

DESCENTE DE GRADIENT ET VARIANTES
SÉMINAIRE MATHS POUR ML

Théo Lopès-Quintas

BPCE Payment Services,
Université Paris Dauphine

16 novembre 2023

1	Rappels	1
2	Descente de gradient	5
2.1	Descente de gradient stochastique	11
2.2	SGD par mini-batch	15
2.3	Descente avec momentum	18
3	Descente de gradient avec pas adaptatif	21
3.1	AdaGrad	22
3.2	RMSProp	24
3.3	Adam	25
4	Descente de gradient et régularisation	29
5	Comment choisir le pas de descente?	32

RAPPELS

1	Rappels	1
2	Descente de gradient	5
2.1	Descente de gradient stochastique	11
2.2	SGD par mini-batch	15
2.3	Descente avec momentum	18
3	Descente de gradient avec pas adaptatif	21
3.1	AdaGrad	22
3.2	RMSProp	24
3.3	Adam	25
4	Descente de gradient et régularisation	29
5	Comment choisir le pas de descente?	32

RAPPELS

FORMULATION D'UN PROBLÈME DE MACHINE LEARNING

Dans le cadre supervisé, nous avons accès à un dataset \mathcal{D} défini comme :

$$\mathcal{D} = \left\{ (x_i, y_i) \mid \forall i \leq n, x_i \in \mathbb{R}^{d'}, y_i \in \mathcal{Y} \right\}$$

Nombre d'observations (pointing to n) Nombre d'informations (pointing to d')

Avec $\mathcal{Y} \subseteq \mathbb{R}$ pour un problème de régression et $\mathcal{Y} \subset \mathbb{N}$ dans le cadre d'une classification. Les problèmes de Machine Learning supervisé peuvent souvent s'écrire sous la forme d'une optimisation d'une fonction de perte $\mathcal{L} : \mathbb{R}^d \times \mathcal{M}_{n,d'} \times \mathbb{R}^n \rightarrow \mathbb{R}_+$ comme :

Vecteur des paramètres optimaux

$$\theta^* = \arg \min_{\theta \in \mathbb{R}^d} \mathcal{L}(\theta, X, y)$$

Dimension du vecteur de paramètres (pointing to d)

Dans la suite, pour simplifier les notations, nous omettrons la dépendance de \mathcal{L} en X (matrice des informations) et y (vecteur réponse). Notons qu'en général, nous avons $d \neq d'$ et dans le cas du deep learning, très souvent $d \gg d'$.

RAPPELS

RÉSEAU DE NEURONES

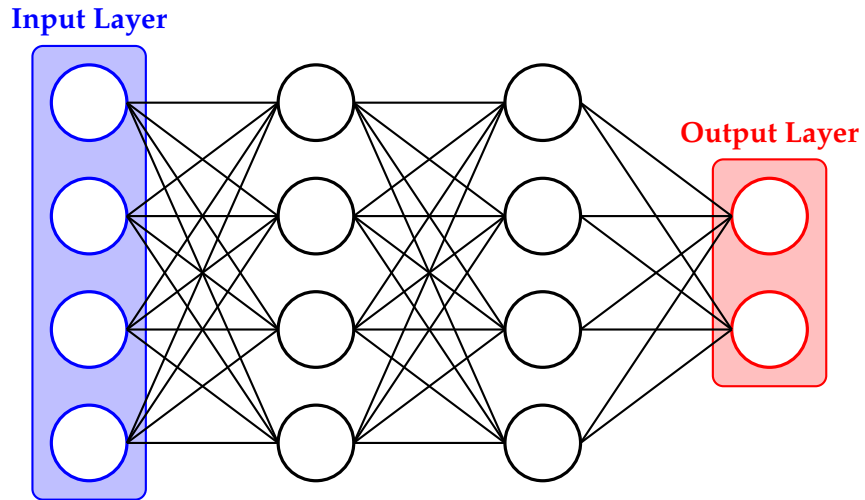


Figure – Exemple d'un réseau de neurones avec 2 couches cachées pour une classification binaire

RAPPELS

NEURONES

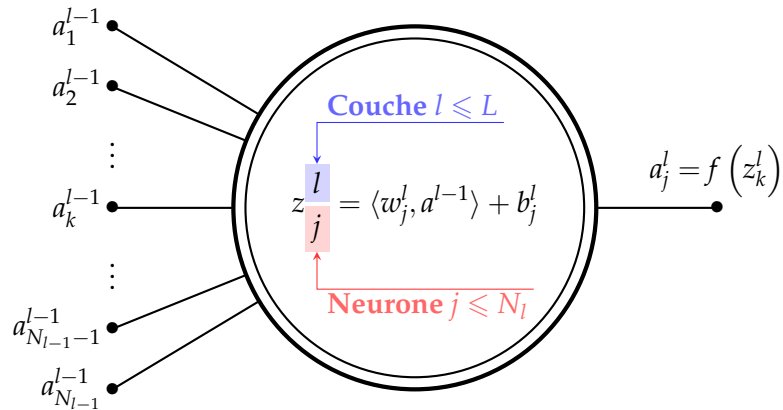


Figure – Neurone j de la couche l paramétré par les poids $w_i^j \in \mathbb{R}^{N_{l-1}}$ et le biais $b_j^l \in \mathbb{R}$

L'objectif est de comprendre comment on peut *apprendre efficacement* ces paramètres.

DESCENTE DE GRADIENT

1	Rappels	1
2	Descente de gradient	5
2.1	Descente de gradient stochastique	11
2.2	SGD par mini-batch	15
2.3	Descente avec momentum	18
3	Descente de gradient avec pas adaptatif	21
3.1	AdaGrad	22
3.2	RMSProp	24
3.3	Adam	25
4	Descente de gradient et régularisation	29
5	Comment choisir le pas de descente?	32

DESCENTE DE GRADIENT

MÉTHODE

La méthode la plus utilisée pour résoudre ce genre de problème est la descente de gradient :

$$\theta_{t+1} = \theta_t - \eta_t \nabla \mathcal{L}(\theta_t)$$

↑
Learning rate

Quand on travaille avec des grands datasets, le coût de calcul/temps est grand si l'on calcule $\nabla \mathcal{L}(\theta_t)$ pour la totalité de la base. On peut donc considérer d'autres approches :

- ▶ **Stochastique (SGD)** : on sélectionne au hasard une observation et on met à jour θ
- ▶ **Stochastique par batch** : on sélectionne aléatoirement une partie de la base (batch) et on met à jour θ à chaque batch

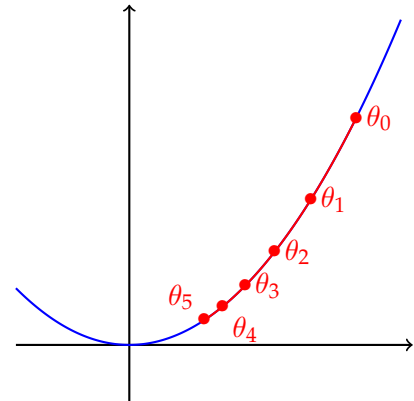


Figure – Exemple d'une descente de gradient pour $f(x) = x^2$

DESCENTE DE GRADIENT

GARANTIE DE CONVERGENCE DANS LE CAS CONVEXE ET β -SMOOTH

Dans le cas d'une fonction convexe, nous savons qu'il existe un minimum global. Avec la descente de gradient, nous avons une garantie de convergence, et on exhibe une vitesse de convergence dans le cas où \mathcal{L} est une fonction convexe et β -smooth.

