Apprentissage des réseaux de neurones

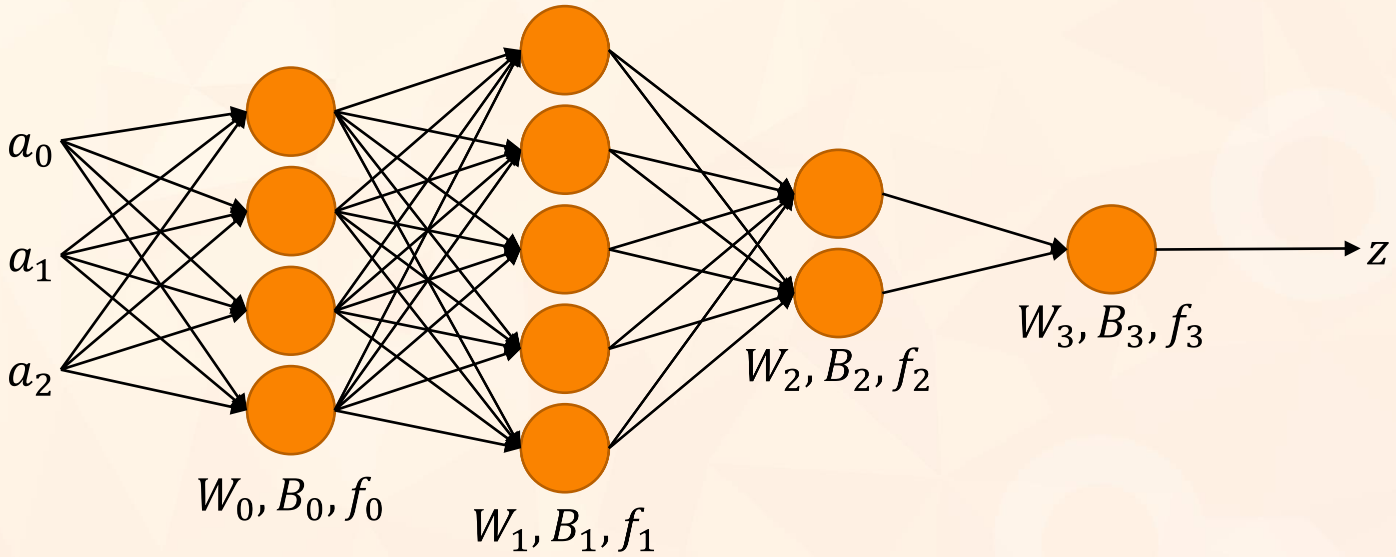
Présenté par Yohann Bosqued

Sommaire

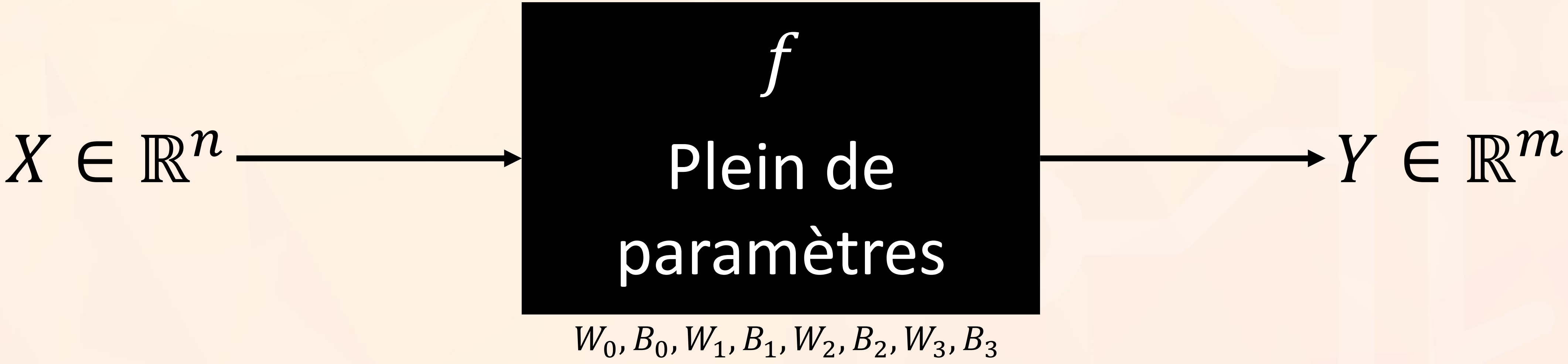
1. **Rappels**
2. Principe de l’apprentissage
3. Backpropagation
4. Conclusion

Rappels

$$\begin{pmatrix} a_0 \\ a_1 \\ a_2 \end{pmatrix} = A$$



$$(z) = Z$$



Principe

Des mathématiques.

