldsymbol{\theta}^{(k-1)} - \gamma_k \times g^{(k-1)}$$

- Cost of each iteration : $\mathcal{O}(nd)$ operations (d- derivatives for each sample) $\rightarrow$ expensive for large datasets
- Convergence to at least a local minimum is guaranteed

Stochastic Gradient Descent

Context
Optimization and supervised ML
Gradient descent
Convergence rates
Generalization risk optimization
Random walks and SGD
Annex

Idea : replace the costly computation of the gradient $\nabla \widehat{\mathcal{R}}\left(\boldsymbol{\theta}^{(k-1)}\right)$ by a noisy "proxy" $g^{(k-1)}$ that is cheaper to compute and is equal to the real gradient "in average".

$$\widehat{\mathcal{R}}(\boldsymbol{\theta}) = \frac{1}{n}\sum_{i=1}^{n} f_i(\boldsymbol{\theta}), \quad f_i : \theta L(y_i, \Phi(x_i)^T\boldsymbol{\theta})$$

Take

$$g^{(k-1)} := \nabla f_{i_k}(\boldsymbol{\theta}^{(k-1)}), i_k \sim \mathcal{U}\{1, ..., n\}$$

Algorithm 2: Stochastic Gradient Descent

Number of iterations N_{iter}, initial state $\boldsymbol{\theta}^{(0)} \in \mathbb{R}^d$

for $k = 1$ **to** N_{iter} **do**

  Draw an index $i_k \sim \mathcal{U}\{1, ..., n\}$

  $\boldsymbol{\theta}^{(k)} \leftarrow \boldsymbol{\theta}^{(k-1)} - \gamma_k \nabla f_{i_k}(\boldsymbol{\theta}^{(k-1)})$

Return $\theta^{N_{\text{iter}}}$

Mini-batch gradient descent

Context
Optimization and supervised ML
Gradient descent
Convergence rates
Generalization risk optimization
Random walks and SGD
Annex

- The best of both worlds

Algorithm 3: Stochastic Gradient Descent

Number of iterations N_{iter}, initial state $\theta^{(0)} \in \mathbb{R}^d$, batch size M

for $k = 1$ **to** N_{iter} **do**

Draw (w/o replacement) M indices $i_k^{(1)}, ..., i_k^{(M)} \sim \mathcal{U}\{1, ..., n\}$

$$g^{(k-1)} = \frac{1}{M} \sum_{m=1}^{M} \nabla f_{i_k^{(m)}}(\theta^{(k-1)})$$

$$\theta^{(k)} \leftarrow \theta^{(k-1)} - \gamma_k \times g^{(k-1)}$$

Return $\theta^{N_{\text{iter}}}$

- Usual batch size : $50 - 256$ samples but for deep learning : 16 samples (due to GPU computations)

Noise reduction

Context
Optimization and supervised ML
Gradient descent
Convergence rates
Generalization risk optimization
Random walks and SGD
Annex

- To reduce noise effect : take the average of all states $\theta^{(k)}$ as an estimator of the optimum, instead of just last one.
- "On line" computation trick of this average $\bar{\theta}^{(k)}$

Algorithm 4: Stochastic Gradient Descent (bis)

Number of iterations N_{iter}, initial state $\theta^{(0)} \in \mathbb{R}^d$

for $k = 1$ **to** N_{iter} **do**

Draw an index $i_k \sim \mathcal{U}\{1, ..., n\}$

$\theta^{(k)} \leftarrow \theta^{(k-1)} - \gamma_k \nabla f_{i_k}(\theta^{(k-1)})$

$\bar{\theta}^{(k)} \leftarrow \frac{1}{k}\left(\theta^{(k)} + (k-1)\bar{\theta}^{(k-1)}\right)$

Return $\bar{\theta}^{N_{\text{iter}}}$

Outline

Context
Optimization and supervised ML
Gradient descent
Convergence rates
Generalization risk optimization
Random walks and SGD
Annex

1 Context

2 Optimization and supervised ML

3 Gradient descent

4 Convergence rates

5 Generalization risk optimization

6 Random walks and SGD

7 Annex

Convergence I

Context
Optimization and supervised ML
Gradient descent
Convergence rates
Generalization risk optimization
Random walks and SGD
Annex

