

INTELLIGENCE ARTIFICIELLE

Partie 3: RESEAUX DE NEURONES ARTIFICIELS

Tarik AL ANI, Département Informatique

ESIEE-Paris

E-mail : t.alani@esiee.fr

Url: <http://www.esiee.fr/~alanit>

Histoire

↻ 1943 McCulloch et Pitts [McCulloch 43] privilégiaient la modélisation du *neurones biologiques* par des *automates linéaires à seuil* (*neurone formel*) interconnectés. Leurs travaux montrent que ces modèles étaient capables de calculer certaines fonctions logiques d'où naquit l'idée de l'ordinateur universel [Minsky 67].

➤ 1948 Von Neumann envisageait les *réseaux d'automates* [Von Neumann 56].

➤ 1949 Hebb souligna l'importance du *couplage synaptique* dans le processus d'apprentissage en se fondant son argumentation sur le comportement psychophysologique [Hebb 49] : première règle d'apprentissage.

➤ 1962 la naissance d'une méthode analytique rigoureuse d'adaptation de poids au sein d'un modèle multi-couches appelée le perceptron [Rosenblatt 62].

Rosenblatt démontra la convergence d'un algorithme itératif d'adaptation des poids pour la classe élémentaire des perceptrons mono-couche.

↳ Inspiré des idées de Hebb, McCulloch et Pitts, le perceptron était en mesure d'apprendre à calculer certaines fonctions logiques par l'exemple en modifiant ses connexions synaptiques.

➤ Des réseaux similaires appelés adalines furent inventés à la même époque [Widrow 60].

↪ 1969 Minsky et Papert [Minsky 69] ont apporté un éclairage théorique sur la structure des problèmes qu'il était impossible de résoudre avec le perceptron.

➡ problème de OU-exclusif (XOR).

➡ Abandonner l'approche connexionniste durant près de 20 ans ➡ Systèmes experts.

➤ Toutefois Rosenblatt, au vu de ses travaux sur les *réseaux multi-couches*, a conclu que :
Les limitations du perceptron pourraient être levées si un algorithme d'apprentissage existait, capable d'ajuster les poids des connexions internes.

↪ les années 1970-1986: Les chercheurs relancent les travaux sur les Réseaux de Neurones:

- réalisation d'une *mémoire associative*,
- un formalisme plus riche a été développé par Hopfield [Hopfield 82], [Hinton 86], [Peretto 84].

↪ En 1974 Werbos [Werbos 74] a inventée l'idée de la *rétro-propagation*.

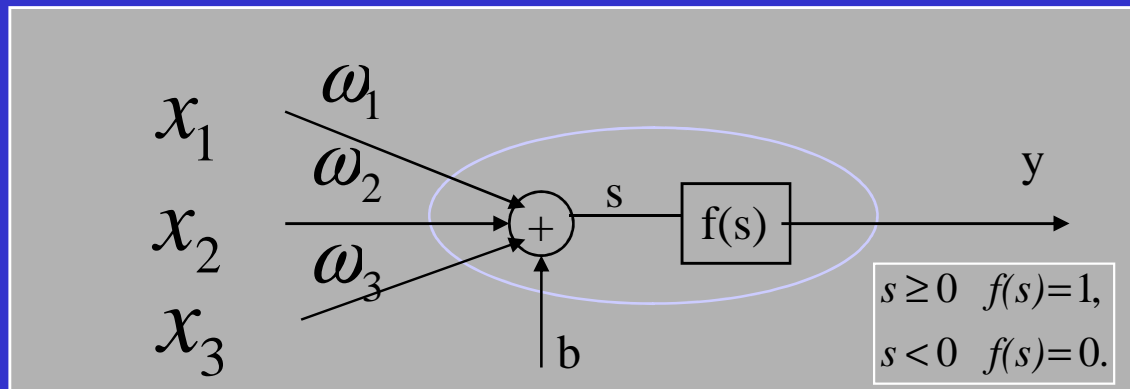
Destiné aux réseaux avec des connexions non-bouclées, *l'algorithme de rétro-propagation* a été développé indépendamment en 1985 par Rumelhart [Rumelhart 86], Parker [Parker 85] et Le Cun [Le Cun 85].

➤ Années 90: Autres modèles et algorithmes ad hoc ont été développés avec l'attention d'étendre les possibilités des réseaux statiques non-bouclés: introduction progressive d'une mémoire interne et de la récurrence au sein du réseau.

Le neurone formel de McCulloch

Un neurone formel fait une somme pondérée des potentiels d'action qui lui parviennent (chacun de ces potentiels est une valeur numérique qui représente l'état du neurone qui l'a émis), puis s'active suivant la valeur de cette sommation pondérée. Si cette somme dépasse un certain seuil, le neurone est activé et transmet une réponse (sous forme de potentiel d'action) dont la valeur est celle de son activation. Si le neurone n'est pas activé, il ne transmet rien.

Le modèle statique de McCulloch [McCulloch 43]



Le neurone formel.