Sommaire

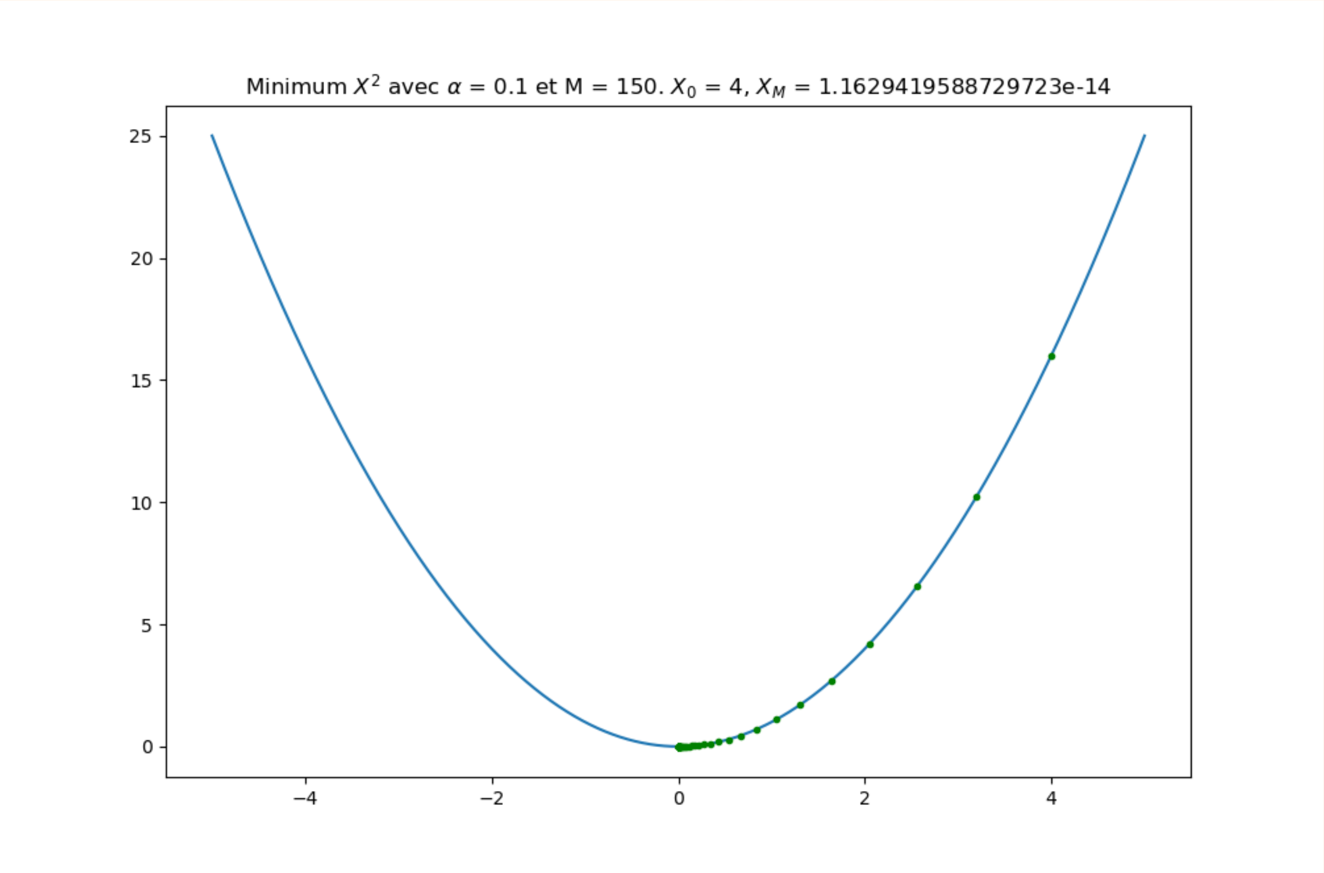
1. Rappels
2. **Principe de l’apprentissage**
3. Backpropagation
4. Conclusion

Principe (recherche de minimum)