Théorème 1 (Vitesse de convergence pour une fonction convexe et β -smooth)

Soit $\mathcal{L} : \mathbb{R}^d \rightarrow \mathbb{R}$ une fonction convexe et β -smooth ayant un minimum $\theta^* = \arg \min_{\theta \in \mathbb{R}^d} \mathcal{L}(\theta)$. De plus, pour

toute étape t on a $\eta_t = \frac{1}{\beta}$. Alors, pour tout $T \geq 1$:

$$\forall t \leq T, \quad \mathcal{L}(\theta_t) - \mathcal{L}(\theta^*) \leq \mathcal{O}\left(\frac{1}{T}\right)$$

Le choix de la valeur du learning rate est empirique en pratique, il n'est donc pas nécessaire de connaître la valeur β .

DESCENTE DE GRADIENT

DESCENTE DE GRADIENT POUR UN RÉSEAU DE NEURONE

Considérons le réseau de neurones suivant avec comme fonction d'activation ReLU :

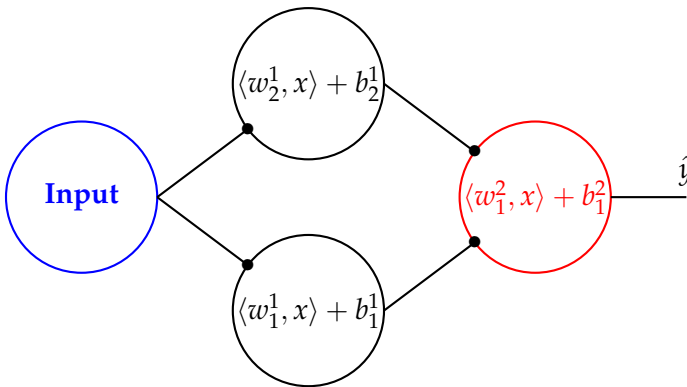


Figure – Réseau de neurone avec une couche caché de deux neurones

Pour le dataset $\mathcal{D} = \{(-1, -1), (1, 1)\}$ et la fonction de perte

$$\mathcal{L}(\theta) = \frac{1}{2} \sum_{i=1}^2 (\hat{y}_i - y_i)^2, \text{ on a que :}$$

- ▶ $\theta_1 : w_1^1 = 1, w_2^1 = -1, w_1^2 = (1, -1)$ et $b_1^1 = b_2^1 = b_1^2 = 0$ donne $\mathcal{L}(\theta_1) = 0$
- ▶ $\theta_2 : w_1^1 = -1, w_2^1 = 1, w_1^2 = (-1, 1)$ et $b_1^1 = b_2^1 = b_1^2 = 0$ donne $\mathcal{L}(\theta_2) = 0$

Si la fonction de perte \mathcal{L} est convexe, alors en prenant une combinaison linéaire entre les deux θ obtenu, on doit avoir encore un minimum. Mais ce n'est pas le cas !

En général dans un réseau de neurone, nous ne sommes pas dans un cadre convexe. Nous souhaitons accélérer la descente de gradient ainsi qu'obtenir des garanties pratique que la méthode classique n'offre pas, pour s'assurer d'une *meilleure convergence*.

DESCENTE DE GRADIENT

GARANTIE DE CONVERGENCE DANS LE CAS β -SMOOTH

Puisque nous ne pouvons pas se reposer sur le cas convexe, nous pouvons exhiber une garantie de convergence dans le cas non-convexe, mais toujours avec l'hypothèse d'une fonction β -smooth.

Théorème 2 (Vitesse de convergence pour une fonction β -smooth)

Soit $\mathcal{L} : \mathbb{R}^d \rightarrow \mathbb{R}$ une fonction β -smooth ayant un minimum $\theta^* = \arg \min_{\theta \in \mathbb{R}^d} \mathcal{L}(\theta)$. De plus, pour toute étape t on

a $\eta_t = \frac{1}{\beta}$. Alors :

$$\forall T \geq 1, \quad \min_{1 \leq t < T} \|\mathcal{L}(\theta_t)\| \leq \mathcal{O}\left(\frac{1}{\sqrt{T}}\right)$$

A la lumière du théorème 1, on peut faire deux remarques :

- ▶ La convergence est plus lente : $\mathcal{O}\left(\frac{1}{T}\right)$ pour le cas convexe
- ▶ On ne garantie pas la convergence vers un minimum, uniquement vers un gradient nul dans le cas non convexe

DESCENTE DE GRADIENT

GARANTIE DE CONVERGENCE DANS LE CAS β -SMOOTH : PREUVE

Puisque \mathcal{L} est une fonction β -smooth, on a :

$$\begin{aligned}\mathcal{L}(\theta_{t+1}) &\leq \mathcal{L}(\theta_t) + \nabla \langle \mathcal{L}(\theta_t), \theta_{t+1} - \theta_t \rangle + \frac{\beta}{2} \|\theta_{t+1} - \theta_t\|^2 \\ &= \mathcal{L}(\theta_t) - \left(\eta_t - \frac{\beta}{2} \eta_t^2 \right) \|\nabla \mathcal{L}(\theta_t)\|^2 \quad \text{et par définition de } \eta_t : \\ &= \mathcal{L}(\theta_t) - \frac{1}{2\beta} \|\nabla \mathcal{L}(\theta_t)\|^2 \\ &\leq \mathcal{L}(\theta_t) - \frac{1}{2\beta} \min_{1 \leq t < T} \|\nabla \mathcal{L}(\theta_t)\|^2\end{aligned}$$

En sommant sur l'ensemble des itérations, et en supprimant les termes commun :

$$\begin{aligned}\mathcal{L}(\theta_T) &\leq \mathcal{L}(\theta_0) - \frac{T}{2\beta} \min_{1 \leq t < T} \|\nabla \mathcal{L}(\theta_t)\|^2 \\ \mathcal{L}(\theta^*) &\leq \mathcal{L}(\theta_0) - \frac{T}{2\beta} \min_{1 \leq t < T} \|\nabla \mathcal{L}(\theta_t)\|^2 \quad \text{par définition de } \theta^* \\ \min_{1 \leq t < T} \|\nabla \mathcal{L}(\theta_t)\|^2 &\leq \frac{2\beta}{T} (\mathcal{L}(\theta_0) - \mathcal{L}(\theta^*))\end{aligned}$$

DESCENTE DE GRADIENT

DESCENTE DE GRADIENT STOCHASTIQUE

On cherche à minimiser une fonction de perte \mathcal{L} qui s'écrit très souvent en Machine Learning comme la somme d'une fonction sur chaque observations du dataset. On note \hat{y}_i la prédiction d'un algorithme pour l'observation x_i avec $i \leq n$ l'index de l'observation i . Pour un problème de **régression** et de **classification**, on a par exemple :

$$\begin{aligned}\mathcal{L}(\theta) &= \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2 \\ &= \frac{1}{n} \sum_{i=1}^n \ell_i(\theta)\end{aligned}$$

$$\begin{aligned}\mathcal{L}(\theta) &= -\frac{1}{n} \sum_{i=1}^n y_i \ln(\hat{y}_i) + (1 - y_i) \ln(1 - \hat{y}_i) \\ &= -\frac{1}{n} \sum_{i=1}^n \ell_i(\theta)\end{aligned}$$

Ainsi, pour une unique mise à jour, on doit appliquer n fonctions ℓ . La descente de gradient stochastique consiste à sélectionner aléatoirement un index i_t et mettre à jour les poids avec uniquement cette observations. La descente de gradient devient alors :

$$\theta_{t+1} = \theta_t - \eta_t \nabla \ell_{i_t}(\theta_t)$$

DESCENTE DE GRADIENT

DESCENTE DE GRADIENT STOCHASTIQUE

Notons que la descente de gradient stochastique n'est pas vraiment une *descente* : nous ne pourrons garantir une descente qu'en espérance. On suppose que \mathcal{L} est différentiable et β -smooth, qu'il existe un minimum pour cette fonction et que chaque ℓ_i soit différentiable continue.