Qu'il soit simulé par logiciel ou réalisé sous une forme d'une puce électronique, le neurone est avant tout un *automate*. A partir des signaux d'entrée qu'il reçoit des automates qui lui sont reliés et de son état interne, il produit un signal de sortie qui devient son nouvel état interne.

Fonctions d'activation:

f est une fonction non-linéaire de la somme pondérée des entrées (a:activation) appelée **fonction d'activation** (état du système)

$$s_i = a + b = \sum_{j=1}^N \omega_j x_j + b = W' \cdot X + b$$
$$y_i = f(s_i)$$

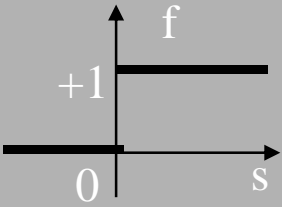
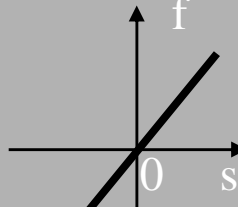
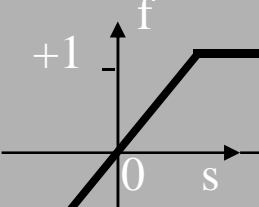
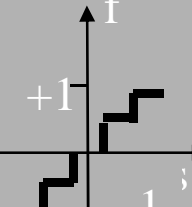
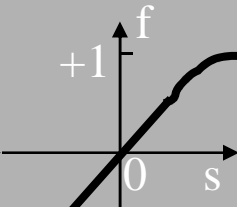
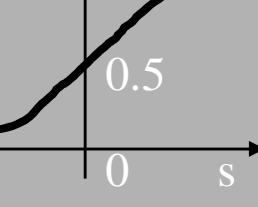
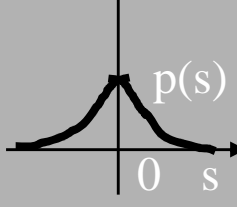
où x_j sont les entrées au neurone qui proviennent de l'environnement externe au réseau ou d'autres neurones. X est le vecteur d'entrée.

ω_{ij} sont les poids synaptiques associés à chaque connexion qui peut varier au cours du temps pour certains modèles. W est le vecteur poids.

b est appelé le biais du neurone = $-\beta$ ou $+\beta$ (seuil) (signal d'inhibition de neurone).

$$a_i(x) = \sum_{j=1}^N \omega_j x_j \quad \text{est l'activation du neurone } i.$$

Fonctions d'activation (fonction de transfert)

Fonction Heaviside	Fonction linéaire sans saturation	Fonction linéaire avec seuil	Fonction à seuils multiples	Fonction sigmoïde $f(s) = \frac{e^{-s} - 1}{e^s + 1}$	Fonction sigmoïde $f(s) = \frac{1}{1 + e^{-s}}$	Fonction stochastique $P(s) = \frac{1}{1 + e^{-\frac{s}{T}}}$ T = Température
						

f peut grossièrement être représentée par une fonction de Heaviside, Voir tableau, ou une fonction seuil du type ($y_i = f(a_i + b_i)$, $b_i = -\beta_i$)

$$y_i = +1 \quad \text{si} \quad a_i \geq \beta_i$$

neurone i est activé

$$y_i = 0 \quad \text{si} \quad a_i < \beta_i$$

neurone i est inactivé

i est l'indice du neurone.

Il est plus commun d'employer une fonction non-linéaire bornée et dérivable sur $]-\infty, +\infty[$ [plutôt qu'une fonction linéaire par morceaux. La fonction *sigmoïde*, voir tableau,

$$f(s) = \frac{1}{1 + e^{-s}}$$

ou la fonction $f(s) = \tanh(s)$, qui est une fonction couramment utilisée.

Ce modèle n'a pas pour un rôle de représenter toutes les caractéristiques de neurone biologique:

- Le processus de transport par l'axone et la modulation synaptique sont cruellement réduits à une simple multiplication de deux vecteurs ωx .
- La dynamique électro-chimique complexe, à la source de la propagation du signal, est ramenée à sa plus simple expression par une fonction statique, f .

- Il n'y a pas de dynamique interne au sein de ce modèle; pour des poids synaptiques fixes, le neurone formel accomplit une transformation statique.

Dans un sens, **ce modèle ne capture que les propriétés spatiales du neurone biologique** et passe outre les relations temporelles.

La capacité représentative du neurone formel est fortement restreinte en raison de la forme rudimentaire de la fonction f utilisée (seuil).

Exemple: Lorsque $f(s)=\text{sign}(s)$, le neurone est associé à un hyperplan « paramétré » par les poids synaptiques.

Au vecteur d'entrée x_1, x_2, \dots, x_N correspond une sortie +1 ou -1 suivant le côté où se situe le vecteur par rapport à cet hyperplan. Aussi, **seul des fonctions linéairement séparables peuvent être réalisées.**

La combinaison linéaire de fonctions linéaires est linéaire: l'interconnexion de neurones linéaires ne pourra jamais approcher une fonction non-linéaire aussi simple soit elle.

