Réseaux de neurones

7 décembre 2017

Objectifs

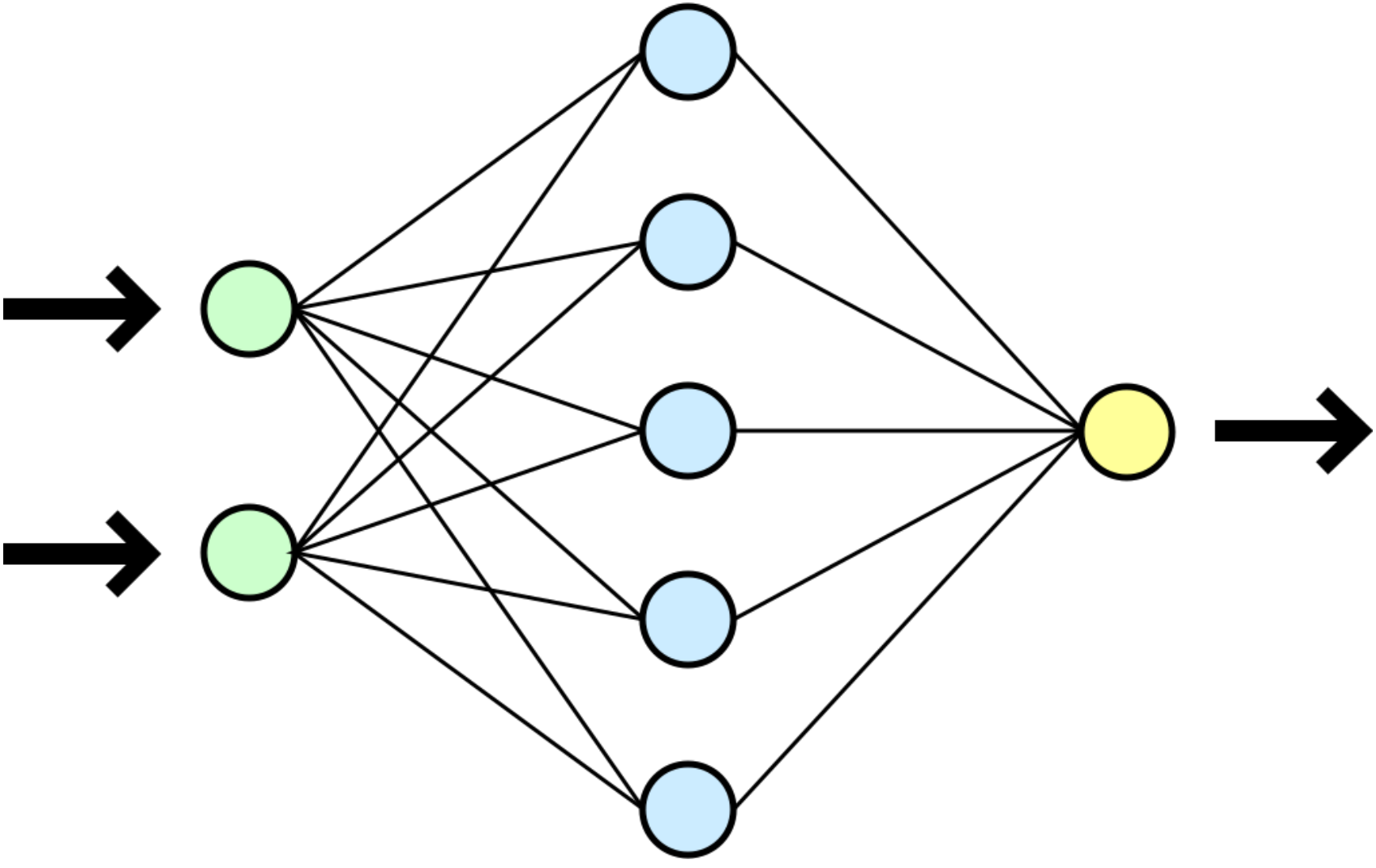
- Comprendre l'architecture de réseaux de neurones simples
- Démystifier la rétropropagation des gradients