Alors, puisque \mathcal{L} est β -smooth :

$$\mathcal{L}(\theta_{t+1}) \leq \mathcal{L}(\theta_t) + \langle \nabla \mathcal{L}(\theta_t), \theta_{t+1} - \theta_t \rangle + \frac{\beta}{2} \|\theta_{t+1} - \theta_t\|^2$$

$\eta_t \nabla \ell_{i_t}(\theta_t)$
↓

Ainsi, en espérance sur le choix aléatoire de i_t , on a :

$$\mathbb{E}[\mathcal{L}(\theta_{t+1})] \leq \mathcal{L}(\theta_t) + \eta_t \langle \nabla \mathcal{L}(\theta_t), \mathbb{E}[\nabla \ell_{i_t}(\theta_t)] \rangle + \frac{\beta \eta_t^2}{2} \mathbb{E}[\|\nabla \ell_{i_t}(\theta_t)\|^2]$$

Cela ne nous assure pas pour autant une garantie de descente en espérance. Nous devons faire des hypothèses supplémentaires.

DESCENTE DE GRADIENT

DESCENTE DE GRADIENT STOCHASTIQUE

Hypothèse 1

Un index i_t d'une mise à jour t est tiré selon :

- ▶ i_t ne dépend pas des index i_0, \dots, i_{t-1}
- ▶ $\nabla \ell_{i_t}(\theta_t)$ est un estimateur non biaisé de $\nabla \mathcal{L}(\theta_t)$
- ▶ $\mathbb{E} \left[\|\nabla \ell_{i_t}(\theta_t)\|^2 \right] \leq \sigma^2 + \|\mathcal{L}(\theta)\|^2$

Si les indices sont tirés uniformément alors les deux premières hypothèses sont vérifiées. Pour le troisième point, s'il existe $M > 0$ tel que pour tout itérations t on a $\|\nabla \ell_{i_t}(\theta_t)\| \leq M$, alors il est vérifié.

Dans ce cas, on a une garantie de descente en espérance :

$$\mathbb{E} [\mathcal{L}(\theta_{t+1})] \leq \mathcal{L}(\theta_t) - \left(\eta_t - \frac{\beta \eta_t^2}{2} \right) \|\nabla \mathcal{L}(\theta_t)\|^2 + \frac{\beta \eta_t^2}{2} \sigma^2 \quad (1)$$

En supposant que $\eta_t \leq \frac{1}{\beta}$.

DESCENTE DE GRADIENT

SGD : GARANTIE DE CONVERGENCE

On se place dans le cadre d'une optimisation d'une fonction de perte \mathcal{L} non convexe, mais β -smooth. On conserve les hypothèses (1) précédentes. On note $\theta^* = \arg \min_{\theta \in \mathbb{R}^d} \mathcal{L}(\theta)$

Théorème 3 (Learning rate fixe)

On considère une descente de gradient stochastique avec $\eta_t = \eta$ tel que $\eta \in \left] 0, \frac{1}{\beta} \right]$. Alors, pour tout $T \geq 1$:

$$\mathbb{E} \left[\frac{1}{T} \sum_{t=0}^{T-1} \|\nabla \mathcal{L}(\theta_t)\|^2 \right] \leq \eta \beta \sigma^2 + \frac{2(\mathcal{L}(\theta_0) - \mathcal{L}(\theta^*))}{\eta T}$$

On a donc que $\lim_{T \rightarrow +\infty} \mathbb{E} \left[\min_{0 \leq t \leq T-1} \|\nabla \mathcal{L}(\theta_t)\|^2 \right] \in [0, \eta \beta \sigma^2]$: le bruit nous empêche de converger vers un point de gradient nul. Ce n'est pas non plus une garantie de convergence vers le minimum global.

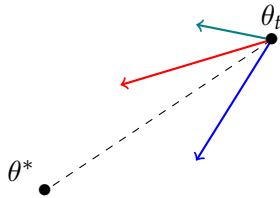
DESCENTE DE GRADIENT

SGD PAR MINI-BATCH

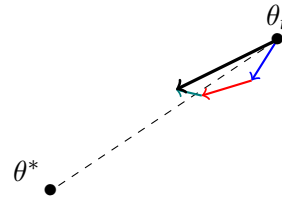
Afin d'obtenir des convergences plus proche du minimum, nous pouvons considérer n_B observations puis mettre à jour. Si $n_B = 1$ on obtient la descente de gradient stochastique et si $n_B = n$ on retrouve la descente de gradient classique. La descente de gradient par mini-batch est donc un compromis entre les deux descentes présentées.

On note \mathcal{B}_t l'ensemble des index choisis aléatoirement tel que $|\mathcal{B}_t| = n_B$, on a :

$$\theta_{t+1} = \theta_t - \eta_t \left(\frac{1}{n_B} \sum_{i \in \mathcal{B}_t} \nabla \ell_i(\theta_t) \right)$$



(a) Calcul des $-\nabla \ell_i(\theta_t)$ pour un batch \mathcal{B}_t



(b) Mise à jour des poids

DESCENTE DE GRADIENT

SGD PAR MINI-BATCH : RÉDUCTION DE VARIANCE

On conserve les hypothèses (1) que l'on a faite sur la fonction de perte et l'aléatoire de tirage des index dont les deux dernières conditions sont :

- ▶ $\nabla \ell_{i_t}(\theta_t)$ est un estimateur non biaisé de $\nabla \mathcal{L}(\theta_t)$
- ▶ $\mathbb{E} \left[\|\nabla \ell_{i_t}(\theta_t)\|^2 \right] \leq \sigma^2 + \|\mathcal{L}(\theta)\|^2$

On obtient le résultat suivant :

Proposition 1 (Variance)

La variance d'une estimation par descente de gradient stochastique par mini-batch utilisant n_B échantillons avec remise vérifie :

$$\mathbb{E} \left[\left\| \frac{1}{n_B} \sum_{i \in \mathcal{B}_t} \nabla \ell_i(\theta_t) \right\|^2 \right] \leq \frac{\sigma^2}{n_B} + \|\nabla \mathcal{L}(\theta_t)\|^2$$

DESCENTE DE GRADIENT

SGD PAR MINI-BATCH : GARANTIE DE CONVERGENCE

A l'aide de la proposition 1 nous pouvons déduire un résultat similaire au théorème 3 qui le généralise.

Théorème 4 (Learning rate fixe pour SGD par mini-batch)

On considère une descente de gradient stochastique par mini-batch avec remise de taille n_B avec $\eta_t = \eta$ tel que $\eta \in \left] 0, \frac{1}{\beta} \right]$. Alors, pour tout $T \geq 1$:

$$\mathbb{E} \left[\frac{1}{T} \sum_{t=0}^{T-1} \|\nabla \mathcal{L}(\theta_t)\|^2 \right] \leq \frac{\eta \beta \sigma^2}{n_B} + \frac{2(\mathcal{L}(\theta_0) - \mathcal{L}(\theta^*))}{\eta T}$$

On obtient alors que $\lim_{T \rightarrow +\infty} \mathbb{E} \left[\min_{0 \leq t \leq T-1} \|\nabla \mathcal{L}(\theta_t)\|^2 \right] \in \left[0, \frac{\eta \beta \sigma^2}{n_B} \right]$.

Il est important de noter que l'on ne traite que du cas où le mini-batch est construit avec remise et que l'on a supposé des indépendances, distributions identiques et des estimations non biaisées. Il est difficile de s'en assurer en pratique.