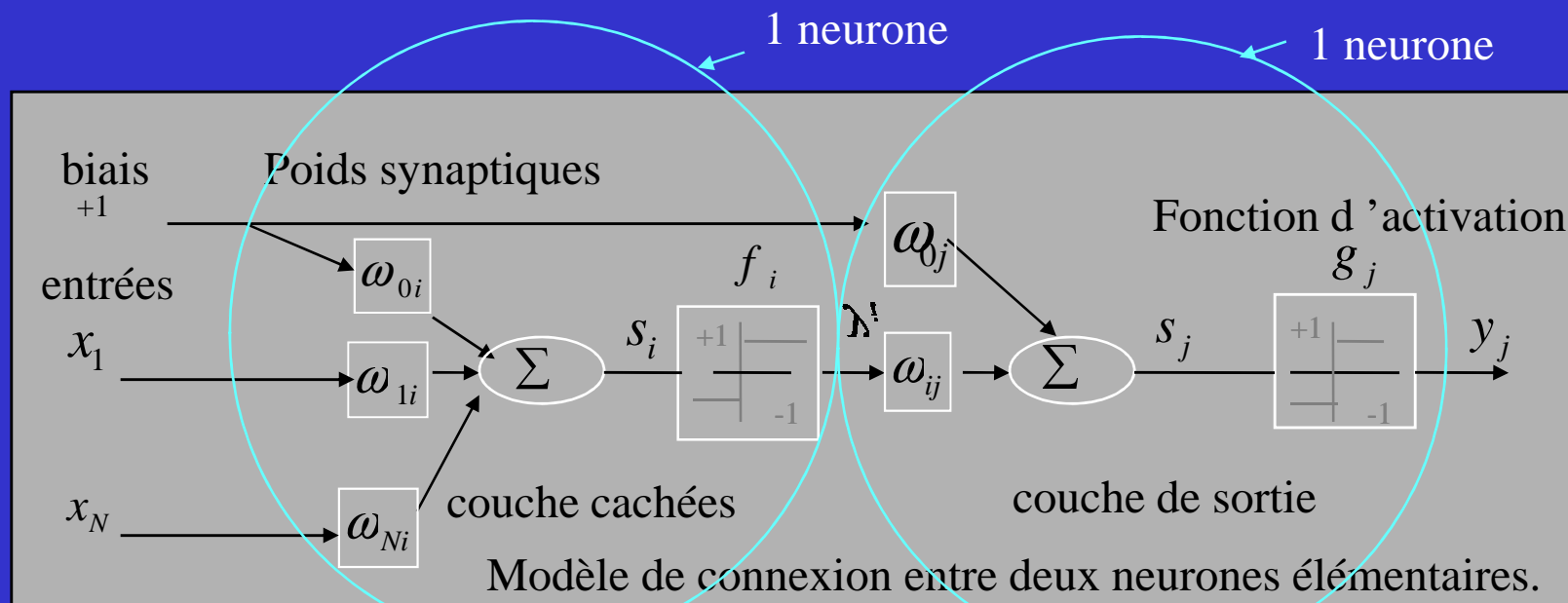
Réseaux de Neurones (RN)

Un RN est un assemblage d'éléments élémentaires de traitement. La capacité de traitement de ce réseau est stocké sous forme de *poids d'interconnexions* obtenus par un processus d'*adaptation* ou d'*apprentissage* à partir d'un ensemble d'exemples d'apprentissage.

Modèle de la connexion entre deux neurones élémentaires

α : le pas d'adaptation; ω_{ij} : poids synaptique

f_i et f_j : fonctions d'activation ; s_i et s_j : valeurs d'activation;
 x_n : une composante d'entrée au réseau



Types de réseaux de neurones:

1. Perceptron (P) un ou plusieurs (adaline) neurones formels sur une seule couche
2. Réseaux statiques type « feed forward » (FFNR) avec multicouches sans bouclage des sorties à la première couche
3. Réseaux statiques type « Radial Basis Functions (RBF) » avec une couche sans bouclage des sorties à la première couche.
 - 3.1 Réseaux à régression généralisées (Generalized regression neural network (GRNN))
 - 3.2 Réseaux probabilistes (Probabilistic neural networks (PNN)).
4. Réseaux partiellement récurrents « feed forward » (Elman ou Jourdan) avec multicouches où uniquement les sorties sont bouclées à la première couche.
5. Réseaux récurrents mono-couche à connexité totale (Réseaux associatifs)
6. Réseaux compétitifs (Self-Organizing and Learning Vector Quantization Nets)
7. Réseaux dynamiques