Assume the following properties for the loss function L :

- L is l-smooth, i.e. l is a lower bound of the eigenvalues of the Hessian matrix of L at any point
- L is μ-strongly convex, i.e. μ is a upper bound of the eigenvalues of the Hessian matrix of L at any point

Assume that the observations $\Phi(x_i)$ have bounded variance and invertible experimental covariance matrix

Importance of the learning rate γ_k

For smooth and convex problems, it was shown that almost surely, $\theta_k \to \theta^*$ if

$$\sum_{k=1}^{\infty} \gamma_k = \infty \quad \sum_{k=1}^{\infty} \gamma_k^2 < \infty$$

Convergence II

Context
Optimization and supervised ML
Gradient descent
Convergence rates
Generalization risk optimization
Random walks and SGD
Annex

Convergence rate = the speed of convergence of towards the solution of the problem, i.e., the minimum of $\widehat{\mathcal{R}}$

Step size	GD Constant	GD Decreasing	SGD Decreasing
Convex function	$\mathcal{O}(\frac{1}{k})$ $(\gamma_k = \frac{1}{L})$	$\mathcal{O}(\frac{1}{\log k})$ $(\gamma_k = \frac{1}{k})$	$\mathcal{O}(\frac{1}{\sqrt{k}})$ $(\gamma_k \propto \frac{1}{\sqrt{k}})$
Strongly convex function	$\mathcal{O}\left(e^{-\frac{4\mu}{L+\mu}k}\right)$ $(\gamma_k = \frac{1}{L+\mu})$	$\mathcal{O}\left(k^{-\frac{2L\mu}{L+\mu}}\right)$ $(\gamma_k = \frac{1}{k})$	$\mathcal{O}\left(\frac{1}{\mu k}\right)$ $(\gamma_k \propto \frac{1}{\mu k})$

Table: Comparison of convergence rates for several algorithms, using a quadratic loss function $L(y, \theta(x)) = (y - \theta^T \Phi(x))^2$

- GD can converge much faster than SGD.
- BUT, each iteration of GD is more costly (by a factor n) than an iteration of SGD

Stochastic averaged gradient descent

Context
Optimization and supervised ML
Gradient descent
Convergence rates
Generalization risk optimization
Random walks and SGD
Annex

- Variation of SGD in which at each iteration, the full gradient

$$g^{(k)} = \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(\theta^{(k_i)}), \quad k_i \in [\![0, k-1]\!]$$

  is updated through i_k-th term of the sum, where $i_k \sim \mathcal{U}\{1, ..., n\}$
- Same update cost as SGD, BUT $g^{(k)}$ has a smaller variance

Algorithm 5: Stochastic Averaged Gradient Descent

Number of iterations N_{iter}, initial state $\theta^{(0)} \in \mathbb{R}^d$

$g^{(0)} = \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(\theta^{(0)})$

for $k = 1$ **to** N_{iter} **do**

  Draw an index $i_k \sim \mathcal{U}\{1, ..., n\}$

  $g^{(k)} = \frac{1}{n}\left(n g^{(k-1)} - \nabla f_{i_k}(\theta^{(k_{i_k})}) + \nabla f_{i_k}(\theta^{(k-1)})\right)$

  $\theta^{(k)} \leftarrow \theta^{(k-1)} - \gamma_k g^{(k)}$

Return $\theta^{N_{\text{iter}}}$

- Convergence rate : $\mathcal{O}((1 - \frac{1}{8Ln})^k)$ (for step size $\gamma_k = \frac{1}{2nL}$)
- Required to store at all time n elementary gradients $\nabla f_i(\theta^{(k_i)})$

Outline

Context
Optimization and supervised ML
Gradient descent
Convergence rates
Generalization risk optimization
Random walks and SGD
Annex

1 Context

2 Optimization and supervised ML

3 Gradient descent

4 Convergence rates

5 Generalization risk optimization

6 Random walks and SGD

7 Annex

Generalization risk I

Context
Optimization and supervised ML
Gradient descent
Convergence rates
Generalization risk optimization
Random walks and SGD
Annex

Some cons of Empirical risk :

- To avoid over-fitting, a regularisation function must be set
- The number of iterations needed to achieve the optimum is hard to predict

$\Rightarrow$ Generalisation risk yields a more general approach given that it measures how accurately we may predict outcome values for previously unseen data (through the expectation).

