IFT 615 – Intelligence Artificielle

Classification linéaire avec le perceptron

Professeur: Froduald Kabanza

Assistants: D’Jeff Nkashama

Université de Sherbrooke

Sujets couverts

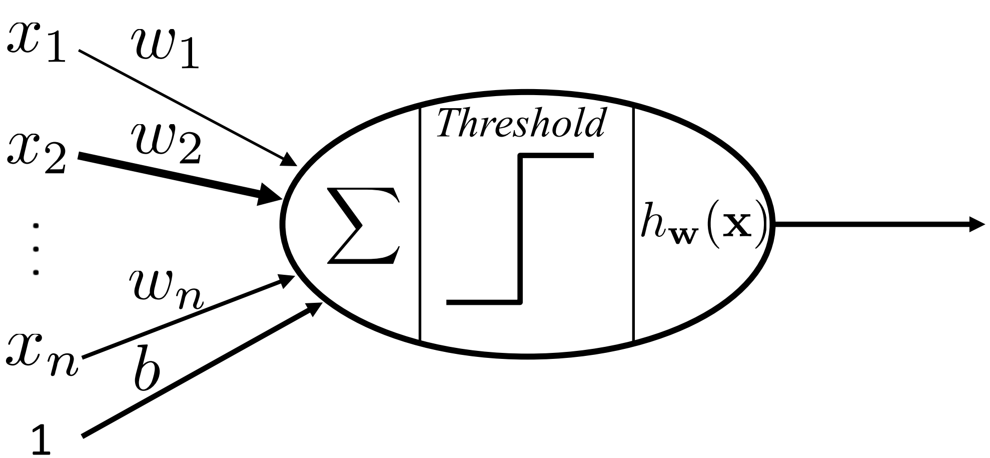
- Classification linéaire avec le perceptron
- Minimisation d'une perte par la descente du gradient

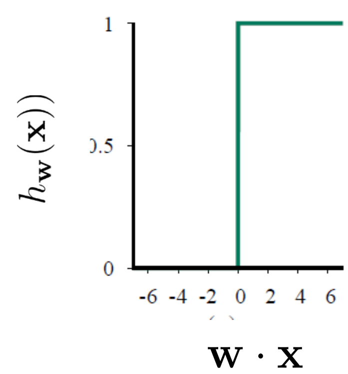
Deuxième algorithme: Perceptron
(Rosenblatt, 1957)

- Un des plus vieux algorithmes de classification
- **Idée**: modéliser la décision à l'aide d'une fonction linéaire, suivi d'un seuil:

$$h_{\mathbf{w}}(\mathbf{x}) = Threshold(\mathbf{w} \cdot \mathbf{x})$$

où $Threshold(z) = 1$ si $z \geq 0$, sinon $Threshold(z) = 0$





- Le **vecteur de poids $\mathbf{w}$** correspond aux **paramètres** du modèle
- On ajoute également un biais b, qui équivaut à ajouter une entrée $x_{n+1} = 1$

Deuxième algorithme: Perceptron
(Rosenblatt, 1957)

- L'algorithme d'apprentissage doit adapter la valeur des paramètres (c'est-à-dire les poids et le biais) de façon à ce que $h_{\mathbf{w}}(\mathbf{x})$ soit la bonne réponse sur les données d'entraînement

- Algorithme du Perceptron:
  1. pour chaque paire $(\mathbf{x}_t, y_t) \in D$
     a. calculer $h_{\mathbf{w}}(\mathbf{x}_t) = Threshold(\mathbf{w} \cdot \mathbf{x}_t)$
     b. si $y_t \neq h_{\mathbf{w}}(\mathbf{x}_t)$
        - $w_i \leftarrow w_i + \alpha(y_t - h_{\mathbf{w}}(\mathbf{x}_t))x_{t,i} \quad \forall i$ (mise à jour des poids et biais)
  2. retourner à 1 jusqu'à l'atteinte d'un critère d'arrêt (nb. maximal d'itérations atteint ou nb. d'erreurs est 0)

- La mise à jour des poids est appelée la **règle d'apprentissage du Perceptron**. La multiplicateur α est appelé le **taux d'apprentissage**

Deuxième algorithme: Perceptron
(Rosenblatt, 1957)

- L'algorithme d'apprentissage doit adapter la valeur des paramètres (c'est-à-dire les poids et le biais) de façon à ce que $h_{\mathbf{w}}(\mathbf{x})$ soit la bonne réponse sur les données d'entraînement

- Algorithme du Perceptron:
  1. pour chaque paire $(\mathbf{x}_t, y_t) \in D$
     a. calculer $h_{\mathbf{w}}(\mathbf{x}_t) = Threshold(\mathbf{w} \cdot \mathbf{x}_t)$
     b. si $y_t \neq h_{\mathbf{w}}(\mathbf{x}_t)$
        - $\mathbf{w} \leftarrow \mathbf{w} + \alpha(y_t - h_{\mathbf{w}}(\mathbf{x}_t))\mathbf{x}_t$ (mise à jour des poids et biais)

        forme vectorielle
  2. retourner à 1 jusqu'à l'atteinte d'un critère d'arrêt (nb. maximal d'itérations atteint ou nb. d'erreurs est 0)

- La mise à jour des poids est appelée la **règle d'apprentissage du Perceptron**. La multiplicateur α est appelé le **taux d'apprentissage**

Exemple

D ensemble entraînement

$\mathbf{x}_t$	y_t
[2,0]	1
[0,3]	0
[3,0]	0
[1,1]	1

- Simulation **avec biais**, $\alpha = 0.1$
- Initialisation : $\mathbf{w} \longleftarrow [0, 0]$, $b = 0.5$
- Paire $(\mathbf{x}_1, y_1)$:
  - $h(\mathbf{x}_1) = Threshold(\mathbf{w} \cdot \mathbf{x}_1 + b) = Threshold(0.5) = 1$
  - puisque $h(\mathbf{x}_1) = y_1$, on ne fait pas de mise à jour de **w** et b

