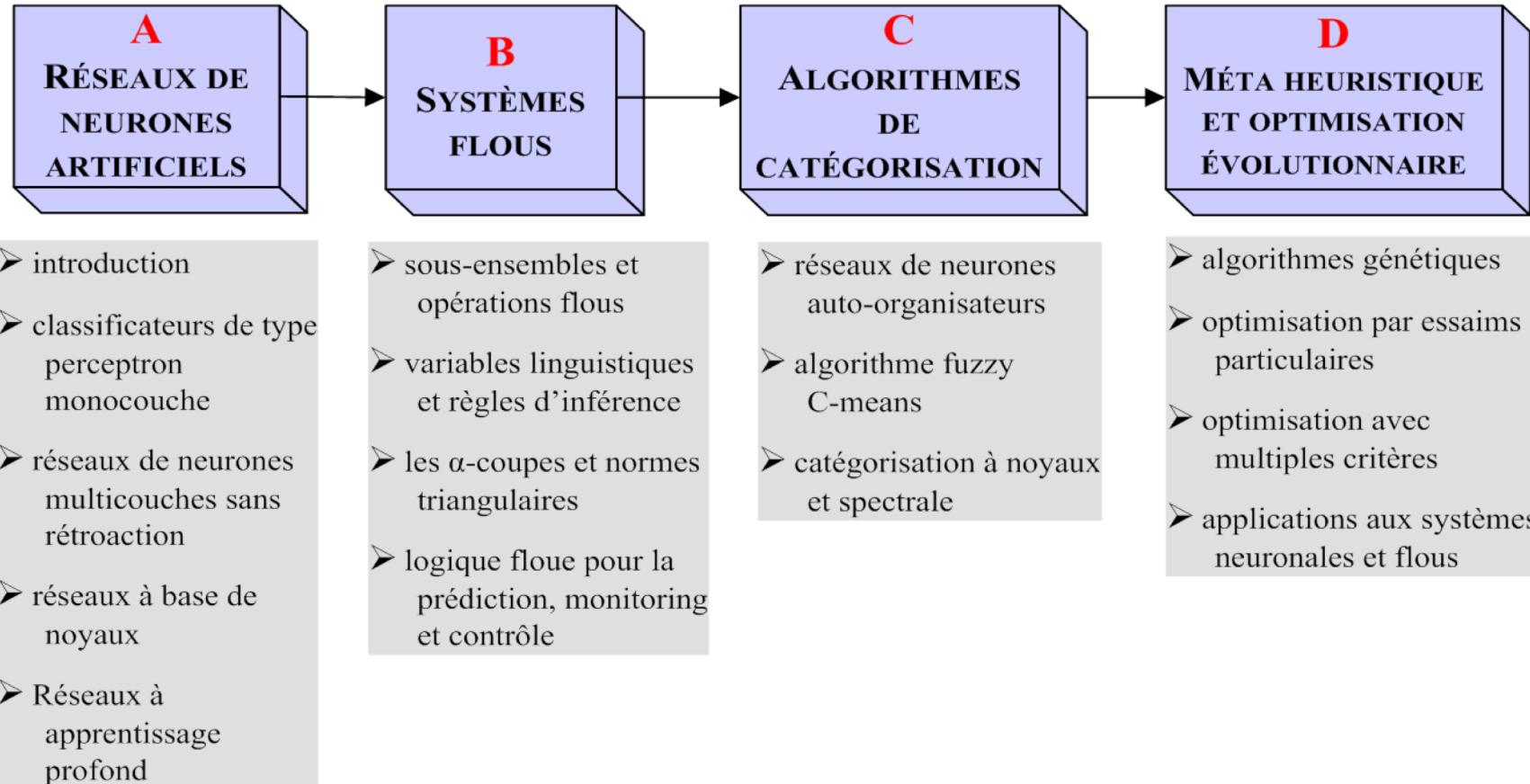


CONTENU DU COURS



CONTENU DU COURS

A.1 Introduction aux RNA:

1. Neurones biologiques et artificiels
2. Modèles de réseaux de neurones artificiels
3. Traitements effectués par les réseaux
4. Apprentissage et adaptation
5. Règles d'apprentissage

1) Neurones biologiques et artificiels

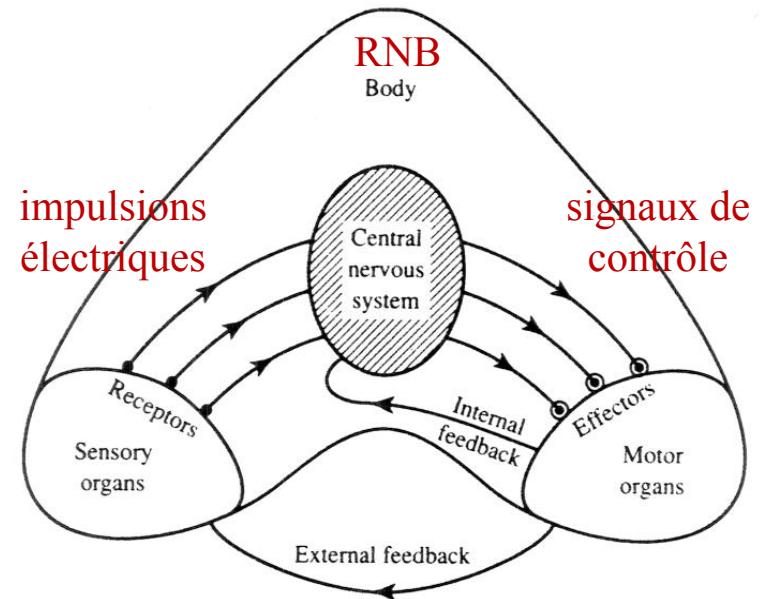
Cerveau humain

- ▶ **Le neurone est l'élément de base du cerveau humain**
 - Le cerveau humain est constitué d'environ 10^{11} neurones
 - Chaque neurone possède en moyenne 10^{14} synapses (connexions)
 - Les neurones communiquent entre eux au moyen d'impulsions électriques, à travers d'axones et de synapses
 - En fait, le cerveau humain peut être considéré comme un vaste **réseau de neurones** élémentaires très fortement connectés

1) Neurones biologiques et artificiels

Cerveau humain

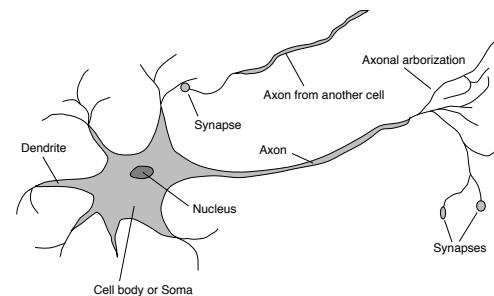
- ▶ **Cheminement de l'information dans le système nerveux**
- Les **entrées du réseau** (récepteurs sensoriels) sont activées par des stimuli (impulsions électriques) qui se chargent de transporter l'information au réseau de neurones.
- Le **système nerveux central (RNB)** traite l'information reçue et alimente ou actionne les organes



1) Neurones biologiques et artificiels

Neurones biologiques

- La cellule nerveuse élémentaire (neurone) est composée de trois régions distinctes:
 - entrées: les dendrites – arbre d'axones qui reçoit l'information d'autres neurones à travers d'axones
 - le corps cellulaire (**soma**) – unité de traitement
 - sortie: l'**axone** – connexion qui transporte l'impulsion du neurone aux voisins



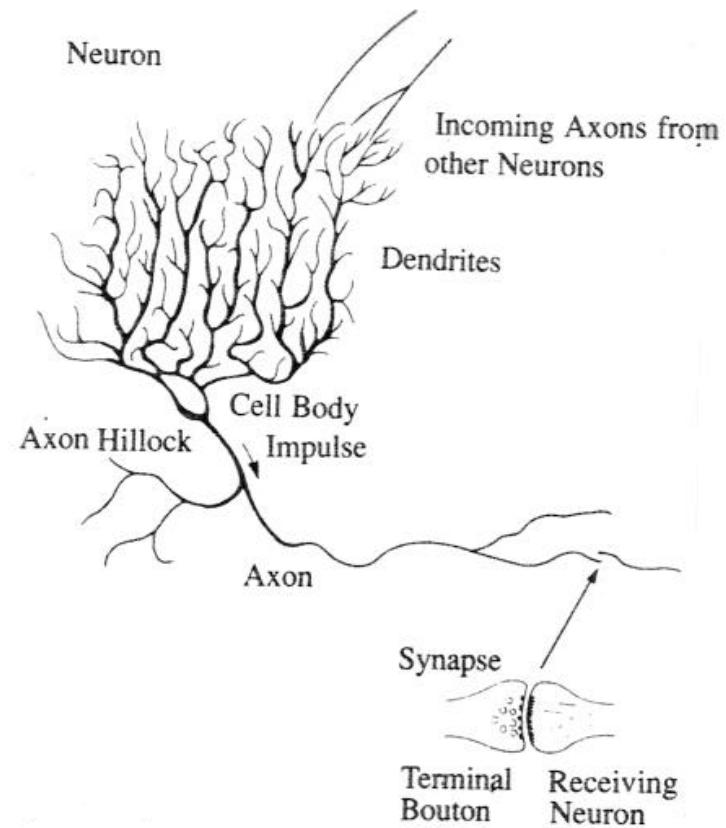
1) Neurones biologiques et artificiels

Neurones biologiques

► Diagramme de la cellule nerveuse élémentaire:

Poids = potentiel qui est associé avec le synapse, l'organe de contact entre axone-dendrites

- synapse excitateur ($w +$)
- synapse inhibiteur ($w -$)

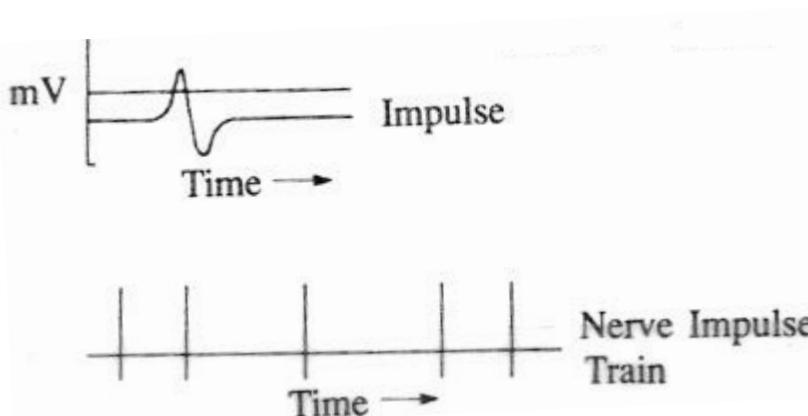


1) Neurones biologiques et artificiels

Neurones biologiques

► Conditions d'activation d'un neurone:

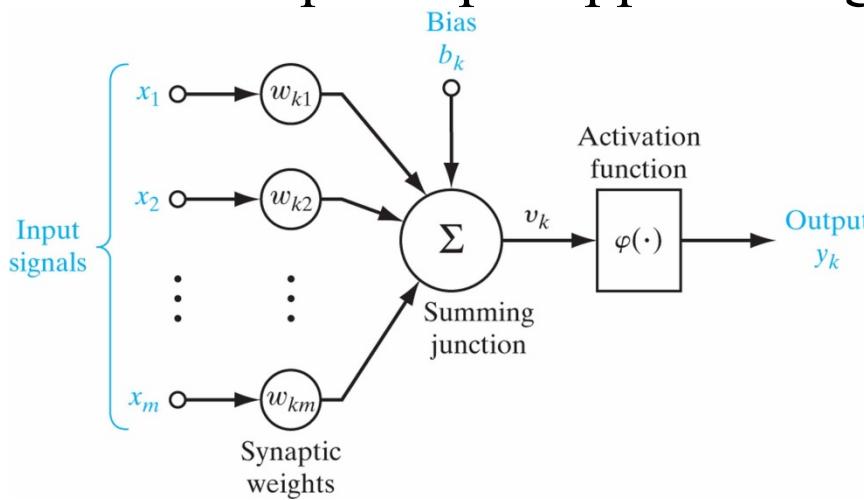
- dépend de l'ensemble d'entrée (dendrites) durant un courte période de temps (ordre de la msec)
- si le potentiel du neurone en mV dépasse un niveau: l'ensemble d'impulsions excitatrices surpassent l'ensemble d'impulsions inhibitrices par un seuil
- le neurone génère une impulsion sur son axone (sortie)



1) Neurones biologiques et artificiels

Neurones artificiels

- **Structure de calcul:** un RNA est composé d'unités de traitement très simples et massivement interconnectés
 - un poids associé à chaque interconnexion permet de stocker les connaissances acquises par apprentissage



Source: S. Haykin, *Neural Networks and Machine Learning*, 3ème ed., Prentice Hall, 2009.

1) Neurones biologiques et artificiels

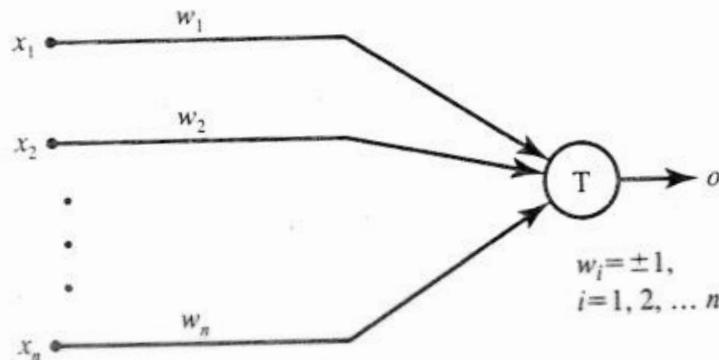
Neurones artificiels

- **Optimisation des paramètres:**
 - processus d'apprentissage – permet de modifier les poids du RNA selon un objectif
 - ce sont des systèmes adaptatifs qui *apprennent* à partir d'exemples
 - ils sont entraînés avec de *données réelles*
- **Fonction:** permettent de résoudre des problèmes de reconnaissance de formes, de contrôle, d'estimation, d'optimisation, de prédiction, de vision artificielle, etc.

1) Neurones biologiques et artificiels

Neurone artificiel de McCulloch-Pitts

- ▶ Ce modèle a été proposé par *McCulloch et Pitts* en 1943:
 - ‘A Logical Calculus of the Ideas Immanent in Nervous Activity,’ *Bulletin of Mathematical Biophysics*, 5:115-133, 1943.
- Soit n entrées x_i pouvant prendre les valeurs binaires $\{0, 1\}$ représentant l'absence ou la présence de stimulus (impulsions) à $k = 1, 2, \dots$ instant discrets.



1) Neurones biologiques et artificiels

Neurone artificiel de McCulloch-Pitts

- La sortie o du neurone à l'instant $k+1$ correspond à la règle d'activation suivante :

$$o^{k+1} = \begin{cases} 1 & \text{si } \sum_{i=1}^n w_i x_i^k \geq T \\ 0 & \text{si } \sum_{i=1}^n w_i x_i^k < T \end{cases}$$

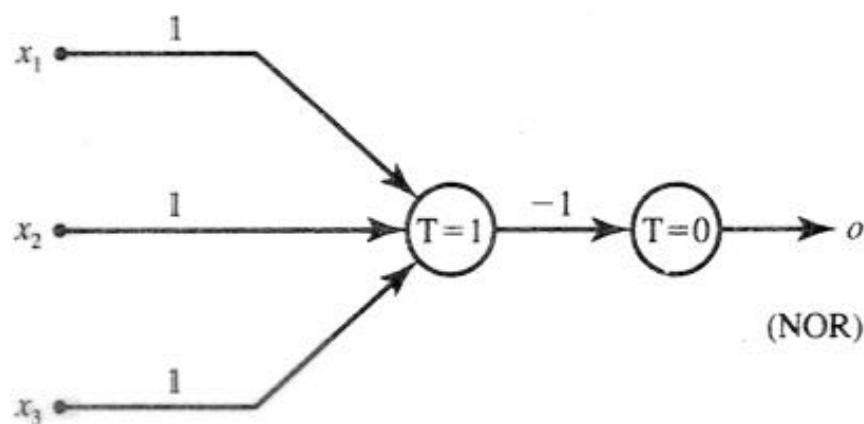
- Un poids $w_i = +1$ indique que le synapse est activable (exitatoire), tandis que $w_i = -1$ indique un synapse inhibiteur.

