Chapitre 3 : Méthodes de gradient stochastique

Optimisation pour l'apprentissage automatique

Dr. EL BENANY Med Mahmoud

SSD

2025/2026

Sommaire

Motivation : Structure des données

- On dispose d'un échantillon de n exemples $\{(x_i, y_i)\}_{i=1}^{n}$ issus d'une distribution[cite : 180].
- But : Trouver un modèle $h(w; x_i) \approx y_i$ en minimisant une fonction de coût[cite : 181, 182].
- Problème de la somme finie :

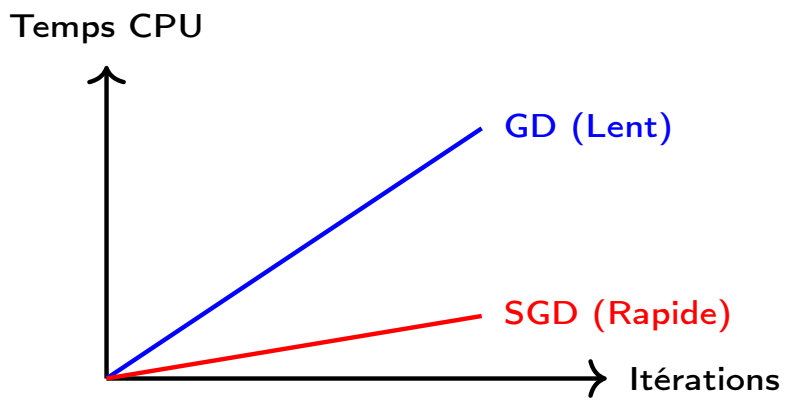
$$\min_{w \in \mathbb{R}^d} f(w) = \frac{1}{n} \sum_{i=1}^{n} f_i(w)$$

- Difficulté : Pour n très grand, le calcul du gradient complet $\nabla f(w)$ est extrêmement coûteux[cite : 199].

Batch vs Stochastique : Coût de calcul

Méthode	Complexité / Itération	Précision du Gradient
Full Batch (GD)	$O(n \cdot d)$	Exacte
Stochastique (SGD)	$O(d)$	Bruitée
Mini-batch	$O(\|S\| \cdot d)$	Intermédiaire

Table – Comparaison des coûts (n = nb d'exemples, d = dimension)



Observation : Pour $n = 10^9$, une seule itération de GD est impossible, tandis que SGD avance dès le premier exemple.

L'algorithme du Gradient Stochastique (SG)

Principe [cite : 200]

Au lieu de calculer la moyenne de tous les gradients, on utilise l'information d'un seul exemple (ou d'un petit lot) tiré aléatoirement.

Algorithme : À chaque itération k, on choisit un indice $i_k \in \{1, \ldots, n\}$ et on met à jour :

$$w_{k+1} = w_k - \alpha_k \nabla f_{i_k}(w_k)$$

où α_k est le taux d'apprentissage (*learning rate*)[cite : 219, 248].

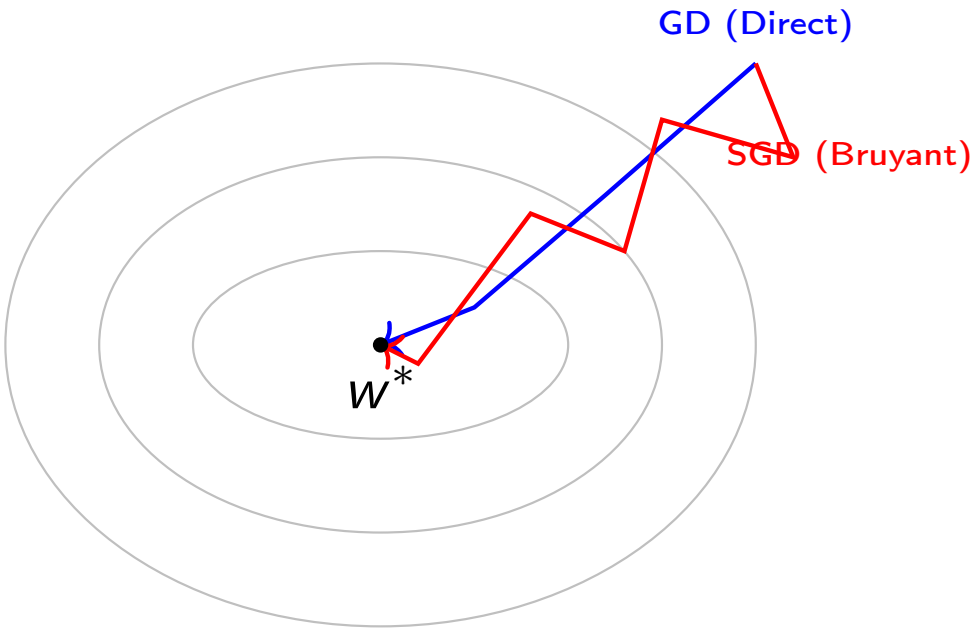
Algorithme : Gradient Stochastique (SGD)