Exemple

D ensemble entraînement

$\mathbf{x}_t$	y_t
[2,0]	1
[0,3]	0
[3,0]	0
[1,1]	1

- Simulation **avec biais**, $\alpha = 0.1$
- Valeur courante : $\mathbf{w} \longleftarrow [0, 0]$, $b = 0.5$
- Paire $(\mathbf{x}_2, y_2)$:
  - $h(\mathbf{x}_2) = \mathit{Threshold}(\ \mathbf{w} \cdot \mathbf{x}_2 + b) = \mathit{Threshold}(0.5) = 1$
  - puisque $h(\mathbf{x}_2) \neq y_2$, on met à jour $\mathbf{w}$ et b
    - » $\mathbf{w} \longleftarrow \mathbf{w} + \alpha\ (\ y_2 - h(\mathbf{x}_2)\ )\ \mathbf{x}_2 = [0, 0] + 0.1 * (0 - 1)\ [0, 3] = [0, -0.3]$
    - » $b\ \longleftarrow b + \alpha\ (\ y_2 - h(\mathbf{x}_2)\ )\ = 0.5 + 0.1\ (0 - 1) = 0.4$

Exemple

D ensemble entraînement

$\mathbf{x}_t$	y_t
[2,0]	1
[0,3]	0
[3,0]	0
[1,1]	1

- Simulation **avec biais**, $\alpha = 0.1$
- Valeur courante : $\mathbf{w} \longleftarrow [0, -0.3]$, $b = 0.4$
- Paire $(\mathbf{x}_3, y_3)$:
  - $h(\mathbf{x}_3) = \mathit{Threshold}(\ \mathbf{w} \cdot \mathbf{x}_3 + b) = \mathit{Threshold}(0.4) = 1$
  - puisque $h(\mathbf{x}_3) \neq y_3$, on met à jour $\mathbf{w}$ et b
    - » $\mathbf{w} \longleftarrow \mathbf{w} + \alpha\ (\ y_3 - h(\mathbf{x}_3)\ )\ \mathbf{x}_3 = [0, -0.3] + 0.1 * (0 - 1)\ [3, 0] = [-0.3, -0.3]$
    - » $b \longleftarrow b + \alpha\ (\ y_3 - h(\mathbf{x}_3)\ )\ = 0.4 + 0.1\ (0 - 1) = 0.3$

Exemple

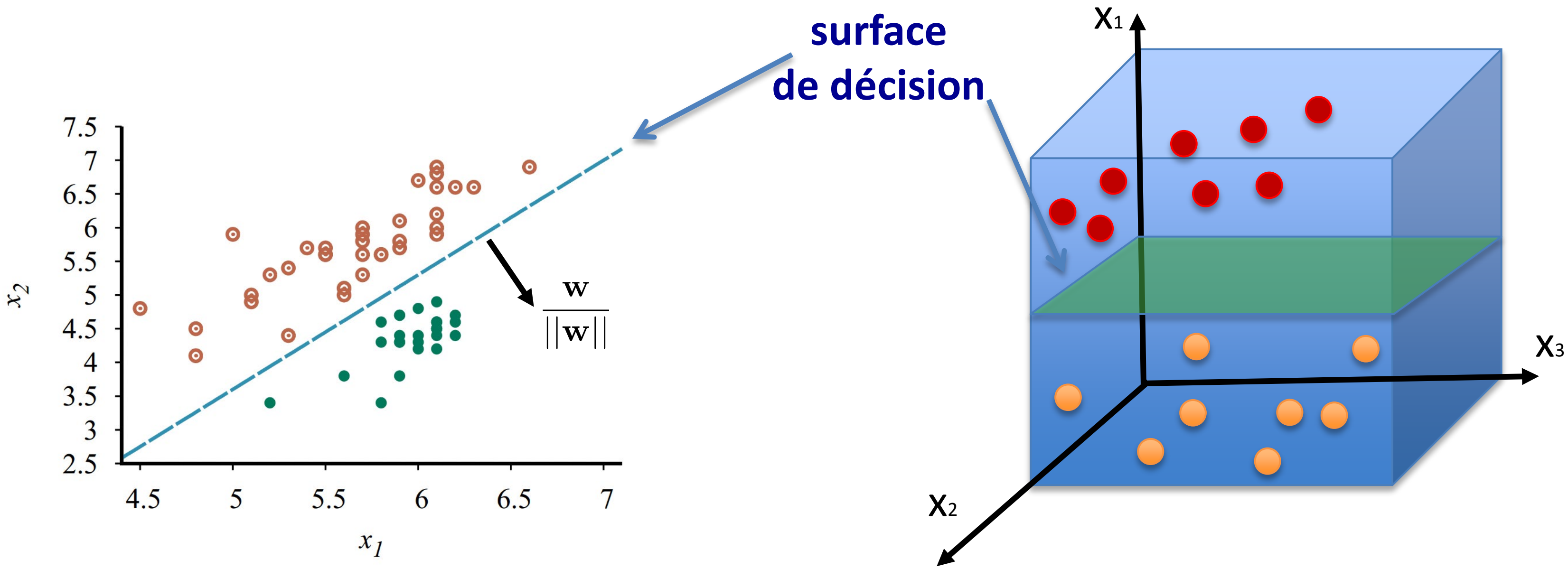
D ensemble entraînement

$\mathbf{x}_t$	y_t
[2,0]	1
[0,3]	0
[3,0]	0
[1,1]	1

- Simulation **avec biais**, $\alpha = 0.1$
- Valeur courante : $\mathbf{w} \longleftarrow [-0.3, -0.3]$, $b = 0.3$
- Paire $(\mathbf{x}_4, y_4)$:
  - $h(\mathbf{x}_4) = \mathit{Threshold}(\ \mathbf{w} \cdot \mathbf{x}_4 + b) = \mathit{Threshold}(-0.3) = 0$
  - puisque $h(\mathbf{x}_4) \neq y_4$, on met à jour **w** et *b*
    - » $\mathbf{w} \longleftarrow \mathbf{w} + \alpha\ (\ y_4 - h(\mathbf{x}_4)\ )\ \mathbf{x}_4 = [-0.3, -0.3] + 0.1 * (1 - 0)\ [1, 1] = [-0.2, -0.2]$
    - » $b\ \longleftarrow b + \alpha\ (\ y_4 - h(\mathbf{x}_4)\ )\ = 0.3 + 0.1\ (1 - 0) = 0.4$
- Et ainsi de suite, jusqu'à l'atteinte d'un critère d'arrêt...