1) Neurones biologiques et artificiels

Neurone artificiel de McCulloch-Pitts

► Réalisation de circuits logiques combinatoires:

- Ce modèle simplifié de neurone permet cependant la mise en oeuvre des opérations logiques de base: NOT, OR, AND, etc.
- Exemple: porte NOR



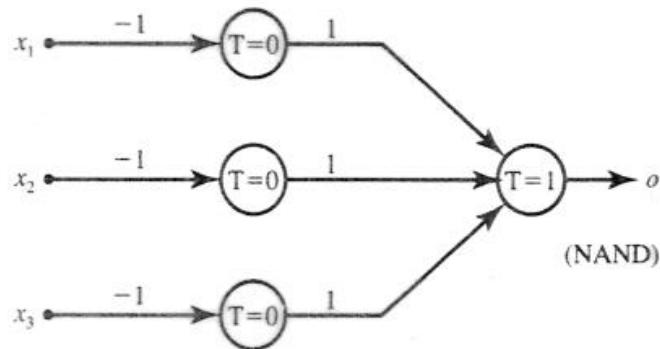
NOR			
x_1	x_2	x_3	o
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

1) Neurones biologiques et artificiels

Neurone artificiel de McCulloch-Pitts

► Réalisation de circuits logiques combinatoires:

- Ce modèle simplifié de neurone permet cependant la mise en oeuvre des opérations logiques de base: NOT, OR, AND, etc.
- Exemple: porte NAND



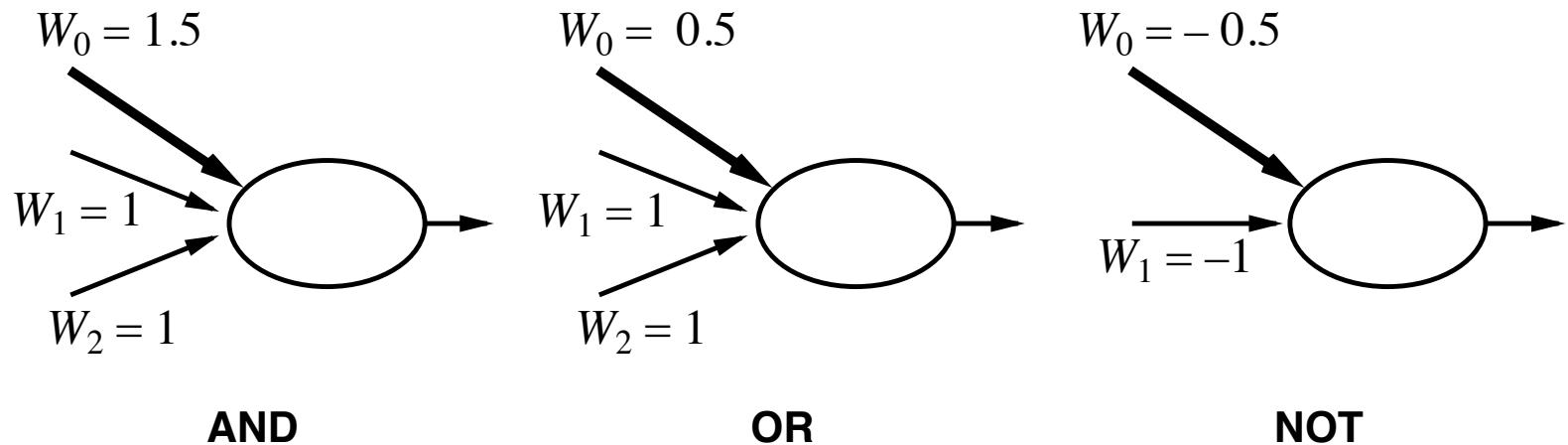
NAND			
x_1	x_2	x_3	o
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

1) Neurones biologiques et artificiels

Neurone artificiel de McCulloch-Pitts

► Réalisation de circuits logiques combinatoires:

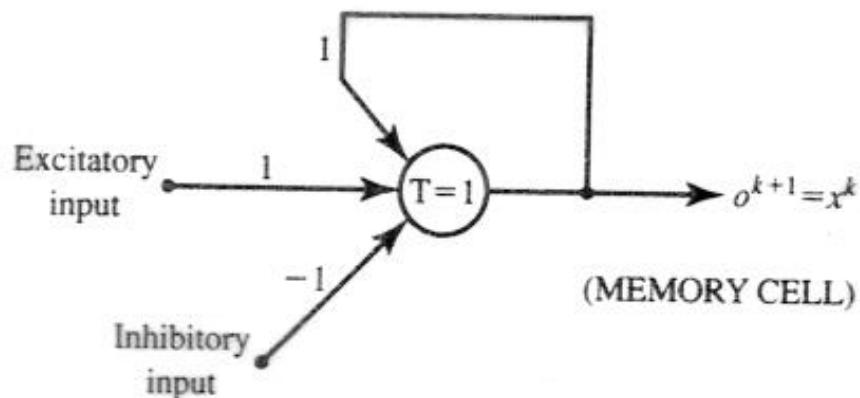
- Ce modèle simplifié de neurone permet cependant la mise en oeuvre des opérations logiques de base: NOT, OR, AND, etc.



1) Neurones biologiques et artificiels

Neurone artificiel de McCulloch-Pitts

- ▶ **Réalisation d'une cellule de mémoire (bascule):**
- **Délai unitaire:** Si le délai de propagation inhérent à ce modèle est considéré, un neurone avec une seule entrée x , un poids w et un seuil T de valeur $w = T = 1$ est équivalent à l'implantation d'un registre avec $o^{k+1} = x^k$
- Ce neurone avec rétroaction représente une mémoire simple:



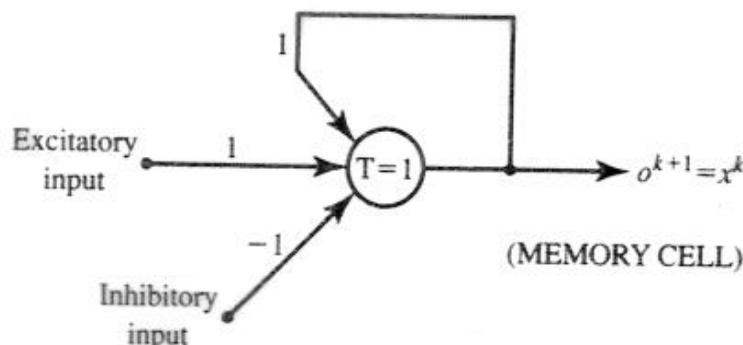
$$o^{k+1} = \begin{cases} 1 & \text{si } x_1^k + (o^k - x_2^k) \geq 1 \\ 0 & \text{si } x_1^k + (o^k - x_2^k) < 1 \end{cases}$$

1) Neurones biologiques et artificiels

Neurone artificiel de McCulloch-Pitts

► Réalisation d'une cellule de mémoire (bascule):

1. une valeur un (1) placée à l'entrée **excitatrice** x_1 initialise la phase **d'activation** de cette cellule,
2. la valeur de sortie de ce neurone demeure constante en **l'absence de stimulus** aux entrées,
3. une valeur un (1) placée à l'entrée **inhibitrice** x_2 initialise la phase d'inaction de cette cellule.



x_1^k	x_2^k	o^k	o^{k+1}
0	0	0	0 ⁽²⁾
1	0	0	1 ⁽¹⁾
0	0	1	1 ⁽²⁾
1	0	1	1 ⁽¹⁾
0	1	0	0 ⁽³⁾
1	1	0	0 ⁽³⁾
0	1	1	0 ⁽³⁾
1	1	1	1 ⁽³⁾

1) Neurones biologiques et artificiels

Neurone artificiel de McCulloch-Pitts

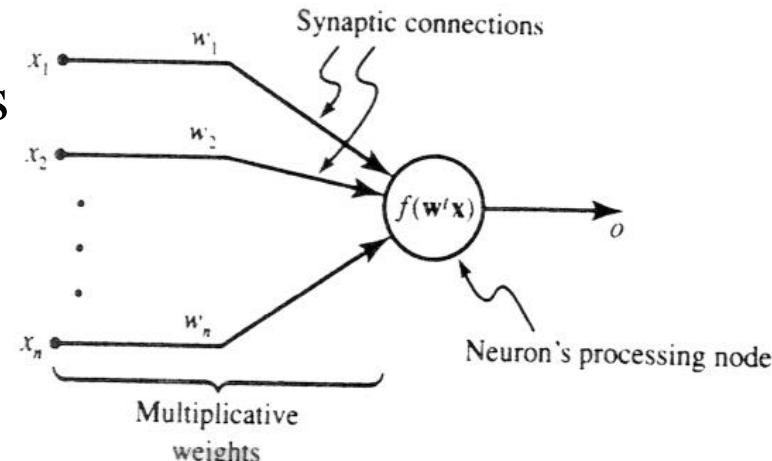
- ▶ **Ce modèle de neurone est simple, et le formalisme est élégant, mais ce modèle possède les limitations suivantes:**
 - les entrées x_i sont binaires (limitées aux valeurs 0 et 1),
 - une base de temps discrète $k = 1, 2, \dots$ est considérée,
 - Le fonctionnement de tous les neurones est supposé synchrone,
 - Les poids w_i et les seuils T sont fixes (aucun apprentissage),
 - Il n'y a pas d'interactions entre les neurones, sauf pour ce qui est du transfert des données.

1) Neurones biologiques et artificiels

Modèle général du neurone artificiel

- ▶ Symbole général d'un neurone artificiel avec plusieurs connexions synaptiques x_n , les poids w_n et une sortie o donné par la relation:

$$o = f(\mathbf{w}^t \mathbf{x}), \quad o = f \left(\sum_{i=1}^n w_i x_i \right)$$



- le vecteur de poids est défini par: $\mathbf{w} \doteq [w_1 \ w_2 \ \dots \ w_n]^t$
- le vecteur d'entrées est représenté par: $\mathbf{x} \doteq [x_1 \ x_2 \ \dots \ x_n]^t$

1) Neurones biologiques et artificiels

Modèle général du neurone artificiel

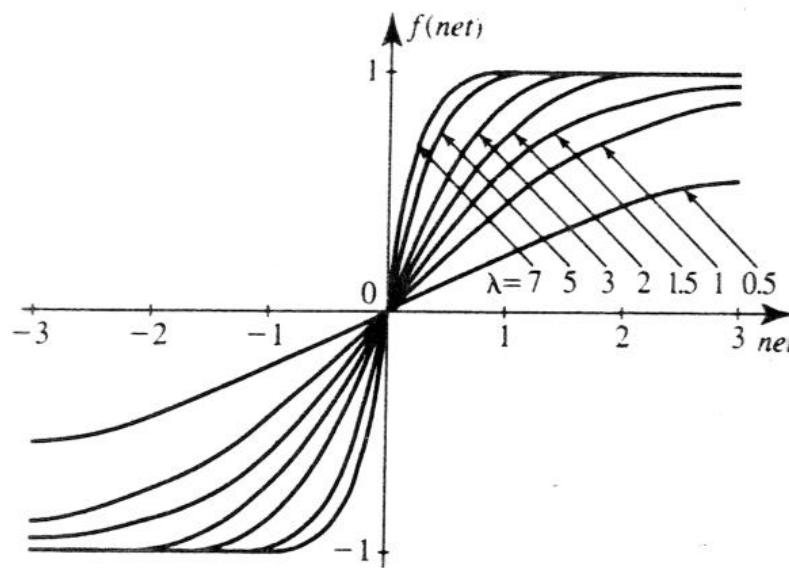
- ▶ La **fonction d'activation** du neurone est $f(\text{net}) = f(\mathbf{w}^t \mathbf{x})$.
 - son domaine est l'ensemble des valeurs d'activation représenté par la variable *net*
 - alors nous avons la fonction $f(\text{net})$ où *net* est le produit scalaire des vecteurs \mathbf{w} et \mathbf{x} : $\text{net} \doteq \mathbf{w}^t \mathbf{x}$
- ▶ Pour le moment, nous supposons que :
 - les arguments de la variable *net* sont continus;
 - les entrées sont constituées des $n-1$ premières connexions, soit, x_1, x_2, \dots, x_{n-1} ;
 - la valeur de l'entrée $x_n = -1$, et celle du poids correspondant $w_n = T$.

1) Neurones biologiques et artificiels

Modèle général du neurone artificiel

► Exemple de fonction d'activation bipolaire continue (\tanh)

$$f(\text{net}) \doteq \frac{2}{1 + \exp(-\lambda \text{net})} - 1 \quad \text{avec} \quad \lambda > 0$$

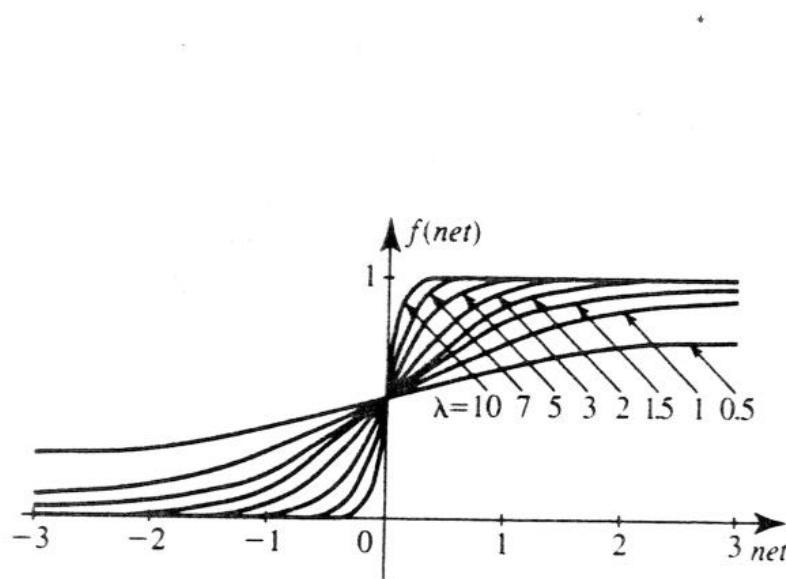


1) Neurones biologiques et artificiels

Modèle général du neurone artificiel

- Exemple de fonction d'activation unipolaire continue (sigmoïde)

$$f(\text{net}) \doteq \frac{1}{1 + \exp(-\lambda \text{net})}, \quad \text{avec } \lambda > 0$$



1) Neurones biologiques et artificiels

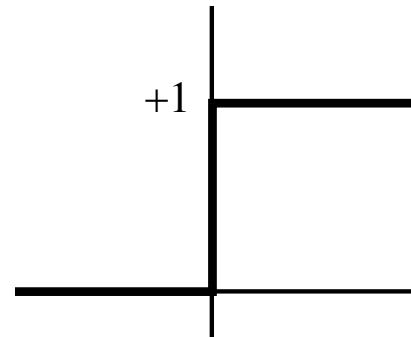
Modèle général du neurone artificiel

- Exemple de fonction d'activation bipolaire binaire (non défini à $net = 0$)

$$f(net) \doteq sgn(net) = \begin{cases} +1, & net > 0 \\ -1, & net < 0 \end{cases}$$

- Exemple de fonction d'activation unipolaire binaire (non défini à $net = 0$)

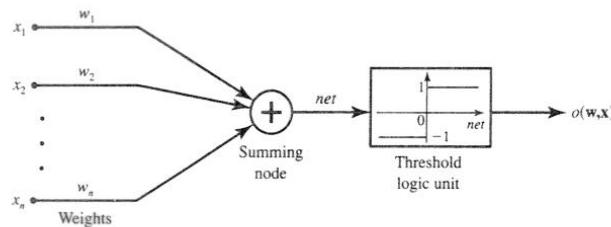
$$f(net) \doteq \begin{cases} 1, & net > 0 \\ 0, & net < 0 \end{cases}$$



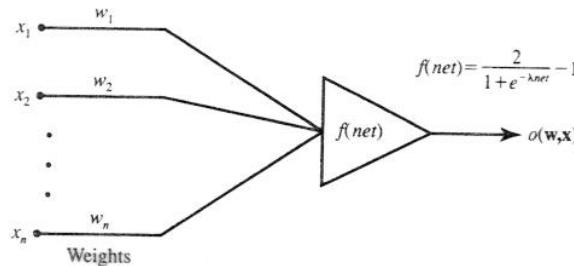
1) Neurones biologiques et artificiels

Modèle du neurone Perceptron

- ▶ Représentation graphique standard des modèles proposée par Zurada (1992)
 - perceptron discret (fonction d'activation binaire bipolaire)



- perceptron continu (fonction d'activation continue bipolaire)



Avec ces modèles, les sorties peuvent être discrètes ou continues.

