Université de Nouakchott Al Aasriya

Faculté des Sciences et Techniques

Département de Mathématiques et Informatique

Rapport du TP 3

Optimisation Stochastique

Jedou Mohamed Bebacar

Matricule : C30824

Master SSD – Statistique et Sciences de Données

Janvier 2026

Table des matières

1 Introduction

Ce TP vise à illustrer les concepts du **Chapitre 3** sur le **Gradient Stochastique (SGD)**. Contrairement à la descente de gradient classique (GD), qui utilise l'ensemble des données, le SGD permet de traiter de grands volumes en mettant à jour le modèle de manière itérative. Nous étudions deux contextes :

— **Partie 1** : Classification binaire sur *Iris* (petit jeu de données),
— **Partie 2** : Régression sur *California Housing* (grand jeu de données).

2 Partie 1 : Classification binaire sur Iris

2.1 Objectif

Implémenter manuellement le **SGD from scratch** pour la perte logistique, et comparer sa convergence à celle de la **descente de gradient classique (GD)**.

2.2 Méthodologie

— **Données** : Iris (150 échantillons, 4 features). Binarisation : Setosa vs autres $\rightarrow$ $y_i \in \{-1, +1\}$.
— **Prétraitement** : Ajout d'un biais, standardisation des features.
— **Fonction objectif** : Perte logistique

$$f_i(w) = \log\left(1 + e^{-y_i x_i^\top w}\right)$$

— **Algorithmes** :
  — **GD** : $w^{k+1} = w^k - \alpha \nabla f(w^k)$
  — **SGD** : $w^{k+1} = w^k - \alpha_k \nabla f_{i_k}(w^k)$, avec $\alpha_k = \frac{\alpha_0}{1+k}$

2.3 Résultats

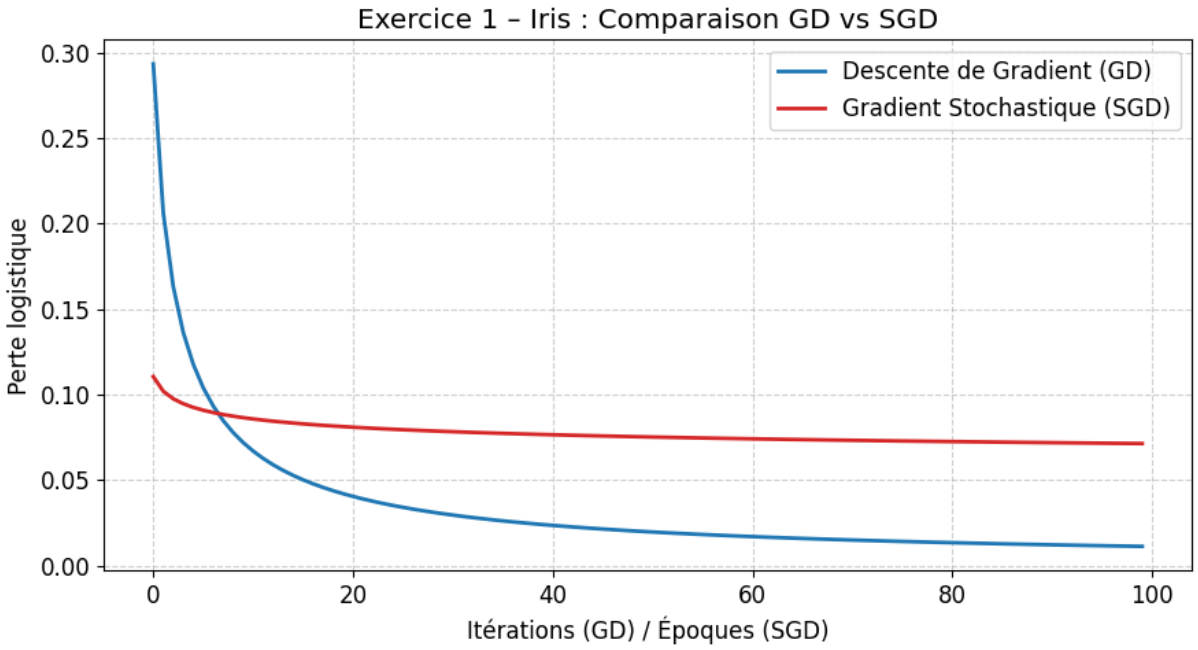

FIGURE 1 – Convergence de la perte logistique pour GD et SGD.