DESCENTE DE GRADIENT

DESCENTE AVEC MOMENTUM

Paramètre du *momentum*

$$\begin{cases} v_{t+1} = \gamma_t v_t + (1 - \gamma_t) \nabla \mathcal{L}(\theta_t) \\ \theta_{t+1} = \theta_t - \eta_t v_{t+1} \end{cases} \quad (2)$$

Vecteur de *vélocité*

Avec cette version, on conserve dans la mise à jour des poids la *tendance* de déplacement des poids dans l'espace des paramètres pour accélérer la descente. A noter que la valeur du momentum γ_t doit être dans l'intervalle $[0, 1]$.

La descente de gradient avec momentum peut parfois *rater* le minimum et faire machine arrière. Ce phénomène peut être mitigé à l'aide d'un choix précis du *learning rate*. Cet hyper-paramètre est de loin le paramètre le plus sensible sur l'ensemble des schémas que nous présenterons

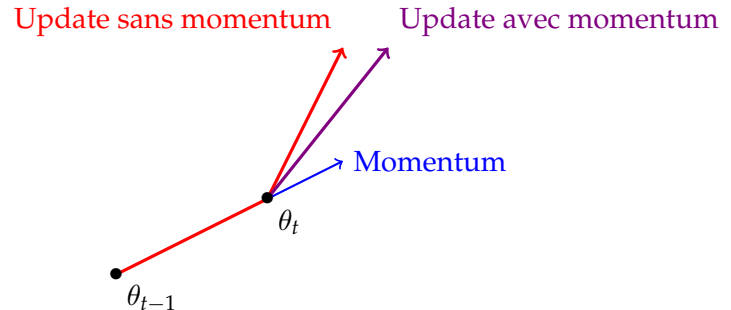


Figure – Effet du momentum sur la mise à jour des poids

DESCENTE DE GRADIENT

ACCÉLÉRATION DE NESTEROV

Une autre manière d'accélérer la descente de gradient et de suivre la variante proposée par [Nesterov, 1983] :

$$\theta_{t+1} = \theta_t + \gamma_t(\theta_t - \theta_{t-1}) - \eta_t \nabla \mathcal{L}(\theta_t + \gamma_t(\theta_t - \theta_{t-1}))$$

Notons que cette fois, on calcule le gradient *un peu plus loin* que le point θ_t où l'on se situe. Visuellement :

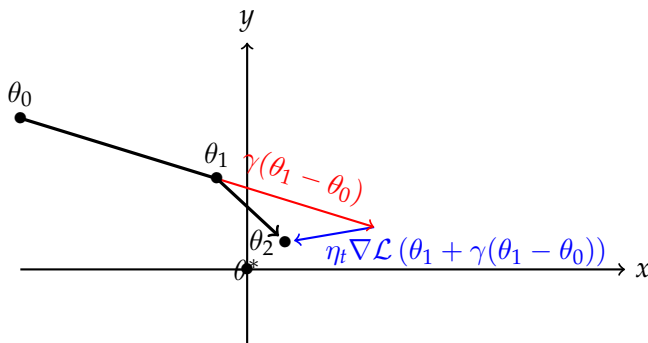


Figure – Accélération de Nesterov avec $f(x, y) = x^2 + \frac{1}{2}y^2$

DESCENTE DE GRADIENT

ACCÉLÉRATION DE NESTEROV

L'intérêt de l'accélération de Nesterov est qu'on atteint des garanties de convergence plus rapide :

Théorème 5 (Accélération de Nesterov pour une fonction convexe β -smooth)

Soit $f : \mathbb{R}^d \rightarrow \mathbb{R}$ une fonction convexe et β -smooth ayant un minimum $\mathcal{L}(\theta^*) = \arg \min_{\theta \in \mathbb{R}^d} \mathcal{L}(\theta)$. On suppose

que $\eta_t = \frac{1}{\beta}$ et que γ_t est calculé comme :

$$y_{t+1} = \frac{1}{2} \left(1 + \sqrt{1 + 4y_t^2} \right) \quad \text{avec } y_0 = 0, \text{ et } \gamma_t = \frac{y_t - 1}{y_{t+1}}$$

Alors :

$$\forall T \geq 1, \quad \mathcal{L}(\theta_T) - \mathcal{L}(\theta^*) \leq \mathcal{O}\left(\frac{1}{T^2}\right)$$

On peut avoir une preuve de ce résultat dans [Bubeck, 2015].

DESCENTE DE GRADIENT AVEC PAS ADAPTATIF

1	Rappels	1
2	Descente de gradient	5
2.1	Descente de gradient stochastique	11
2.2	SGD par mini-batch	15
2.3	Descente avec momentum	18
3	Descente de gradient avec pas adaptatif	21
3.1	AdaGrad	22
3.2	RMSProp	24
3.3	Adam	25
4	Descente de gradient et régularisation	29
5	Comment choisir le pas de descente?	32

DESCENTE DE GRADIENT AVEC PAS ADAPTATIF

ADAGRAD : UNE PREMIÈRE RÉPONSE

$$\begin{cases} g_{t+1} = g_t + \nabla \mathcal{L}(\theta_t)^2 & \text{avec } g_0 = 0 \\ \theta_{t+1} = \theta_t - \eta \frac{\nabla \mathcal{L}(\theta_t)}{\sqrt{g_{t+1}} + \varepsilon} \end{cases} \quad (\text{AdaGrad, 2011})$$

Nombre pour éviter des problèmes numériques

Si le gradient a une magnitude plus importante dans une direction, elle sera privilégiée pendant l'optimisation.

AdaGrad [Duchi et al., 2011] propose de normaliser le gradient pour avoir un apprentissage *uniforme*.

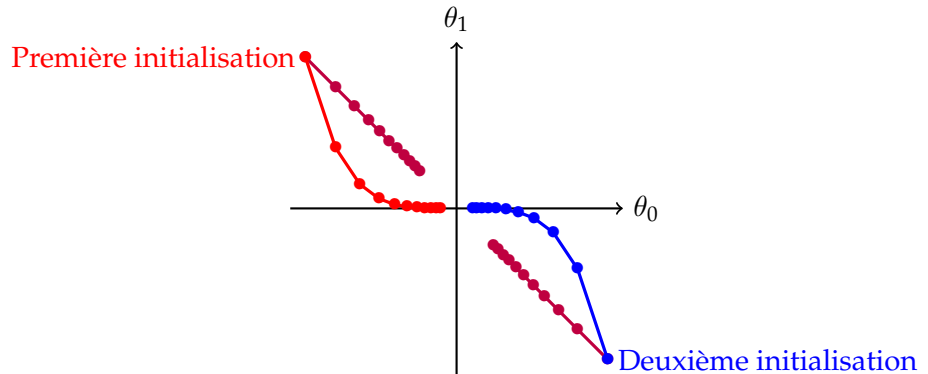


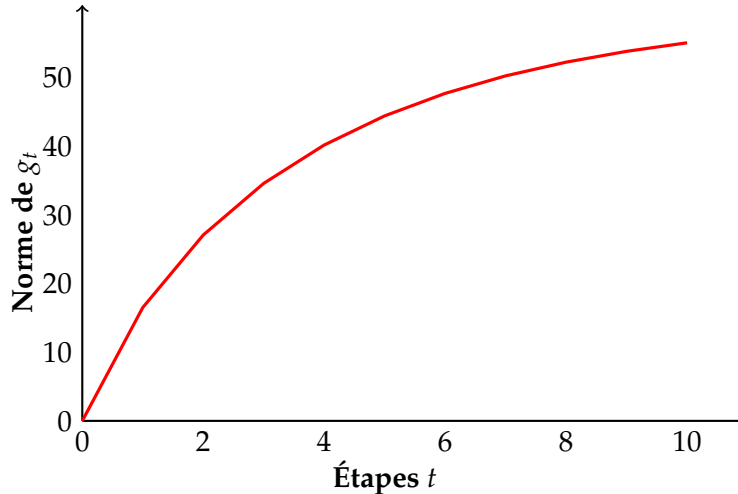
Figure – 10 étapes de descente de gradient pour la fonction $f(x, y) = x^2 + 3y^2$ avec **AdaGrad** pour deux initialisations différentes

DESCENTE DE GRADIENT AVEC PAS ADAPTATIF

ADAGRAD : RÉPONSE IMPARFAITE

$$\begin{cases} g_{t+1} = g_t + \nabla \mathcal{L}(\theta_t)^2 & \text{avec } g_0 = 0 \\ \theta_{t+1} = \theta_t - \eta \frac{\nabla \mathcal{L}(\theta_t)}{\sqrt{g_{t+1}} + \epsilon} \end{cases} \quad (\text{AdaGrad, 2011})$$

Cependant avec ce schéma, AdaGrad tend à avoir un apprentissage qui ralentit au fil de l'entraînement puisque $(g_t)_t$ est croissante.