1) Neurones biologiques et artificiels

Modèle du neurone Perceptron

► Domaine du vecteur o de sortie d'une couche de neurones:

- Étant donnée une couche de m neurones, les sorties o_1, o_2, \dots, o_m sont représentées par le vecteur des sorties: $\mathbf{o} \equiv [o_1, o_2, \dots, o_m]^t$
- le **domaine des vecteurs de sorties o** correspondant aux fonctions d'activation *continues* peut être représenté par *l'intérieur d'un hypercube* défini dans R^m tel que : $(-1, 1)^m \equiv \{o \in R^m, o_i \in (-1,1)\}$ ou

$$(0, 1)^m \equiv \{o \in R^m, o_i \in (0,1)\}$$

- le **domaine des vecteurs de sorties o** correspondant aux fonctions d'activation *binaires* peut être représenté par *les coins d'un hypercube* défini dans R^m tel que : $\{-1, 1\}^m \equiv \{o \in R^m, o_i \in \{-1,1\}\}$ ou

$$\{0, 1\}^m \equiv \{o \in R^m, o_i \in \{0,1\}\}$$

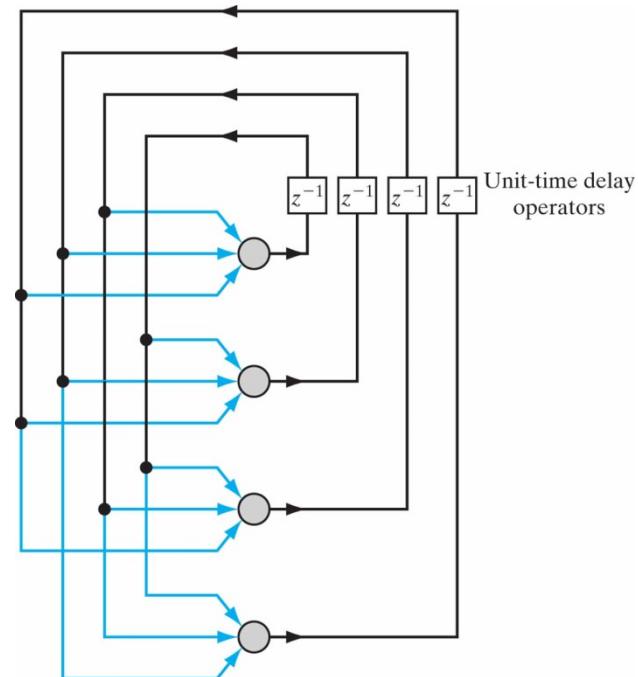
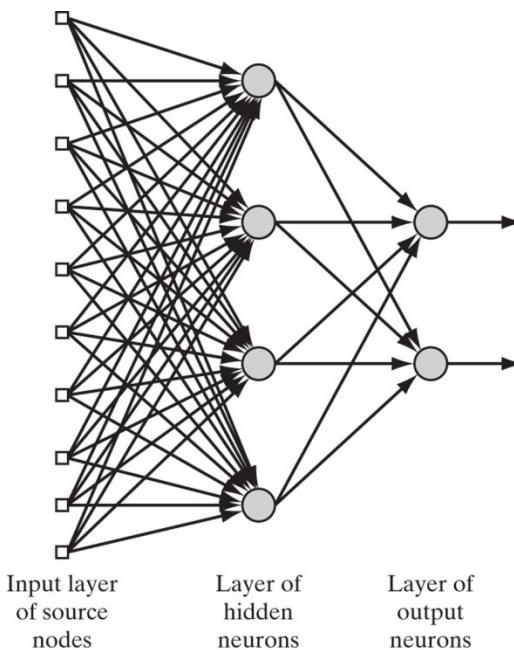
CONTENU DU COURS

A.1 Introduction aux RNA:

1. Neurones biologiques et artificiels
2. Modèles de réseaux de neurones artificiels
3. Traitements effectués par les réseaux
4. Apprentissage et adaptation
5. Règles d'apprentissage

2) Modèles de réseaux de neurones

- ▶ Réseaux de neurones artificiels (RNA):
- Structure de calcul: réseaux récurrents et non récurrents



Source: S. Haykin, *Neural Networks and Machine Learning*, 3ème ed., Prentice Hall, 2009.

2) Modèles de réseaux de neurones

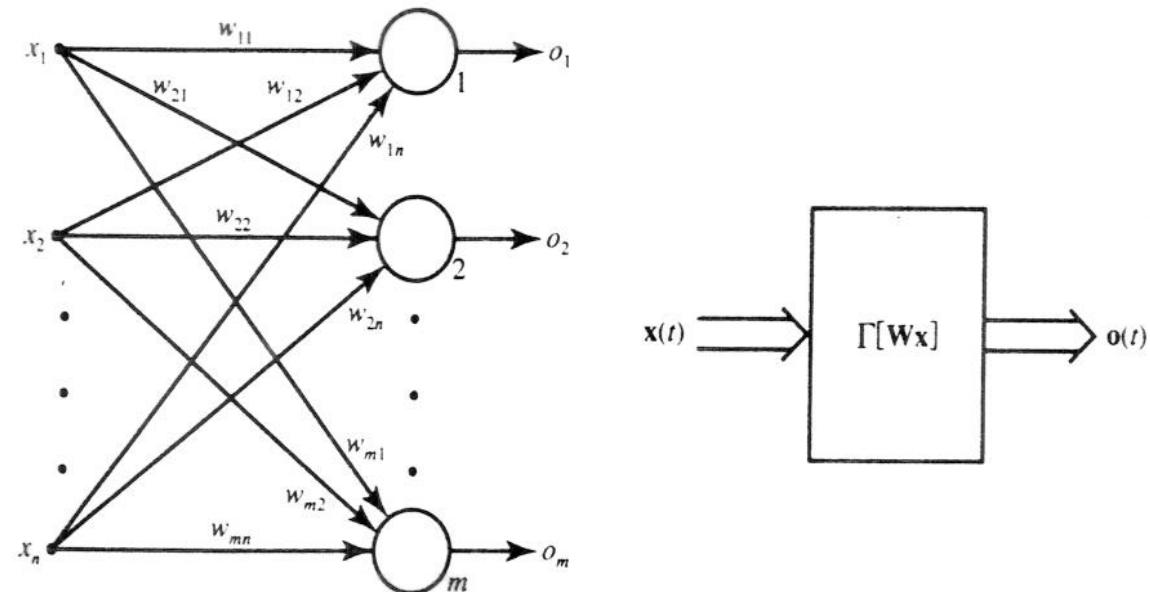
Réseau de neurones sans rétroaction (feedforward)

► Structure générale:

- Soit l'architecture du réseau constitué de m neurones et n entrées. Alors nous avons les vecteurs d'entrées x et de sorties o suivants:

$$x = [x_1, x_2, \dots, x_n]^t$$

$$o = [o_1, o_2, \dots, o_m]^t$$



2) Modèles de réseaux de neurones

Réseau de neurones sans rétroaction (*feedforward*)

► **Poids synaptiques:**

- le poids w_{ij} est associé à la connexion qui relie le $i^{\text{ième}}$ neurone à la $j^{\text{ième}}$ entrée (notation standard utilisée : $w_{\text{destination-source}}$).
- le vecteur de poids \mathbf{w}_i correspondant au neurone i est:

$$\mathbf{w}_i = [w_{i1}, w_{i2}, \dots, w_{in}]^t$$

- où \mathbf{W} représente la matrice des poids :

$$\mathbf{W} \doteq \begin{bmatrix} w_{11} & w_{12} & \dots & w_{1n} \\ w_{21} & w_{22} & \dots & w_{2n} \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & \dots & \cdot \\ \cdot & \cdot & & \cdot \\ w_{m1} & w_{m2} & \dots & w_{mn} \end{bmatrix}$$

2) Modèles de réseaux de neurones

Réseau de neurones sans rétroaction (*feedforward*)

► Fonction d'activation:

- La valeur d'intégration du $i^{\text{ième}}$ neurone est obtenue par:

$$net_i = \sum_{j=1}^n w_{ij} x_j, \quad \text{pour } i = 1, 2, \dots, m$$

- La transformation non-linéaire $f(net_i)$ est appliquée aux m neurones. Alors la sortie du $i^{\text{ième}}$ neurone est obtenue par:

$$o_i = f(w_i^t x), \quad \text{pour } i = 1, 2, \dots, m$$

2) Modèles de réseaux de neurones

Réseau de neurones sans rétroaction (*feedforward*)

► Description compact:

- Soit l'opérateur matriciel non-linéaire Γ implanté par le réseau de neurones précédent. Cet opérateur permet d'effectuer la correspondance entre l'espace d'entrée x et l'espace de sortie o .

$$o = \Gamma[Wx]$$

- Γ est la matrice des transformations non-linéaires associées à chaque neurone du réseau:

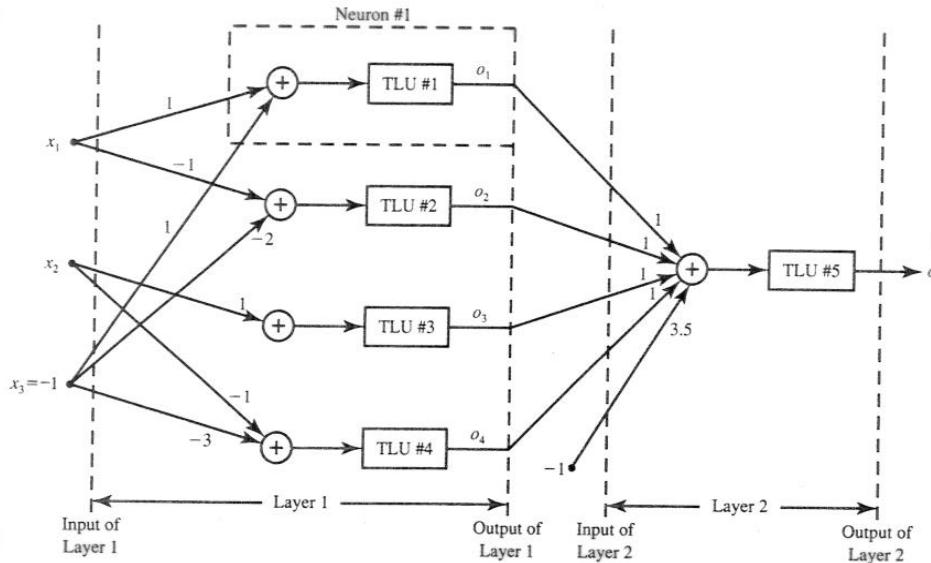
$$\Gamma[\cdot] \doteq \begin{bmatrix} f(\cdot) & 0 & \dots & 0 \\ 0 & f(\cdot) & \dots & 0 \\ \vdots & \vdots & & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \dots & f(\cdot) \end{bmatrix}$$

2) Modèles de réseaux de neurones

Réseau de neurones sans rétroaction (*feedforward*)

- **Exemple:** Réseau de neurones sans rétroaction (2 couches), avec la fonction d'activation binaire bipolaire:

$$f(\text{net}) \doteq \text{sgn}(\text{net}) = \begin{cases} +1, & \text{net} > 0 \\ -1, & \text{net} < 0 \end{cases}$$



2) Modèles de réseaux de neurones

Réseau de neurones sans rétroaction (*feedforward*)

- **Question:** Déterminez la sortie o_5 analytiquement en fonction de l'entrée x , sachant que chaque couche du réseau est décrite par $\mathbf{o} = \Gamma[\mathbf{Wx}]$

couche 1

$$\mathbf{o} = [o_1 \ o_2 \ o_3 \ o_4]^t$$

$$\mathbf{x} = [x_1 \ x_2 \ -1]^t$$

$$\mathbf{W}_1 = \begin{bmatrix} 1 & 0 & 1 \\ -1 & 0 & -2 \\ 0 & 1 & 0 \\ 0 & -1 & -3 \end{bmatrix}$$

couche 2

$$\mathbf{o} = [o_5]$$

$$\mathbf{x} = [o_1 \ o_2 \ o_3 \ o_4 \ -1]^t$$

$$\mathbf{W}_2 = [1 \ 1 \ 1 \ 1 \ 3.5]$$

2) Modèles de réseaux de neurones

Réseau de neurones sans rétroaction (*feedforward*)

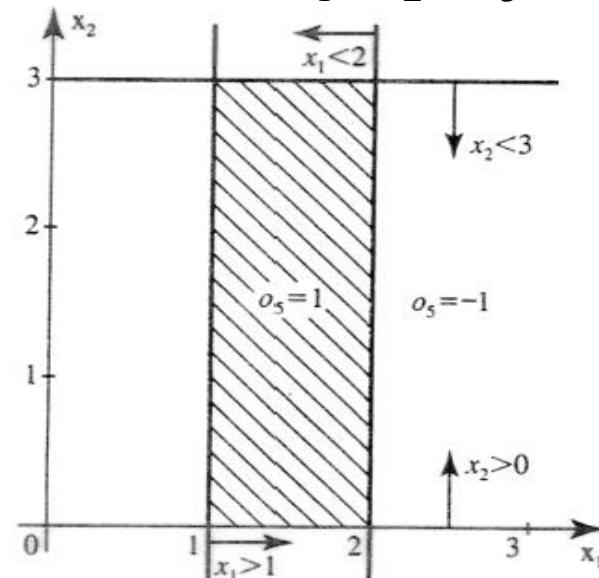
- La sortie de chaque couche est obtenue par les relations:

couche 1: $\mathbf{o} = [sgn(x_1 - 1) \quad sgn(-x_1 + 2) \quad sgn(x_2) \quad sgn(-x_2 + 3)]^t$

couche 2: $o_5 = sgn(o_1 + o_2 + o_3 + o_4 - 3.5)$

∴ la sortie o_5 du réseau prend pour valeur +1 ssi $o_1=o_2=o_3=o_4=1$

- Interprétation géométrique de la fonction implantée par le réseau :



2) Modèles de réseaux de neurones

Réseau de neurones sans rétroaction (*feedforward*)

- **Exemple:** Réseau de neurones sans rétroaction (2 couches), avec la fonction d'activation bipolaire continue:

$$f(\text{net}) \doteq \frac{2}{1 + \exp(-\lambda \text{net})} - 1 \quad \text{avec} \quad \lambda > 0$$

- La sortie de chaque couche est obtenue par les relations:

couche 1

couche 2

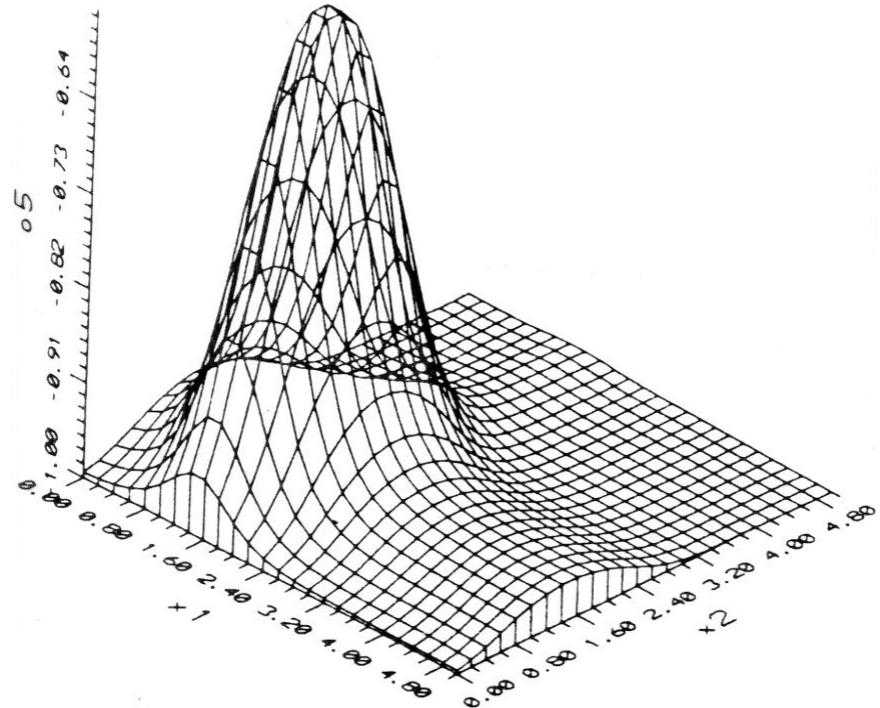
$$\boldsymbol{o} = \begin{bmatrix} \frac{2}{1 + \exp(1 - x_1)\lambda} - 1 \\ \frac{2}{1 + \exp(x_1 - 2)\lambda} - 1 \\ \frac{2}{1 + \exp(-x_2)\lambda} - 1 \\ \frac{2}{1 + \exp(x_2 - 3)\lambda} - 1 \end{bmatrix}$$

$$o_5 = \frac{2}{1 + \exp(3.5 - o_1 - o_2 - o_3 - o_4)\lambda} - 1$$

2) Modèles de réseaux de neurones

Réseau de neurones sans rétroaction (*feedforward*)

- L'espace de représentation de la fonction implantée par le réseau de neurones avec l'utilisation de fonctions d'activation bipolaires continues et $\lambda=2.5$ correspond à:
- Interprétation géométrique de la fonction implantée par le réseau :

