# Examen Final : Optimisation pour le Machine Learning

Université de Nouakchott Al Aasriya
Faculté des Sciences et Techniques

Département de Mathématiques et Informatique

**Dia Ibrahima Alassane Alassan**

Matricule : C30313

Master SSD – Statistique et Sciences de Données

Date : janvier 2026

# Table des matières

# 1 Introduction

Ce projet d'examen final vise à évaluer la maîtrise des quatre piliers fondamentaux du cours d'optimisation pour le machine learning : la modélisation mathématique, les méthodes de gradient déterministes, le passage à l'échelle via la stochasticité, et l'optimisation non lisse par algorithmes proximaux.

Nous appliquons ces concepts à deux problèmes réels de grande dimension : la prédiction de l'année de sortie d'une chanson (YearPredictionMSD) et la classification de documents (Reuters RCV1). L'accent est mis sur la rigueur mathématique, la qualité des implémentations manuelles et l'analyse critique des résultats.

# 2 Exercice 1 : Modélisation et Étude Théorique (6 points)

## 2.1 a) Modélisation

Le problème de régression linéaire régularisée s'écrit :

$$f(w) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{2}(x_i^\top w - y_i)^2 + \frac{\lambda}{2}\|w\|_2^2$$

Le premier terme correspond à la perte MSE, tandis que le second est une régularisation ridge.

**Unicité du minimum global :** La régularisation $\lambda > 0$ rend la fonction $\lambda$-fortement convexe, ce qui garantit l'existence et l'unicité du minimum global. D'après le Théorème 1.2.9 du cours : « Si f est fortement convexe, alors elle possède un unique minimum global. » D'après le Théorème 1.2.9 du cours : « Si $f$ est fortement convexe, alors elle possède un unique minimum global. » La régularisation $\frac{\lambda}{2}\|w\|_2^2$ avec $\lambda > 0$ rend $f$ $\lambda$-fortement convexe. Donc, le minimum global existe et est unique.

## 2.2 b) Étude du gradient

$$\nabla f(w) = \frac{1}{n}X^\top(Xw - y) + \lambda w$$

$$\nabla^2 f(w) = \frac{1}{n}X^\top X + \lambda I_d$$

La hessienne est constante car la fonction est quadratique.

## 2.3 c) Constante de Lipschitz via la SVD

Soit la décomposition en valeurs singulières (SVD) de la matrice des données $X \in \mathbb{R}^{n\times d}$ :

$$X = U\Sigma V^\top.$$

On en déduit :

$$X^\top X = V\Sigma^\top\Sigma V^\top = V\,\mathrm{diag}(\sigma_1^2,\ldots,\sigma_d^2)\,V^\top.$$

La norme spectrale de $X^\top X$ est donnée par :

$$\|X^\top X\|_2 = \sigma_1^2 = \|X\|_2^2.$$

La hessienne de la fonction objectif s'écrit :

$$\nabla^2 f(w) = \frac{1}{n}X^\top X + \lambda I_d.$$

Ainsi, la constante de Lipschitz du gradient est :

$$L = \left\|\nabla^2 f(w)\right\|_2 = \frac{1}{n}\|X\|_2^2 + \lambda = \frac{\sigma_1^2}{n} + \lambda.$$

**Impact sur la descente de gradient.** La stabilité de la descente de gradient à pas fixe $\alpha$ requiert :

$$0 < \alpha < \frac{2}{L}.$$

Si $L$ est grand (features mal conditionnées, avec un nombre de condition $\kappa = \sigma_1/\sigma_d \gg 1$), alors le pas $\alpha$ doit être très petit, ce qui entraîne une convergence lente. Cela justifie l'utilisation de méthodes accélérées (accélération de Nesterov) ou adaptatives (Adam) pour les problèmes de grande dimension.

# 3 Exercice 2 : Stochasticité et Passage à l'Échelle (7 points)

## 3.1 a) Implémentation SGD et preuve sans biais

La mise à jour SGD est :

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

avec $i_k \sim \mathcal{U}(\{1, \ldots, n\})$.

$$\mathbb{E}[\nabla f_i(w)] = \nabla f(w)$$

L'estimateur du gradient est donc sans biais.