DESCENTE DE GRADIENT AVEC PAS ADAPTATIF

RMSPROP : DEUXIÈME RÉPONSE

Contrôle la mémoire des précédents gradients, $\alpha \geq 0$

$$\begin{cases} g_{t+1} = \alpha g_t + (1 - \alpha) \nabla \mathcal{L}(\theta_t)^2 & \text{avec } g_0 = 0 \\ \theta_{t+1} = \theta_t - \eta \frac{\nabla \mathcal{L}(\theta_t)}{\sqrt{g_{t+1} + \epsilon}} \end{cases} \quad (\text{RMSProp, 2012})$$

RMSProp [Hinton et al., 2012] cherche aussi à permettre un apprentissage uniforme dans toutes les directions, mais le fait avec une moyenne mobile exponentielle. Cela permet à $(g_t)_t$ de pouvoir décroître.

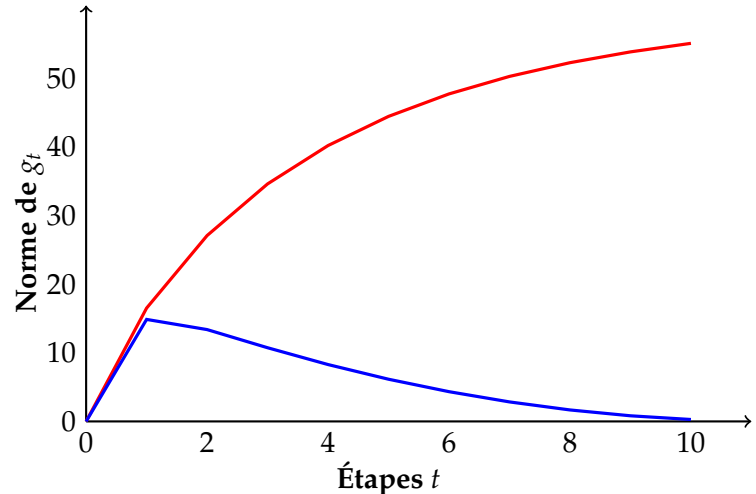


Figure – Norme de g_t pour **RMSProp** et **AdaGrad**

DESCENTE DE GRADIENT AVEC PAS ADAPTATIF

ADAM : COMBINER ADA GRAD ET RMSPROP

$$\left\{ \begin{array}{l} m_{t+1} = \beta_1 m_t + (1 - \beta_1) \nabla \mathcal{L}(\theta_t) \quad \text{avec } m_0 = 0 \\ \hat{m}_{t+1} = \frac{m_{t+1}}{1 - \beta_1^{t+1}} \\ v_{t+1} = \beta_2 v_t + (1 - \beta_2) \nabla \mathcal{L}(\theta_t)^2 \quad \text{avec } v_0 = 0 \\ \hat{v}_{t+1} = \frac{v_{t+1}}{1 - \beta_2^{t+1}} \\ \theta_{t+1} = \theta_t - \eta \frac{\hat{m}_{t+1}}{\sqrt{\hat{v}_{t+1} + \epsilon}} \end{array} \right. \quad (\text{Adam, 2014})$$

Adam [Kingma and Ba, 2015] s'est rapidement imposé comme un excellent schéma d'optimisation pour l'apprentissage d'un réseau de neurones. Il ressemble à une version accélérée de RMSProp, étudions plus en détail ses moyennes mobiles.

DESCENTE DE GRADIENT AVEC PAS ADAPTATIF

ADAM : CORRECTION DU BIAIS DE L'INITIALISATION

Si l'on reprend une partie d'Adam, par exemple pour la suite $(v_t)_{t \in \mathbb{N}}$:

$$\begin{cases} v_{t+1} &= \beta_2 v_t + (1 - \beta_2) \nabla \mathcal{L}(\theta_t)^2 \quad \text{avec } v_0 = 0 \\ \hat{v}_{t+1} &= \frac{v_{t+1}}{1 - \beta_2^{t+1}} \end{cases}$$

On remarque que :

$$\begin{aligned} v_t &= \beta_2 v_{t-1} + (1 - \beta_2) \mathcal{L}(\theta_t)^2 \\ &= \beta_2^2 v_{t-2} + (1 - \beta_2) \left[\beta_2 \mathcal{L}(\theta_{t-1})^2 + \mathcal{L}(\theta_t)^2 \right] \\ &= \beta_2^t v_0 + (1 - \beta_2) \sum_{i=1}^t \beta_2^{t-i} \mathcal{L}(\theta_i)^2 \end{aligned}$$

Si l'on suppose que les gradients sont identiquement distribués, on obtient :

$$\mathbb{E}[v_t] = (1 - \beta_2) \mathbb{E} \left[\mathcal{L}(\theta_t)^2 \right] \sum_{i=1}^t \beta_2^{t-i} \iff \mathbb{E}[v_t] = \mathbb{E} \left[\mathcal{L}(\theta_t)^2 \right] (1 - \beta_2^t)$$

D'où on déduit la correction apportée par la suite $(\hat{v}_t)_{t \in \mathbb{N}}$. Le calcul est le même pour la suite $(\hat{m}_t)_{t \in \mathbb{N}}$.

DESCENTE DE GRADIENT AVEC PAS ADAPTATIF

ADAM : INTERPRÉTATION DU LEARNING RATE

Puisque les suites $(\hat{v}_t)_{t \in \mathbb{N}}$ et $(\hat{m}_t)_{t \in \mathbb{N}}$ sont des estimateurs non biaisés de $\mathbb{E}[\mathcal{L}(\theta_t)]$ et $\mathbb{E}[\mathcal{L}(\theta_t)^2]$, on doit pouvoir obtenir une information sur la mise à jour des paramètres.

On considère une variable aléatoire X telle que $\mathbb{E}[X]$ existe et $\mathbb{E}[X^2]$ existe et n'est pas nulle. Alors, puisque la fonction $x \mapsto x^2$ est une fonction convexe, avec l'inégalité de Jensen on a :

$$\mathbb{E}[X^2] \geq \mathbb{E}[X]^2 \iff \frac{\mathbb{E}[X]^2}{\mathbb{E}[X^2]} \leq 1 \iff \frac{|\mathbb{E}[X]|}{\sqrt{\mathbb{E}[X^2]}} \leq 1$$

Dans notre cas, avec les mêmes hypothèses pour $\mathcal{L}(\theta_t)$ que pour X , on définit $\Delta_t = \theta_t - \theta_{t-1}$ et on a :

$$\begin{aligned} \Delta_t &= \eta \frac{\hat{m}_t}{\sqrt{\hat{v}_t}} \quad \text{donc} \quad |\Delta_t| = \eta \frac{|\hat{m}_t|}{\sqrt{\hat{v}_t}} \\ |\Delta_t| &\sim \eta \frac{|\mathbb{E}[\mathcal{L}(\theta_t)]|}{\sqrt{\mathbb{E}[\mathcal{L}(\theta_t)^2]}} \quad \text{ainsi} \quad |\Delta_t| \lesssim \eta \end{aligned}$$

DESCENTE DE GRADIENT AVEC PAS ADAPTATIF

QUEL OPTIMIZER CHOISIR ?