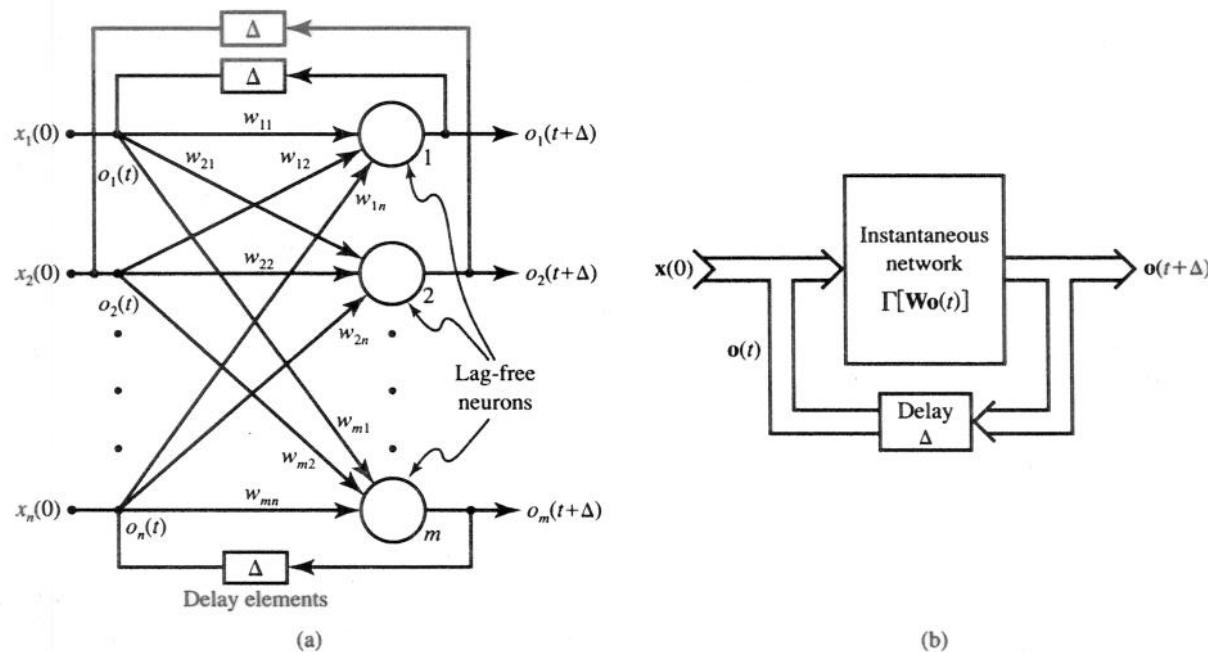


2) Modèles de réseaux de neurones

Réseau de neurones avec rétroaction (*feedback*)

► Structure générale:

- Comme le cas sans rétroaction, sauf qu'on connecte les sorties de l'état précédent aux entrées



2) Modèles de réseaux de neurones

Réseau de neurones avec rétroaction (*feedback*)

► Fonctionnement général:

- l'entrée $x(t)$ est requise simplement pour initialiser le réseau à la valeur $o(0) = x(0)$
- l'entrée est ensuite enlevée, et le système agit de façon autonome pour $t > 0$.
- la sortie est évaluée par la relation $o(t + \Delta) = \Gamma [W o(t)]$
- le réseau est *récurrent* car la réponse au temps $t + \Delta$ dépend de toutes les réponses données par le réseau depuis $t = 0$.
- Habituellement, les réseaux récurrents utilisent des entrées et des fonctions d'activation binaires.

2) Modèles de réseaux de neurones

Réseau de neurones avec rétroaction (*feedback*)

► Système à temps discret (on observe à des Δ fixes):

- Lorsque le réseau utilise une base de temps discrète, le symbole Δ représente un délai unitaire. La notation suivante est couramment utilisée pour modéliser ce type de réseaux :

$$\mathbf{o}^{k+1} = \Gamma [W \mathbf{o}^k]$$

- Le réseau est *récurrent* car la réponse au temps $k+1$ dépend de toutes les réponses données par le réseau depuis $k=0$

$$\mathbf{o}^1 = \Gamma[W\mathbf{x}^0]$$

$$\mathbf{o}^2 = \Gamma[W\Gamma[W\mathbf{x}^0]]$$

...

$$\mathbf{o}^{k+1} = \Gamma[W\Gamma[\dots\Gamma[W\mathbf{x}^0]\dots]]$$

2) Modèles de réseaux de neurones

Réseau de neurones avec rétroaction (*feedback*)

- ▶ **Système à temps discret (on observe à des Δ fixes):**
 - L'équation précédente décrit l'état \mathbf{o}^k du réseau aux instants $k = 1, 2, \dots$, et produit la séquence des transitions d'états du réseau.
 - La séquence des transitions d'états débute à l'instant 0 avec \mathbf{x}^0 , et passe par les transitions d'états \mathbf{o}^k pour $k = 1, 2, \dots$, jusqu'à ce que le réseau trouve éventuellement un état d'équilibre.
 - L'état d'équilibre s'appelle communément *attracteur*, qui peut être associé à un seul état, ou à un nombre limité d'états.

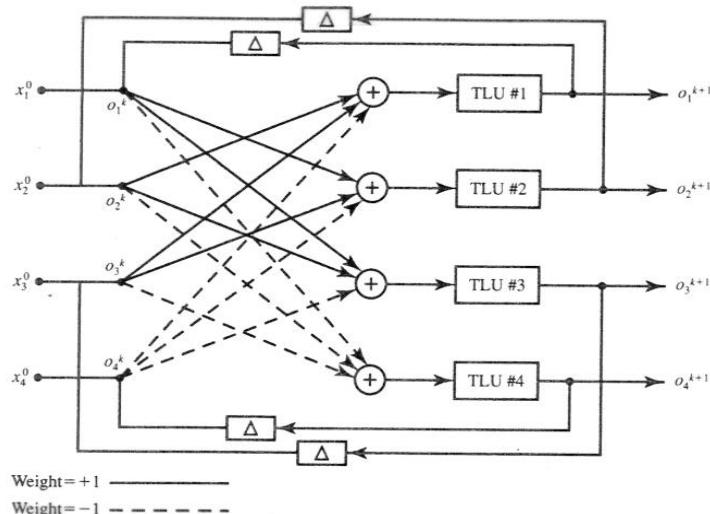
2) Modèles de réseaux de neurones

Réseau de neurones avec rétroaction (*feedback*)

- **Exemple:** RNA récurrent discret à 4 neurones caractérisé par les deux états attracteurs (états d'équilibre) suivants:

$$\mathbf{o}_1 = [1 \ 1 \ 1 \ -1]^t \text{ et } \mathbf{o}_2 = [-1 \ -1 \ -1 \ 1]^t$$

- supposons que nous savons comment entraîner le réseau pour forcer la transition entre les états marqués par les flèches



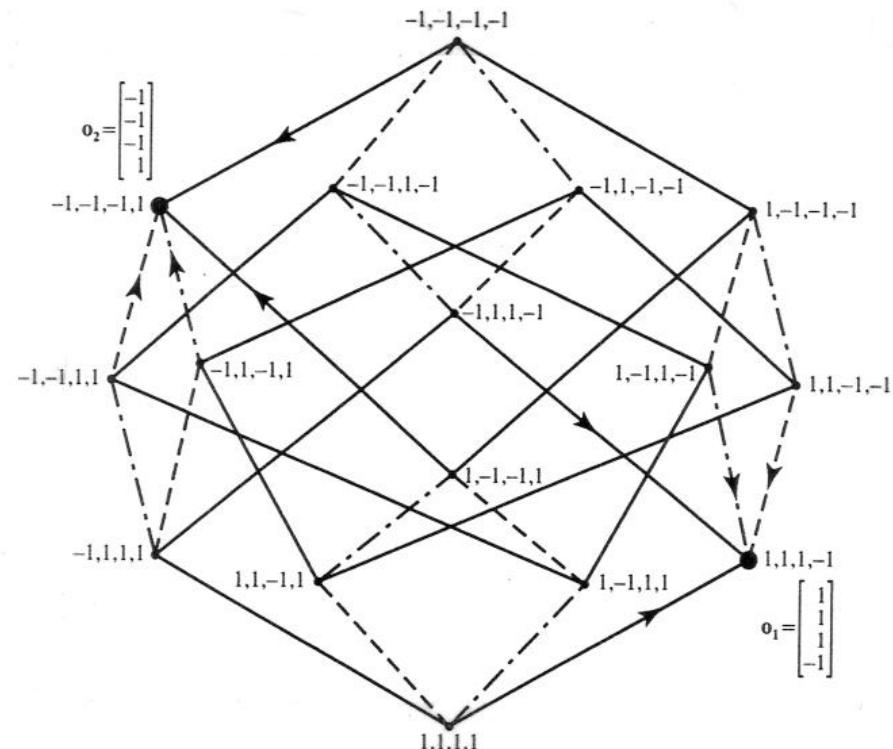
2) Modèles de réseaux de neurones

Réseau de neurones avec rétroaction (*feedback*)

- ▶ **Exemple:** illustre la notion des transitions d'états et l'analyse d'un réseau de neurones récurrent et discret

Domain des vecteurs de sortie \mathbf{o}

- Hypercube dans R^4 avec deux états d'équilibre (\mathbf{o}_1 et \mathbf{o}_2). Cette figure représente le domaine des sorties d'un réseau de 4 neurones binaires et bipolaires.
- Les transitions (arcs) entre deux états (vertices) du réseau sont possibles ssi la valeur des vecteurs d'état diffère d'un seul bit.



2) Modèles de réseaux de neurones

Réseau de neurones avec rétroaction (*feedback*)

- L'analyse du réseau montre, par inspection, que la matrice des poids est égale à:

$$\mathbf{W} = \begin{bmatrix} 0 & 1 & 1 & -1 \\ 1 & 0 & 1 & -1 \\ 1 & 1 & 0 & -1 \\ -1 & -1 & -1 & 0 \end{bmatrix}$$

- Premièrement, il est facile de constater que les états \mathbf{o}_1 et \mathbf{o}_2 sont des états d'équilibre. Supposons que $\mathbf{o}_1 = \mathbf{x}^0$, alors à la première itération nous avons:

$$\mathbf{o}^1 = \begin{bmatrix} sgn(\cdot) & 0 & 0 & 0 \\ 0 & sgn(\cdot) & 0 & 0 \\ 0 & 0 & sgn(\cdot) & 0 \\ 0 & 0 & 0 & sgn(\cdot) \end{bmatrix} \begin{bmatrix} net_1^0 \\ net_2^0 \\ net_3^0 \\ net_4^0 \end{bmatrix} = [sgn(3) \quad sgn(3) \quad sgn(3) \quad sgn(-3)]^t$$

∴ il n'y a pas d'autres transitions possibles étant donné que $\mathbf{o}^1 = \mathbf{o}^2 = \dots = \mathbf{o}_1$. Nous pouvons vérifier également que si $\mathbf{o}_2 = \mathbf{x}^0$, alors la séquence $\mathbf{o}^1 = \mathbf{o}^2 = \dots = \mathbf{o}_2$ est toujours vraie.

2) Modèles de réseaux de neurones

Réseau de neurones avec rétroaction (*feedback*)

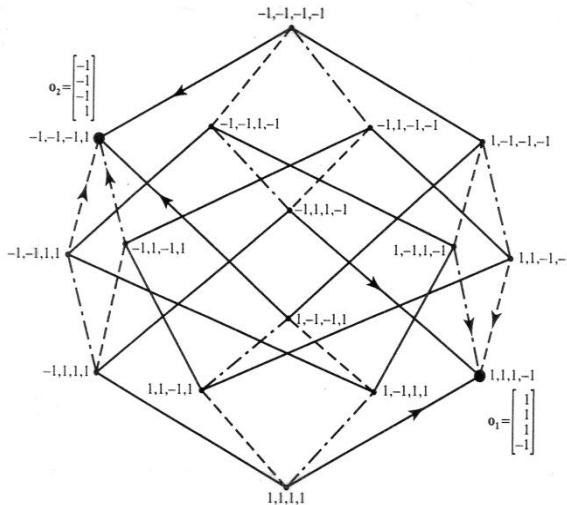
- ▶ **État d'équilibre le plus proche (après une itération):**
- Supposons que le réseau est initialisé à l'état $\mathbf{x}_0 = [1 \ 1 \ 1 \ 1]^t$, lequel est adjacent à \mathbf{o}_1 . Alors nous obtenons que:
$$\mathbf{o}^1 = [sgn(1) \ sgn(1) \ sgn(1) \ sgn(-3)]^t$$
- La transition suivante a eu lieu, soit $[1 \ 1 \ 1 \ 1] \rightarrow [1 \ 1 \ 1 \ -1]$, et le réseau a atteint immédiatement un état d'équilibre.
- Nous pouvons facilement montrer que pour les états initiaux suivants, soient $\mathbf{x}_0 = [1 \ -1 \ 1 \ -1]^t$, $\mathbf{x}_0 = [1 \ 1 \ -1 \ -1]^t$, et $\mathbf{x}_0 = [-1 \ 1 \ 1 \ -1]^t$, le réseau atteindra toujours l'état $\mathbf{o}^1 = \mathbf{o}_1$, et le réseau sera toujours stabilisé à \mathbf{o}_1 .

2) Modèles de réseaux de neurones

Réseau de neurones avec rétroaction (*feedback*)

► **État d'équilibre le plus proche:**

- Dans les cas où l'état initial diffère de 2 bits de l'un ou l'autre des points d'équilibre, les vecteurs de départ sont équidistants des points o_1 et o_2 . Alors le réseau convergera vers l'un ou l'autre des états attracteurs.
 \therefore le réseau cherchera toujours à atteindre l'état d'équilibre qui ressemble le plus au vecteur de départ.



CONTENU DU COURS

A.1 Introduction aux RNA:

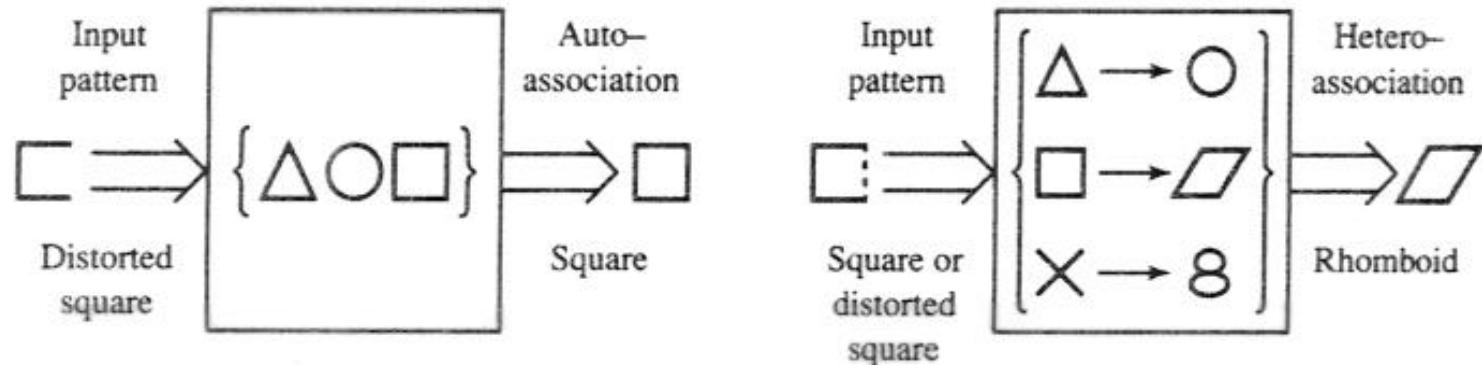
1. Neurones biologiques et artificiels
2. Modèles de réseaux de neurones artificiels
- 3. Traitements effectués par les réseaux**
4. Apprentissage et adaptation
5. Règles d'apprentissage

3) Traitements effectués par les réseaux

- Il existe 2 processus principaux liés à un RNA
 - 1. **rappel:** décoder ou évaluer la sortie la sortie o pour un vecteur d'entrées x donné
 - 2. **apprentissage:** modifier la matrice des poids W selon une base d'exemples $\{x\}$ référence
- Les exemples de la section précédente illustrent le **mécanisme de rappel** par lequel un réseau de neurones évalue la sortie o pour un vecteur x donné. On distingue:
 - autoassociation *vs* hétéroassociation
 - classification *vs* reconnaissance
 - mémorisation *vs* généralisation
- On suppose ici que le réseau de neurones est déjà entraîné

3) Traitements effectués par les réseaux

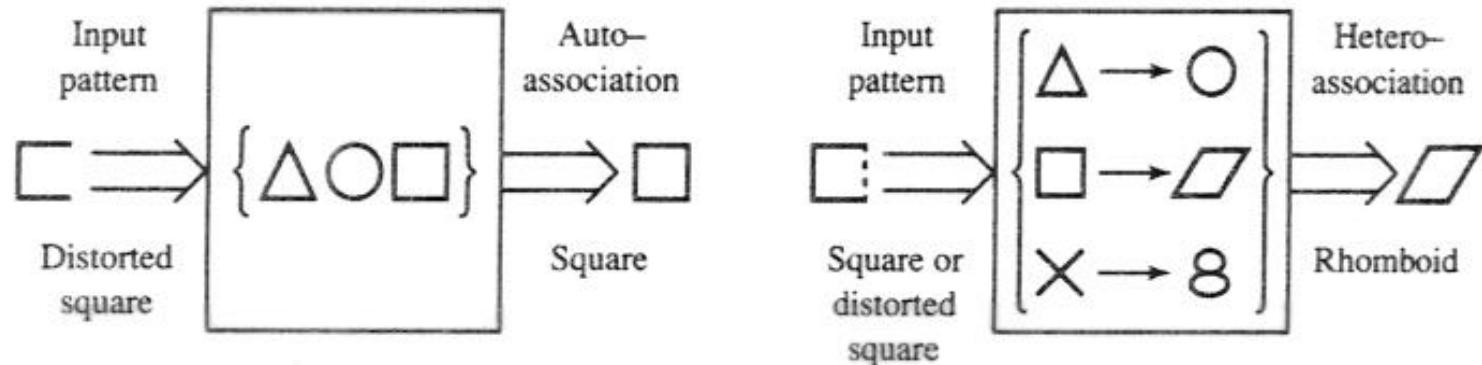
Autoassociation vs Hétéroassociation



- **Autoassociation:** rappel le patron le plus semblable
- Supposons qu'un réseau soit en mesure d'apprendre des exemples (prototypes). Un réseau de neurones fonctionne en **mode autoassociatif** lorsque la sortie du réseau représente le prototype le plus ressemblant au vecteur d'entrées.