Réseau de neurones

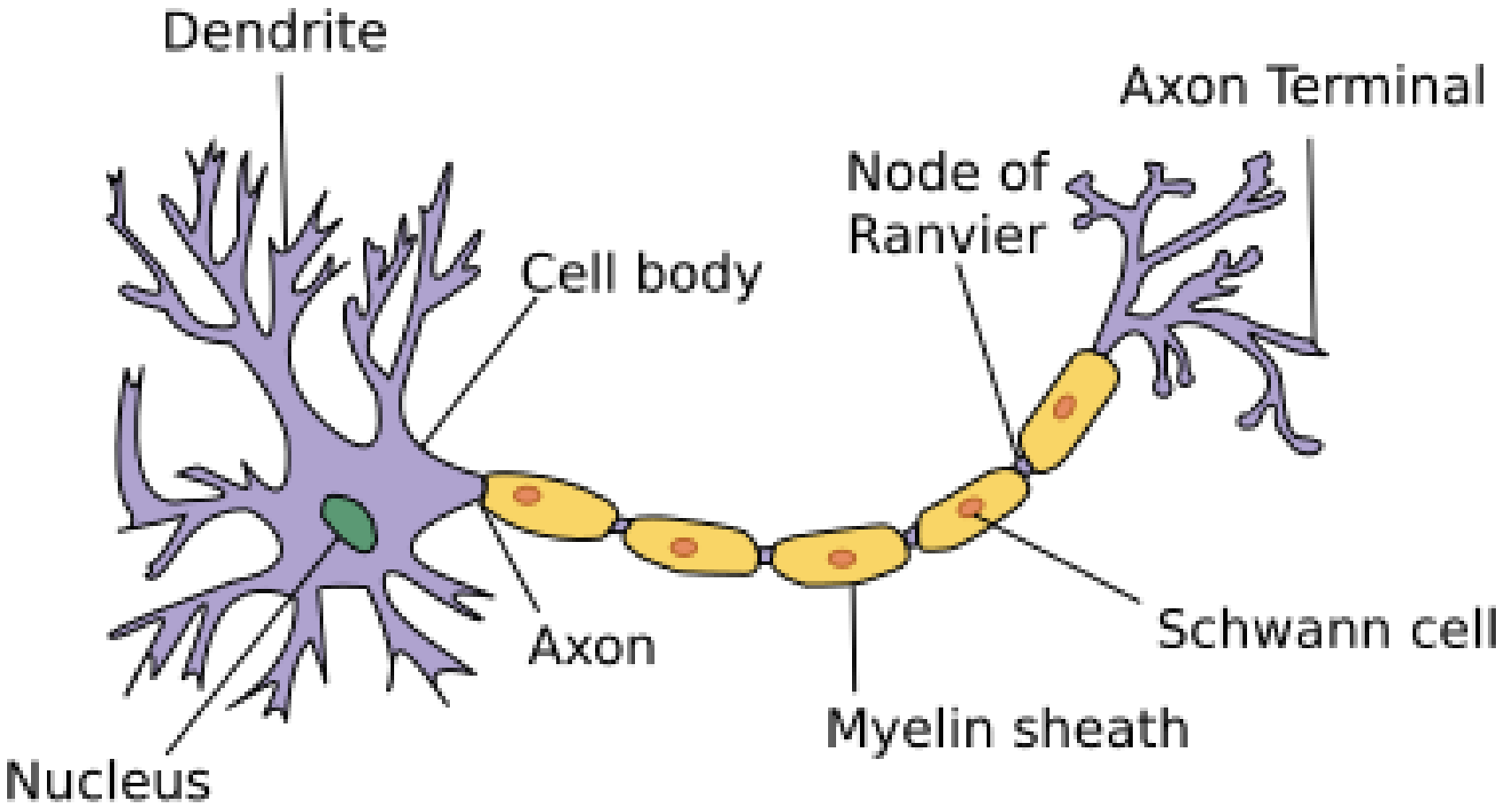
Intuition :

1 neurone $=$ 1 calcul simple
1M neurones $=$ 1M calculs simples
$=$ 1 calcul complexe