De nombreuses autres possibilités existent, et chaque année plusieurs optimiseurs sont proposés. C'est pourquoi [Schmidt et al., 2021] propose de comparer équitablement quinze optimiseurs sur différentes tâches. Il en ressort deux informations :

- ▶ Adam est l'optimiseur le plus performant sur le plus de tâches, sans pour autant être clairement supérieur. RMSProp et l'accélération de Nesterov restent des alternatives très intéressantes.
- ▶ Choisir les meilleurs hyperparamètres est tout aussi efficace voire plus efficace que de changer d'optimiseur.

Perhaps the most important takeaway from our study is hidden in plain sight : the field is in danger of being drowned by noise. Different optimizers exhibit a surprisingly similar performance distribution compared to a single method that is re-tuned or simply re-run with different random seeds. It is thus questionable how much insight the development of new methods yields, at least if they are conceptually and functionally close to the existing population.

— Robin Schmidt, Frank Schneider et Philipp Hennig (2021)

DESCENTE DE GRADIENT ET RÉGULARISATION

1	Rappels	1
2	Descente de gradient	5
2.1	Descente de gradient stochastique	11
2.2	SGD par mini-batch	15
2.3	Descente avec momentum	18
3	Descente de gradient avec pas adaptatif	21
3.1	AdaGrad	22
3.2	RMSProp	24
3.3	Adam	25
4	Descente de gradient et régularisation	29
5	Comment choisir le pas de descente?	32

DESCENTE DE GRADIENT ET RÉGULARISATION

PROBLÈME DE LA RÉGULARISATION

[Krogh and Hertz, 1991] montre qu'avoir un réseau de neurones avec une magnitude de poids faible permet de limiter le sur-apprentissage.

Une manière classique pour régulariser un réseau de neurones est de modifier la fonction de perte que l'on optimise. Le plus souvent on exploite la régularisation \mathcal{L}_2 :

$$\mathcal{L}_\lambda(w) = \mathcal{L}(w) + \frac{\lambda}{2} \|w\|_2^2 \quad \text{avec } \lambda \geq 0$$

Dans le cas d'une descente de gradient classique, cela donne le schéma d'optimisation :

$$\begin{aligned} w_{t+1} &= w_t - \eta_t \nabla \mathcal{L}_\lambda(w_t) && \text{par définition de la descente de gradient} \\ &= w_t - \eta_t \nabla \mathcal{L}(w_t) - \eta_t \lambda w_t && \text{par définition de } \mathcal{L}_\lambda \end{aligned}$$

Le terme supplémentaire $\eta_t \lambda w_t$ est appelé le weight decay.

DESCENTE DE GRADIENT ET RÉGULARISATION

PROBLÈME DE LA RÉGULARISATION

Mais si l'on considère une descente de gradient avec momentum :

$$\begin{aligned}v_{t+1} &= \beta v_t + (1 - \beta) \nabla \mathcal{L}_\lambda(w_t) && \text{par définition} \\ &= \beta v_t + (1 - \beta) [\nabla \mathcal{L}(w_t) + \lambda w_t] && \text{par définition de } \mathcal{L}_\lambda\end{aligned}$$

$$\begin{aligned}w_{t+1} &= w_t - \eta_t (\beta v_t + (1 - \beta) [\nabla \mathcal{L}(w_t) + \lambda w_t]) \\ &= w_t - \eta_t \beta v_t - \eta_t (1 - \beta) \nabla \mathcal{L}(w_t) - \eta_t (1 - \beta) \lambda w_t\end{aligned}$$

Autrement dit, nous avons une propagation de la régularisation dans le schéma de descente, ce qui n'est pas souhaité. Pour conserver le comportement observé dans une descente de gradient classique, [Loshchilov and Hutter, 2019] propose simplement de modifier le schéma de descente plutôt que la fonction de perte.

Ainsi, dès lors que l'on utilise une méthode *adaptive*, il faut préférer le **weight decay** à la régularisation \mathcal{L}_2 . L'omniprésence d'Adam dans la littérature a amené à la création d'AdamW qui est Adam intégrant le weight decay.

COMMENT CHOISIR LE PAS DE DESCENTE ?

1	Rappels	1
2	Descente de gradient	5
2.1	Descente de gradient stochastique	11
2.2	SGD par mini-batch	15
2.3	Descente avec momentum	18
3	Descente de gradient avec pas adaptatif	21
3.1	AdaGrad	22
3.2	RMSProp	24
3.3	Adam	25
4	Descente de gradient et régularisation	29
5	Comment choisir le pas de descente ?	32

COMMENT CHOISIR LE PAS DE DESCENTE ?

IL FAUT UN ÉCHÉANCIER POUR SGD

On se place dans le cadre d'une optimisation d'une fonction de perte \mathcal{L} non convexe, mais β -smooth. On conserve les hypothèses (1) précédentes. On note $\theta^* = \arg \min_{\theta \in \mathbb{R}^d} \mathcal{L}(\theta)$

Théorème 6 (Pas de descente décroissant)

On considère une descente de gradient stochastique avec η_t une suite décroissante telle que $\eta_t \in \left] 0, \frac{1}{\beta} \right]$ et que :

$$\sum_{t=0}^{+\infty} \eta_t = +\infty \quad \text{et} \quad \sum_{t=0}^{+\infty} \eta_t^2 < +\infty$$

Alors, pour tout $T \geq 1$:

$$\lim_{T \rightarrow +\infty} \mathbb{E} \left[\frac{1}{\sum_{t=0}^{T-1} \eta_t} \sum_{t=0}^{T-1} \eta_t \|\nabla \mathcal{L}(\theta_t)\|^2 \right] = 0$$

Nous n'avons plus cette fois une convergence dans un intervalle proche du minimum, mais une convergence vers un point de gradient nul. D'où la nécessité d'avoir un échéancier pour le choix du learning rate. De même pour Nesterov, nous avons besoin d'un échéancier pour le momentum.

COMMENT CHOISIR LE PAS DE DESCENTE ?

ÉCHÉANCIERS CLASSIQUES

Les succès de l'ensemble des schémas précédents sont conditionnés à un bon choix du learning rate η , même dans le cas adaptatif. Il est possible de le conserver constant pour la totalité de l'entraînement mais il existe des **échéanciers** pour modifier η au cours de l'entraînement.