$\Rightarrow$ SGD can be used to minimise the generalisation risk, even though its expression is unknown.

Idea : Considering that the examples $(x_i, y_i)_{1 \leq i \leq n}$ are i.i.d from ρ,

$$g^{(k)}(\theta^{(k-1)}) = \nabla L(y_k, \langle \boldsymbol{\theta}^{(k-1)}, \Phi(x_k) \rangle)$$

is a good proxy for

$$\begin{aligned} \nabla \mathcal{R}(\theta^{(k-1)}) &= \nabla \mathbb{E}_\rho[L(Y, \langle \boldsymbol{\theta}^{(k-1)}, \Phi(X) \rangle)] \\ &= \mathbb{E}_\rho[\nabla L(Y, \langle \boldsymbol{\theta}^{k-1}, \Phi(X) \rangle)] \end{aligned}$$

Generalization risk II

Context
Optimization and supervised ML
Gradient descent
Convergence rates
Generalization risk optimization
Random walks and SGD
Annex

Algorithm 6: Stochastic Gradient Descent for Generalisation risk

Number of iterations N_{iter}, initial state $\theta^{(0)} \in \mathbb{R}^d$

for $k = 1$ **to** N_{iter} **do**

$\theta^{(k)} \leftarrow \theta^{(k-1)} - \gamma_k \nabla L(y_k, \langle \theta^{(k-1)}, \Phi(x_k) \rangle)$

Return $\theta^{N_{\text{iter}}}$

- There is a **single** pass through the data : sometimes, due to the size of the problem, we can't do more than that
- Well fitted for online learning
- With a convex (resp. strongly convex) loss function and a constant step size, it has a convergence rate of $\mathcal{O}(1/\sqrt{n})$ (resp. $\mathcal{O}(1/(\mu n))$)
- Working hypothesis when minimizing Generalisation risk : you have enough data sample to accurately represent the distribution ρ, i.e. every possibility of values for the couple (X, Y)

Outline

Context
Optimization and supervised ML
Gradient descent
Convergence rates
Generalization risk optimization
Random walks and SGD
Annex

1 Context

2 Optimization and supervised ML

3 Gradient descent

4 Convergence rates

5 Generalization risk optimization

6 Random walks and SGD

7 Annex

A random walk interpretation I

Context
Optimization and supervised ML
Gradient descent
Convergence rates
Generalization risk optimization
Random walks and SGD
Annex

For general loss functions, SGD does not necessarily converge towards the optimum of the risk. Why is that?
Updates of SGD with constant step size γ can be written :

$$\theta_\gamma^{(k)} = \theta_\gamma^{(k-1)} - \gamma \underbrace{\left(\nabla \mathcal{R}(\theta_\gamma^{(k-1)}) + \varepsilon^{(k)}(\theta_\gamma^{(k-1)}) \right)}_{=g^{(k)}(\theta_\gamma^{(k-1)})}$$

where $(\varepsilon^{(k)})_{k \geq 0}$ are i.i.d. "zero-mean" random noises.

$\Rightarrow (\theta_\gamma^{(k)})_{k \geq 0}$ is a homogeneous Markov chain $\Rightarrow$ There exists a distribution π_γ such that asymptotically, $\theta_\gamma^{(k)} \sim \pi_\gamma, \forall k$
Some consequences :

- $(\theta_\gamma^{(k)})_{k \geq 0}$ does not converge to a point, but rather will (asymptotically) oscillate around the mean $\bar{\theta}_\gamma = \mathbb{E}_{\pi_\gamma}[\theta]$ with an average magnitude $\gamma^{1/2}$
- The average $\bar{\theta}_\gamma^{(k)}$ of the $k+1$ states $\{\theta^{(i)} : 0 \leq i \leq k\}$, converges towards this mean with rate $\mathcal{O}(1/\sqrt{k})$ (central limit theorem)
- This mean is not necessarily equal to the optimum parameters!