Réseau de neurones

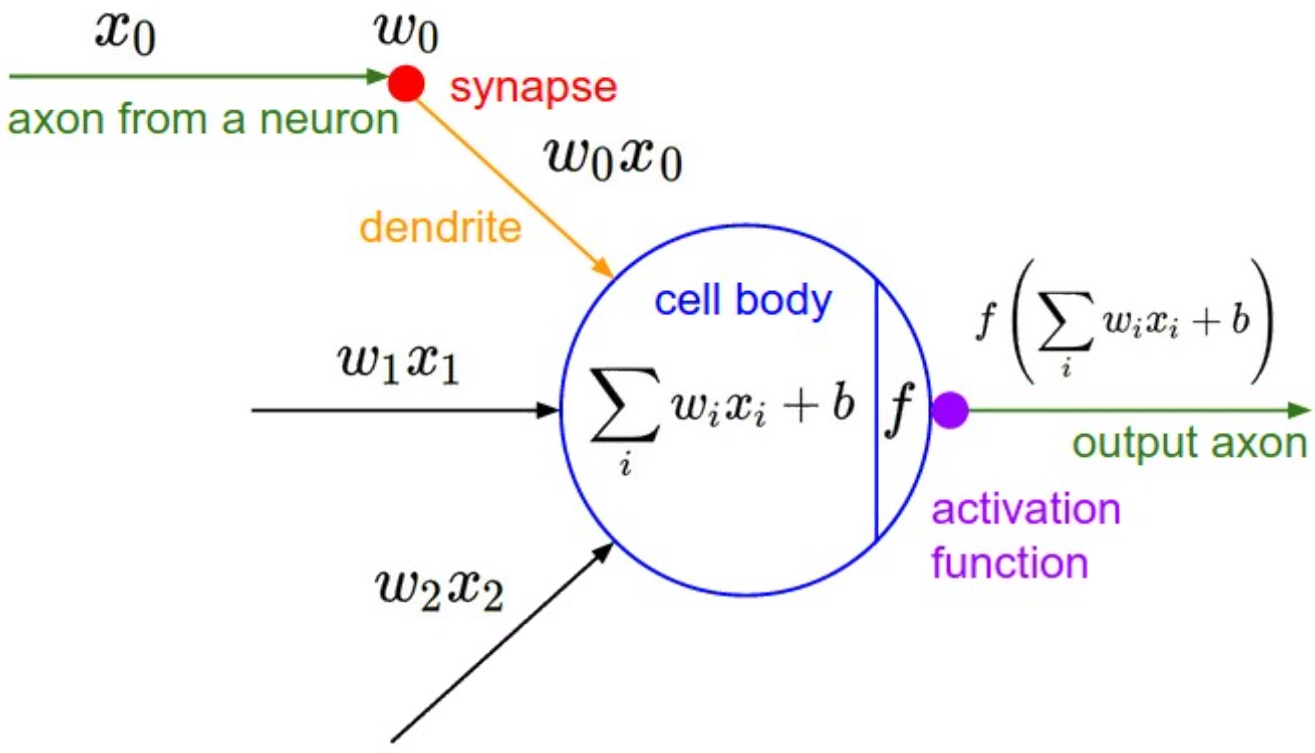


Neurone biologique



Neurone artificiel

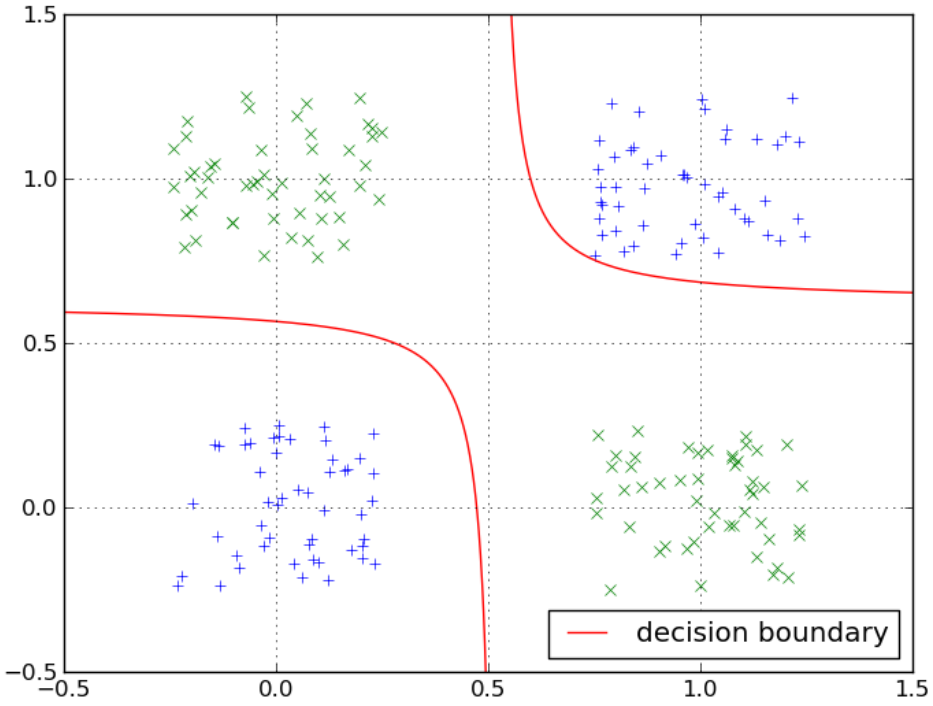
Simulation extrêmement basique d'un neurone : somme pondérée + activation.