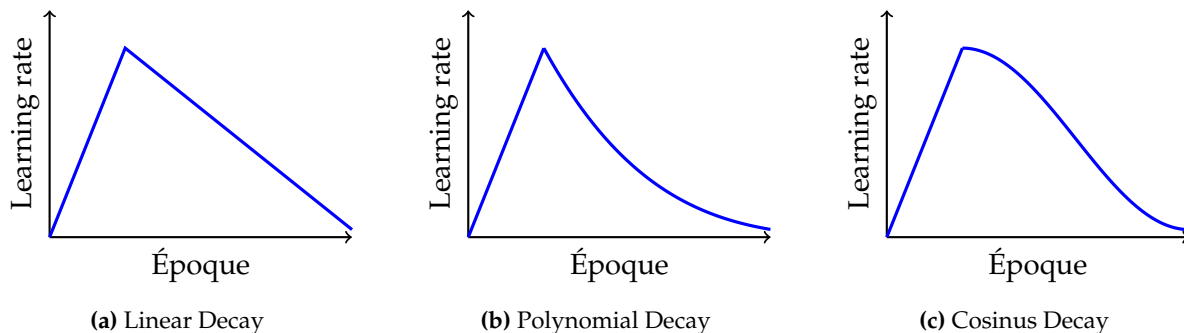












Figure – Exemples d'échéanciers du learning rate avec échauffements

L'échéancier cosinus est devenu standard suite à l'utilisation presque exclusive de cet échéancier par les *Large Language Models*. Cela fait suite à une étude menée par Deepmind [Hoffmann et al., 2022] qui vise à définir les *Scaling laws*, et qui traite en particulier des échéanciers cycliques cosinus. La conclusion a donc été de ne l'utiliser que sur un seul cycle.




BIBLIOGRAPHIE I

-  Bubeck, S. (2015).
Convex optimization : Algorithms and complexity.
Foundations and Trends® in Machine Learning.
-  Chen, X., Liang, C., Huang, D., Real, E., Wang, K., Liu, Y., Pham, H., Dong, X., Luong, T., Hsieh, C.-J., et al. (2023).
Symbolic discovery of optimization algorithms.
arXiv preprint arXiv :2302.06675.
-  Duchi, J., Hazan, E., and Singer, Y. (2011).
Adaptive subgradient methods for online learning and stochastic optimization.
Journal of machine learning research.
-  Hinton, G., Srivastava, N., and Swersky, K. (2012).
Neural networks for machine learning lecture 6a overview of mini-batch gradient descent.
Cited on.
-  Hoffmann, J., Borgeaud, S., Mensch, A., Buchatskaya, E., Cai, T., Rutherford, E., Casas, D. d. L., Hendricks, L. A., Welbl, J., Clark, A., et al. (2022).
Training compute-optimal large language models.
arXiv preprint arXiv :2203.15556.

BIBLIOGRAPHIE II

-  Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., and Amodei, D. (2020).
Scaling laws for neural language models.
arXiv preprint arXiv :2001.08361.
-  Kingma, D. P. and Ba, J. (2015).
Adam : A method for stochastic optimization.
In International Conference on Learning Representations (ICLR).
-  Krogh, A. and Hertz, J. (1991).
A simple weight decay can improve generalization.
Advances in neural information processing systems.
-  Loshchilov, I. and Hutter, F. (2016).
Sgdr : Stochastic gradient descent with warm restarts.
arXiv preprint arXiv :1608.03983.
-  Loshchilov, I. and Hutter, F. (2019).
Decoupled weight decay regularization.
In International Conference on Learning Representations (ICLR).

BIBLIOGRAPHIE III

-  [Nesterov, Y. \(1983\).](#)
A method of solving a convex programming problem with convergence rate $o(1/k^2)$.
Soviet Mathematics Doklady.
-  [Schmidt, R. M., Schneider, F., and Hennig, P. \(2021\).](#)
Descending through a crowded valley-benchmarking deep learning optimizers.
In *International Conference on Machine Learning*. PMLR.
-  [Smith, L. N. \(2017\).](#)
Cyclical learning rates for training neural networks.
In *2017 IEEE winter conference on applications of computer vision (WACV)*, pages 464–472. IEEE.

ANNEXE

FONCTION LIPSCHITZIENNE ET β -SMOOTH

On considère une fonction $f : \mathcal{D} \rightarrow \mathbb{R}$ une fonction avec $\mathcal{D} \subset \mathbb{R}^d$.

Définition 1 (Fonction Lipschitzienne)

On dit que f est *L-Lipschitzienne* si il existe un nombre L tel que :

$$\forall u, v \in \mathcal{D}, \quad |f(u) - f(v)| \leq L \|u - v\|$$

Définition 2 (Fonction β -smooth)

On suppose que f est différentiable. On dit que f est *β -smooth* si :

$$\forall u, v \in \mathcal{D}, \quad f(v) \leq f(u) + \langle \nabla f(u), v - u \rangle + \frac{\beta}{2} \|v - u\|^2$$


On peut montrer que f est une fonction *β -smooth* si, et seulement si, le gradient de f est *β -Lipschitzienne*.

ANNEXE : COMPLÉMENTS

OPTIMIZER LION

[Chen et al., 2023] Une équipe de chercheur de Google et UCLA propose un nouvel optimizer nommé Lion (EvoLved Sign Momentum) [Chen et al., 2023]. Il est le fruit d'une recherche symbolique du meilleur optimizer possible. AdamW est le point de départ de la recherche. Cet optimizer s'écrit avec nos notations sous la forme :

$$\begin{cases} c_{t+1} &= \beta_1 m_t + (1 - \beta_1) \nabla \mathcal{L}(\theta_t) & \text{avec } m_0 = 0 \\ m_{t+1} &= \beta_2 m_t + (1 - \beta_2) \nabla \mathcal{L}(\theta_t) \\ \theta_{t+1} &= \theta_t - \eta_t (\text{sign}(c_t) + \lambda \theta_t) \end{cases} \quad (\text{Lion, 2023})$$

Weight decay, $\lambda \geq 0$

Lion ne travaille pas avec l'amplitude des gradients mais seulement le signe. On a également deux hyperparamètre β_1 et β_2 qui n'ont pas un rôle aussi clair que pour Adam. L'approche globale reste très proche de celle d'Adam et le comportement important du weight decay a été conservé.

Des améliorations sensibles sont décrites dans l'article, mais la communauté n'a pas trouvé les mêmes résultats de manière consistante, nous renvoyant à la conclusion de [Schmidt et al., 2021].

L'approche est cependant suffisamment innovante et l'utilisation uniquement du signe suffisent, à notre avis, pour que l'on connaisse ce travail.

ANNEXE : COMPLÉMENTS

ÉCHÉANCIERS CYCLIQUES

[Smith, 2017] propose de définir des échéanciers cycliques en partant du postulat suivant :

The essence of this learning rate policy comes from the observation that increasing the learning rate might have a short term negative effect and yet achieve a longer term beneficial effect

— Leslie Smith (2015)

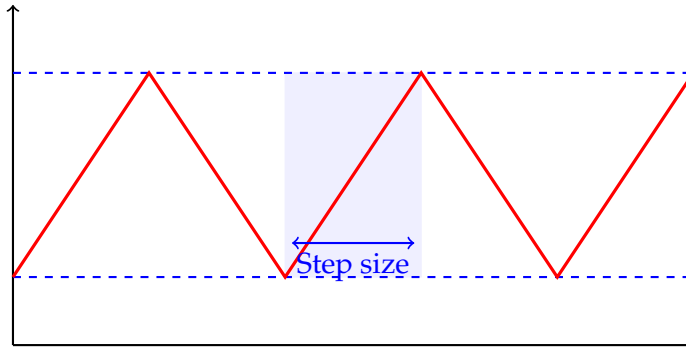


Figure – Échéancier triangulaire pour le learning rate

Nous avons 3 hyperparamètres à régler : les deux bornes et la longueur d'un cycle. Nous pouvons également introduire un effet de rétrécissement à chaque époque.

L'article propose de fixer la longueur d'un cycle comme 2 à 10 fois le nombre d'itérations qui seront réalisées à chaque époque. Concernant les deux bornes c'est plus expérimental.

ANNEXE : COMPLÉMENTS

ÉCHÉANCIERS CYCLIQUES

[Loshchilov and Hutter, 2016] poursuit la même idée et propose de ne pas modifier l'échéancier de manière continue mais de faire un *restart*.