Surface de séparation

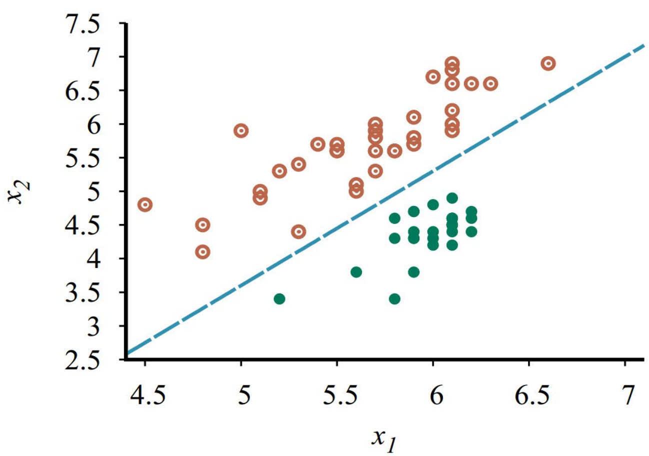
- Le Perceptron cherche donc un **séparateur linéaire** entre les deux classes

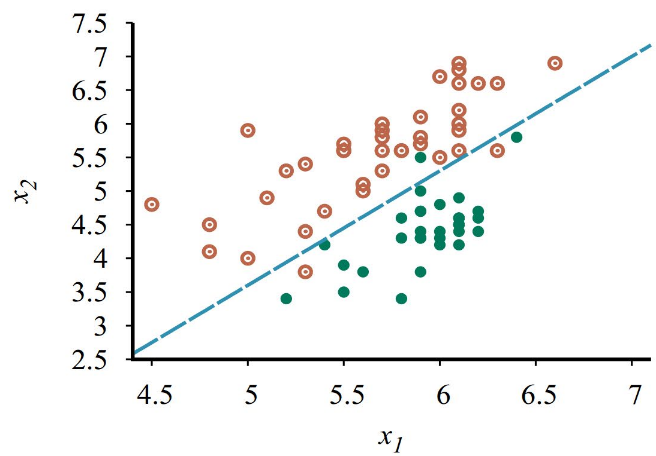


- La **surface de décision** d'un classifieur est la surface (dans le cas du perceptron en 2D, une droite) qui sépare les deux régions classifiées dans les deux classes différentes

Convergence et séparabilité

- Si les exemples d'entraînement sont **linéairement séparables** (gauche), l'algorithme est garanti de converger à **une solution avec une erreur nulle** sur l'ensemble d'entraînement, quel que soit le choix de α

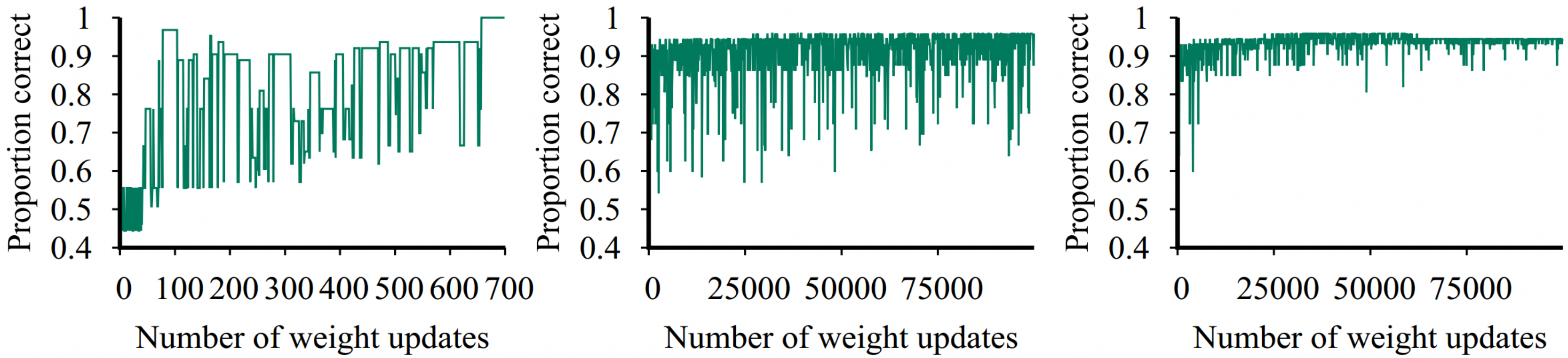




- Si non-séparable linéairement (droite), pour garantir la convergence à une **solution avec la plus petite erreur possible en entraînement**, on doit décroître le taux d'apprentissage, par ex. selon $\alpha_k = \dfrac{\alpha}{1 + \beta k}$

Courbe d'apprentissage

- Pour visualiser la progression de l'apprentissage, on peut regarder la **courbe d'apprentissage**, c'est-à-dire la courbe du taux d'erreur (ou de succès) en fonction du nombre de mises à jour des paramètres



linéairement séparable, avec taux d'app. constant

pas linéairement séparable, avec aux d'app. constant

pas linéairement séparable, avec taux d'app. décroissant

Apprentissage vue comme la minimisation d'une perte

- Le problème de l'apprentissage peut être formulé comme un problème d'optimisation
  - pour chaque exemple d'entraînement, on souhaite minimiser une certaine distance $Loss(y_t, h_{\mathbf{w}}(\mathbf{x}_t))$ entre la cible y_t et la prédiction $h_{\mathbf{w}}(\mathbf{x}_t)$
  - on appelle cette distance une **perte**