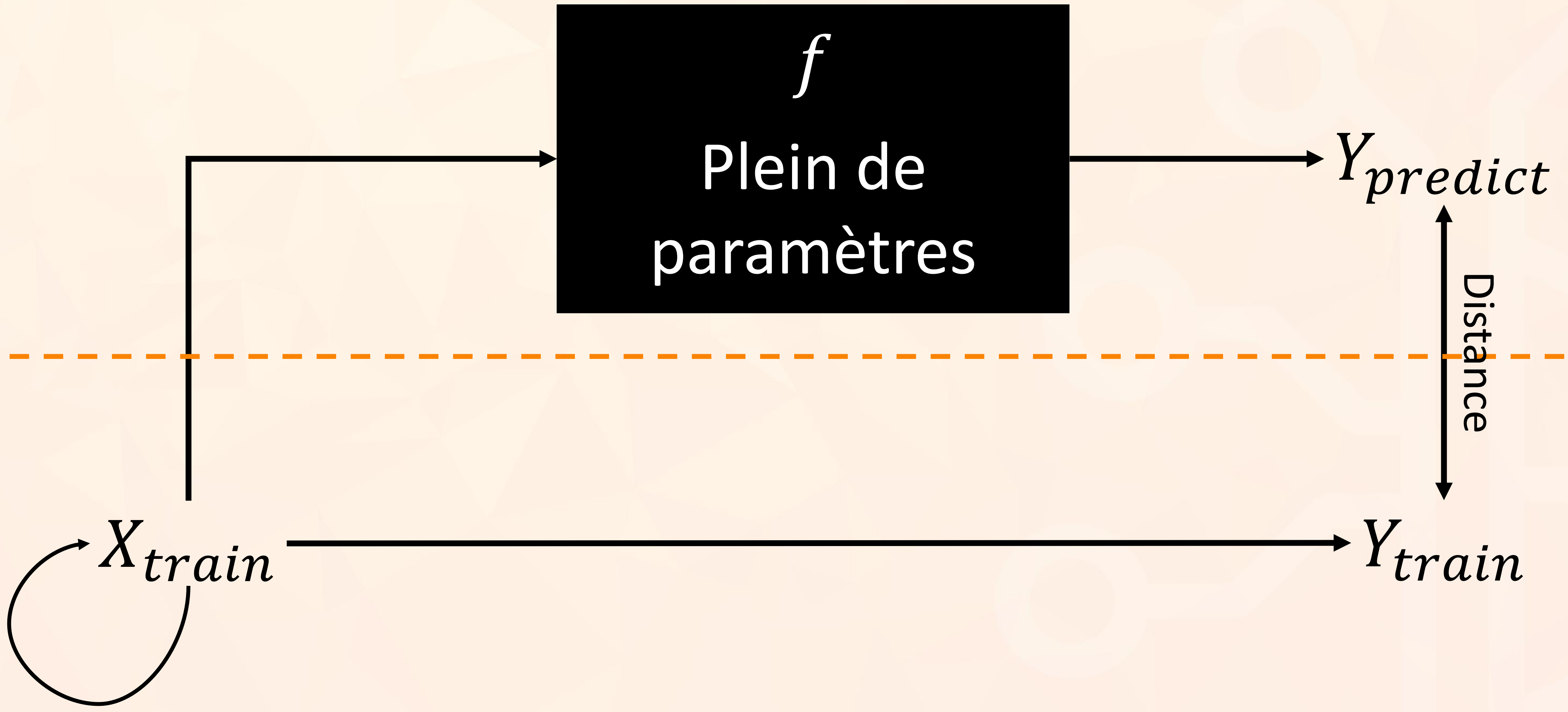
$f: \mathbb{R} \to \mathbb{R}$

$x_{n+1} = x_n - \alpha f'(x_n)$

$\alpha > 0$ est un hyperparamètre



Principe (apprentissage)



Distance / Fonction de coût

Permet d'évaluer une erreur entre 2 vecteurs

$$L2 :$$
$$h(X,Y) = \frac{1}{2}\|X - Y\|_2^2$$

$$L1 :$$
$$h(X,Y) = \|X - Y\|_1$$

$$Cross - entropy :$$
$$h(X,Y) = -(\mathrm{Y} * \log(\mathrm{X}) + (1 - Y) * \log(1 - X))$$

Principe

$$g(X_{train}, Y_{train}) = h(f(X_{train}), Y_{train})$$

$$g(w_{0,11}, w_{0,12}, \dots, b_{N,n}, X, Y) = h(f(w_{0,11}, w_{0,12}, \dots, b_{N,n}, X), Y)$$

$w_{0,11}, w_{0,12}, \dots, b_{N,n}$ sont les paramètres du réseau de neurones.