Procédure SGD

1. **Initialisation :** $w_0 \in \mathbb{R}^d$, suite de pas $\{\alpha_k\}_{k\geq 0}$.
2. **Pour** $k = 0, 1, 2, \ldots$ **faire :**
   - Tirer un indice i_k uniformément dans $\{1, \ldots, n\}$.
   - Calculer le gradient local : $g_k = \nabla f_{i_k}(w_k)$.
   - **Mise à jour :** $w_{k+1} = w_k - \alpha_k g_k$.
   - **Optionnel :** Tester la convergence sur un jeu de validation.
3. **Fin Pour**
4. **Retourner :** w_{k+1} (ou la moyenne des itérés).

Note : Le coût par itération est indépendant de n.

Visualisation de la trajectoire : GD vs SGD



- **GD** : Suit la pente exacte, chemin direct.
- **SGD** : Direction aléatoire à chaque pas, oscille autour de la solution.

Analyse et Convergence

- **Biais :** On suppose que l'estimateur est sans biais : $\mathbb{E}_{i_k}[\nabla f_{i_k}(w_k)] = \nabla f(w_k)$[cite : 216].
- **Décroissance :** On ne garantit pas une décroissance à chaque pas, mais seulement une décroissance **en moyenne**[cite : 217].
- La variance de l'estimateur joue un rôle crucial dans la vitesse de convergence[cite : 218].

Variantes à lots (Mini-batch)

- **Principe :** Utiliser un ensemble d'indices $S_k \subset \{1, \ldots, n\}$ au lieu d'un seul[cite : 238, 239].
- **Avantage :** Réduit la variance de l'estimateur du gradient et permet de mieux exploiter le calcul parallélisé.
- **Mise à jour :**

$$w_{k+1} = w_k - \alpha_k \frac{1}{|S_k|} \sum_{i \in S_k} \nabla f_i(w_k)$$

Optimiseurs Modernes

Dans l'apprentissage profond, on utilise des variantes adaptatives[cite : 246, 250] :

- **Momentum :** Ajoute une fraction du déplacement précédent pour accélérer dans les directions constantes.
- **AdaGrad :** Adapte le pas pour chaque composante du vecteur w[cite : 2].
- **RMSProp / Adam :** Combinent le momentum et l'adaptation du taux d'apprentissage par composante pour une meilleure robustesse[cite : 2].

Algorithme : ADAM (Adaptive Moment Estimation)

Mise à jour à l'étape k

- **Gradients :** $g_k = \nabla f_{i_k}(w_k)$
- **Moments d'ordre 1 :** $m_k = \beta_1 m_{k-1} + (1-\beta_1) g_k$
- **Moments d'ordre 2 :** $v_k = \beta_2 v_{k-1} + (1-\beta_2) g_k^2$
- **Correction du biais :** $\hat{m}_k = \frac{m_k}{1-\beta_1^k}, \quad \hat{v}_k = \frac{v_k}{1-\beta_2^k}$
- **Mise à jour finale :**

$$w_{k+1} = w_k - \frac{\alpha}{\sqrt{\hat{v}_k} + \epsilon} \hat{m}_k$$

Utilisation : Quasi-universelle en Deep Learning (vision, NLP).

Comparatif des Algorithmes Adaptatifs

Méthode	Caractéristique Clé	Usage recommandé
Momentum	$v_k = \beta v_{k-1} + \nabla f_i$	Éviter les oscillations
AdaGrad	α_k divisé par $\sqrt{\sum g^2}$	Données creuses (*sparse*)
RMSProp	Moyenne mobile des carrés	Séries temporelles / RNN
Adam	Momentum + RMSProp	**Standard par défaut**

Le rôle de la variance

Ces méthodes tentent de compenser le bruit du gradient stochastique par une estimation de sa variance passée.