3) Traitements effectués par les réseaux

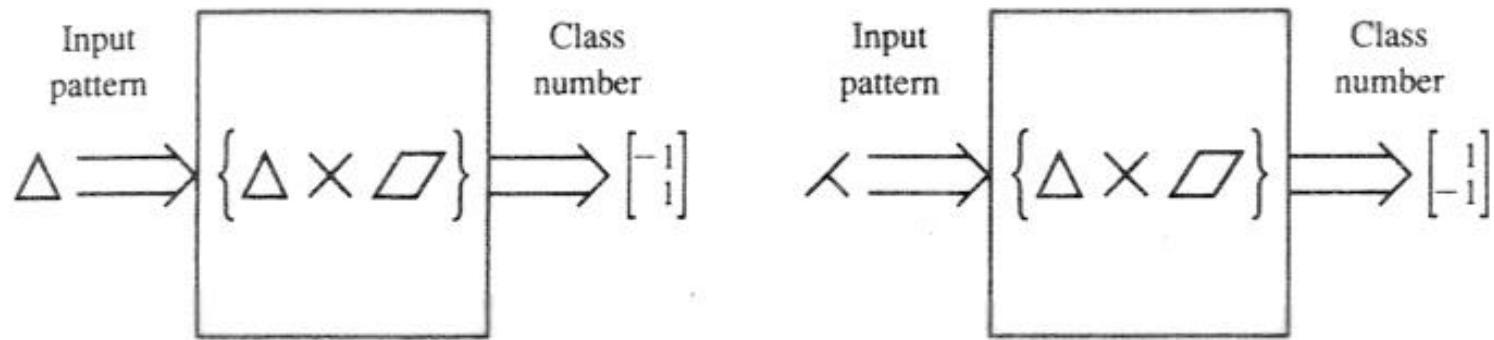
Autoassociation vs Hétéroassociation



- **Hétéroassociation:** rappel un patron lié au patron le plus semblable
- Supposons que le réseau soit en mesure d'apprendre l'association qui existe entre plusieurs paires de patrons différents. Ce type de réseau de neurones fonctionne alors en **mode hétéroassociatif**.

3) Traitements effectués par les réseaux

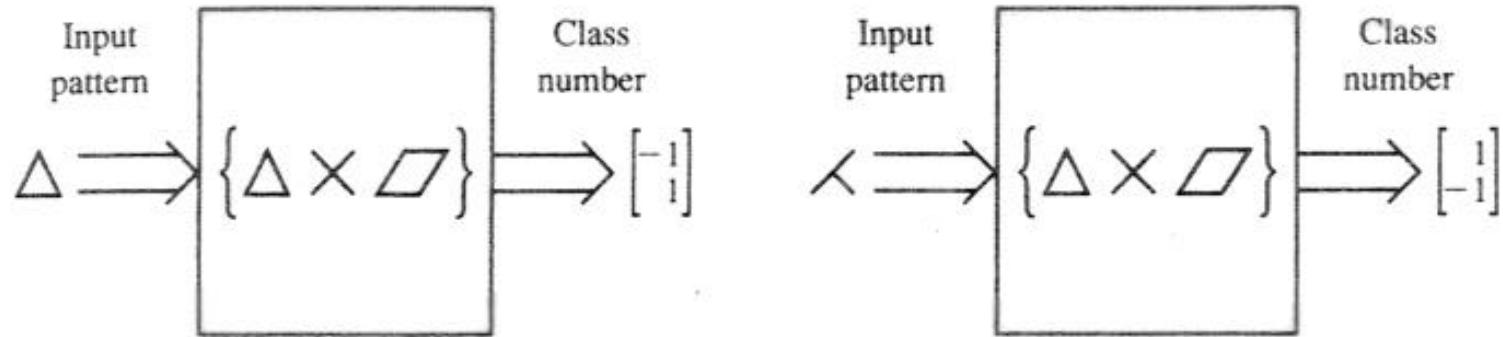
Classification vs Reconnaissance



- Supposons que les patrons d'entrées sont subdivisés en catégories d'objets (i.e., appartiennent à différentes classes).
- Un réseau de neurones est en mesure de **classer** des objets si la sortie correspondant à un vecteur d'entrées donné représente le numéro de la classe à laquelle il appartient.
- Un réseau classificateur est un cas particulier de hétéroassociation.

3) Traitements effectués par les réseaux

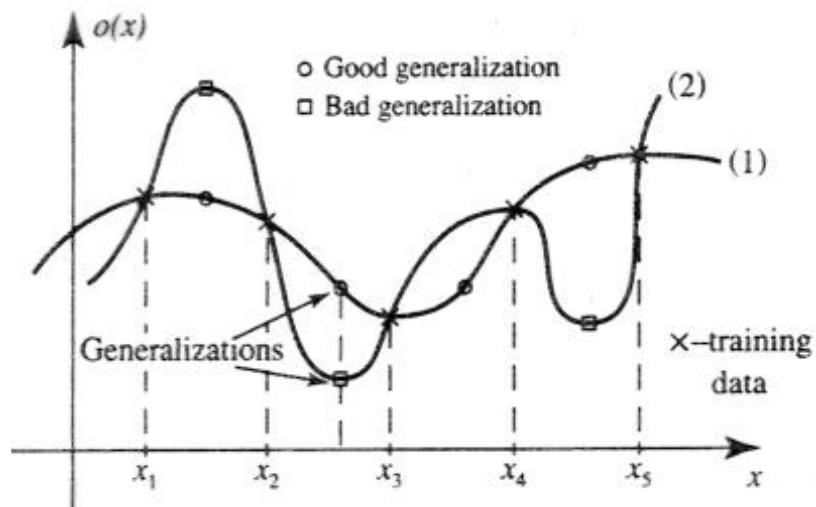
Classification vs Reconnaissance



- Si le vecteur présenté à l'entrée du réseau ne correspond pas parfaitement à l'un des objets de l'ensemble d'apprentissage, alors le traitement effectué par le réseau en est un de **reconnaissance**.
- Lorsque la sortie du réseau correspond au numéro d'identification d'une classe, alors la **reconnaissance** devient identique à la **classification**.

3) Traitements effectués par les réseaux

Mémorisation vs Généralisation

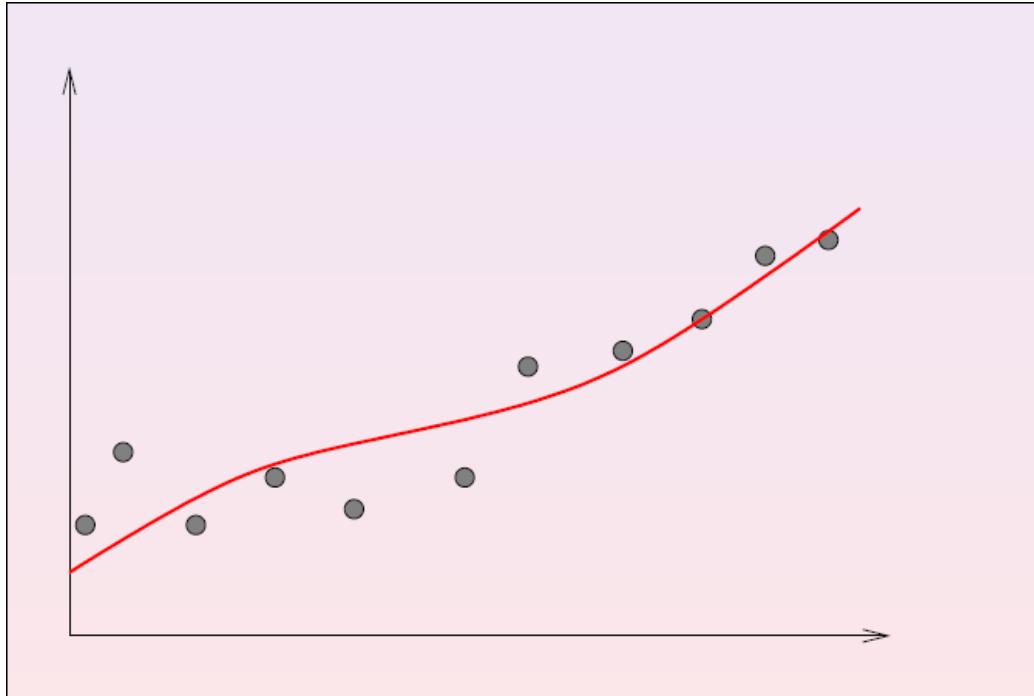


- Une caractéristique importante des réseaux de neurones est la capacité de **généralisation**.
- Un réseau généralise bien lorsqu'il est en mesure d'extrapoler à partir de vecteurs d'entrées différents de ceux utilisés pour l'entraînement.

3) Traitements effectués par les réseaux

Types de tâches

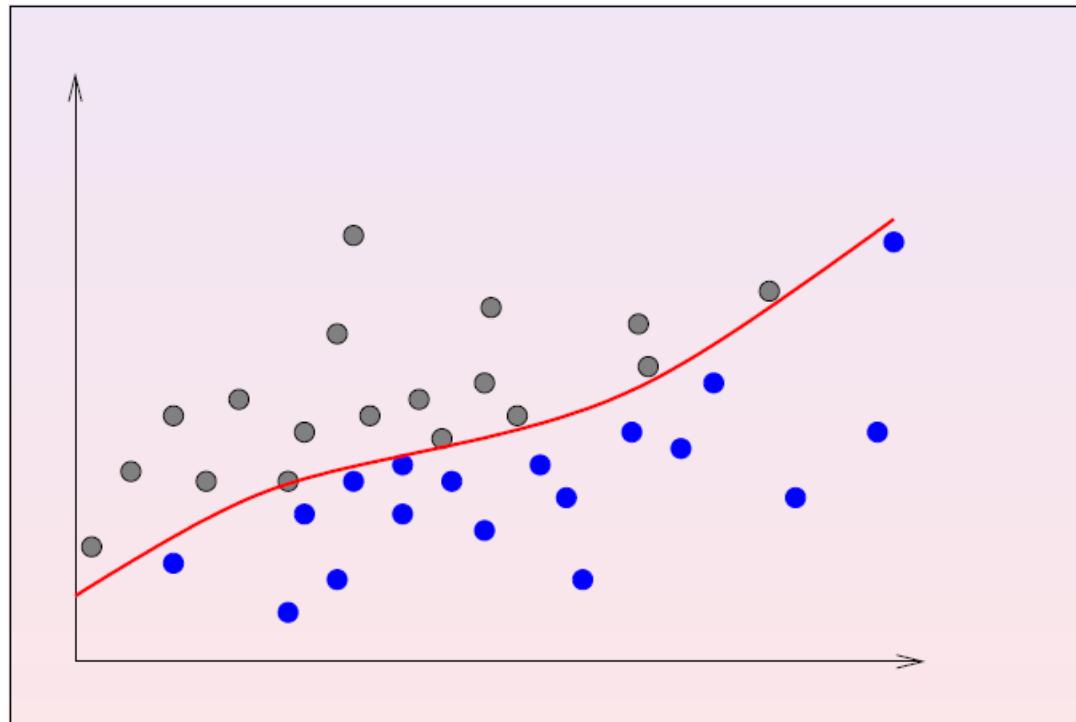
- ▶ **Régression:** prédiction de la relation entre 2+ variables



3) Traitements effectués par les réseaux

Types de tâches

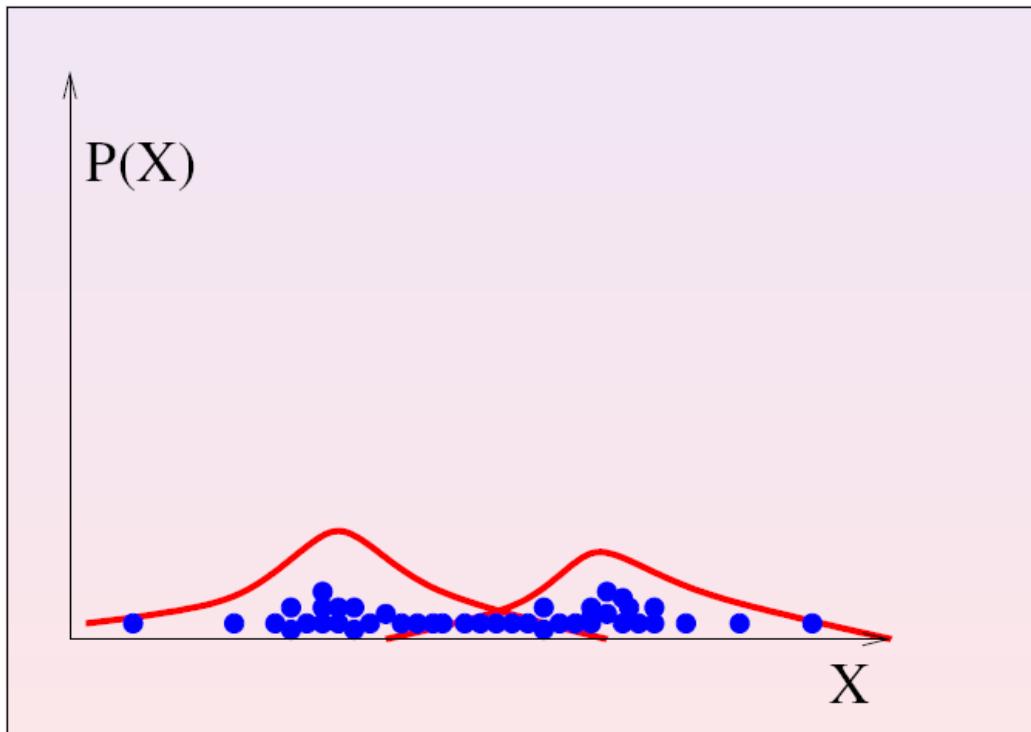
- ▶ **Classification:** assignation de classes aux observations



3) Traitements effectués par les réseaux

Types de tâches

- ▶ **Estimation de densité:** estimation d'une PDF non-observable



CONTENU DU COURS

A.1 Introduction aux RNA:

1. Neurones biologiques et artificiels
2. Modèles de réseaux de neurones artificiels
3. Traitements effectués par les réseaux
4. **Apprentissage et adaptation**
5. Règles d'apprentissage

4) Apprentissage et adaptation

► Apprentissage chez l'être humain:

- En psychologie, l'apprentissage peut être définie comme des modifications durables du comportement d'un sujet (humain ou animal) grâce à expériences répétées.
- L'apprentissage est un processus d'inférence – nous ne pouvons pas le voir directement mais nous pouvons nous assurer qu'il s'est produit en observant les changements de performance.