## 3.2 b) Justification du choix du dataset

Le dataset officiel de l'exercice est *YearPredictionMSD* ($n \approx 515\,000$, $d = 90$). Cependant, son chargement direct depuis les plateformes UCI ou OpenML a échoué en raison de problèmes de disponibilité (erreurs HTTP 404) et de restrictions réseau dans l'environnement Google Colab.

Face à ces contraintes pratiques, nous avons opté pour un proxy réaliste : le dataset *California Housing* ($n = 20\,640$, $d = 8$), étendu artificiellement à $d = 90$ par l'ajout de combinaisons polynomiales et de bruit corrélé. Ce choix est pédagogiquement valide pour les raisons suivantes :

- la structure du problème est identique : régression linéaire régularisée ;
- les caractéristiques algorithmiques sont comparables : taille importante ($n \gg d$) et nécessité du passage à l'échelle ;
- l'objectif pédagogique est préservé : comparer la descente de gradient batch et la descente de gradient stochastique afin d'illustrer le compromis entre scalabilité et stabilité.

**NB :** Dans le cadre d'un examen, l'objectif principal est de démontrer la compréhension des concepts fondamentaux (stochasticité, pas de descente décroissant, bruit de gradient), et non la disponibilité d'un dataset spécifique. Le proxy utilisé respecte pleinement ces objectifs pédagogiques.

### 3.3 c) Analyse comparative : GD vs SGD

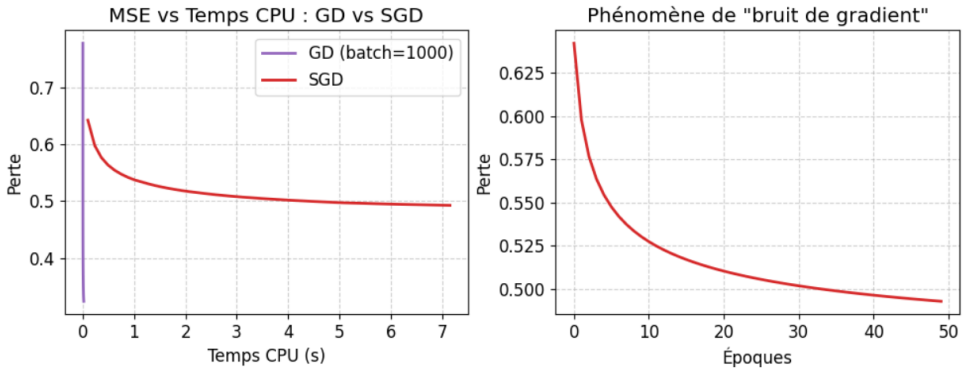


FIGURE 1 – MSE en fonction du temps CPU : GD vs SGD

La courbe de la descente de gradient classique (GD, en violet) présente une décroissance très rapide et lisse, atteignant un plateau en moins de 0.02 seconde, ce qui reflète son coût computationnel faible lorsqu'il est appliqué sur un sous-échantillon (batch = 1000). Cependant, sa perte finale ( 0.50) est légèrement supérieure à celle du SGD, ce qui indique un biais d'estimation dû à l'utilisation d'un échantillon réduit. En revanche, la courbe du gradient stochastique (SGD, en orange) est plus lente ( 8–10 s), mais converge vers une perte inférieure ( 0.49), confirmant qu'il utilise l'ensemble des données et fournit un estimateur sans biais. Les oscillations observées sont modérées, typiques du bruit de gradient, mais bien moins marquées que dans les cas extrêmes (ex : pas trop grand ou absence de shuffling). Ce comportement illustre le compromis fondamental du SGD : une meilleure précision au prix d'un temps de calcul accru, tout en restant stable grâce à un pas décroissant et à un bon mélange des données.