Synthèse : Comparatif des Algorithmes d'Optimisation

Algorithme	Complexité / Itér.	Vitesse (Conv.)	Usage Principal
GD (Batch)	$O(n \cdot d)$	Rapide (itérations)	Petits jeux de données
SGD	$O(d)$	Lente / Bruitée	Big Data, Online Learning
Nesterov	$O(n \cdot d)$	Accélérée	Problèmes convexes lisses
Adam	$O(d)$	Très rapide	**Réseaux de neurones**

Table – Tableau comparatif global

Critères de choix

- **Précision :** Privilégier le GD ou Mini-batch avec grand b.
- **Vitesse CPU/GPU :** Privilégier le SGD ou Adam.
- **Mémoire :** SGD est le plus économe (1 seul exemple à la fois).

Conclusion

- Le gradient stochastique est indispensable pour le *Big Data*[cite : 179].
- Le choix du *learning rate* α_k est le défi majeur de ces méthodes[cite : 219].
- Les méthodes adaptatives (comme Adam) sont les standards actuels pour l'entraînement des réseaux de neurones[cite : 2].

Exercice 1 : Estimateur sans biais

Soit $f(w) = \frac{1}{n}\sum_{i=1}^{n} f_i(w)$. On choisit i uniformément dans $\{1, \ldots, n\}$.

1. Rappelez la définition de l'espérance $\mathbb{E}[\nabla f_i(w)]$.
2. Montrez que $\mathbb{E}[\nabla f_i(w)] = \nabla f(w)$.
3. Pourquoi cette propriété est-elle fondamentale pour la convergence du SGD ?

Exercice 2 : Mini-batch et Variance

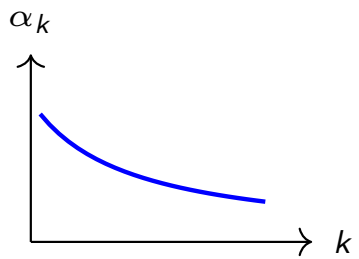
On compare le gradient stochastique pur ($|S| = 1$) et le mini-batch ($|S| = b$).

- Comment évolue la variance de l'estimateur quand b augmente ?
- Si $b = n$, quelle méthode retrouve-t-on ?
- Quel est l'impact de b sur la parallélisation (GPU) ?

Exercice 3 : Décroissance du pas (Learning Rate)

Pour le SGD, on utilise souvent $\alpha_k = \frac{\alpha_0}{k+1}$.

- Pourquoi n'utilise-t-on pas un pas fixe comme dans le GD ?
- Expliquez le phénomène de "stagnation dans une zone de bruit" si le pas est trop grand.



Correction Exercice 1 : Biais du gradient

Preuve de l'estimateur sans biais

Soit I une variable aléatoire suivant une loi uniforme sur $\{1, \ldots, n\}$.

$$\mathbb{E}_I[\nabla f_I(w)] = \sum_{i=1}^{n} P(I = i) \nabla f_i(w)$$

Comme $P(I = i) = \frac{1}{n}$ pour tout i :

$$\mathbb{E}_I[\nabla f_I(w)] = \sum_{i=1}^{n} \frac{1}{n} \nabla f_i(w) = \frac{1}{n} \sum_{i=1}^{n} \nabla f_i(w)$$

On retrouve exactement $\nabla f(w)$.

Conséquence : En moyenne, le SGD suit la direction du vrai gradient.

Références bibliographiques

- **L. Bottou**, *Large-Scale Machine Learning with Stochastic Gradient Descent*, COMPSTAT, 2010.
- **Robbins & Monro**, *A Stochastic Approximation Method*, Annals of Mathematical Statistics, 1951. (L'article fondateur).
- **Kingma & Ba**, *Adam : A Method for Stochastic Optimization*, ICLR, 2015.
- **Goodfellow et al.**, *Deep Learning* (Chapitre 8), MIT Press, 2016.
- **[180, 200, 217]** Bottou, L. (2010). *Large-Scale Machine Learning with SGD*. COMPSTAT.
- **[216, 218]** Robbins, H., & Monro, S. (1951). *A Stochastic Approximation Method*. Annals of Math. Stats.
- **[2, 250]** Kingma, D. P., & Ba, J. (2015). *Adam : A Method for Stochastic Optimization*. ICLR.
- **[181, 246]** Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.
- **[199, 219]** Bottou, L., Curtis, F. E., & Nocedal, J. (2018). *Optimization Methods for Large-Scale ML*. SIAM Review.