4) Apprentissage et adaptation

► Apprentissage avec les RNA :

- Un processus plus direct, et nous pouvons capturer chaque étape d'apprentissage dans une relation de cause à effet distincte.
- La conception d'un réseau associateur ou d'un réseau classificateur nécessite l'apprentissage d'une relation (*mapping*) d'entrée-sortie à partir d'un ensemble d'entraînement.
- Une approche classique pour résoudre ce type de problème repose sur la théorie de l'approximation (Poggio & Girosi 1990)
 - *on veut apprendre l'approximation de relations entre des vecteurs d'entrées-sorties*

4) Apprentissage et adaptation

-
- ▶ **Apprentissage par approximation de relations:**
 - Nous voulons approximer une fonction continue multi variable $h(\mathbf{x})$ par une autre fonction $H(\mathbf{w}, \mathbf{x})$, où $\mathbf{x} = [x_1 \ x_2 \ \dots \ x_n]^t$ représente un vecteur d'entrée et $\mathbf{w} = [w_1 \ w_2 \ \dots \ w_m]^t$ un vecteur de poids
 - la tâche d'apprentissage consiste à trouver la meilleure approximation possible de $h(\mathbf{x})$ à partir d'un ensemble d'entraînement $\{\mathbf{x}\}$
 - un choix important à faire est celui du type de fonction d'approximation $H(\mathbf{w}, \mathbf{x})$ qui permettra la meilleure approximation de $h(\mathbf{x})$ selon les paramètres \mathbf{w}
 - **Problème d'apprentissage:** processus d'estimation des paramètres \mathbf{W} de RNA de tel sorte à minimiser un coût

4) Apprentissage et adaptation

► Apprentissage par approximation de relations:

- Une formulation plus précise de notre **problème d'apprentissage** peut être énoncée comme la recherche de w^* tel que :

$$\min \{ \rho[H(w, x), h(x)] \}$$

où la fonction de distance, $\rho[H(w, x), h(x)]$, est une mesure de coût d'approximation de $h(x)$ par $H(w, x)$.

- Lorsque la somme des différences au carré est utilisée pour juger de le coût de l'approximation à partir de l'ensemble d'entraînement $\{x_i : i = 1, 2, \dots, n\}$, la distance précédente a la forme de la somme des erreurs au carré.

$$\rho[H(w, x), h(x)] = \sum_{i=1}^n (o(x_i) - d_i)^2$$

- REM: les réseaux de neurones sans rétroaction, monocouche et multicouches, utilisent ce principe d'apprentissage.

4) Apprentissage et adaptation

- ▶ **L'apprentissage:** le processus qui consiste à guider un RNA afin de produire une réponse particulière pour une entrée spécifique:
- L'apprentissage est nécessaire lorsque la relation entre les entrées-sorties est inconnue a priori ou incomplète; conséquemment la conception ne peut être effectuée à l'avance.
- **L'apprentissage par lot** (batch) d'un réseau est effectué lorsque les poids sont ajustés itérativement, après chaque présentation complète (epoch) des données de l'ensemble d'entraînement.
- **L'apprentissage incrémental** (séquentiel) est généralement effectué lorsque les poids sont ajustés itérativement, après présentation de chaque exemple de l'ensemble d'entraînement.
- **Mode d'apprentissage:** Une réponse particulière peut être spécifiée (supervisé) ou non (non-supervisé) dans le but d'effectuer une correction externe.

4) Apprentissage et adaptation

Algorithmes d'apprentissage

► **Apprentissage chez l'humain:**

- une propriété essentielle
- consiste à changer de telle façon à améliorer son comportement (selon un critère donné) lorsque qu'on fait face à une situation semblable
- ne consiste pas à mémoriser par cœur

► **Défi principal pour l'apprentissage automatique par machines:**

- la capacité de *généraliser* un comportement pour de nouvelles situations

4) Apprentissage et adaptation

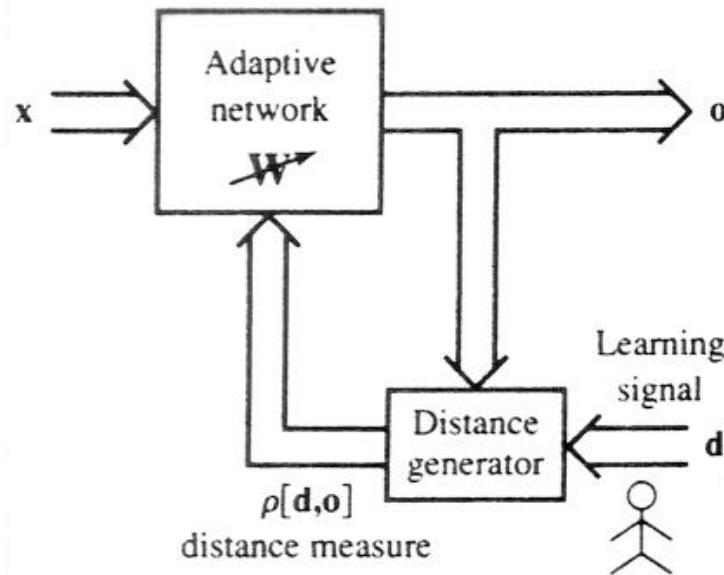
Algorithmes d'apprentissage

- ▶ **Objectif de l'apprentissage automatique:** concevoir un système pour résoudre des relations d'e/s ('mappings') à partir:
 1. d'un ensemble fini de données réelles D (exemples de référence) pour l'entraînement
 2. de connaissances a priori limitées h_0 sur l'ensemble des solutions possibles
- ▶ **Approche générale:** dériver une relation d'e/s via un processus d'optimisation

4) Apprentissage et adaptation

► L'apprentissage supervisé:

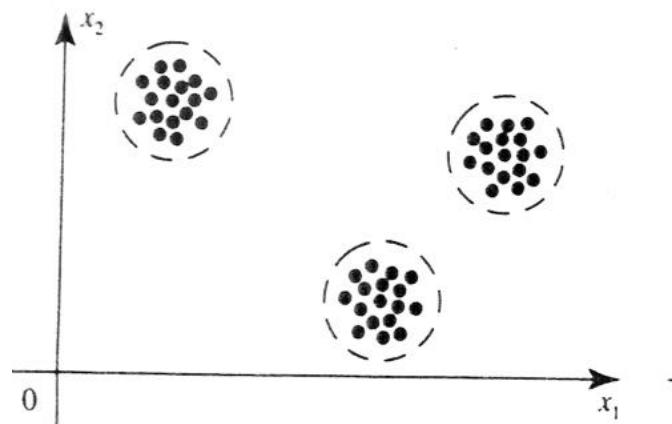
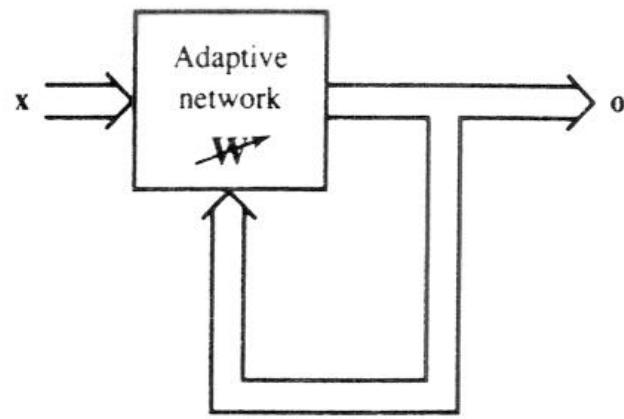
- Lors de l'apprentissage supervisé d'un réseau, pour chaque entrée appliquée, la réponse désirée d du système est spécifiée **par un professeur**.
- La distance $\rho[d, o]$ entre la réponse actuelle et la réponse désirée sert pour évaluer une mesure de l'erreur et le résultat est utilisé pour corriger les paramètres du réseau.



4) Apprentissage et adaptation

► L'apprentissage non-supervisé:

- Lors de l'apprentissage non supervisé, la réponse désirée n'est pas connue, alors une mesure de l'erreur faite par le réseau n'est pas disponible pour améliorer le comportement du réseau.
- Ce type de réseau utilise habituellement la présence de redondance dans les données de l'ensemble d'entraînement pour découvrir la structure des classes sur la base d'indices de ressemblance, de régularité, etc.



CONTENU DU COURS

A.1 Introduction aux RNA:

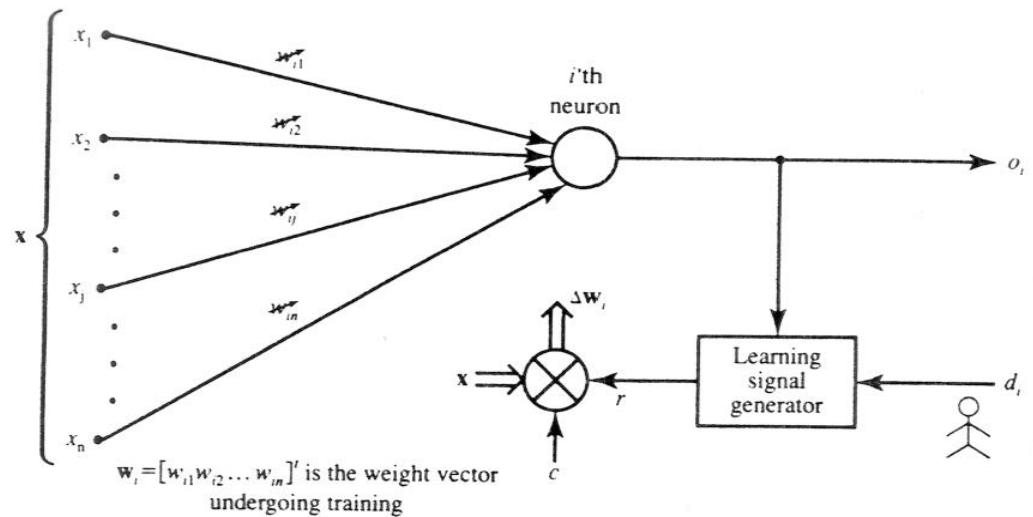
1. Neurones biologiques et artificiels
2. Modèles de réseaux de neurones artificiels
3. Traitements effectués par les réseaux
4. Apprentissage et adaptation
5. Règles d'apprentissage

5) Règles d'apprentissage

Règle d'apprentissage généralisée

► **Illustration générale pour l'apprentissage des poids associés aux connexions d'un neurone artificiel (Amari, 1990).**

- la sortie désirée d_i est utilisée dans le mode d'apprentissage supervisé seulement.
- le seuil T correspond au poids w_{in} , conséquemment la valeur de l'entrée x_n est fixée à -1.



5) Règles d'apprentissage

Règle d'apprentissage généralisée

- ▶ Cette règle généralisée, proposée par Amari (1990) est couramment utilisée dans les travaux portant sur les RNA:
 - Le vecteur de poids $\mathbf{w}_i = [w_{i1} \ w_{i2} \dots \ w_{in}]^t$ varie proportionnellement au produit de l'entrée \mathbf{x} et du signal d'apprentissage r .
 - Le signal d'apprentissage r est en général une fonction de \mathbf{w}_i , \mathbf{x} , et quelquefois du signal fourni par le professeur, soit d_i .
 - Alors, nous obtenons pour le réseau de la page précédent

$$r = r(\mathbf{w}_i, \mathbf{x}, d_i)$$

5) Règles d'apprentissage

Règle d'apprentissage généralisée

- Selon cette règle, l'ajustement du vecteur w_i produit lors de l'étape d'apprentissage au temps t est égale à :

$$\Delta w_i(t) = c \ r[w_i(t), x(t), d_i(t)] x(t)$$

où c est un nombre positif appelé constante d'apprentissage qui détermine le taux d'apprentissage effectué à chaque itération

- Le vecteur de poids du $i^{\text{ième}}$ neurone adapté au temps t deviendra, à la prochaine étape $t+1$, égal à

$$w_i(t+1) = w_i(t) + \Delta w_i(t) = w_i(t) + c \ r[w_i(t), x(t), d_i(t)] x(t)$$

- Par convention, la $k^{\text{ième}}$ étape d'apprentissage sera notée

$$w_i^{k+1} = w_i^k + \Delta w_i^k = w_i^k + c \ r[w_i^k, x^k, d_i^k] x^k$$

5) Règles d'apprentissage

Règle d'apprentissage de Hebb

► Apprentissage non supervisé et sans rétroaction:

- Dans ce cas, le signal d'apprentissage r est simplement égal à la sortie du neurone

$$r = f(\mathbf{w}_i^t \mathbf{x})$$

- Alors, l'incrément $\Delta \mathbf{w}_i$ du vecteur de poids devient

$$\Delta \mathbf{w}_i = c r \mathbf{x} = c f(\mathbf{w}_i^t \mathbf{x}) \mathbf{x}$$

- Conséquemment, le poids w_{ij} est adapté en utilisant la relation

$$\Delta w_{ij} = c f(\mathbf{w}_i^t \mathbf{x}) x_j = c o_i x_j \text{ pour } j = 1, 2, \dots, n$$

5) Règles d'apprentissage

Règle d'apprentissage de Hebb

- Cette règle d'apprentissage nécessite l'initialisation des poids. Les valeurs ($w_i \approx 0$) sont choisies aléatoirement avant le démarrage de la phase d'apprentissage.
- Cette règle stipule que si le produit de l'entrée par la sortie (corrélation $o_i x_j$) est positive, il résulte une augmentation du poids w_{ij} ; autrement, la valeur du poids w_{ij} doit diminuer.
- Les vecteurs d'entrées les plus fréquents auront le plus d'influence sur le vecteur de poids du neurone et produiront éventuellement la plus grande valeur de sortie.
- La règle d'apprentissage de Hebb s'applique indépendamment du choix de la fonction d'activation $f(net)$, soit binaire ou continue, unipolaire ou bipolaire.

5) Règles d'apprentissage

Règle d'apprentissage de Hebb

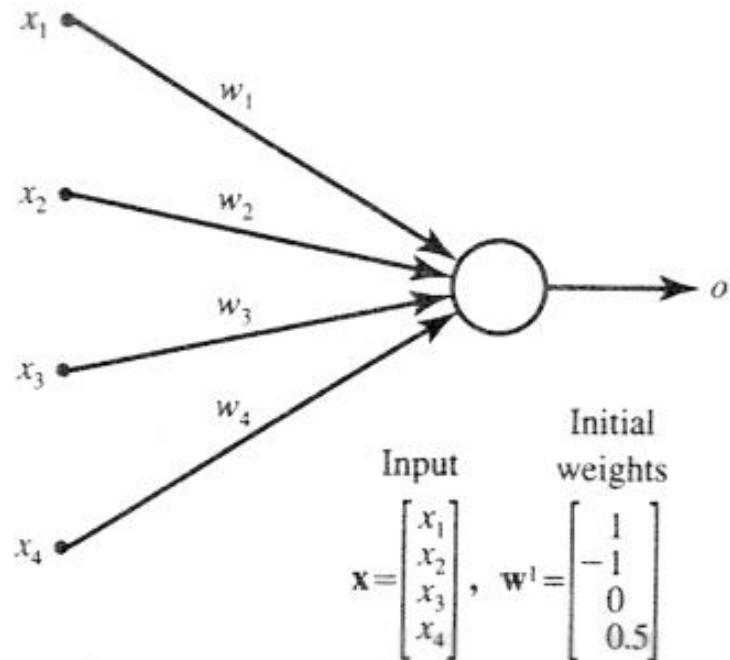
► **Exemple:** soit l'architecture du réseau suivant

- Posons arbitrairement la constante d'apprentissage $c = 1$, et un vecteur de poids initial:

$$\mathbf{w}^1 = \begin{bmatrix} 1 \\ -1 \\ 0 \\ 0.5 \end{bmatrix}$$

- et l'ensemble d'entraînement:

$$\mathbf{x}_1 = \begin{bmatrix} 1 \\ -2 \\ 1.5 \\ 0 \end{bmatrix}, \quad \mathbf{x}_2 = \begin{bmatrix} 1 \\ -0.5 \\ -2 \\ -1.5 \end{bmatrix}, \quad \mathbf{x}_3 = \begin{bmatrix} 0 \\ 1 \\ -1 \\ 1.5 \end{bmatrix}$$



5) Règles d'apprentissage

Règle d'apprentissage de Hebb

- ▶ **Exemple:** activation binaire bipolaire $f(net) = sgn(net)$
- **Étape 1:** Le vecteur x_1 est appliqué à l'entrée du réseau et la valeur d'activation net^1 correspond à:
$$\mathbf{w}^2 = \mathbf{w}^1 + sgn(net^1)\mathbf{x}_1 = \mathbf{w}^1 + \mathbf{x}_1 = \begin{bmatrix} 1 \\ -1 \\ 0 \\ 0.5 \end{bmatrix} + \begin{bmatrix} 1 \\ -2 \\ 1.5 \\ 0 \end{bmatrix} = \begin{bmatrix} 2 \\ -3 \\ 1.5 \\ 0.5 \end{bmatrix}$$
- **Étape 2:** Le vecteur x_2 est ensuite appliqué à l'entrée du réseau, alors
$$\mathbf{w}^3 = \mathbf{w}^2 + sgn(net^2)\mathbf{x}_2 = \mathbf{w}^2 - \mathbf{x}_2 = \begin{bmatrix} 1 \\ -2.5 \\ 3.5 \\ 2 \end{bmatrix}$$
- **Étape 3:** Finalement, le vecteur x_3 est appliqué à l'entrée du réseau, alors

$$\mathbf{w}^4 = \mathbf{w}^3 + sgn(net^3)\mathbf{x}_3 = \mathbf{w}^3 - \mathbf{x}_3 = \begin{bmatrix} 1 \\ -3.5 \\ 4.5 \\ 0.5 \end{bmatrix}$$