On souhaite minimiser g par rapport aux paramètres.

$$k \in \{w_{0,11}, w_{0,12}, \dots, b_{N,n}\} \rightarrow k = k - \alpha \left.\frac{\partial g}{\partial k}\right|_{w_{0,11}, w_{0,12}, \dots, b_{N,n}, X, Y}$$

Rappels de maths

$$f(x+h) \approx f(x) + df(x).h$$

$$h = g \circ f = g(f_1, \dots, f_n)$$

$$g \circ f(x+h) \approx g(f(x) + df(x).h)$$
$$\approx g \circ f(x) + dg\big(f(x)\big).df(x).h$$

$$h'(x) = g'\big(f(x)\big)f'(x)$$

$$\left.\frac{\partial h}{\partial x}\right|_{x=a} = \sum_k \left.\frac{\partial g}{\partial z_k}\right|_{z=f(a)} \left.\frac{\partial f_k}{\partial x}\right|_{x=a}$$

$$J(h)(a) = J(g)\big(f(a)\big).J(f)(a)$$

$$f: \mathbb{R} \to \mathbb{R} \Rightarrow \text{dérivée classique} \Rightarrow f'(x)h$$

$$f: \mathbb{R} \to \mathbb{R}^m \Rightarrow f = \begin{pmatrix} f_1 \\ \vdots \\ f_m \end{pmatrix} \Rightarrow f'(x) = \begin{pmatrix} f_1{}'(x) \\ \vdots \\ f_m{}'(x) \end{pmatrix} \Rightarrow \begin{pmatrix} f_1'(x)h \\ \vdots \\ f_m'(x)h \end{pmatrix}$$

$$f: \mathbb{R}^n \to \mathbb{R} \Rightarrow df(x) = \left(\left.\frac{\partial f}{\partial x_1}\right|_x, \dots, \left.\frac{\partial f}{\partial x_n}\right|_x \right) \Rightarrow \left.\frac{\partial f}{\partial x_1}\right|_x h_1 + \dots + \left.\frac{\partial f}{\partial x_n}\right|_x h_n = \nabla f(x).h$$

$$f: \mathbb{R}^n \to \mathbb{R}^m \Rightarrow f = \begin{pmatrix} f_1 \\ \vdots \\ f_m \end{pmatrix} \Rightarrow df(x) = \begin{pmatrix} \nabla f_1(x) \\ \vdots \\ \nabla f_m(x) \end{pmatrix} \Rightarrow \begin{pmatrix} \nabla f_1(x).h \\ \vdots \\ \nabla f_m(x).h \end{pmatrix} = J(f)(x).h$$

Backpropagation

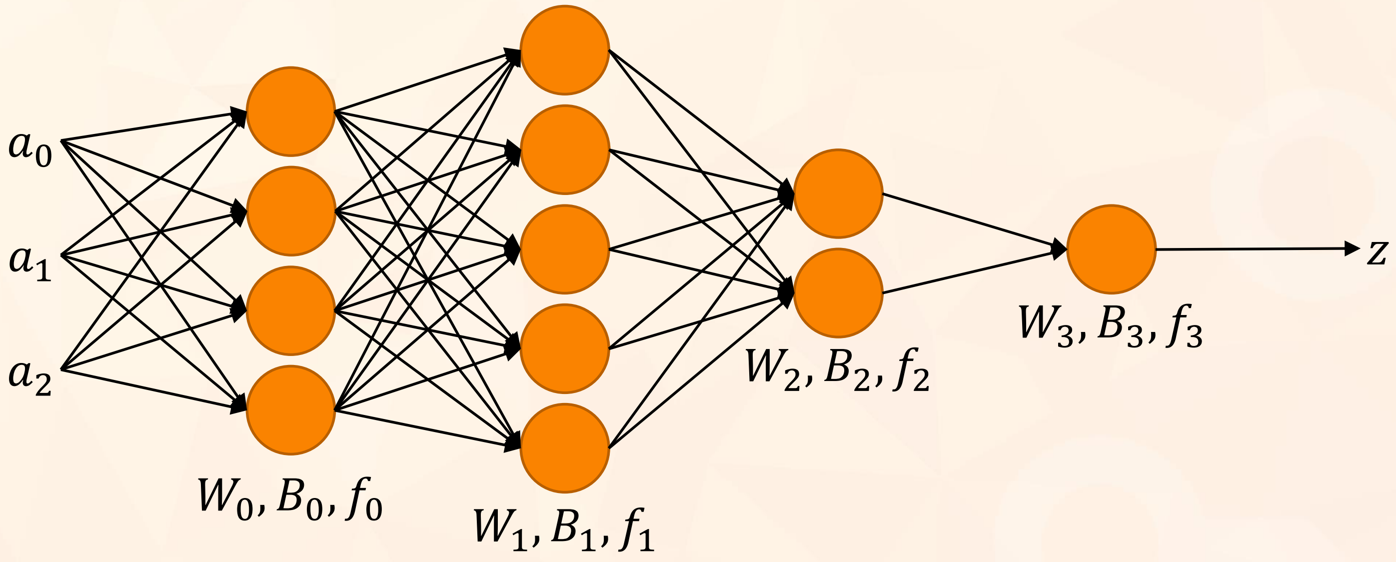
L'enfer commence.