1. Perceptron (P) un ou plusieurs neurones formels sur une seule couche

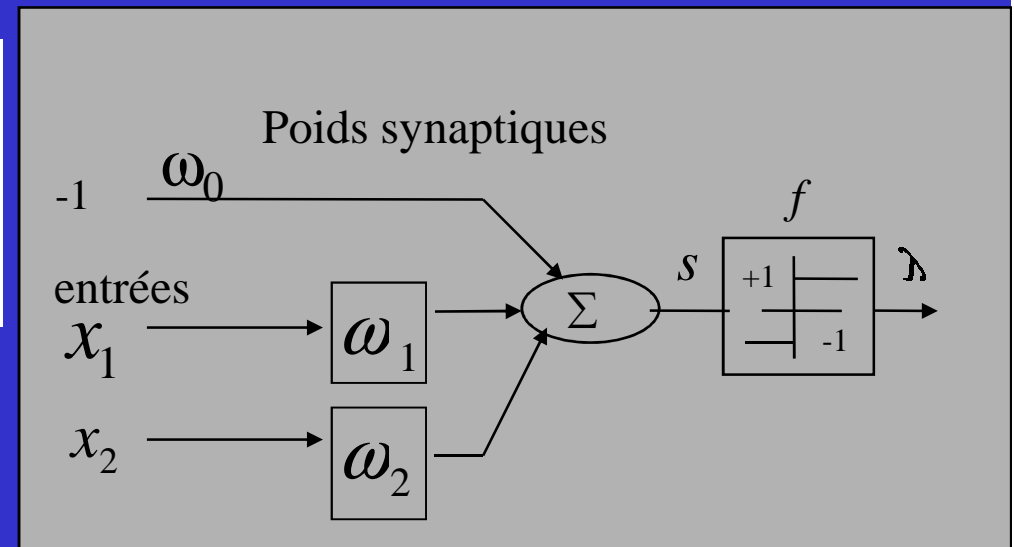
$b = \text{biais} = -\omega_0$, ω_0 est un seuil

$$\text{activation} : a(x) = X^T \bullet W = \sum_{j=1}^2 \omega_j x_j$$

$$W = \text{vecteur de poids} = [\omega_1 \ \omega_2]^T$$

$$s = X^T \bullet W + b$$

$$y = f(s)$$



Séparation linéaire des classes

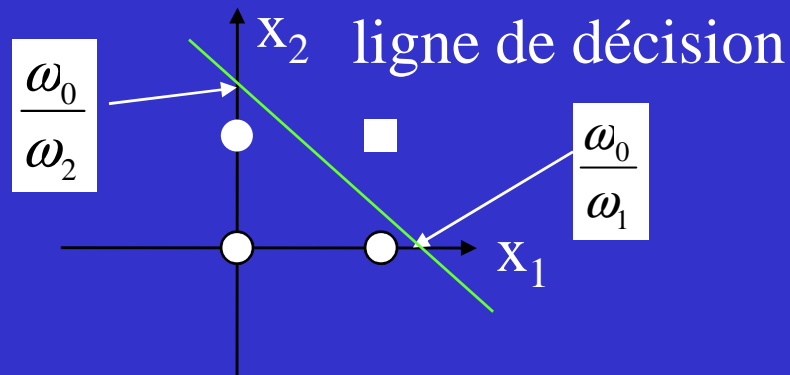
Condition critique de séparation : activation = seuil

$$b = \text{biais} = -\omega_0$$

$X^T \bullet W = \omega_0$ équation d'une ligne droite de décision

- Dans l'espace 2D si $\omega_0 > 0$ nous obtenons une équation d'une ligne droite avec une pente $p = -\frac{\omega_1}{\omega_2}$

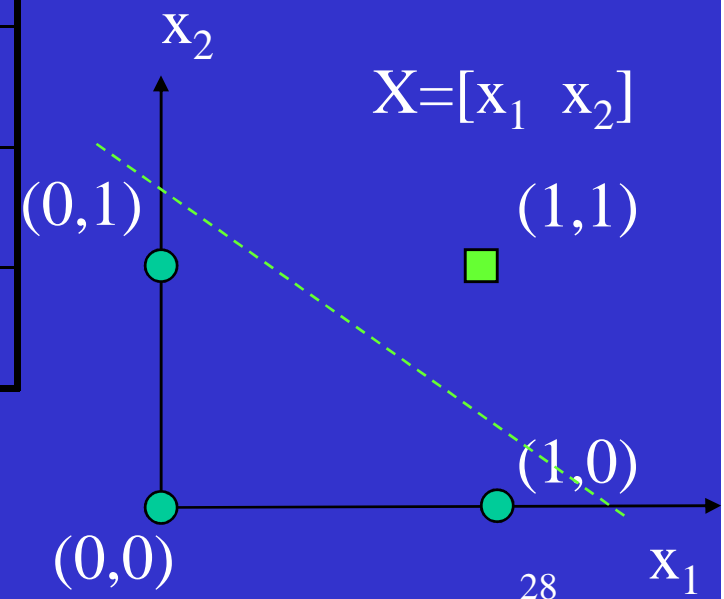
$$x_2 = -\frac{\omega_1}{\omega_2} x_1 + \frac{\omega_0}{\omega_2}$$



Exemple d'un espace d'entrée = 2 avec 2 classes
(classe 1: ● , classe 2: ■)