5) Règles d'apprentissage

Règle d'apprentissage de Hebb

- Exemple: activation bipolaire continue $f(net) = \tanh(net)$

$$f(net) \doteq \frac{2}{1 + \exp(-\lambda net)} - 1 \quad \text{avec} \quad \lambda > 0$$

Étape 1

$$f(net^1) = 0.905$$

$$w^2 = \begin{bmatrix} 1.905 \\ -2.81 \\ 1.357 \\ 0.5 \end{bmatrix}$$

Étape 2

$$f(net^2) = -0.077$$

$$w^3 = \begin{bmatrix} 1.828 \\ -2.772 \\ 1.512 \\ 0.616 \end{bmatrix}$$

Étape 3

$$f(net^3) = -0.932$$

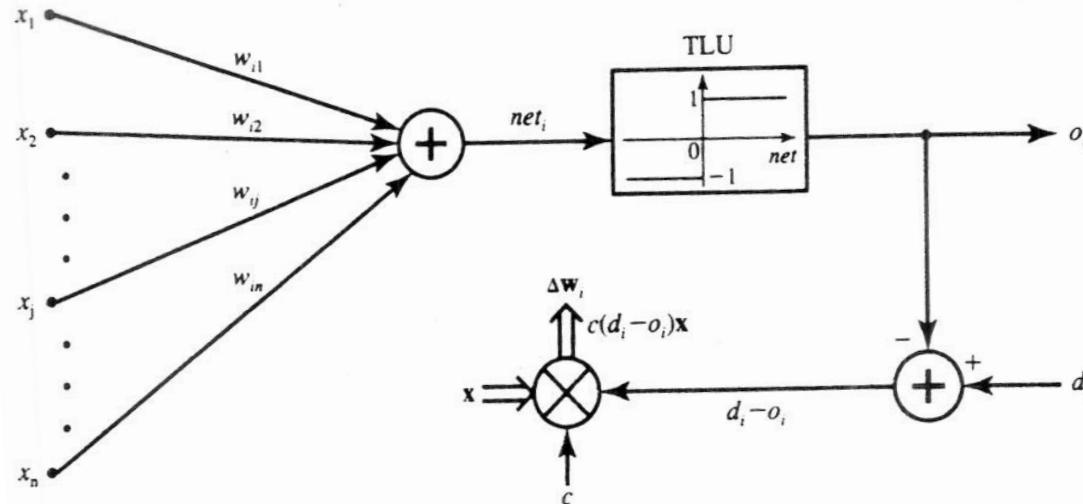
$$w^4 = \begin{bmatrix} 1.828 \\ -3.70 \\ 2.44 \\ -0.783 \end{bmatrix}$$

Ici, l'ajustement des poids est plus fin comparé à la fonction d'activation binaire utilisée précédemment.

5) Règles d'apprentissage

Règle d'apprentissage du Perceptron

- Architecture de réseau permettant l'apprentissage de type Perceptron:



- Cette règle repose sur l'utilisation d'une fonction d'activation binaire seulement (unipolaire ou bipolaire).

5) Règles d'apprentissage

Règle d'apprentissage du Perceptron

► Apprentissage en mode supervisé:

- Le signal d'apprentissage est égal à $r = d_i - o_i$, soit la différence entre la sortie du neurone $o_i = \text{sgn}(\mathbf{w}_i^t \mathbf{x})$, et la réponse désirée d_i
- Alors, l'ajustement du vecteur de poids est réalisé à l'aide de la relation: $\Delta \mathbf{w}_i = c r \mathbf{x} = c [d_i - \text{sgn}(\mathbf{w}_i^t \mathbf{x})] \mathbf{x}$
- Conséquemment, le poids w_{ij} est adapté en utilisant la relation

$$\Delta w_{ij} = c [d_i - \text{sgn}(\mathbf{w}_i^t \mathbf{x})] x_j \quad \text{pour } j = 1, 2, \dots, n$$

5) Règles d'apprentissage

Règle d'apprentissage du Perceptron

► Apprentissage en mode supervisé:

- Suivant cette règle, les poids sont modifiés si et seulement si la valeur de la sortie o_i est incorrecte.
- Étant donné que la réponse désirée $d_i = 1$ ou $d_i = -1$, alors l'ajustement des poids se résume à $\Delta w_i = \pm 2 c x$

Nous avons: $\Delta w_i = 2 c x$ si $d_i = 1$ et $\text{sgn}(w_i^t x) = -1$

$\Delta w_i = -2 c x$ si $d_i = -1$ et $\text{sgn}(w_i^t x) = 1$

$\Delta w_i = 0$ si $d_i = \text{sgn}(w_i^t x)$

- Les poids sont initialisés de façon arbitraire dans cette méthode

5) Règles d'apprentissage

Règle d'apprentissage du Perceptron

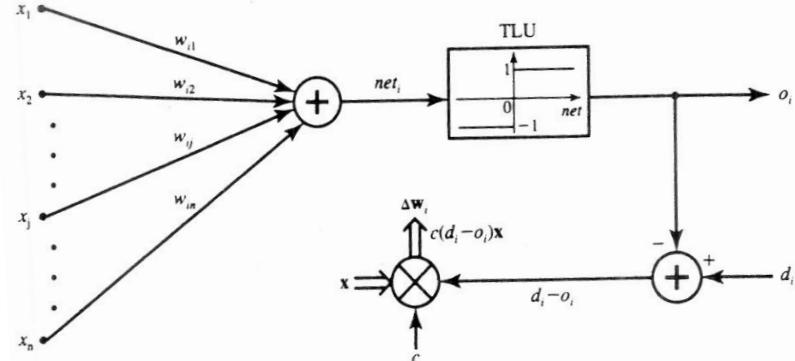
- **Exemple:** soit l'entraînement du réseau montré à la figure précédente, avec le vecteur de poids initial:

$$\mathbf{w}^1 = \begin{bmatrix} 1 \\ -1 \\ 0 \\ 0.5 \end{bmatrix}$$

- et l'ensemble d'entraînement:

$$\mathbf{x}_1 = \begin{bmatrix} 1 \\ -2 \\ 0 \\ -1^{(\dagger)} \end{bmatrix}, \quad \mathbf{x}_2 = \begin{bmatrix} 0 \\ 1.5 \\ -0.5 \\ -1^{(\dagger)} \end{bmatrix}, \quad \mathbf{x}_3 = \begin{bmatrix} -1 \\ 1 \\ 0.5 \\ -1^{(\dagger)} \end{bmatrix}$$

- Ici, la constante d'apprentissage est fixée arbitrairement à $c = 0.1$
- Les réponses désirées pour \mathbf{x}_1 , \mathbf{x}_2 et \mathbf{x}_3 sont égales à $d_1 = -1$, $d_2 = -1$ et $d_3 = 1$
- (\dagger) La règle du perceptron requiert que la valeur d'un élément des vecteurs de l'ensemble d'entraînement soit fixée à $x_n = -1$.



5) Règles d'apprentissage

Règle d'apprentissage du Perceptron

► **Exemple:** activation binaire bipolaire $f(net) = sgn(net)$

– **Étape 1:** L'entrée est x_1 , et la réponse désirée est $d_1 = -1$. L'ajustement des poids est nécessaire car $d_1 \neq sgn(2.5)$.

$$\mathbf{w}^2 = \mathbf{w}^1 + 0.1(-1 \ -1)x_1 = \begin{bmatrix} 1 \\ -1 \\ 0 \\ 0.5 \end{bmatrix} - 0.2 \begin{bmatrix} 1 \\ -2 \\ 0 \\ -1 \end{bmatrix} = \begin{bmatrix} 0.8 \\ -0.6 \\ 0 \\ 0.7 \end{bmatrix}$$

– **Étape 2:** L'entrée est x_2 , et la réponse désirée est $d_2 = -1$. L'ajustement des poids n'est pas nécessaire car $d_2 = sgn(-1.6)$.

– **Étape 3:** L'entrée est x_3 , et la réponse désirée est $d_3 = 1$. L'ajustement des poids est nécessaire car $d_3 \neq sgn(-2.1)$.

$$\mathbf{w}^4 = \mathbf{w}^3 + 0.1(1+1)x_3 = \begin{bmatrix} 0.8 \\ -0.6 \\ 0 \\ 0.7 \end{bmatrix} + 0.2 \begin{bmatrix} -1 \\ 1 \\ 0.5 \\ -1 \end{bmatrix} = \begin{bmatrix} 0.6 \\ -0.4 \\ 0.1 \\ 0.5 \end{bmatrix}$$

5) Règles d'apprentissage

Règle d'apprentissage du Perceptron

- ▶ **Exemple:** activation binaire bipolaire $f(\text{net}) = \text{sgn}(\text{net})$

Processus itératif:

- L'apprentissage prend fin à moins que l'ensemble d'entraînement ne soit examiné de nouveau.
- D'une manière générale, plusieurs itérations sur l'ensemble d'entraînement fait en sorte que l'erreur faite par le réseau est de plus en plus petite.
- *Ce type d'apprentissage permet habituellement au réseau de mémoriser parfaitement les exemples qui lui sont présentés.*

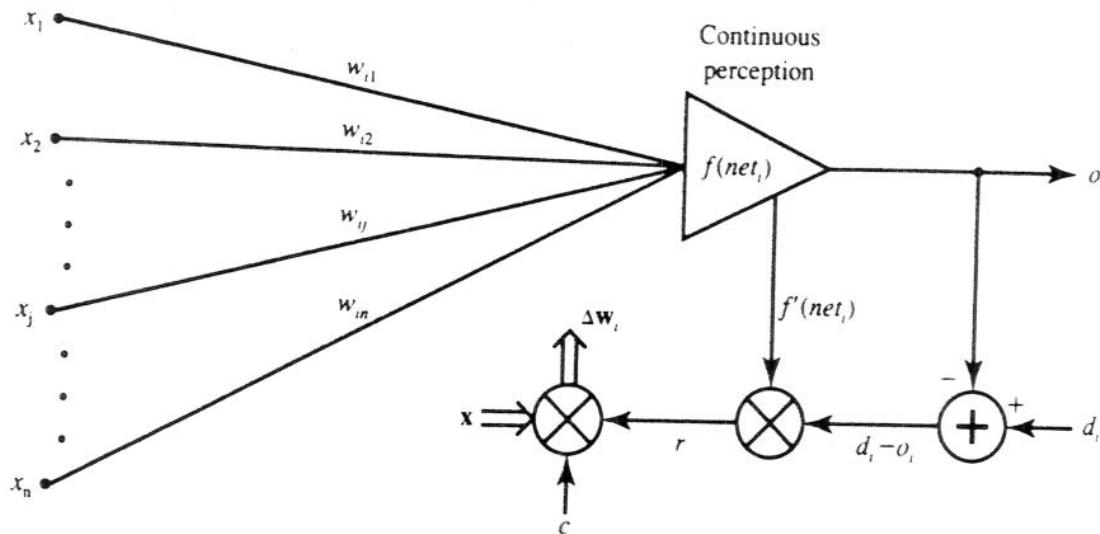
Q: Stratégies – critère pour éviter le sur-apprentissage?

- arrêter quand les poids stabilisent pour l'ensemble d'entraînement?
- arrêter quand la fonction de coût fixe reste $r = d_i - o_i$?
- utiliser une base de validation indépendante?

5) Règles d'apprentissage

Règle d'apprentissage delta

► Architecture de réseau avec apprentissage delta:



- Cette règle est valide seulement pour les fonctions d'activation $f(\text{net})$ continues (unipolaires ou bipolaires), et un mode d'entraînement supervisé.

5) Règles d'apprentissage

Règle d'apprentissage *delta*

► Apprentissage en mode supervisé:

- Le signal d'apprentissage est égal à $r = [d_i - f(\mathbf{w}_i^t \mathbf{x})] f'(\mathbf{w}_i^t \mathbf{x})$.
 - Le terme $f'(\mathbf{w}_i^t \mathbf{x})$ est la dérivée de la fonction d'activation $f(net)$ évaluée avec $net = \mathbf{w}_i^t \mathbf{x}$.
- **Dérivation:** Cette règle d'apprentissage découle de l'erreur quadratique moyen E évaluée entre o_i et d_i :

$$E \doteq \frac{1}{2}(d_i - o_i)^2 = \frac{1}{2}[d_i - f(\mathbf{w}_i^t \mathbf{x})]^2$$

5) Règles d'apprentissage

Règle d'apprentissage *delta*

–Le vecteur gradient de E évalué par rapport à \mathbf{w}_i :

$$\nabla E = -(d_i - o_i)f'(\mathbf{w}_i^t \mathbf{x})\mathbf{x}$$

où

$$\frac{\partial E}{\partial w_{ij}} = -(d_i - o_i) f'(\mathbf{w}_i^t \mathbf{x}) x_j, \text{ pour } j = 1, 2, \dots, n$$

–Étant donné que la minimisation de l'erreur requiert que les changements de poids soient effectués dans la direction négative du gradient, nous prenons

$$\Delta w_{ij} = \eta(d_i - o_i) f'(net_i) x_j, \text{ pour } j = 1, 2, \dots, n$$

où η est une constante positive, avec:

$$\Delta \mathbf{w}_i = -\eta \nabla E = \eta(d_i - o_i)f'(net_i)\mathbf{x}$$

5) Règles d'apprentissage

Règle d'apprentissage *delta*

- L'ajustement des poids à la formule précédente est basé sur la minimisation de l'erreur quadratique E . Après remplacement dans la formule d'apprentissage généralisée, l'ajustement des poids devient égal à

$$\Delta \mathbf{w}_i = c \ r \ \mathbf{x} = c [d_i - o_i] f'(net_i) \ \mathbf{x}$$

- Les poids sont initialisés à des valeurs arbitraires choisies aléatoirement dans cette méthode.
- Cette règle nécessite des valeurs de c très petites étant donné qu'elle repose sur le déplacement du vecteur de poids (dans l'espace des poids) dans la direction négative du gradient.

5) Règles d'apprentissage

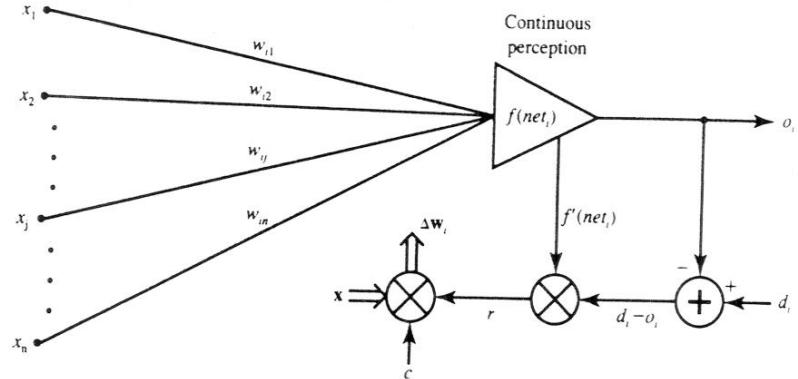
Règle d'apprentissage *delta*

- **Exemple:** soit l'entraînement du réseau montré à la figure précédente, avec le vecteur de poids initial:

$$\mathbf{w}^1 = \begin{bmatrix} 1 \\ -1 \\ 0 \\ 0.5 \end{bmatrix}$$

- et l'ensemble d'entraînement:

$$\mathbf{x}_1 = \begin{bmatrix} 1 \\ -2 \\ 0 \\ -1^{(\dagger)} \end{bmatrix}, \quad \mathbf{x}_2 = \begin{bmatrix} 0 \\ 1.5 \\ -0.5 \\ -1^{(\dagger)} \end{bmatrix}, \quad \mathbf{x}_3 = \begin{bmatrix} -1 \\ 1 \\ 0.5 \\ -1^{(\dagger)} \end{bmatrix}$$



- Ici, la constante d'apprentissage est fixée arbitrairement à $c = 0.1$
- Les réponses désirées pour \mathbf{x}_1 , \mathbf{x}_2 et \mathbf{x}_3 sont égales à $d_1 = -1$, $d_2 = -1$ et $d_3 = 1$
- (\dagger) La règle delta requiert que la valeur d'un élément des vecteurs de l'ensemble d'entraînement soit fixée à $x_n = -1$.