Sommaire

1. Rappels
2. Principe de l'apprentissage
3. **Backpropagation**
4. Conclusion

Principe

$$\begin{pmatrix} a_0 \\ a_1 \\ a_2 \end{pmatrix} = A$$



$$(z) = Z$$

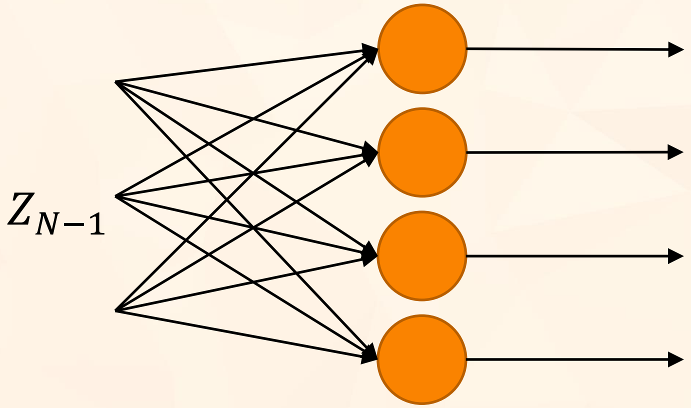
$\longrightarrow h(Z, Y)$

A $\quad Z_0 = f_0(A_0)$ $\quad Z_1 = f_1(A_1)$ $\quad Z_2 = f_2(A_2)$ $\quad Z = Z_3 = f_3(A_3)$

$\longleftarrow J(h)(Z)$

$\delta_0, \Delta W_0, \Delta B_0$ $\quad \delta_1, \Delta W_1, \Delta B_1$ $\quad \delta_2, \Delta W_2, \Delta B_2$ $\quad \delta_3, \Delta W_3, \Delta B_3$

Dernière couche



W_N, B_N, f_N

$Z_N = f_N(A_N)$

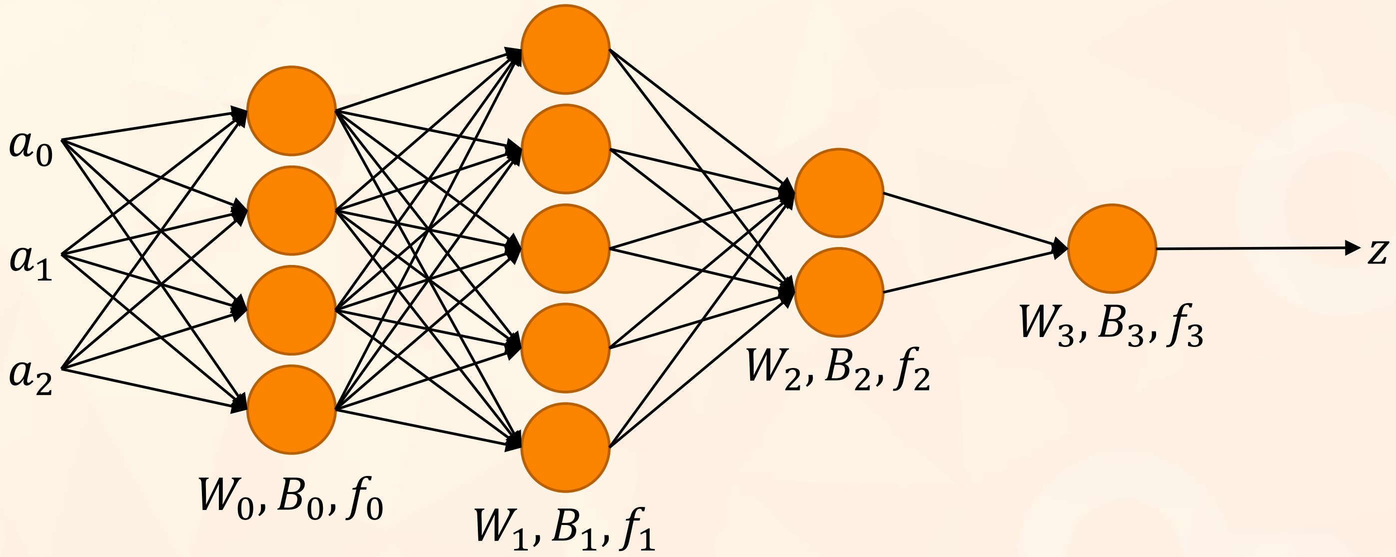
$h(Z_N, Y) = h(f_N(A_N), Y) = h(f_N(W_N . Z_{N-1} + B_N), Y)$