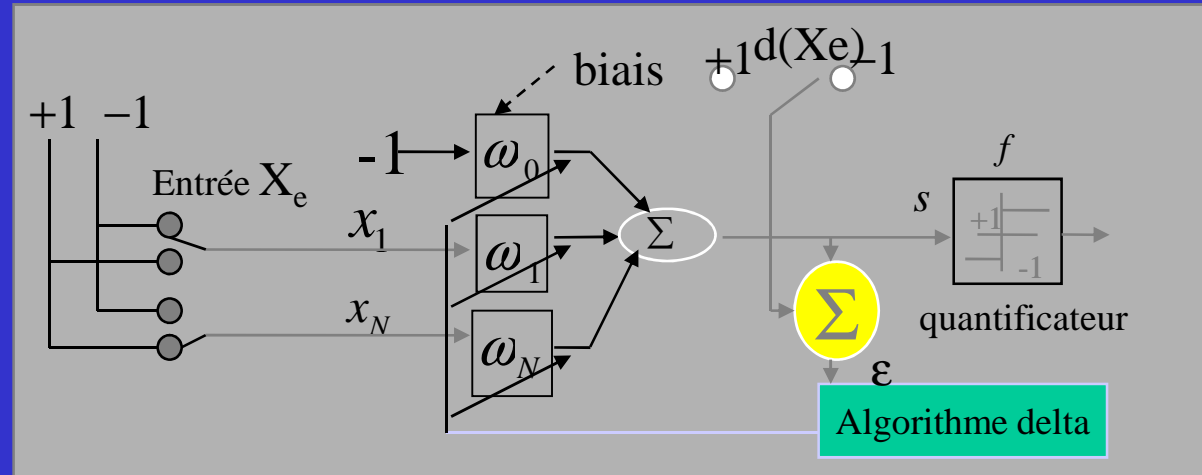
$\omega_1 = \omega_2 = 1$, seuil $\omega_0 = 1,5$

x_1	x_2	activation	sortie
0	0	0	0 ●
0	1	1	0 ●
1	0	1	0 ●
1	1	2	1 ■

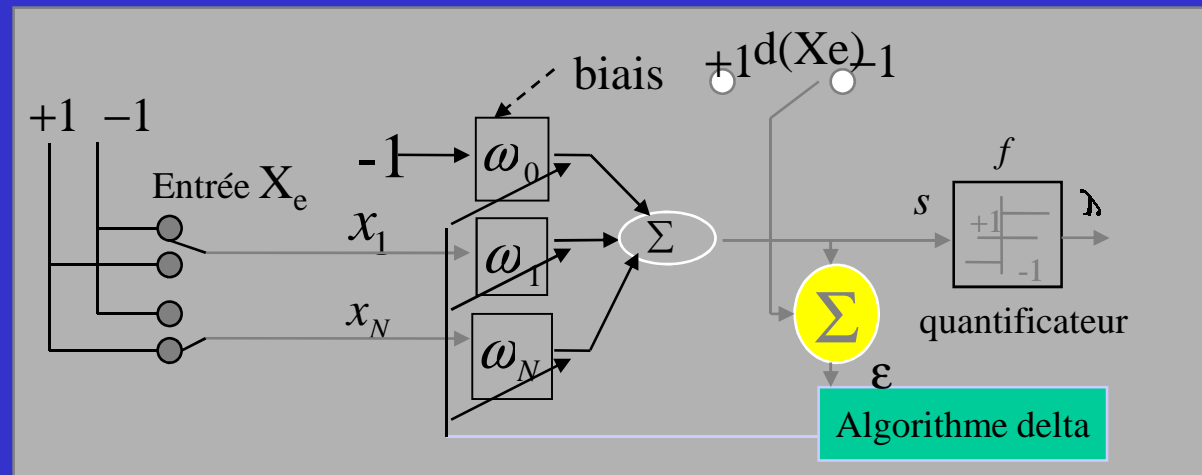


ADALINE (Single Layer Feedforward Networks)