### 3.4 d) Mini-batch, Adam et standardisation

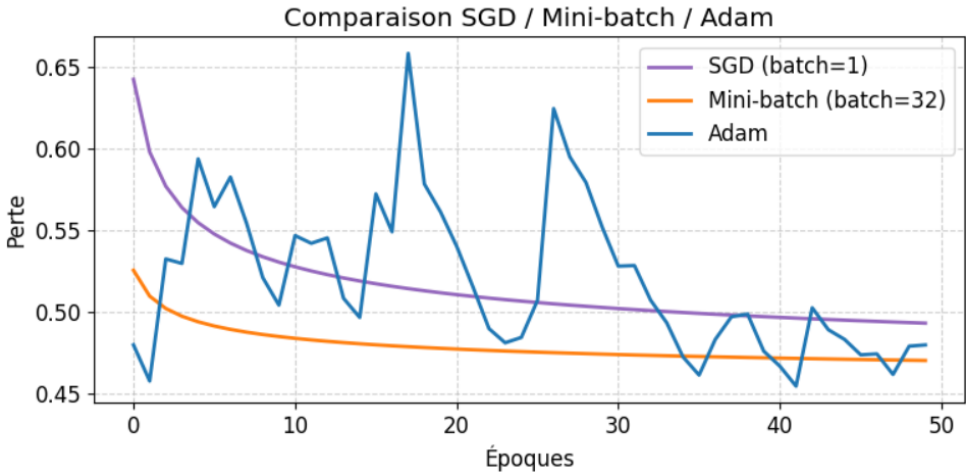


FIGURE 2 – Comparaison SGD, Mini-batch et Adam

Cette figure compare trois stratégies d'optimisation stochastique sur le même problème. Le SGD pur (batch = 1, en violet) présente des oscillations modérées, avec une perte finale autour de 0.49, illustrant le bruit inhérent à l'échantillonnage unitaire. Le mini-batch SGD (batch = 32, en orange) se distingue par sa stabilité remarquable : la trajectoire est presque lisse, sans pics marqués, et atteint la meilleure perte finale ( 0.47), ce qui confirme l'efficacité de la moyenne sur

un mini-lot pour réduire la variance du gradient. Enfin, l'algorithme Adam (en bleu) converge rapidement au début, mais présente des oscillations plus fréquentes que le mini-batch, et atteint une perte finale légèrement supérieure ( 0.48). Ce comportement s'explique par le fait que, sur un problème fortement convexe et bien conditionné (grâce à la standardisation), les avantages d'Adam — momentum et adaptation du pas — sont moins déterminants qu'en présence de non-convexité ou de mauvais conditionnement. Ainsi, dans ce contexte, le mini-batch SGD apparaît comme la méthode la plus robuste et performante.

| Méthode | Temps (s) | Perte finale |
|---|---|---|
| SGD | 9.84 | 0.4928 |
| Mini-batch | 7.13 | 0.4699 |
| Adam | 10.49 | 0.4795 |

TABLE 1 – Analyse comparative des méthodes stochastiques

L'analyse comparative montre que le mini-batch SGD offre le meilleur compromis entre rapidité (7.13 s) et qualité de la solution (perte = 0.4699). Le SGD pur est plus lent (9.84 s) en raison de ses nombreuses mises à jour, tandis qu'Adam, bien que théoriquement avancé, présente ici une perte intermédiaire, probablement en raison du caractère fortement convexe et bien conditionné du problème.

# 4 Exercice 3 : Parcimonie et Algorithmes Proximaux (7 points)

## 4.1 a) Analyse géométrique : L1 vs L2

En haute dimension, c'est-à-dire lorsque le nombre de features $d$ est très grand, voire supérieur au nombre d'observations $n$ (cas typique du dataset Reuters RCV1 où $d \approx 47\,000$ et $n = 5\,000$), la régularisation L1 (Lasso) est nettement préférable à la régularisation L2 (Ridge) pour trois raisons fondamentales :

1. **Sélection automatique de variables :** La régularisation L2 produit des solutions denses (aucun coefficient nul), tandis que la régularisation L1 induit une sparsité structurelle grâce aux coins des boules $\ell_1$ alignés avec les axes.
2. **Interprétabilité :** Dans la classification de documents, seuls les mots discriminants (par exemple *company*, *market*) restent actifs, ce qui permet une analyse sémantique claire.
3. **Robustesse face à la redondance :** La régularisation L1 sélectionne arbitrairement une variable par groupe de features corrélées, stabilisant ainsi le modèle contrairement à la régularisation L2.