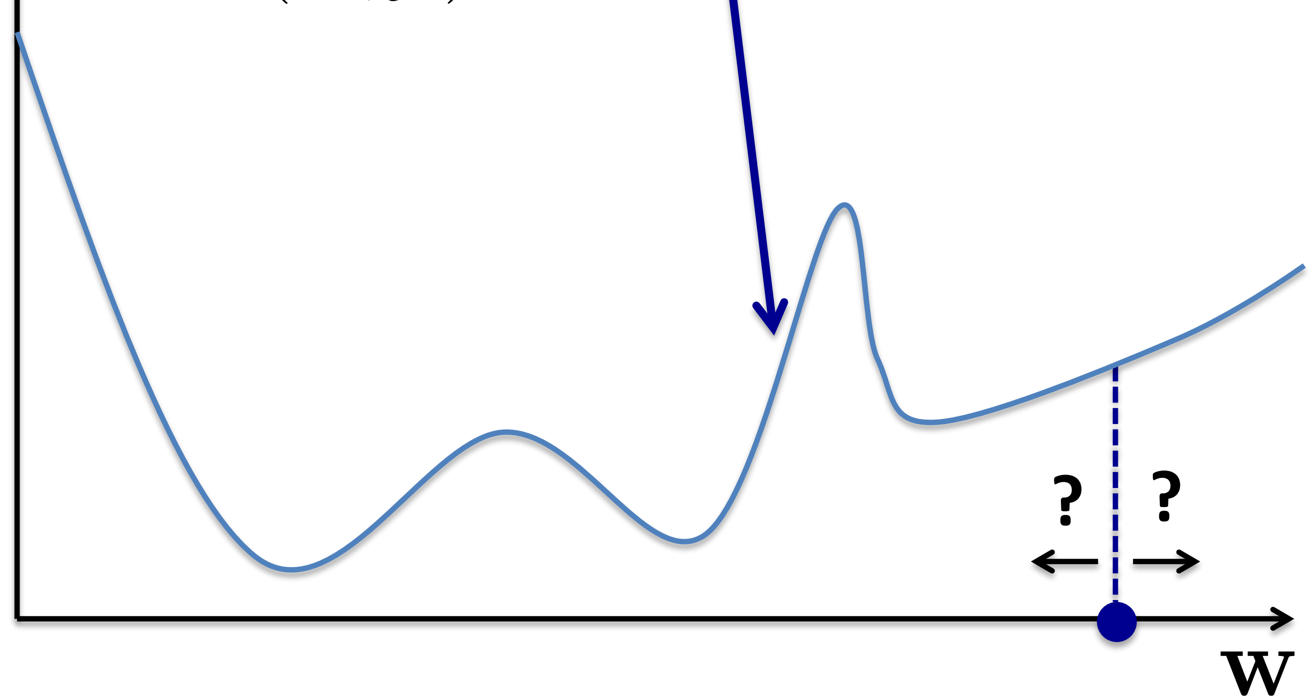
- Dans le cas du perceptron:

$$Loss(y_t, h_{\mathbf{w}}(\mathbf{x}_t)) = -(y_t - h_{\mathbf{w}}(\mathbf{x}_t))\mathbf{w} \cdot \mathbf{x}_t$$

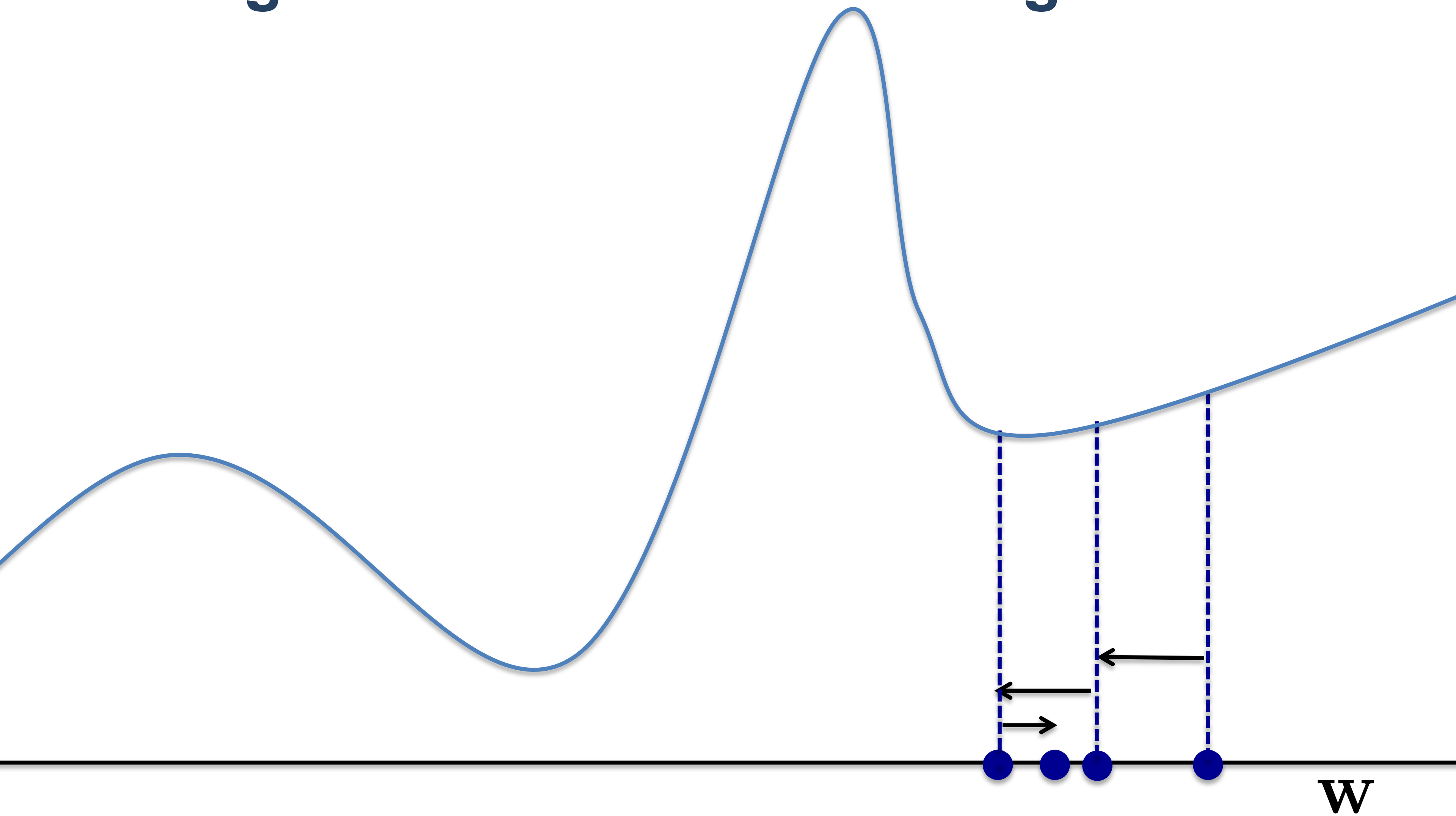
  - si la prédiction est bonne, le coût est 0
  - si la prédiction est mauvaise, le coût est la distance entre $\mathbf{w} \cdot \mathbf{x}_t$ et le seuil à franchir pour que la prédiction soit bonne