$$\left.\frac{\partial h}{\partial k}\right|_{Z_N} = \nabla h(Z_N).\left.\frac{\partial f_N}{\partial k}\right|_{A_N} = \begin{cases} \nabla h(Z_N).J(f_N)(A_N).\frac{\partial W_N}{\partial k}.Z_{N-1} \text{ } si \text{ } k \text{ } paramètre \text{ } de \text{ } W_N \\ \nabla h(Z_N).J(f_N)(A_N).\frac{\partial B_N}{\partial k} \text{ } si \text{ } k \text{ } paramètre \text{ } de \text{ } B_N \end{cases}$$

$$= \begin{cases} \delta_{N,i} z_j \text{ } si \text{ } k = w_{N,ij} \\ \delta_{N,i} \text{ } si \text{ } k = b_{N,i} \end{cases} avec \text{ } \delta_N = \nabla h(Z_N).J(f_N)(A_N)$$

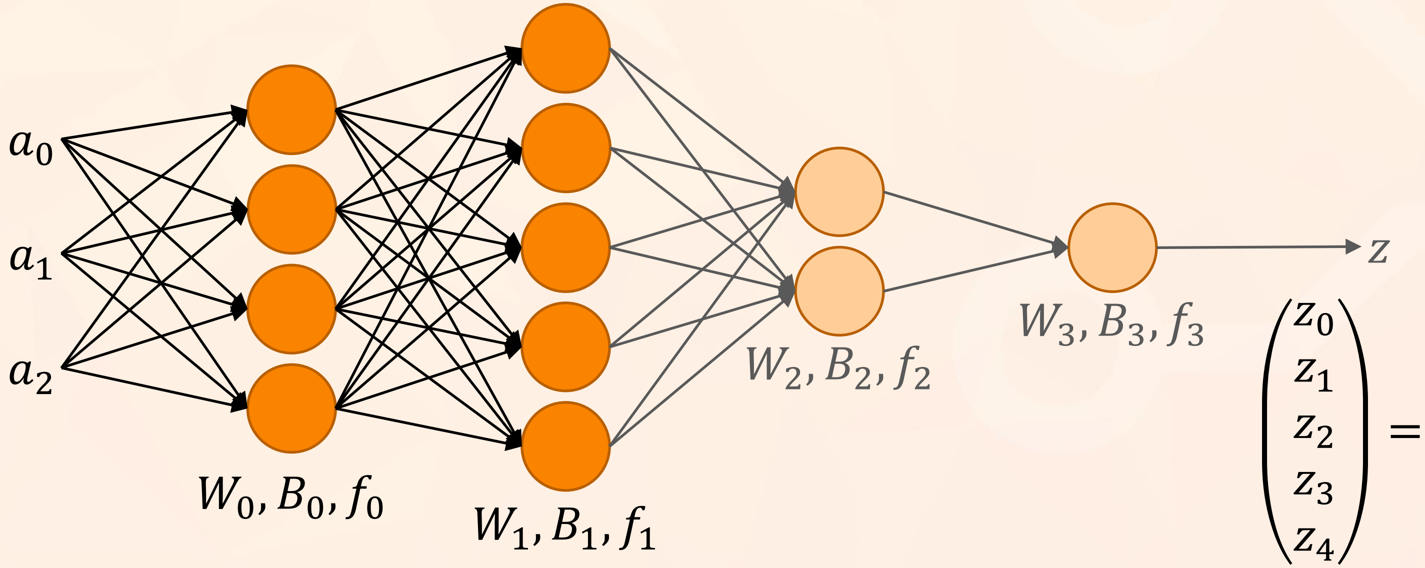
Principe pour les couches suivantes

$$\begin{pmatrix} a_0 \\ a_1 \\ a_2 \end{pmatrix} = A$$



$$(z) = Z \rightarrow h(Z, Y)$$

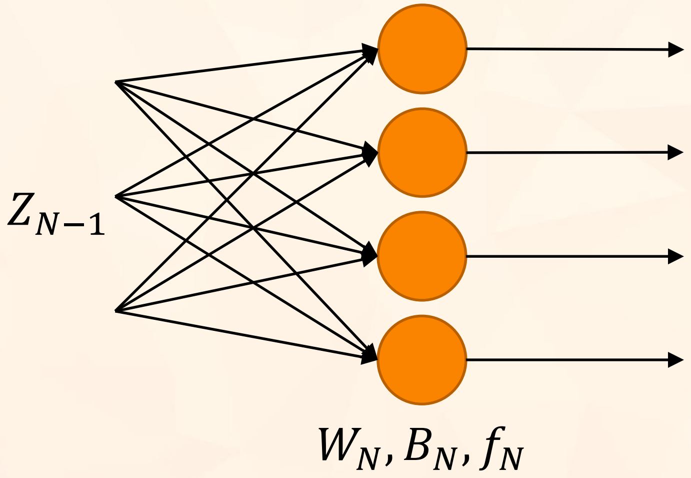
$$\begin{pmatrix} a_0 \\ a_1 \\ a_2 \end{pmatrix} = A$$



$$(z) = Z \rightarrow h(Z, Y)$$

$$\begin{pmatrix} z_0 \\ z_1 \\ z_2 \\ z_3 \\ z_4 \end{pmatrix} = Z_1 \rightarrow \widetilde{h_1}(Z_1, Y) = h(Z, Y)$$

Couche N avec couche N+1



$$Z_N = f_N(A_N)$$

$$\widetilde{h_N}(Z_N, Y) = \widetilde{h_N}(f_N(A_N), Y) = \widetilde{h_N}(f_N(W_N . Z_{N-1} + B_N), Y)$$
$$\widetilde{h_N}(Z_N, Y) = \widetilde{h_{N+1}}(f_{N+1}(W_{N+1} . Z_N + B_{N+1}), Y)$$