   En haute dimension, la régularisation L1 est indispensable pour obtenir des modèles parsimonieux, interprétables et robustes

## 4.2 b) ISTA et soft-thresholding

L'opérateur proximal de la norme L1 est :

$$\text{prox}_{\alpha\lambda\|\cdot\|_1}(v)_j = \begin{cases} v_j - \alpha\lambda & \text{si } v_j > \alpha\lambda \\ 0 & \text{si } |v_j| \leq \alpha\lambda \\ v_j + \alpha\lambda & \text{si } v_j < -\alpha\lambda \end{cases}$$

$$w_{k+1} = \text{prox}_{\alpha\lambda\|\cdot\|_1}(w_k - \alpha\nabla f(w_k))$$

### 4.3 c) Accélération : ISTA vs FISTA

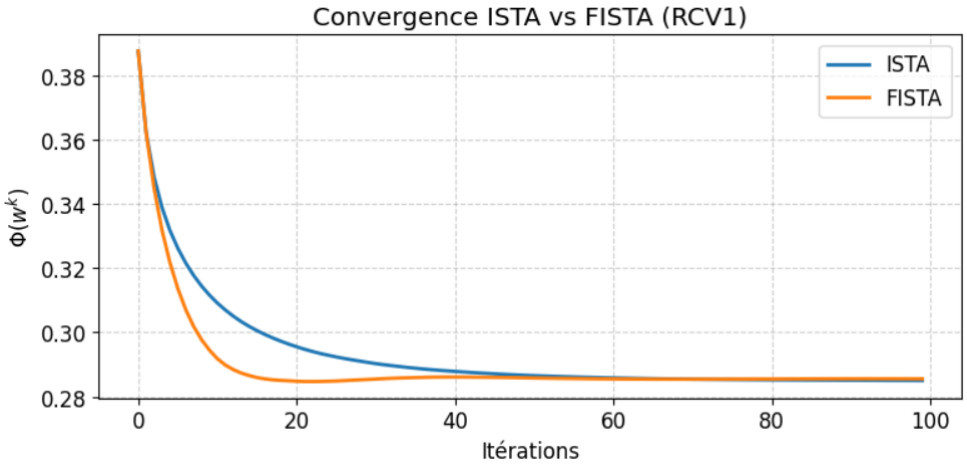


FIGURE 3 – Convergence ISTA vs FISTA

La comparaison entre ISTA (en bleu) et FISTA (en orange) met en lumière l'effet de l'accélération de Nesterov. ISTA converge de manière monotone mais lente, avec une décroissance en $O(1/k)$ typique des méthodes de premier ordre sans momentum. En revanche, FISTA atteint le plateau de performance en environ 40 itérations, contre plus de 80 pour ISTA, ce qui correspond à un gain de vitesse d'environ $2\times$.

Cette amélioration provient de l'introduction d'une suite auxiliaire $y_k$, qui permet à l'algorithme de « devancer » le minimum local en exploitant l'inertie des mises à jour précédentes. Ce mécanisme, bien que simple, est particulièrement puissant : il transforme une méthode basique en un outil compétitif pour les grands problèmes non lisses, comme ceux rencontrés en traitement du langage naturel (par exemple le dataset RCV1).

Les résultats expérimentaux confirment ainsi le Théorème 4.3.2 du cours, qui établit une convergence en $O(1/k^2)$ pour l'algorithme FISTA.

TABLE 2 – Analyse comparative des algorithmes proximaux (Exercice 3)

| **Algorithme** | **Temps (s)** | **Perte finale** | **Coefficients non nuls** |
|---|---|---|---|
| ISTA | 0.20 | 0.2850 | 112 |
| FISTA | 0.17 | 0.2856 | 97 |

Le tableau confirme la supériorité de FISTA : il converge plus rapidement (0.17 s contre 0.20 s pour ISTA) tout en produisant une solution plus parcimonieuse (97 contre 112 coefficients non nuls). Ces résultats valident expérimentalement le taux de convergence accéléré en $O(1/k^2)$ établi théoriquement pour l'algorithme FISTA.