Recherche locale pour la minimisation d'une perte

$$\frac{1}{|D|} \sum_{(\mathbf{x}_t, y_t) \in D} Loss(y_t, h_{\mathbf{w}}(\mathbf{x}_t))$$

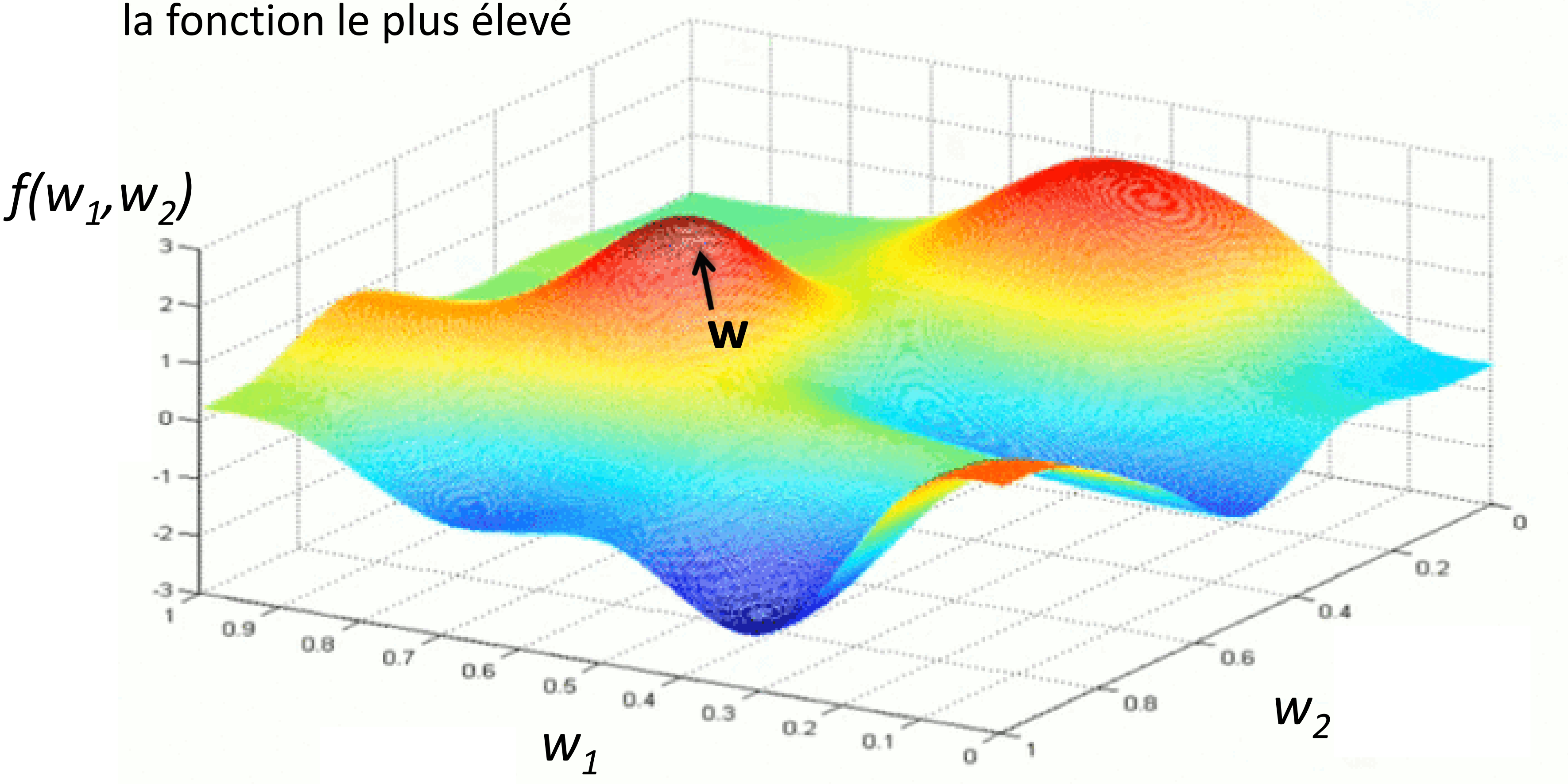


Algorithme de descente de gradient



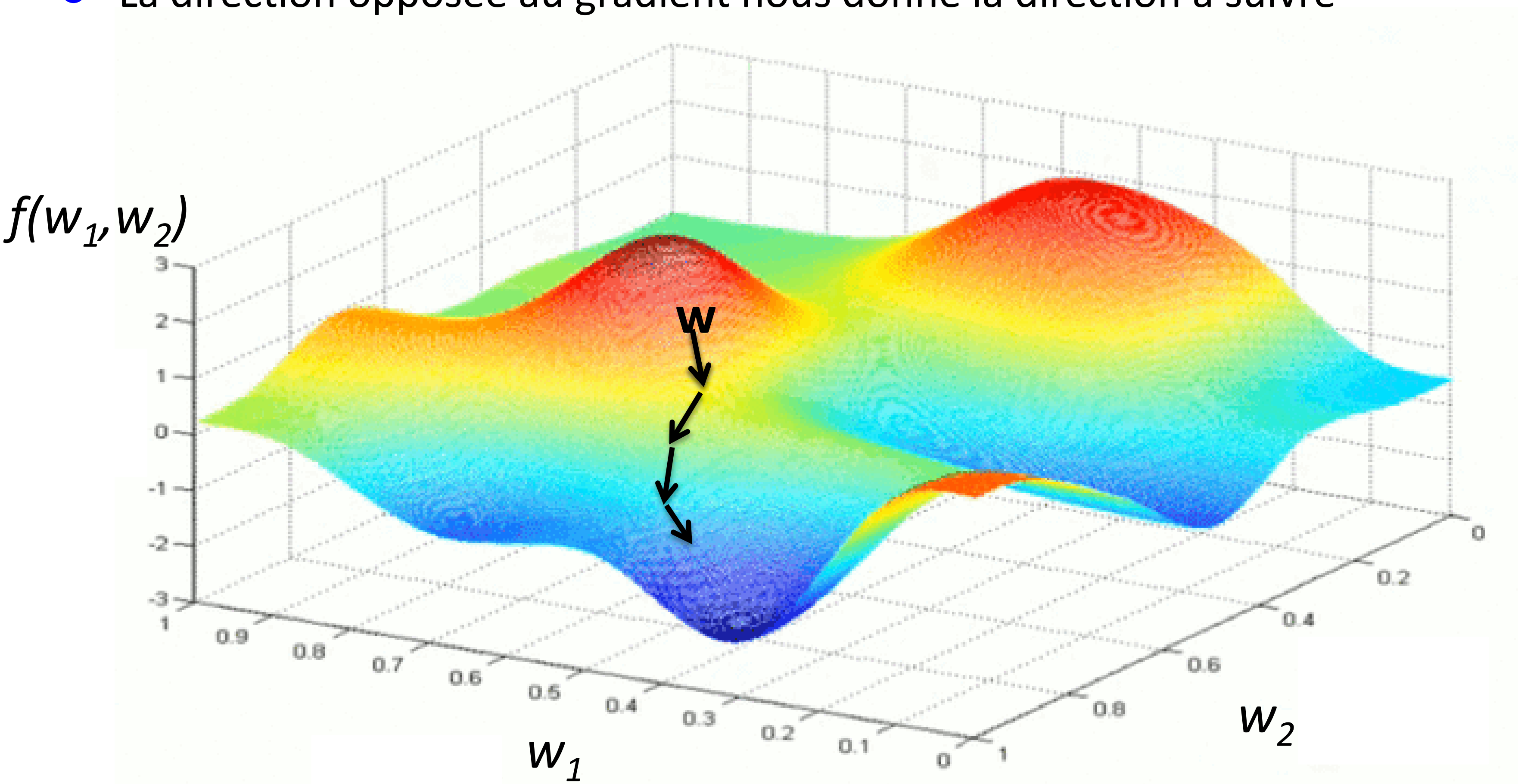
Descente de gradient

- Le gradient donne la direction (vecteur) ayant le taux d'accroissement de la fonction le plus élevé



Descente de gradient

- La direction opposée au gradient nous donne la direction à suivre



Apprentissage vue comme la minimisation d'une perte

- En apprentissage automatique, on souhaite optimiser:

$$\frac{1}{|D|} \sum_{(\mathbf{x}_t, y_t) \in D} Loss(y_t, h_{\mathbf{w}}(\mathbf{x}_t))$$

- Le gradient par rapport à la perte moyenne contient les dérivées partielles:

$$\frac{1}{|D|} \sum_{(\mathbf{x}_t, y_t) \in D} \frac{\partial}{\partial w_i} Loss(y_t, h_{\mathbf{w}}(\mathbf{x}_t)) \quad \forall i$$

- Devrait calculer la moyenne des dérivées sur tous les exemples d'entraînement avant de faire une mise à jour des paramètres!

Descente de gradient stochastique

- **Descente de gradient stochastique**: mettre à jour les paramètres à partir du (c.-à-d. des dérivées partielles) d'un seul exemple, choisi aléatoirement:

> - Initialiser $\mathbf{w}$ aléatoirement
> - Répéter jusqu'à la convergence
>   - Pour chaque exemple d'entraînement $(\mathbf{x}_t, y_t)$
>   - $$w_i \leftarrow w_i - \alpha \frac{\partial}{\partial w_i} Loss(y_t, h_{\mathbf{w}}(\mathbf{x}_t)) \quad \forall i$$

- Cette procédure est beaucoup plus efficace lorsque l'ensemble d'entraînement D est grand
    - on fait |D| mises à jour des paramètres après chaque parcours de l'ensemble d'entraînement, plutôt qu'une seule mise à jour avec la descente de gradient normale

Retour sur le Perceptron

Rappel $\qquad \frac{\partial f(g(x))}{\partial x} = \frac{\partial f(x)}{\partial g(x)} \frac{\partial g(x)}{\partial x} \qquad \frac{\partial}{\partial x} x^2 = 2x \qquad \frac{\partial}{\partial x} x = 1.$