1

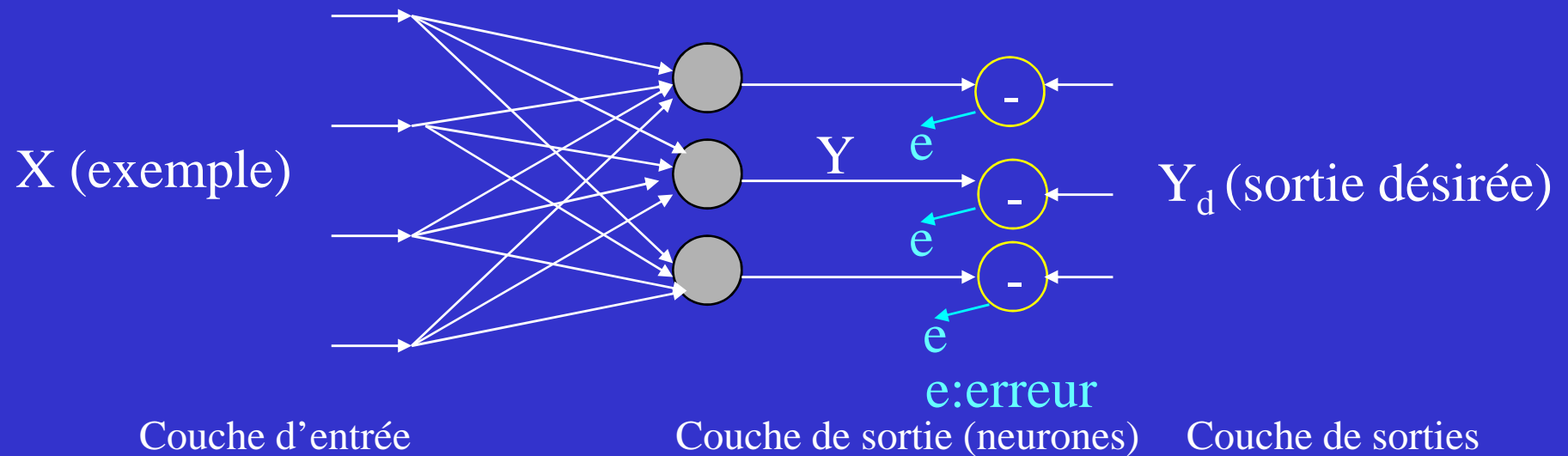


n



ADALINE (Single Layer Feedforward Networks)

Type d'apprentissage: supervisé (X, Y_d)



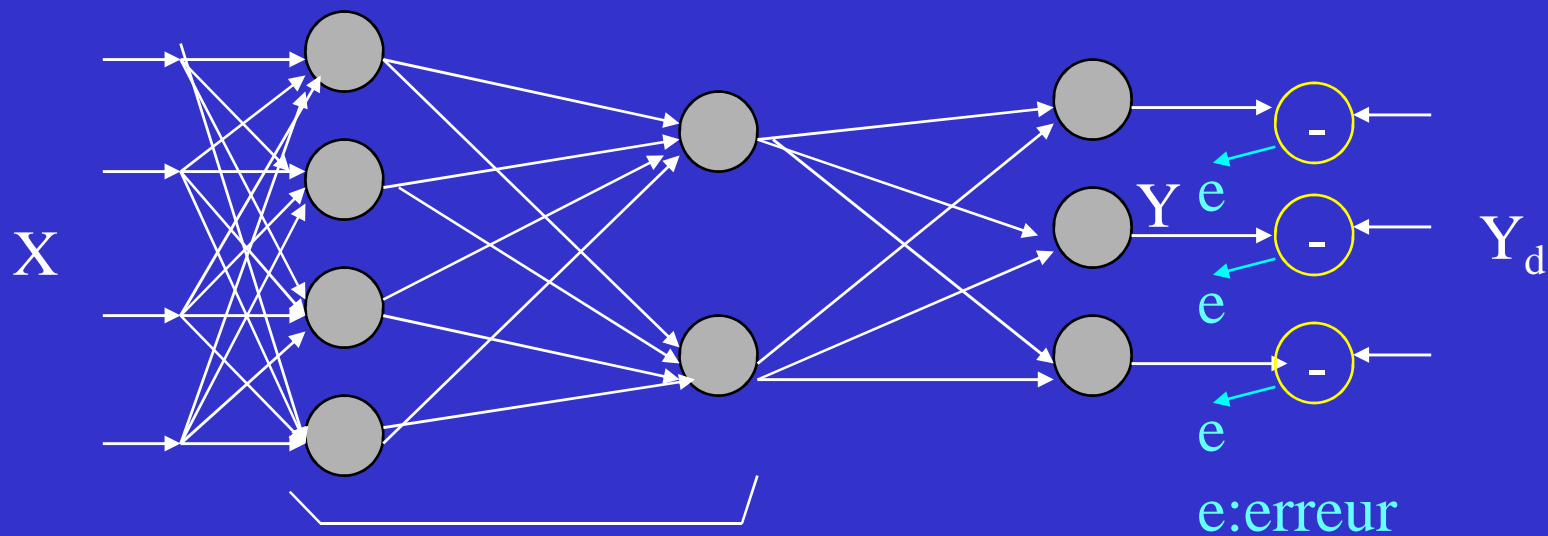
ADALINE (suite)

1. L'entrée et la sortie sont vectorielles
2. Une fois l'entrée X (exemple d'apprentissage) est présentée aux réseaux, les activations de neurones sont calculées une fois les réseaux sont stabilisés.
3. Les erreurs ($E = X - Y_d$) sont calculées.
4. Les erreurs (E) sont minimisées par un algorithme d'apprentissage.

2. Réseaux statiques type « feed forward » (FFNR) avec multicouches sans bouclage des sorties à la première couche

- Certains problèmes non linéaires (ou logiques, e.g. ou exclusive) ne sont pas résolubles par des réseaux à une couche :
 - Une ou plusieurs couches intermédiaires (cachées) sont ajoutées entre la couche d'entrée et la couche de sortie pour permettre aux réseaux de créer sa propre représentation des entrées. Dans ce cas il est possible d'effectuer une approximation d'une fonction non linéaire quelconque ou réaliser une fonction logique quelconque.