Linéarité

Présence exclusive de sommes pondérées (combinaisons linéaires) $\rightarrow$ linéarité du réseau quelque soit sa profondeur (une combinaison linéaire de combinaisons linéaires est une combinaison linéaire).

$\rightarrow$ nécessité d'introduire des non-linéarités (sigmoid, tanh, ReLU, ...)



Apprentissage

Modification des poids des neurones pour que les sommes pondérées activées expriment la fonction souhaitée.

Métrique

Pour quantifier la qualité des poids du réseau, définition d'une fonction de perte à partir de données annotées.

$$L(\hat{y}, y)$$

Plus cette perte est proche de 0, meilleurs sont les poids de notre réseau.

Apprentissage — modélisation

Apprendre = minimiser la fonction de perte.

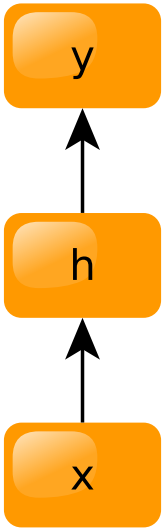
$$\arg\min_{\hat{y}} L(\hat{y}, y)$$

Apprentissage — optimisation par descente du gradient

Minimisation de L en soustrayant pour chaque poids w une partie du gradient par rapport à la perte (α est appelé pas d'apprentissage).

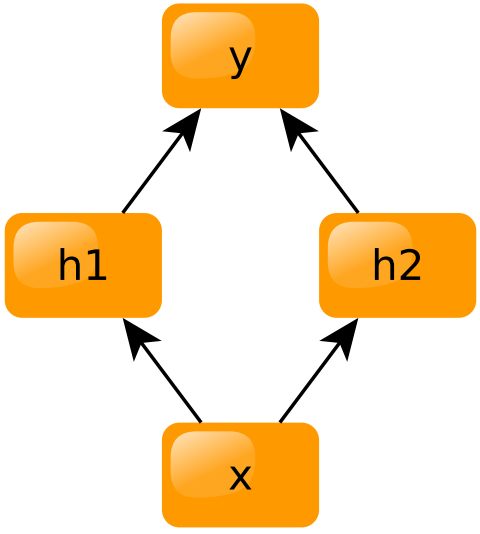
$$w \leftarrow w - \alpha \frac{\partial L}{\partial w}$$

Règle de chainage — cas simple



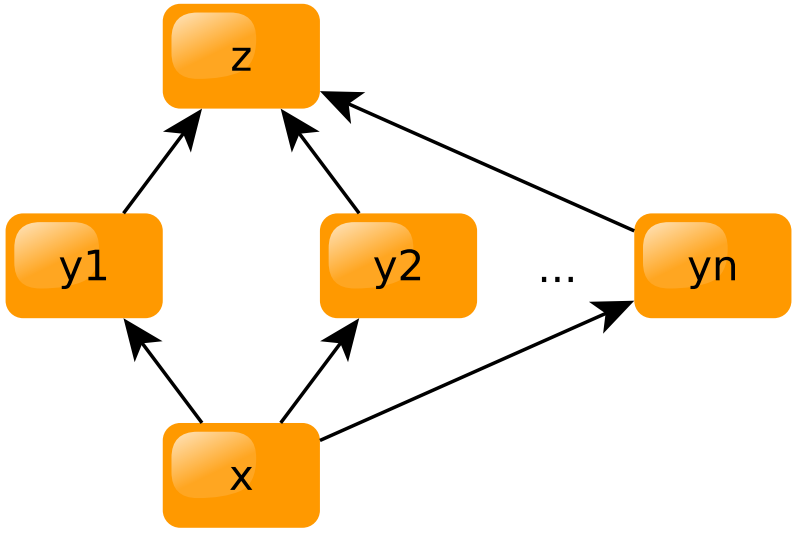
$$\frac{\partial z}{\partial x} = \frac{\partial z}{\partial y}\frac{\partial y}{\partial x}$$