- Utilisons le gradient (dérivée partielle) pour déterminer une direction de mise à jour des paramètres:

$$\frac{\partial}{\partial w_i} Loss(y_t, h_{\mathbf{w}}(\mathbf{x}_t)) = \frac{\partial}{\partial w_i} (y_t - h_{\mathbf{w}}(x_t))^2 = 2(y_t - h_{\mathbf{w}}(x_t)) \times \frac{\partial}{\partial w_i} (y_t - \underbrace{h_{\mathbf{w}}(x_t)}_{\sum_j \omega_j x_{t,j}})$$

$$= -2(y_t - h_{\mathbf{w}}(x_t)) \times x_{t,i}$$

- Rappel de la règle de mise à jour de la descente du gradient

$$w_i \leftarrow w_i - \alpha \frac{\partial}{\partial w_i} Loss(y_t, h_{\mathbf{w}}(\mathbf{x}_t)) \quad \forall i$$

- On obtient à nouveau la règle d'apprentissage du Perceptron

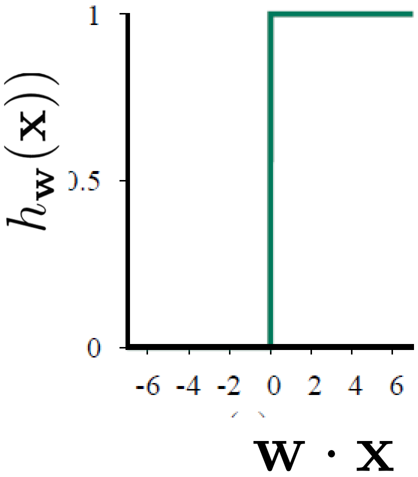
$$w_i \leftarrow w_i + \alpha(y_t - h_{\mathbf{w}}(\mathbf{x}_t))x_{t,i} \quad \forall i$$

Apprentissage vue comme la minimisation d'une perte

- La procédure de descente de gradient stochastique est applicable à n'importe quelle perte dérivable partout

- Dans le cas du Perceptron, on a un peu triché:
  - la dérivée de $h_{\mathbf{w}}(\mathbf{x}))$ n'est pas définie lorsque $\mathbf{w} \cdot \mathbf{x} = 0$

- L'utilisation de la fonction *Threshold* (qui est constante par partie) fait que la courbe d'entraînement peut être instable



Vous devriez être capable de...

- Définir et simuler l'algorithme d'apprentissage du perceptron
- Dériver l'algorithme d'apprentissage du perceptron en utilisant la descente stochastique du gradiant

RAPPEL SUR LES DÉRIVÉES PARTIELLES ET LE GRADIENT

Dérivées

- On peut obtenir la direction de descente via la **dérivée**

$$f'(a) = \frac{df(a)}{da} = \lim_{\Delta \to 0} \frac{f(a + \Delta) - f(a)}{\Delta}$$

- Le signe de la dérivée est la **direction d'augmentation** de f
  - signe positif indique que $f(a)$ augmente lorsque a augmente
  - signe négatif indique que $f(a)$ diminue lorsque a augmente
- La valeur absolue de la dérivée est le **taux d'augmentation** de f

- Plutôt que d, je vais utiliser le symbole ∂

Dérivées

- Les dérivées usuelles les plus importantes sont les suivantes:

$$\frac{\partial a}{\partial x} = 0$$

$$\frac{\partial x^n}{\partial x} = n x^{n-1}$$

$$\frac{\partial \log(x)}{\partial x} = \frac{1}{x}$$

$$\frac{\partial \exp(x)}{\partial x} = \exp(x)$$

a et n sont des constantes

Dérivées

- On peut obtenir des dérivées de composition de fonctions

$$\frac{\partial a f(x)}{\partial x} = a \frac{\partial f(x)}{\partial x}$$

$$\frac{\partial f(x)^n}{\partial x} = n f(x)^{n-1} \frac{\partial f(x)}{\partial x}$$

$$\frac{\partial \exp(f(x))}{\partial x} = \exp(f(x)) \frac{\partial f(x)}{\partial x}$$

$$\frac{\partial \log(f(x))}{\partial x} = \frac{1}{f(x)} \frac{\partial f(x)}{\partial x}$$

a et n sont des constantes

Dérivées

- Exemple 1: $f(x) = 3x^4$

$$\frac{\partial f(x)}{\partial x} = \frac{\partial 3x^4}{\partial x} = 3\frac{\partial x^4}{\partial x} = 12x^3$$

Dérivées

- Exemple 2: $f(x) = \exp\left(\frac{x^2}{3}\right)$

$$\frac{\partial f(x)}{\partial x} = \frac{\partial \exp\left(\frac{x^2}{3}\right)}{\partial x} = \exp\left(\frac{x^2}{3}\right) \frac{\partial \frac{x^2}{3}}{\partial x}$$

$$= \frac{1}{3} \exp\left(\frac{x^2}{3}\right) \frac{\partial x^2}{\partial x} = \frac{2}{3} \exp\left(\frac{x^2}{3}\right) x$$

Dérivées

- Pour des combinaisons plus complexes:

$$\frac{\partial g(x) + h(x)}{\partial x} = \frac{\partial g(x)}{\partial x} + \frac{\partial h(x)}{\partial x}$$

$$\frac{\partial g(x)h(x)}{\partial x} = \frac{\partial g(x)}{\partial x}h(x) + g(x)\frac{\partial h(x)}{\partial x}$$

$$\frac{\partial \frac{g(x)}{h(x)}}{\partial x} = \frac{\partial g(x)}{\partial x}\frac{1}{h(x)} - \frac{g(x)}{h(x)^2}\frac{\partial h(x)}{\partial x}$$