La courbe de **GD** est **lisse et monotone**, car elle utilise le gradient complet à chaque itération → direction exacte vers l'optimum. La courbe de **SGD** est **très instable**, car elle se base sur un seul échantillon aléatoire → **bruit élevé** dans le gradient. Cela illustre une propriété fondamentale du SGD : il ne garantit **pas une décroissance monotone**, mais seulement une **convergence en moyenne** vers un voisinage de l'optimum [Chap. 3, p. 7].

3 Partie 2 : Régression sur California Housing

3.1 Contexte

Le jeu *California Housing* contient plus de 20 000 lignes. Le calcul du gradient complet est coûteux → le SGD est motivé.

3.2 Exercice 2 : Importance de la Standardisation

3.2.1 Méthodologie

— **Données brutes** : Features à échelles très différentes (ex : `Population` 100–3500, `MedInc` 1–5).
— **Données standardisées** : $x_j \leftarrow \frac{x_j - \mu_j}{\sigma_j}$.
— **Fonction objectif** : MSE $f_i(w) = (x_i^\top w - y_i)^2$.

3.2.2 Résultats

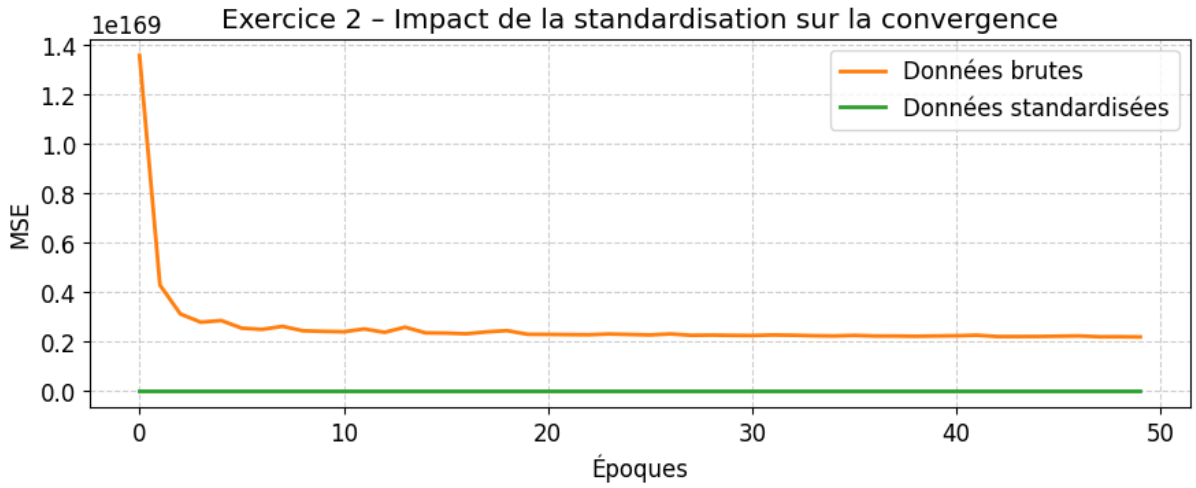

FIGURE 2 – Convergence du SGD sur données brutes vs standardisées.

Sur les **données brutes**, la convergence est **extrêmement lente** et oscillante. Cela s'explique par la **forme elliptique étirée** des lignes de niveau de la fonction de coût. Le gradient pointe perpendiculairement aux lignes de niveau, ce qui force l'algorithme à suivre un chemin en "zigzag". Après **standardisation**, les lignes de niveau deviennent **quasi-circulaires** → la descente est directe et rapide. Ce phénomène est explicitement mentionné dans le **Chapitre 3** : *« Le SGD est extrêmement sensible à l'échelle des données »* [Chap. 3, p. 1].