$$\left.\frac{\partial \tilde{h}_N}{\partial k}\right|_{Z_N} = \nabla\widetilde{h_N}(Z_N) . \left.\frac{\partial f_N}{\partial k}\right|_{A_N} = \begin{cases} \nabla\widetilde{h_N}(Z_N) . J(f_N)(A_N) . \dfrac{\partial W_N}{\partial k} . Z_{N-1} \text{ } si\ k\ paramètre\ de\ W_N \\ \nabla\widetilde{h_N}(Z_N) . J(f_N)(A_N) . \dfrac{\partial B_N}{\partial k} \text{ } si\ k\ paramètre\ de\ B_N \end{cases}$$

$$= \begin{cases} \delta_{N,i} z_j \ si\ k = w_{N,ij} \\ \delta_{N,i} \ si\ k = b_{N,i} \end{cases} avec\ \delta_N = \delta_{N+1} . W_{N+1} . J(f_N)(A_N)$$

$$\nabla\widetilde{h_N}(Z_N) = \nabla\widetilde{h_{N+1}}(Z_{N+1}) . J(f_{N+1})(A_{N+1}) . W_{N+1} = \delta_{N+1} . W_{N+1} \Rightarrow \delta_N = \delta_{N+1} . W_{N+1} . J(f_N)(A_N)$$

Finalité

1) Réseau de neurones de profondeur k :

- $Poids = \{W_0, \dots, W_{k-1}\}$
- $Biais = \{B_0, \dots, B_{k-1}\}$
- $Fcts. = \{f_0, \dots, f_{k-1}\}$

2) On effectue un feed-forward en conservant des valeurs :

- $Sums. = \{A_0, \dots, A_{k-1}\}$
- $Acts. = \{Z_{-1}, Z_0, \dots, Z_{k-1}\} = \{A, f_0(A_0), \dots, f_{k-1}(A_{k-1})\}$

3) Enfin, on effectue la backpropagation $n \in [\![0, k-1]\!]$:

$$\begin{cases} \delta_{k-1} = \nabla h(Z_{k-1}).J(f_{k-1})(A_{k-1}) \; si \; n = k-1 \\ \delta_n = \delta_{n+1}.W_{n+1}.J(f_n)(A_n) \; sinon \\ \Delta W_n = \left(dw_{n,ij}\right) = \left(\delta_{n,i} z_{n-1,j}\right) = \delta_n^t Z_{n-1}^t \\ \Delta B_n = \left(db_{n,i}\right) = \left(\delta_{n,i}\right) = \delta_n^t \end{cases}$$

4) Et mise à jour des poids et biais :