A random walk interpretation II

The error between the output of the SGD $\theta_\gamma^{(k)}$ and the optimum can be decomposed:

$$\bar{\theta}_\gamma^{(k)} - \theta^* = \underbrace{\bar{\theta}_\gamma^{(k)} - \bar{\theta}_\gamma}_{\mathcal{O}(1/\sqrt{k})} + \underbrace{\bar{\theta}_\gamma - \theta^*}_{\text{independent of k}}$$

We can prove that :

- For quadratic loss function : $\bar{\theta}_\gamma = \theta^*$, i.e. the output of SGD is the risk optimum
- In the general case, we have :
$$\bar{\theta}_\gamma = \theta^* + \gamma C + \mathcal{O}(\gamma^2)$$
where C is a constant independent of γ. Therefore the averaged SGD will only converge to a point "near" the optimum.

Note *: Sometimes it is sufficient to remain "near" the optimum, as the optimum can be an over-fitted solution of the problem. Moreover, for large dimension problems, SGD tends to converge to a solution that is quite near the optimum...*

A random walk interpretation III

Context
Optimization and supervised ML
Gradient descent
Convergence rates
Generalization risk optimization
Random walks and SGD
Annex

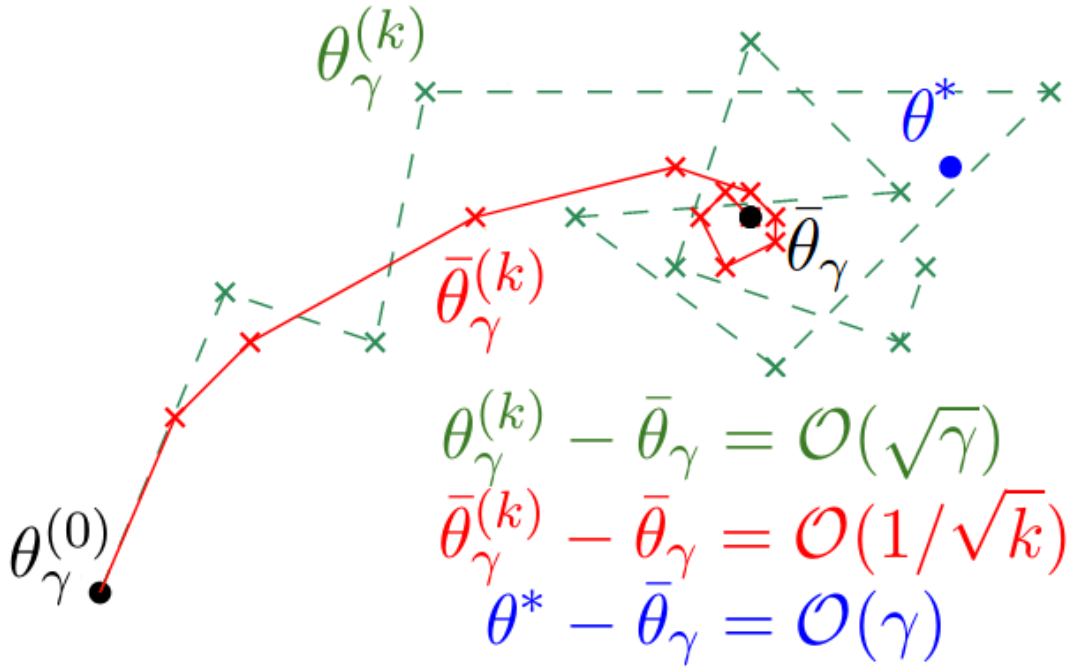

Figure: Convergence of iterates $\theta_\gamma^{(k)}$ and averaged iterates $\bar{\theta}_\gamma^{(k)}$ to the mean $\bar{\theta}_\gamma$ under the stationary distribution π_γ.

Richardson-Romberg Extrapolation

Context
Optimization and supervised ML
Gradient descent
Convergence rates
Generalization risk optimization
Random walks and SGD
Annex

Goal : To get a better estimate of θ^* in the general case.