2. Réseaux statiques type « feed forward » (FFNR) avec multicouches sans bouclage des sorties à la première couche (suite)



Couche d'entrée Une ou plusieurs couches cachées

Couche de sortie Couche de
sorties
désirées

3. Réseaux statiques type « Radial Basis Functions (RBF) » avec une couche sans bouclage des sorties à la première couche.

Les RBF peuvent exiger plus de neurones que les FFNR, mais souvent ils peuvent être conçus dans une fraction du temps par rapport au temps nécessaire pour entraîner les FFNR. Ces réseaux fonctionnent mieux quand beaucoup de vecteurs d'apprentissage sont disponibles.

Nécessitent plus de neurones par rapport aux *FFNN*

Réseaux RBF (suite)

Utilisés pour :

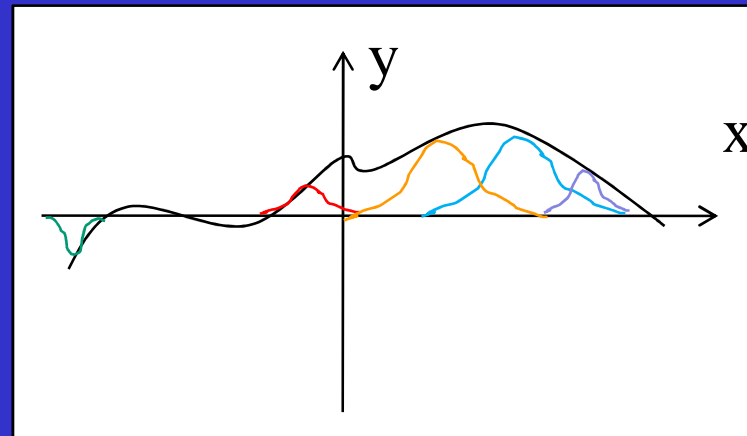
- Régression : *Generalized regression networks (GRNN)*
- Classification : *Probabilistic neural networks (PNN)*

Réseaux RBF (suite)

3.1 Generalized regression networks (**GRNN**)

Une fonction continue quelconque peut être approchée par une combinaison linéaires des fonctions gaussiennes bien choisies.

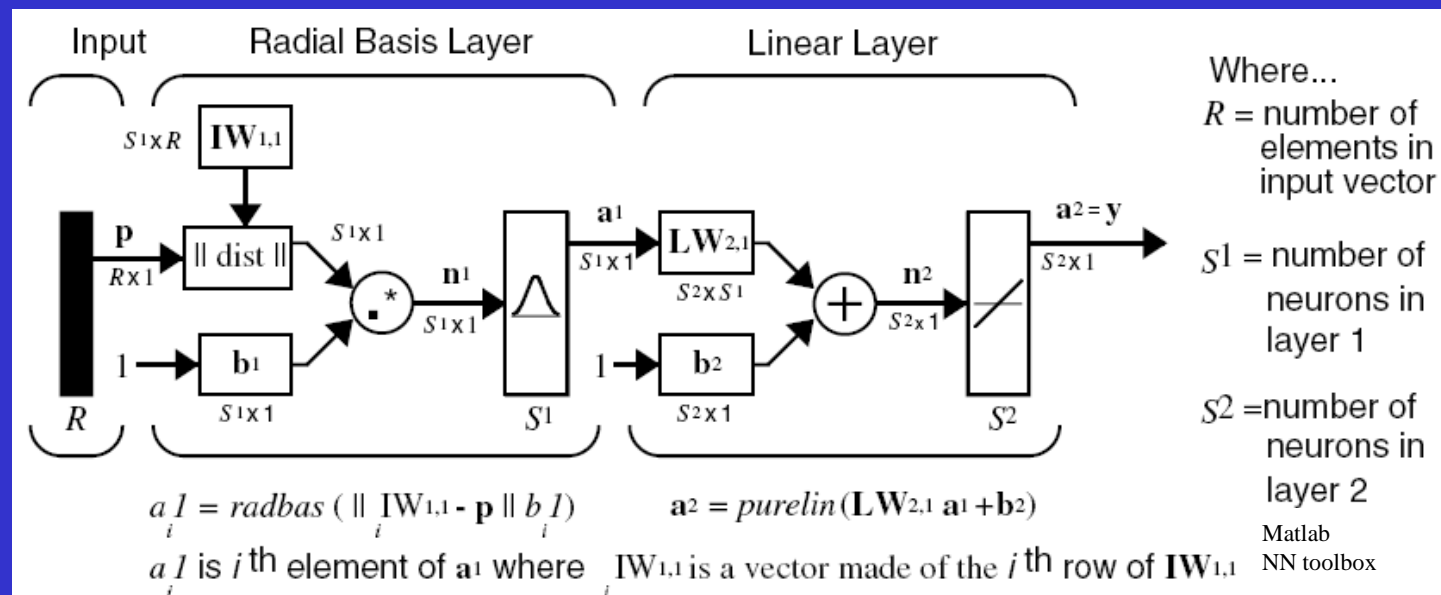
Objectif : Régression : construire une bonne approximation d'une fonction qui n'est connue que par un nombre fini de points "expérimentaux" faiblement bruités $\{x_i, y_i\}$



Régression locale : les gaussiennes de base n'influent que sur des petites zones autour de leurs valeurs moyennes.