- $W_n = W_n - \alpha \Delta W_n$
- $B_n = B_n - \alpha \Delta B_n$

Batchs

$X \in (\mathbb{R}^n)^k$ → **f** Plein de paramètres → $Y \in (\mathbb{R}^m)^k$

Backpropagation

$\Delta W_n^{1,\ldots,k}, \Delta B_n^{1,\ldots,k}$

↓

$$W_n = W_n - \frac{\alpha}{k}\sum_{i=1}^{k} \Delta W_n^i$$

$$B_n = B_n - \frac{\alpha}{k}\sum_{i=1}^{k} \Delta B_n^i$$

Batchs et régularisation

$X \in (\mathbb{R}^n)^k$ → **f Plein de paramètres** → $Y \in (\mathbb{R}^m)^k$

Backpropagation

$\Delta W_n^{1,..,k}, \Delta B_n^{1,..,k}$

↓

$$W_n = W_n - \frac{\alpha}{k}\sum_{i=1}^{k} \Delta W_n^i - \lambda W_n$$

$$B_n = B_n - \frac{\alpha}{k}\sum_{i=1}^{k} \Delta B_n^i$$

Conclusion

Fini les maths.

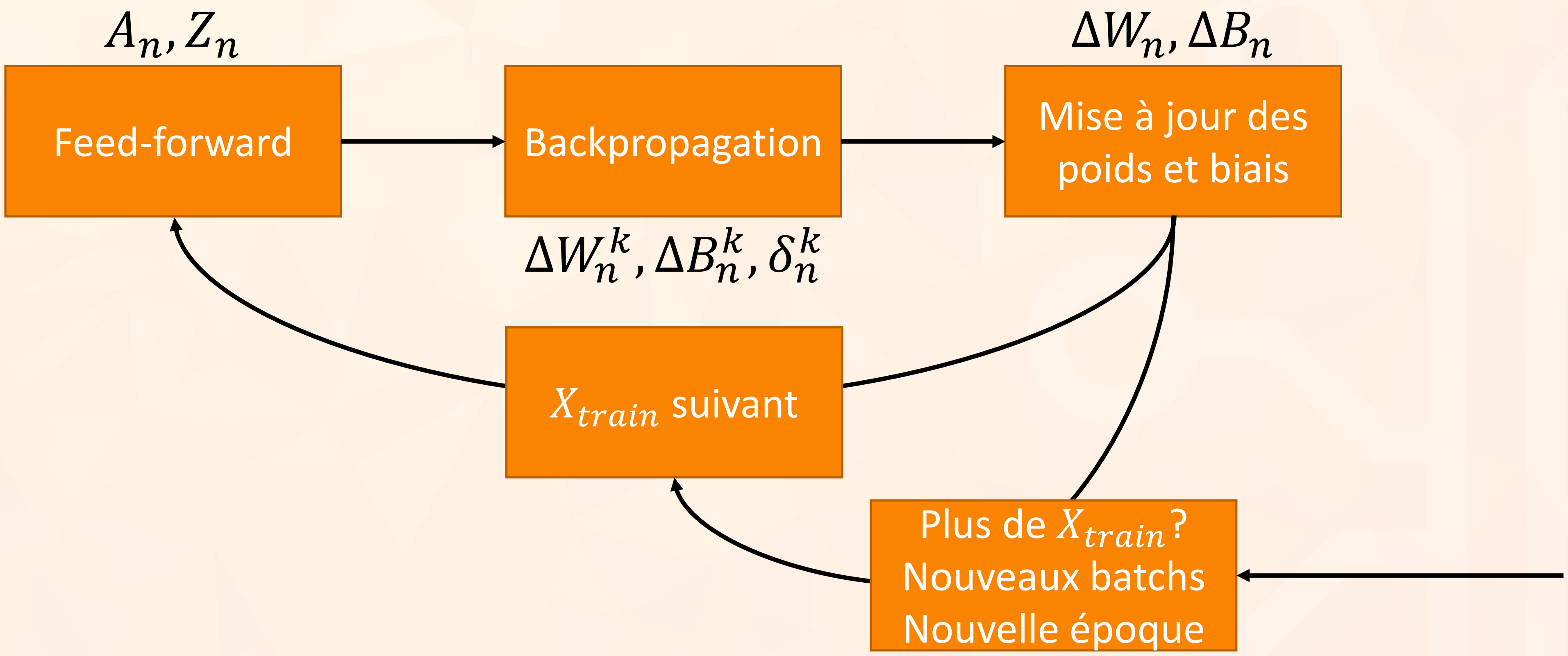
Sommaire

1. Rappels
2. Principe de l’apprentissage
3. Backpropagation
4. **Conclusion**

Pré-apprentissage

- Sélections des X_{train} et du roulement des valeurs d'entrainement.
- Mise en place des différents hyperparamètres :
  - α, la taille du pas d'apprentissage.
  - λ, l'importance de la régularisation.
  - k, la taille des batchs.
  - Le nombre d'époques.

Apprentissage



Merci d'avoir suivi cette formation

Des questions ?