Règle de chainage — deux chemins



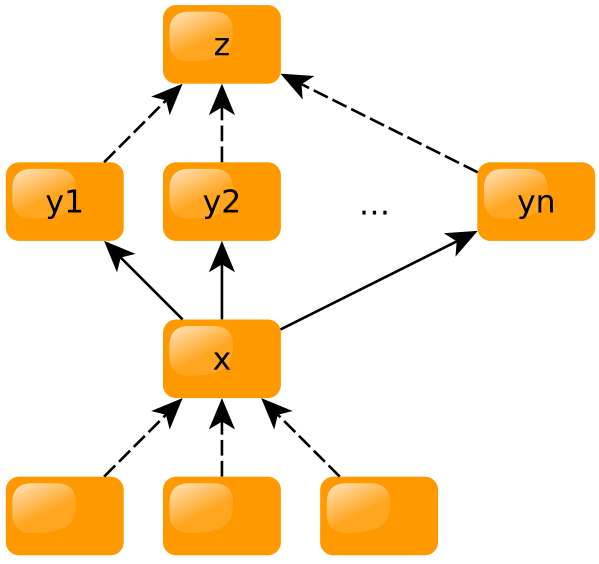
$$\frac{\partial z}{\partial x} = \frac{\partial z}{\partial y_1}\frac{\partial y_1}{\partial x} + \frac{\partial z}{\partial y_2}\frac{\partial y_2}{\partial x}$$

Règle de chainage — chemins multiples



$$\frac{\partial z}{\partial x} = \sum_{i=1}^{n} \frac{\partial z}{\partial y_i} \frac{\partial y_i}{\partial x}$$

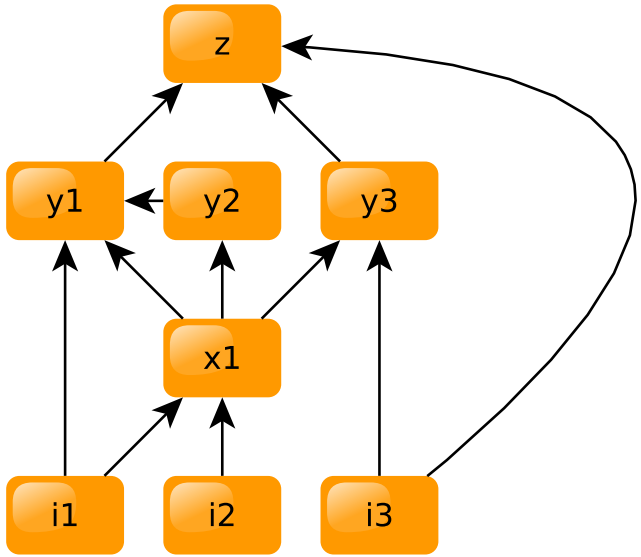
Règle de chainage — graphe dirigé acyclique



Pointillés = dépendance indirecte

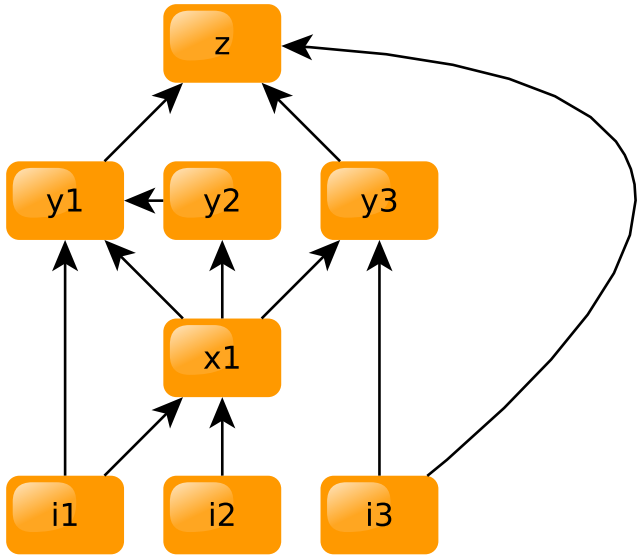
$$\frac{\partial z}{\partial x} = \sum_{i=1}^{n} \frac{\partial z}{\partial y_i} \frac{\partial y_i}{\partial x}$$

Règle de chainage — Exercice



1. Dans quel ordre doit-on calculer les dérivées partielles ?
2. Quelle est la chaîne de calcul pour trouver la dérivée du poids x_1 par rapport à l'erreur z ?