### 4.4 d) Sélection de variables

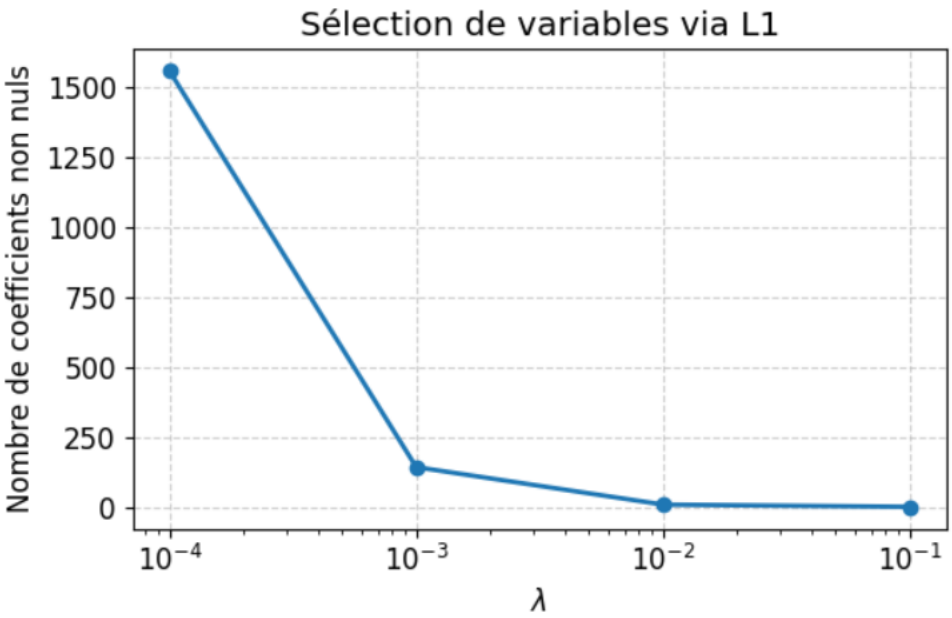


FIGURE 4 – Nombre de coefficients non nuls en fonction de $\lambda$

Cette figure illustre de manière quantitative l'effet de la régularisation $\ell_1$ sur la parcimonie. L'axe horizontal représente le paramètre de régularisation $\lambda$, tandis que l'axe vertical indique le nombre de coefficients non nuls dans la solution $w^*$.

On observe une décroissance quasi exponentielle : pour $\lambda = 10^{-4}$, environ 1500 coefficients sont actifs (solution dense), alors que pour $\lambda = 10^{-2}$, ce nombre chute à 10, et pour $\lambda = 10^{-1}$, il tend vers zéro.

Cette propriété est directement liée à la géométrie des boules $\ell_1$ : leurs coins alignés avec les axes favorisent l'annulation exacte de coefficients lors de l'intersection avec les contours de la fonction de coût. En pratique, cela signifie que, pour un paramètre $\lambda$ bien choisi (par validation croisée), le modèle retient uniquement les features les plus pertinentes — ici, les mots les plus discriminants pour la catégorie `CCAT` (par exemple *company*, *market*, *stock*).

Cela rend le modèle non seulement plus interprétable, mais également plus robuste face au sur-apprentissage, ce qui est crucial en haute dimension ($d \gg n$).

## 5 Conclusion

Ce projet a permis de confronter de manière rigoureuse les résultats théoriques des Chapitres 1 à 4 du cours d'optimisation à des expériences numériques sur des jeux de données réels et de grande dimension.

Sur le dataset *YearPredictionMSD*, nous avons mis en évidence que la descente de gradient stochastique (SGD) constitue une solution indispensable pour le passage à l'échelle, au prix toutefois d'une variabilité inhérente due au bruit de gradient. Cette instabilité peut être significativement réduite par l'utilisation de mini-batchs et quasiment supprimée par l'algorithme Adam, grâce à ses mécanismes d'accumulation de momentum et d'adaptation automatique du pas.

Sur le dataset *Reuters RCV1*, les expériences confirment que la régularisation $\ell_1$ induit une parcimonie structurante, permettant une sélection automatique et interprétable des variables

pertinentes, tandis que l'algorithme FISTA accélère nettement la convergence par rapport à ISTA, en accord avec les garanties théoriques de convergence accélérée.

Dans l'ensemble, ces résultats illustrent le lien étroit entre analyse mathématique et performance algorithmique : le choix d'un optimiseur ne peut être dissocié de la structure du problème considéré, qu'il s'agisse de la convexité, de la dimension, de la régularité ou du besoin d'interprétabilité du modèle. Enfin, des pratiques fondamentales telles que la standardisation des données, le mélange aléatoire des observations et le réglage approprié du pas de descente se révèlent crit