5) Règles d'apprentissage

Règle d'apprentissage *delta*

► **Exemple:** fonction activation bipolaire continue

$$f(\text{net}) \doteq \frac{2}{1 + \exp(-\lambda \text{net})} - 1 \quad \text{avec} \quad \lambda = 1$$

$$f'(\text{net}) = \frac{2 \exp(-\text{net})}{[1 + \exp(-\text{net})]^2} = \frac{1}{2}(1 - o^2)$$

- Ce résultat reflète la pente de la fonction d'activation évaluée à la sortie du neurone.
- Étant donné que les valeurs désirées sont égales à ± 1 , les poids sont modifiés à chaque étape car $d_i \neq f(\text{net}_i)$ pour toutes les données d'entraînement.

5) Règles d'apprentissage

Règle d'apprentissage *delta*

► Exemple: fonction activation bipolaire continue

Étape 1

entrées x_1

poids initial w^1

$$net^1 = w^{1t}x_1 = 2.5$$

$$o^1 = f(net^1) = 0.848$$

$$f'(net^1) = \frac{1}{2}[1-(o^1)^2] = 0.140$$

$$w^2 = c(d_1 - o^1) f'(net^1)x_1 + w^1$$

$$w^2 = [0.974 \quad -0.948 \quad 0 \quad 0.526]$$

Étape 2

entrées x_2

poids initial w^2

$$net^2 = w^{2t}x_2 = -1.948$$

$$o^2 = f(net^2) = -0.75$$

$$f'(net^2) = \frac{1}{2}[1-(o^2)^2] = 0.218$$

$$w^3 = c(d_2 - o^2) f'(net^2)x_2 + w^2$$

$$w^3 = [0.974 \quad -0.956 \quad 0.002 \quad 0.531]$$

Étape 3

entrées x_3

poids initial w^3

$$net^3 = w^{3t}x_3 = -2.46$$

$$o^3 = f(net^3) = -0.842$$

$$f'(net^3) = \frac{1}{2}[1-(o^3)^2] = 0.145$$

$$w^4 = c(d_3 - o^3) f'(net^3)x_3 + w^3$$

$$w^4 = [0.947 \quad -0.929 \quad 0.016 \quad 0.505]$$

5) Règles d'apprentissage

Règle d'apprentissage de Widrow-Hoff

► Propriétés de cette règle:

- Également appelée *Least Mean Square* (LMS)
- Cette règle d'apprentissage est applicable pour l'entraînement supervisé des réseaux de neurones
- Elle est indépendante du choix de la fonction d'activation des neurones étant donné qu'elle minimise le carré de l'erreur entre la valeur désirée d_i et la valeur d'activation du neurone

$$net_i = \mathbf{w}_i^t \mathbf{x} .$$

5) Règles d'apprentissage

Règle d'apprentissage de Widrow-Hoff

- Le signal d'apprentissage pour cette règle est défini par

$$r = d_i - \text{net}_i = d_i - \mathbf{w}_i^t \mathbf{x}$$

- Alors, l'ajustement du vecteur de poids est réalisé à l'aide de la relation: $\Delta \mathbf{w}_i = c r \mathbf{x} = c [d_i - \mathbf{w}_i^t \mathbf{x}] \mathbf{x}$

où $\Delta w_{ij} = c [d_i - \mathbf{w}_i^t \mathbf{x}] x_j$ pour $j = 1, 2, \dots, n$

- Les poids sont initialisés à des valeurs arbitraires choisies aléatoirement dans cette méthode.
- Cette règle peut être considérée comme un cas particulier de la règle d'apprentissage delta avec $f(\mathbf{w}_i^t \mathbf{x}) = \mathbf{w}_i^t \mathbf{x}$. Si $f(\text{net}) = \text{net}$, et $f'(\text{net}) = 1$ et nous obtenons le signal d'apprentissage $r = d_i - \mathbf{w}_i^t \mathbf{x}$.

5) Règles d'apprentissage

Règle d'apprentissage de type *corrélation*

► Propriétés de cette règle:

- Cette règle est utilisée pour enregistrer les données dans les RNA de type mémoires associatives, utilisant des neurones avec des fonctions d'activation binaire.
- Elle est applicable pour l'entraînement en mode supervisé
- Elle peut être considérée comme un cas particulier de la règle entraînement de Hebb (mode non supervisé) avec une fonction d'activation binaire et $o_i = d_i$. Sinon, la méthode d'ajustement des poids entre les deux méthodes est la même.

5) Règles d'apprentissage

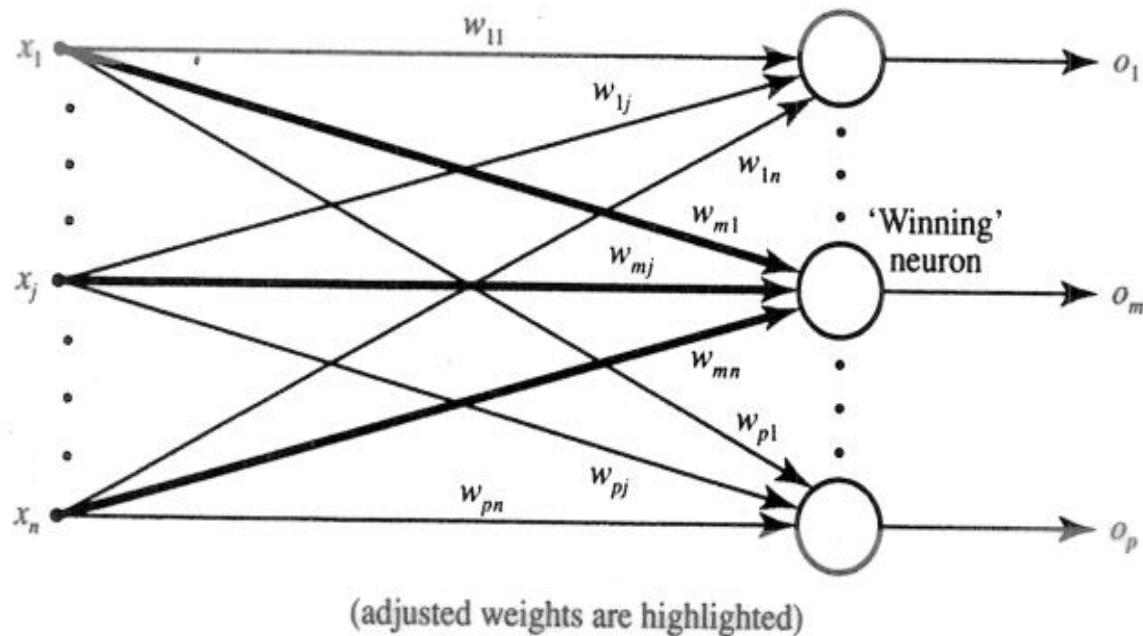
Règle d'apprentissage de type *corrélation*

- En substituant $r = d_i$ dans la règle d'apprentissage généralisée, nous obtenons la règle d'apprentissage de type corrélation.
- Elle stipule que si d_i est la réponse désirée due à x_j , le poids w_{ij} correspondant augmente proportionnellement à la valeur de leur produit.
- Alors, l'ajustement du vecteur de poids est réalisé à l'aide de la relation: $\Delta \mathbf{w}_i = c r \mathbf{x} = c d_i \mathbf{x}$
où $\Delta w_{ij} = c d_i x_j$ pour $j = 1, 2, \dots, n$
- La règle d'apprentissage par corrélation nécessite l'initialisation des poids avec $\mathbf{W} = \mathbf{0}$.

5) Règles d'apprentissage

Règle d'apprentissage de type *winner-take-all*

- **Architecture:** Cette règle d'apprentissage s'applique à une couche composée de p neurones. A titre d'exemple:



5) Règles d'apprentissage

Règle d'apprentissage de type *winner-take-all*

► Propriétés de cette règle:

- Cette règle est utilisée pour enregistrer les données dans les RNA de type mémoires associatives, utilisant des neurones avec des fonctions d'activation binaire.
- Elle est applicable pour l'entraînement en mode supervisé
- Elle peut être considérée comme un cas particulier de la règle entraînement de Hebb (mode non supervisé) avec une fonction d'activation binaire et $o_i = d_i$. Sinon, la méthode d'ajustement des poids entre les deux méthodes est la même.

5) Règles d'apprentissage

Règle d'apprentissage de Widrow-Hoff

- Le résultat de cette compétition est que le vecteur de poids $\mathbf{w}_m = [w_{m1}, w_{m2}, \dots, w_{mp}]^t$ est le seul vecteur qui sera modifié à ce stage de l'entraînement à l'aide de l'équation suivante:

$$\Delta \mathbf{w}_m = \alpha (\mathbf{x} - \mathbf{w}_m)$$

où $\Delta w_{mj} = \alpha (x_j - w_{mj})$ pour $j = 1, 2, \dots, n$

- Dans la formule précédente, $\alpha > 0$ est une constante d'apprentissage qui normalement décroît en fonction du temps.
- Les poids du réseaux sont normalement initialisés à des valeurs choisies aléatoirement
- De plus, les données sont normalisées dans cette méthode.

5) Règles d'apprentissage

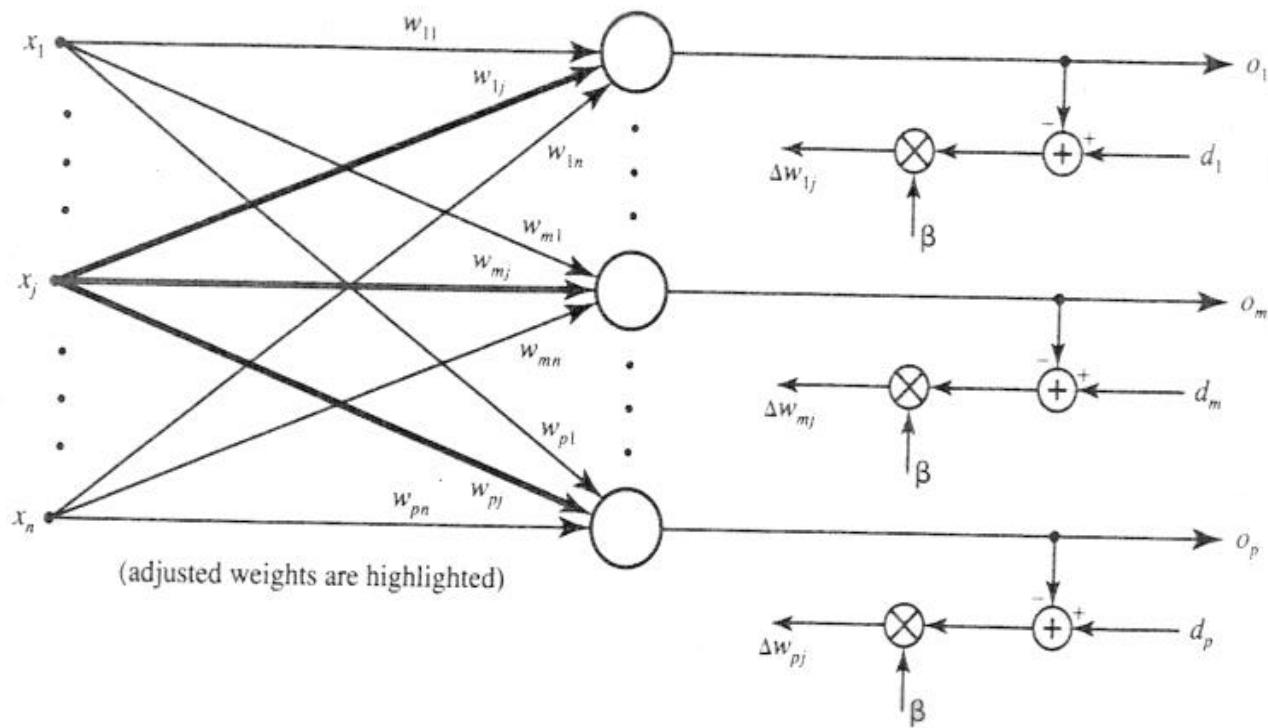
Règle d'apprentissage de Widrow-Hoff

- La sélection du gagnant m de cette compétition est basée sur un critère d'activation maximale parmi les p neurones participants:
$$m = \operatorname{argmax} \{ \mathbf{w}_i^T \mathbf{x} : i = 1, 2, \dots, p \}$$
- En fait, ce critère correspond à trouver le vecteur de poids \mathbf{w}_m le plus proche du vecteur \mathbf{x} .
- Cette règle d'apprentissage se résume à modifier le vecteur \mathbf{w}_m par une fraction de $(\mathbf{x} - \mathbf{w}_m)$.
- Certains types de réseaux compétitifs ajustent non seulement les poids du neurone gagnant, mais également les poids des neurones localisés dans le voisinage immédiat du gagnant.

5) Règles d'apprentissage

Règle d'apprentissage de type *outstar*

► **Architecture:**



5) Règles d'apprentissage

Règle d'apprentissage de type *outstar*

► Propriétés de cette règle:

- Cette règle d'apprentissage est utilisée en mode d'entraînement supervisé et appliquée sur une couche de neurones.
- Elle est conçue pour produire la réponse désirée d par le réseau monocouche constituée de p neurones.
- Elle permet l'apprentissage des propriétés intrinsèques et répétitives des relations d'entrées-sorties.
- Enfin, elle fait en sorte que le patron de sortie du réseau devient semblable au patron désiré (sans distorsion) après avoir présenté au réseau de façon répétitive des versions bruitées de la sortie désirée.

5) Règles d'apprentissage

Règle d'apprentissage de type *outstar*

- Il est intéressant de noter que les poids sont associés à la $j^{\text{ième}}$ entrée et sont distribués aux p noeuds du réseau suivant la définition suivante: $\mathbf{w}_j = [w_{1j}, w_{2j}, \dots, w_{pj}]^t$
- **Les poids sont ajustés selon** l'équation suivante:

$$\Delta \mathbf{w}_j = \beta (\mathbf{d} - \mathbf{w}_j)$$

où $\Delta w_{mj} = \alpha (d_m - w_{mj})$ pour $m = 1, 2, \dots, p$

- Dans la formule précédente, $\beta > 0$ est une constante d'apprentissage qui normalement décroît en fonction du temps.

5) Règles d'apprentissage

► Synthèse: les règles d'apprentissage et leurs propriétés

Learning rule	Single weight adjustment Δw_{ij}	Initial weights	Learning	Neuron characteristics	Neuron / Layer
Hebbian	$c o_i x_j$ $j = 1, 2, \dots, n$	0	U	Any	Neuron
Perceptron	$c [d_i - \text{sgn}(\mathbf{w}_i^T \mathbf{x})] x_j$ $j = 1, 2, \dots, n$	Any	S	Binary bipolar, or Binary unipolar*	Neuron
Delta	$c(d_i - o_i)f'(net_i)x_j$ $j = 1, 2, \dots, n$	Any	S	Continuous	Neuron
Widrow-Hoff	$c(d_i - \mathbf{w}_i^T \mathbf{x})x_j$ $j = 1, 2, \dots, n$	Any	S	Any	Neuron
Correlation	$c d_i x_j$ $j = 1, 2, \dots, n$	0	S	Any	Neuron
Winner-take-all	$\Delta w_{mj} = \alpha(x_j - w_{mj})$ $m\text{-winning neuron number}$ $j = 1, 2, \dots, n$	Random Normalized	U	Continuous	Layer of p neurons
Outstar	$\beta(d_i - w_{ij})$ $i = 1, 2, \dots, p$	0	S	Continuous	Layer of p neurons

c, α, β are positive learning constants
 S — supervised learning, U — unsupervised learning
 * — Δw_{ij} not shown

5) Règles d'apprentissage

► Synthèse: principaux réseaux de neurones

Network Architecture	Learning Mode S, U, R	Recall Mode FF, REC	Recall Time Domain CT, DT
Single-layer Network of Discrete and Continuous Perceptrons (Figure 1.1)	S	FF	—
Multilayer Network of Discrete and Continuous Perceptrons (EEG spike detectors, ALVINN, Figures 1.3 and 2.9)	S	FF	—
Gradient-type Network (Figures 1.8 and 2.15)	R	REC	CT
Linear Associative Memory	R	FF	—
Autoassociative Memory [Figures 1.5, 1.7, 2.12, and 2.16(a)]	R	REC	DT or CT
Bidirectional Associative Memory [Figure 2.16(b)] (also Multidirectional Associative Memory)	R	REC	DT or CT
Temporal Associative Memory	R	REC	DT or CT
Hamming Network	R	FF	—
MAXNET	R (fixed)	REC	DT or CT
Clustering Network (Figure 2.25)	U	FF	—
Counterpropagation Network (Figure 2.25 and 2.26)	U + S	FF	—
Self-Organizing Neural Array (Figure 1.11)	U	FF	—
Adaptive Resonance Theory 1 Network	U	REC	DT or CT

Learning Mode
S—Supervised
U—Unsupervised
R—Recording (Batch)

Recall Mode
FF—Feedforward
REC—Recurrent

Recall Time Domain (only for recurrent networks)
CT—Continuous-time
DT—Discrete-time