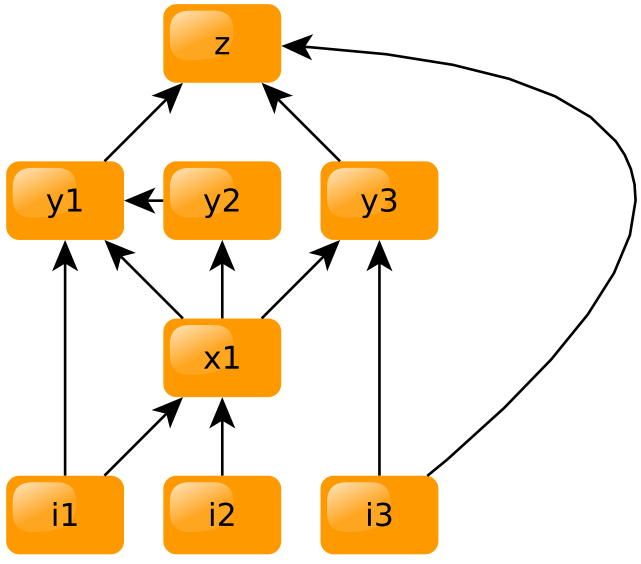
Règle de chainage — question 1



Dans quel ordre doit-on calculer les dérivées partielles ?

1. $\frac{\partial z}{\partial y_1}, \frac{\partial z}{\partial y_3}, \frac{\partial z}{\partial i_3}$
2. $\frac{\partial y_1}{\partial y_2}, \frac{\partial y_1}{\partial x_1}, \frac{\partial y_1}{\partial i_1}, \frac{\partial y_3}{\partial x_1}, \frac{\partial y_3}{\partial i_3}$
3. $\frac{\partial y_2}{\partial x_1}$
4. $\frac{\partial x_1}{\partial i_1}, \frac{\partial x_1}{\partial i_2}, \frac{\partial x_1}{\partial i_3}$

Règle de chainage — question 2



Quelle est la chaîne de calcul pour trouver la dérivée du poids x_1 par rapport à l'erreur z ?

$$\frac{\partial z}{\partial y_1}\frac{\partial y_1}{\partial x_1} + \frac{\partial z}{\partial y_1}\frac{\partial y_1}{\partial y_2}\frac{\partial y_2}{\partial x_1} + \frac{\partial z}{\partial y_3}\frac{\partial y_3}{\partial x_1}$$

Zoom sur les non-linéarités

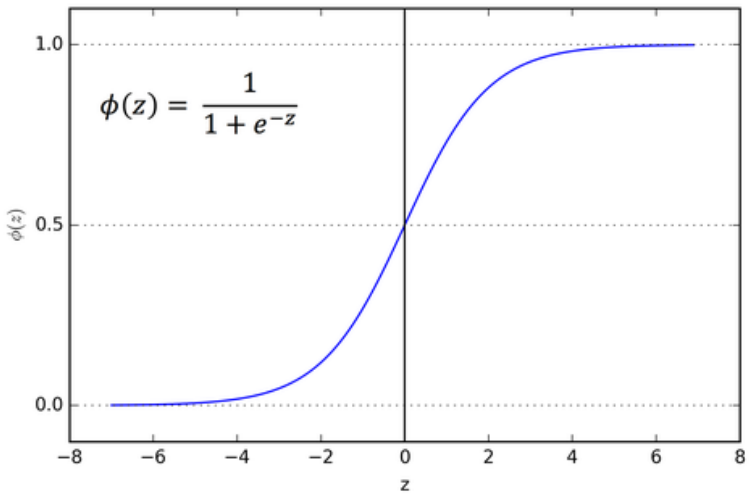
Analyser une non-linéarité :