For instance, for two terms :

1. Run SGD with rate $\gamma \to \bar{\theta}_\gamma$
2. Run SGD with rate $2\gamma \to \bar{\theta}_{2\gamma}$
3. Return
$$2\bar{\theta}_\gamma - \bar{\theta}_{2\gamma} = \theta^* + \mathcal{O}(\gamma^2)$$
which is a better estimate of θ^*.

This approach can simply be generalized to more terms (runs of the SGD).

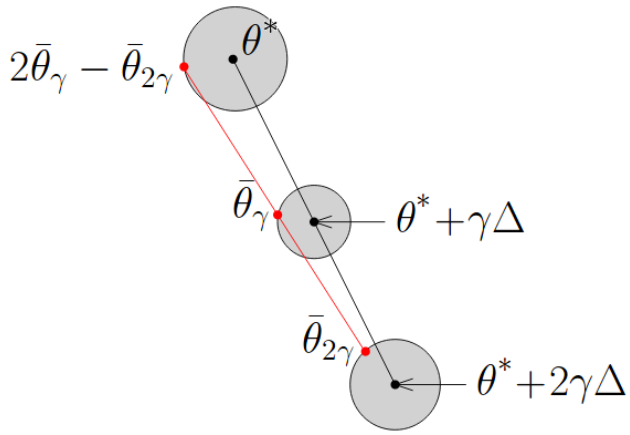

Figure: Richardson-Romberg extrapolation, the disks are of radius $\mathcal{O}(\gamma^2)$.

Outline

Context
Optimization and supervised ML
Gradient descent
Convergence rates
Generalization risk optimization
Random walks and SGD
Annex

1 Context
2 Optimization and supervised ML
3 Gradient descent
4 Convergence rates
5 Generalization risk optimization
6 Random walks and SGD
7 Annex

Proof that $\bar{\theta}_\gamma = \theta^*$ for quadratic loss

Notice that if we take $\theta_\gamma^{(0)} \sim \pi_\gamma$, then by definition of π_γ,

$$\theta_\gamma^{(0)} - \gamma \left(\nabla \mathcal{R}(\theta_\gamma^{(0)}) + \varepsilon^{(1)}(\theta_\gamma^{(0)}) \right) = \theta_\gamma^{(1)} \sim \pi_\gamma$$

Hence, by taking the expectation under π_γ we get :

$$\mathbb{E}_{\pi_\gamma}[\nabla \mathcal{R}(\theta_\gamma^{(0)})] = 0$$

and recalling the expression of $\mathcal{R}$ we get

$$\begin{aligned} \mathbb{E}_{\pi_\gamma}[\nabla \mathcal{R}(\theta)] &= \mathbb{E}_{\pi_\gamma}\left[\mathbb{E}_\rho[\nabla l(Y, \Phi(X)^T\theta)]\right] \\ &= \mathbb{E}_\rho\left[\mathbb{E}_{\pi_\gamma}[\nabla l(Y, \Phi(X)^T\theta)]\right] = 0 \end{aligned}$$

In particular, in the quadratic loss case, given that

$$\theta \mapsto \nabla l(Y, \Phi(X)^T\theta) = 2(Y - \Phi(X)^T\theta)\Phi(X)$$

is a linear function of θ, we have

$$\begin{aligned} \mathbb{E}_\rho\left[\mathbb{E}_{\pi_\gamma}[\nabla l(Y, \Phi(X)^T\theta)]\right] &= \mathbb{E}_\rho\left[\nabla l(Y, \Phi(X)^T\mathbb{E}_{\pi_\gamma}[\theta])\right] \\ &= \nabla \mathcal{R}(\mathbb{E}_{\pi_\gamma}[\theta]) = 0 \end{aligned}$$

And therefore,

$$\bar{\theta}_\gamma = \mathbb{E}_{\pi_\gamma}[\theta] = \theta^*$$

We retrieve the fact that in the quadratic loss case, the averaged SGD converges to the optimum at rate $\mathcal{O}(1/\sqrt{k})$.