3.3 Exercice 3 : Mini-batch et Optimiseurs Modernes

3.3.1 Méthodologie

— **SGD pur** : batch_size = 1
— **Mini-batch SGD** : batch_size = 32
— **Adam** : Implémentation manuelle avec $\beta_1 = 0.9$, $\beta_2 = 0.999$

3.3.2 Résultats

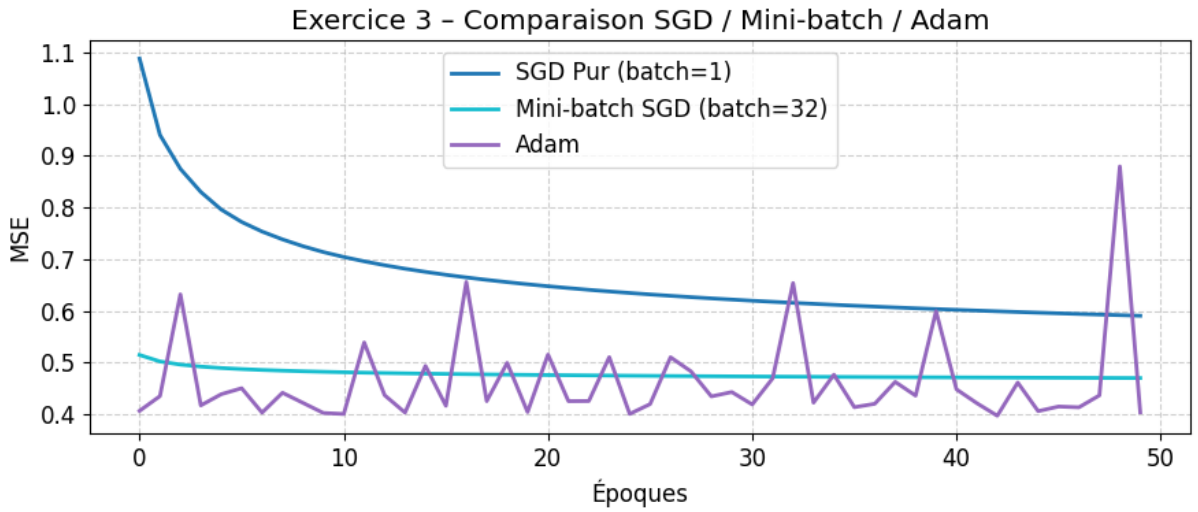

FIGURE 3 – Évolution de la MSE pour les trois algorithmes.

— Le **SGD pur** converge lentement avec un bruit important.
— Le **Mini-batch SGD** réduit significativement la variance grâce à la moyenne sur 32 échantillons → trajectoire plus stable.
— **Adam** atteint le **plateau de performance le plus rapidement**. Cela s'explique par deux mécanismes combinés [Chap. 3, p. 10] :
  1. Le **momentum** accumule les directions de descente cohérentes,
  2. L'**adaptation du pas** ajuste localement le learning rate pour chaque composante.

Adam est donc particulièrement adapté aux problèmes de grande dimension avec des gradients hétérogènes.

4 Synthèse et Réponses aux Questions

Q1. Pourquoi ne faut-il jamais utiliser un pas α trop grand avec le SGD ? :
Un pas trop grand **amplifie le bruit du gradient stochastique**, ce qui cause des **oscillations importantes** ou même une **divergence**. Comme le montre le **Théorème 3.2.1** [Chap. 3, p. 37], un pas constant trop grand ne garantit la convergence que vers un **voisinage** de l'optimum, dont la taille dépend de α.

Q2. Quel est l'avantage computationnel du Mini-batch par rapport au SGD pur sur GPU ? : Le mini-batch permet de **vectoriser** le calcul des gradients sur b exemples simultanément. Sur GPU, cela exploite le **parallélisme massif**, rendant chaque itération **plus rapide** qu'une séquence de b mises à jour SGD pures [Chap. 3, p. 40].

Q3. Expliquez le concept de Shuffling (mélange), pourquoi est-il crucial pour le gradient stochastique ? : Le shuffling garantit que les échantillons sont présentés dans un **ordre aléatoire** à chaque époque. Cela évite les **biais d'ordre** (ex : toutes les Setosa d'abord) et assure que l'estimateur du gradient reste **sans biais** :

$$\mathbb{E}[\nabla f_{i_k}(w^k)] = \nabla f(w^k)$$

Sans shuffling, la convergence peut être **biaisée** ou **instable** [Chap. 3, p. 5].

5 Conclusion

Ce TP a permis de :

— Valider expérimentalement les propriétés théoriques du **Chapitre 3**,

— Montrer que **SGD** et **mini-batch** sont indispensables pour les grands jeux de données,

— Confirmer que la **standardisation** et le **choix du pas** sont critiques pour la stabilité.

Les résultats confirment que **le choix de l'algorithme doit s'adapter à la taille des données** :

— **Petit** $n \rightarrow$ GD,

— **Grand** $n \rightarrow$ SGD, mini-batch ou Adam.