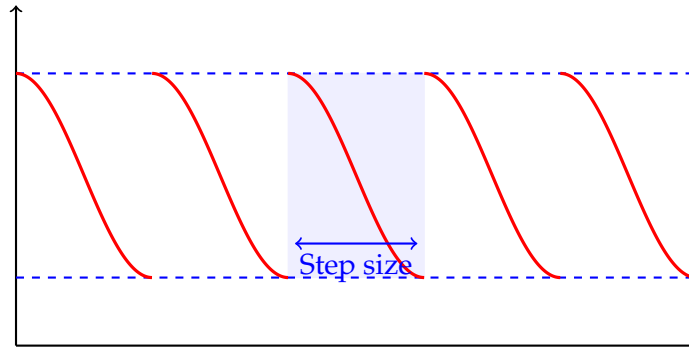


Figure – Échéancier cosinus pour le learning rate avec restart

Pour [Kaplan et al., 2020] une décroissance jusqu'à zéro n'est pas vraiment un problème et ne semble pas changer énormément l'apprentissage tant que l'évolution du learning rate n'est pas trop brusque et que les valeurs ne sont pas trop petites.

L'optimal (selon Deepmind [Hoffmann et al., 2022]) semble être d'adopter une période de chauffe du modèle en passant d'un learning rate très faible (de l'ordre 10^{-7}) au plus haut learning rate que l'on souhaite, puis de le réduire de 10% avec un cosinus. La taille d'un cycle étant celle de l'entraînement, voire légèrement plus.

ANNEXE : PREUVE

THÉORÈME 3 : CONVERGENCE POUR SGD AVEC LEARNING RATE FIXE

Soit $T \leq 1$ et $t \leq T$ une étape. L'inéquation 1 se réécrit comme :

$$\begin{aligned}\mathbb{E}[\mathcal{L}(\theta_{t+1}) - \mathcal{L}(\theta_t)] &\leq -\left(\eta - \frac{\eta^2\beta}{2}\right) \|\nabla\mathcal{L}(\theta_t)\|^2 + \frac{\eta^2\beta}{2}\sigma^2 \\ &\leq -\frac{\eta}{2}\|\nabla\mathcal{L}(\theta_t)\|^2 + \frac{\eta^2\beta}{2}\sigma^2 \quad \text{car } \eta \in \left]0, \frac{1}{\beta}\right[\end{aligned}$$

Puisque $\mathcal{L}(\theta_t) \geq \mathcal{L}(\theta^*)$ pour toute étape t , en sommant sur toute les étapes, on a en espérance :

$$\begin{aligned}\mathcal{L}(\theta^*) - \mathcal{L}(\theta_0) &\leq \mathbb{E}[\mathcal{L}(\theta_T) - \mathcal{L}(\theta_0)] \leq -\frac{\eta}{2} \sum_{t=0}^{T-1} \mathbb{E}[\|\nabla\mathcal{L}(\theta_t)\|^2] + T\frac{\eta^2\beta}{2}\sigma^2 \\ \mathbb{E}\left[\frac{1}{T} \sum_{t=0}^{T-1} \|\nabla\mathcal{L}(\theta_t)\|^2\right] &\leq \eta\beta\sigma^2 + \frac{2(\mathcal{L}(\theta_0) - \mathcal{L}(\theta^*))}{\eta T} \end{aligned}$$

D'où $\lim_{T \rightarrow +\infty} \mathbb{E}\left[\min_{0 \leq t \leq T-1} \|\nabla\mathcal{L}(\theta_t)\|^2\right] \in [0, \eta\beta\sigma^2]$

ANNEXE : PREUVE

PROPOSITION 1 : RÉDUCTION DE VARIANCE POUR SGD AVEC MINI-BATCH

On rappelle que l'on a fait les hypothèses suivante sur la constitution du mini-batch pour $i \in \mathcal{B}_t$ à une époque t :

- ▶ $\nabla \ell_i(\theta_t)$ est un estimateur non biaisé de $\nabla \mathcal{L}(\theta_t)$
- ▶ $\mathbb{E} \left[\|\nabla \ell_i(\theta_t)\|^2 \right] \leq \sigma^2 + \|\mathcal{L}(\theta_t)\|^2$

On a supposé de plus qu'il y a indépendance et identique distribution pour les $\nabla \ell_i$. Ainsi, on a :

$$\begin{aligned} \mathbb{E} \left[\left\| \frac{1}{n_B} \sum_{i \in \mathcal{B}} \nabla \ell_i(\theta_t) \right\|^2 \right] - \left\| \mathbb{E} \left[\frac{1}{n_B} \sum_{i \in \mathcal{B}} \nabla \ell_i(\theta_t) \right] \right\|^2 &= \frac{1}{n_B} \left(\mathbb{E} \left[\|\nabla \ell_i(\theta_t)\|^2 \right] - \|\mathcal{L}(\theta_t)\|^2 \right) \\ &\leq \frac{\sigma^2}{n_B} \quad \text{avec le second point} \end{aligned}$$

Le premier point nous indique que : $\mathbb{E} \left[\frac{1}{n_B} \sum_{i \in \mathcal{B}} \nabla \ell_i(\theta_t) \right] = \frac{1}{n_B} \sum_{i \in \mathcal{B}} \mathbb{E} [\nabla \ell_i(\theta_t)] = \nabla \mathcal{L}(\theta_t)$

On a donc finalement que :

$$\mathbb{E} \left[\left\| \frac{1}{n_B} \sum_{i \in \mathcal{B}} \nabla \ell_i(\theta_t) \right\|^2 \right] \leq \frac{\sigma^2}{n_B} + \|\mathcal{L}(\theta_t)\|^2$$

ANNEXE : PREUVE

THÉORÈME 4 : CONVERGENCE POUR SGD PAR MINI-BATCH AVEC LEARNING RATE FIXE

La démonstration du théorème 4 est similaire à celle du théorème 3 mais il faut prendre en compte le mini-batch. Puisque \mathcal{L} est β -smooth :

$$\begin{aligned}\mathcal{L}(\theta_{t+1}) &\leq \mathcal{L}(\theta_t) + \langle \nabla \mathcal{L}(\theta_t), \theta_{t+1} - \theta_t \rangle + \frac{\beta}{2} \|\theta_{t+1} - \theta_t\|^2 \\ &\leq \mathcal{L}(\theta_t) - \eta_t \langle \nabla \mathcal{L}(\theta_t), \frac{1}{n_B} \sum_{i \in \mathcal{B}} \nabla \ell_i(\theta_t) \rangle + \frac{\beta \eta_t^2}{2} \left\| \frac{1}{n_B} \sum_{i \in \mathcal{B}} \nabla \ell_i(\theta_t) \right\|^2\end{aligned}$$

Ainsi, en espérance sur le choix aléatoire de \mathcal{B}_t :

$$\mathbb{E}[\mathcal{L}(\theta_{t+1})] \leq \mathcal{L}(\theta_t) - \eta_t \|\nabla \mathcal{L}(\theta_t)\|^2 + \frac{\beta \eta_t^2}{2} \mathbb{E} \left[\left\| \frac{1}{n_B} \sum_{i \in \mathcal{B}_t} \nabla \ell_i(\theta_t) \right\|^2 \right]$$

A l'aide de la proposition 1 on obtient finalement :

$$\mathbb{E}[\mathcal{L}(\theta_{t+1})] \leq \mathcal{L}(\theta_t) - \left(\eta_t - \frac{\beta \eta_t^2}{2} \right) \|\nabla \mathcal{L}(\theta_t)\|^2 + \frac{\beta \eta_t^2}{2n_B} \sigma^2 \quad (3)$$

On utilise l'inéquation 3 comme point de départ à la place de l'inéquation 1, puis *mutatis mutandis* dans la démonstration du théorème 4.