Dérivées

- Exemple 3: $f(x) = x \exp(x)$

$$\frac{\partial f(x)}{\partial x} = \frac{\partial x}{\partial x} \exp(x) + x \frac{\partial \exp(x)}{\partial x}$$

$$= \exp(x) + x \exp(x)$$

Dérivées

- Exemple 4: $f(x) = \dfrac{\exp(x)}{x}$

$$\frac{\partial f(x)}{\partial x} = \frac{\partial \exp(x)}{\partial x}\frac{1}{x} - \frac{\exp(x)}{x^2}\frac{\partial x}{\partial x}$$

$$= \frac{\exp(x)}{x} - \frac{\exp(x)}{x^2}$$

Dérivées

- Exemple 4: $f(x) = \frac{\exp(x)}{x}$

$\boxed{\text{dérivation alternative!}}$

$$\frac{\partial f(x)}{\partial x} = \frac{\partial \exp(x)}{\partial x} \frac{1}{x} + \exp(x) \frac{\partial \frac{1}{x}}{\partial x}$$

$$= \frac{\exp(x)}{x} - \frac{\exp(x)}{x^2}$$

Dérivée partielle et gradient

- Dans notre cas, la fonction à optimiser dépend de plus d'une variable
  - elle dépend de tout le vecteur $\mathbf{w}$
- Dans ce cas, on va considérer les **dérivées partielles**, c.-à-d. la dérivée par rapport à chacune des variables en supposant que les autres sont constantes:

$$\frac{\partial f(a,b)}{\boxed{\partial a}} = \lim_{\Delta \to 0} \frac{f(\boxed{a+\Delta}, b) - f(a,b)}{\Delta}$$

$$\frac{\partial f(a,b)}{\boxed{\partial b}} = \lim_{\Delta \to 0} \frac{f(a, \boxed{b+\Delta}) - f(a,b)}{\Delta}$$

Dérivée partielle et gradient

- Exemple de fonction à deux variables:

$$f(x,y) = \frac{x^2}{y}$$

- Dérivées partielles:

$$\frac{\partial f(x,y)}{\partial x} = \underbrace{\frac{2x}{y}}_{\text{traite } y \text{ comme une constante}} \qquad \frac{\partial f(x,y)}{\partial y} = \underbrace{\frac{-x^2}{y^2}}_{\text{traite } x \text{ comme une constante}}$$

Dérivée partielle et gradient

- Un deuxième exemple:

$$f(\mathbf{x}) = \frac{\exp(x_2)}{\exp(x_1) + \exp(x_2) + \exp(x_3)}$$

- Dérivée partielle $\dfrac{\partial f(\mathbf{x})}{\partial x_1}$:

  équivaut à faire la dérivée de $f(x) = \dfrac{a}{\exp(x) + b}$

  où $x = x_1$
  et on a des **constantes** $a = \exp(x_2)$ et $b = \exp(x_2) + \exp(x_3)$

Dérivée partielle et gradient

- Un deuxième exemple: $f(x) = \dfrac{a}{\exp(x) + b}$

$$\frac{\partial f(x)}{\partial x} = \frac{\partial}{\partial x} \frac{a}{\exp(x) + b} = a \frac{\partial}{\partial x} \frac{1}{\exp(x) + b}$$

$$= \frac{-a}{(\exp(x) + b)^2} \frac{\partial}{\partial x} (\exp(x) + b)$$

$$= \frac{-a \exp(x)}{(\exp(x) + b)^2}$$

Dérivée partielle et gradient

- Un deuxième exemple:

$$\frac{\partial f(x)}{\partial x} = \frac{-a\exp(x)}{(\exp(x)+b)^2}$$

où $x = x_1$, $a = \exp(x_2)$, $b = \exp(x_2) + \exp(x_3)$

- On remplace:

$$\frac{\partial f(\mathbf{x})}{\partial x_1} = \frac{-\exp(x_2)\exp(x_1)}{(\exp(x_1)+\exp(x_2)+\exp(x_3))^2}$$

Dérivée partielle et gradient

- Un troisième exemple:

$$f(\mathbf{x}) = \frac{\exp(x_2)}{\exp(x_1) + \exp(x_2) + \exp(x_3)}$$

- Dérivée partielle $\dfrac{\partial f(\mathbf{x})}{\partial x_2}$:

  équivaut à faire la dérivée de $f(x) = \dfrac{\exp(x)}{\exp(x) + b}$

  où $x = x_2$
  et on a une constante $b = \exp(x_1) + \exp(x_3)$

Dérivée partielle et gradient

- Un troisième exemple: $$f(x) = \frac{\exp(x)}{\exp(x) + b}$$

$$\frac{\partial f(x)}{\partial x} = \frac{\partial \exp(x)}{\partial x} \frac{1}{\exp(x) + b} - \frac{\exp(x)}{(\exp(x) + b)^2} \frac{\partial (\exp(x) + b)}{\partial x}$$

$$= \frac{\exp(x)}{\exp(x) + b} - \frac{\exp(x) \exp(x)}{(\exp(x) + b)^2}$$

Dérivée partielle et gradient

- Un troisième exemple:

$$\frac{\partial f(x)}{\partial x} = \frac{\exp(x)}{\exp(x) + b} - \frac{\exp(x)\exp(x)}{(\exp(x) + b)^2}$$

où $x = x_2$, $b = \exp(x_1) + \exp(x_3)$

- On remplace:

$$\frac{\partial f(\mathbf{x})}{\partial x_2} = \frac{\exp(x_2)}{\exp(x_2) + \exp(x_1) + \exp(x_3)} - \frac{\exp(x_2)\exp(x_2)}{(\exp(x_2) + \exp(x_1) + \exp(x_3))^2}$$

Dérivée partielle et gradient

- On va appeler **gradient** ∇f d'une fonction f le vecteur contenant les dérivées partielles de f par rapport à toutes les variables

- Dans l'exemple avec la fonction $f(x,y)$:

$$\begin{aligned}\nabla f(x,y) &= \left[\frac{\partial f(x,y)}{\partial x}, \frac{\partial f(x,y)}{\partial y}\right] \\ &= \left[\frac{2x}{y}, \frac{-x^2}{y^2}\right]\end{aligned}$$