- saturante ou non
- propriétés de la dérivée
- blocage possible ou non

Sigmoid

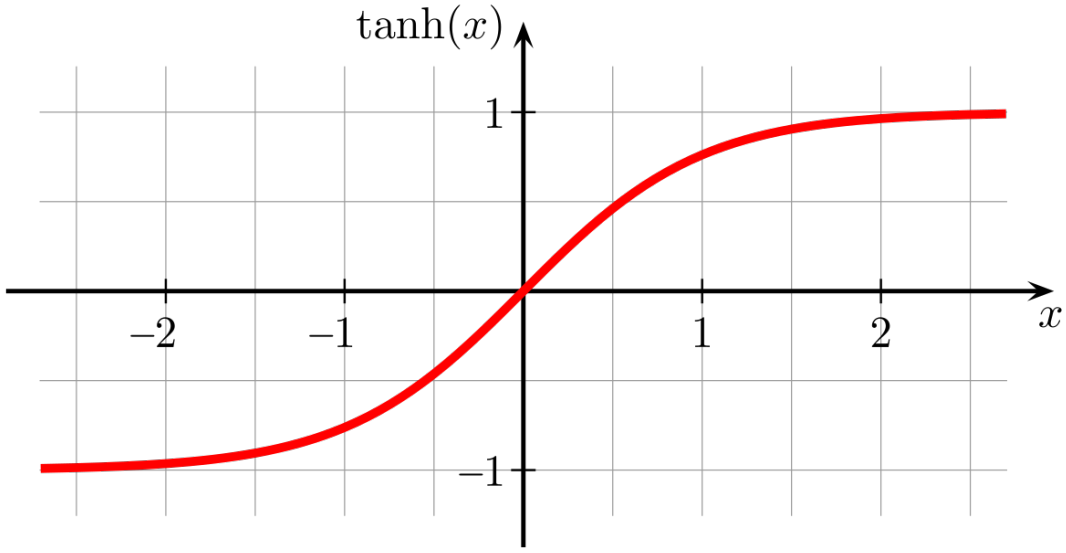
- Définition : $\sigma(x) = \frac{1}{1+e^{-x}}$
- Dérivée : $\sigma'(x) = \sigma(x).(1 - \sigma(x))$
- Fonction saturante

$\sigma'(x) \leq 0.25$



Tanh

- Définition : $\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$
- Dérivée : $\tanh'(x) = 1 - \tanh^2(x)$
- Fonction saturante



ReLU

- Définition : $\text{ReLU}(x) = \max(0, x)$
- Dérivée : $\text{ReLU}'(x) = \begin{cases} 0 & \text{si } x < 0 \\ 1 & \text{si } x \geq 0 \end{cases}$
- Fonction semi-saturante

Zoom sur les fonctions de perte

Analyser une fonction de perte :

- Classification ou régression
- Log ou linéaire

Perte L1

$$\mathrm{L1}(x, y) = \frac{\sum_{i=1}^{n} |y_i - x_i|}{n}$$

- Régression
- Peu utilisée car pénalise de la même manière toutes les amplitudes d'erreur

Perte L2

$$\mathrm{L2}(x, y) = \frac{\sum_{i=1}^{n} |y_i - x_i|^2}{n}$$

- Régression
- Perte la plus utilisée
- Pénalise plus fortement les grandes erreurs

Entropie croisée binaire

$y \in \{0,1\}^n$, $x \in [0,1]^n$.

$$\mathrm{ECB}(x,y) = -\frac{\sum_{i=1}^{n} (y_i \log x_i + (1-y_i) \log 1 - x_i)}{n}$$

- Classification
- Utilisée quand les sorties (x_i) sont indépendantes
- Logarithmique

Entropie croisée

c index de la classe correcte, x distribution de proba sur les classes.

$$\begin{aligned} \mathrm{EC}(x, c) &= -\frac{\log e^{x_c}}{\sum_j e^{x_j}} \\ &= -x_c + \log \sum_j e^{x_j} \end{aligned}$$

- Classification
- Utilisée quand les sorties sont liées
- Logarithmique

Et bien d'autres

- Pour gérer des embeddings (distance cosinus)
- Pour gérer des classements
- Pertes multi-termes

Conclusion

- Neurone = somme pondérée + activation
- Apprentissage = trouver les bons poids des sommes
- Métrique = fonction de perte
- Technique = rétropropagation des gradients
- Non linéarités et pertes classiques ont quelques propriétés à connaître