Réseaux RBF (suite)

Ce réseau peut être employé pour des problèmes de classification. Quand une entrée est présentée, la première couche calcule les distances entre le vecteur d'entrée et le vecteur de poids et produit un vecteur, multiplié par le biais.



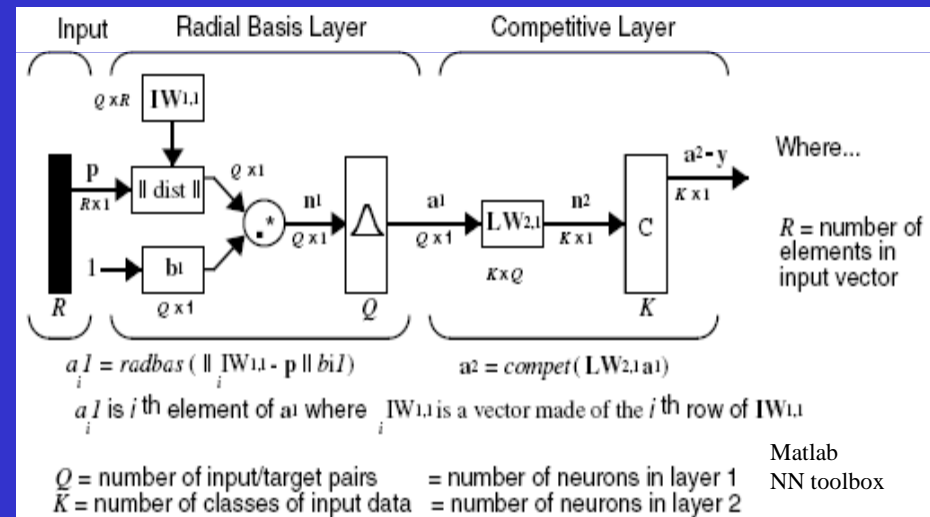
Réseaux RBF (suite)

3.2 Réseau probabiliste

Ce réseau peut être employé pour des problèmes de classification. Quand une entrée est présentée,

- la première couche calcule les distances entre le vecteur d'entrée et tous les vecteurs d'apprentissage et produit un vecteur dont les éléments indiquent comment ce vecteur d'entrée est proche de chaque vecteur d'apprentissage.

- la deuxième couche additionne ces contributions pour chaque classe des entrées pour produire à la sortie du réseau un vecteur des probabilités. Enfin, une fonction de transfert de concurrence sur la sortie de la deuxième couche sélectionne le maximum de ces probabilités, et produit un 1 pour cette classe et un 0 pour les autres classes.



Réseaux RBF (suite)

Remarques générales :

- Reproduire la fonction dans tout l'espace de données
→ paver l'espace par un très grand nombre de gaussiennes
- En pratique, la fonction RBF est normalisée et centrée a :

$$f(\mathbf{x}) = \exp(-\mathbf{x}^T \mathbf{x})$$

Réseaux RBF (suite)

- Les réseaux RBF sont peu performants :
 - Sur un espace de données (m) de grande dimension
 - Sur des données très bruitées. La reconstruction locale de la fonction empêche le réseau de "moyenner" le bruit sur tout l'espace (comparez avec la Régression Linéaire, dont l'objectif est justement de moyenner le bruit sur les données.).

Réseaux RBF (suite)

- Apprentissage des réseaux RBF
 - Apprentissage par des procédures d'optimisation : temps de calcul considérable → apprentissage très lent voir impossible en pratique
 - Apprentissage par des techniques **heuristiques** : **approximation** → la construction d'un réseau RBF est rapide et facile mais ils sont peu performants comparés aux réseaux de perceptrons multicouches (MLP).

Réseaux RBF (suite)

Conclusion sur les réseaux RBF :

- Une alternative crédible aux MLP sur des problèmes pas trop difficiles.
- Rapidité et facilité d'utilisation

[1] Chen, S., C.F.N. Cowan, and P.M. Grant, "Orthogonal Least Squares Learning Algorithm for Radial Basis Function Networks," IEEE Transactions on Neural Networks, Vol. 2, No. 2, March 1991, pp. 302-309.

<http://eprints.ecs.soton.ac.uk/1135/1/00080341.pdf>

[2] P.D. Wasserman, Advanced Methods in Neural Computing, New York: Van Nostrand Reinhold, 1993, on pp. 155-61 and pp. 35-55, respectively.

Réseaux récurrents (suite)

4. Réseaux partiellement récurrents « feed forward » (Elman ou Jourdan) avec multicouches où uniquement les sorties sont bouclées à la première couche.

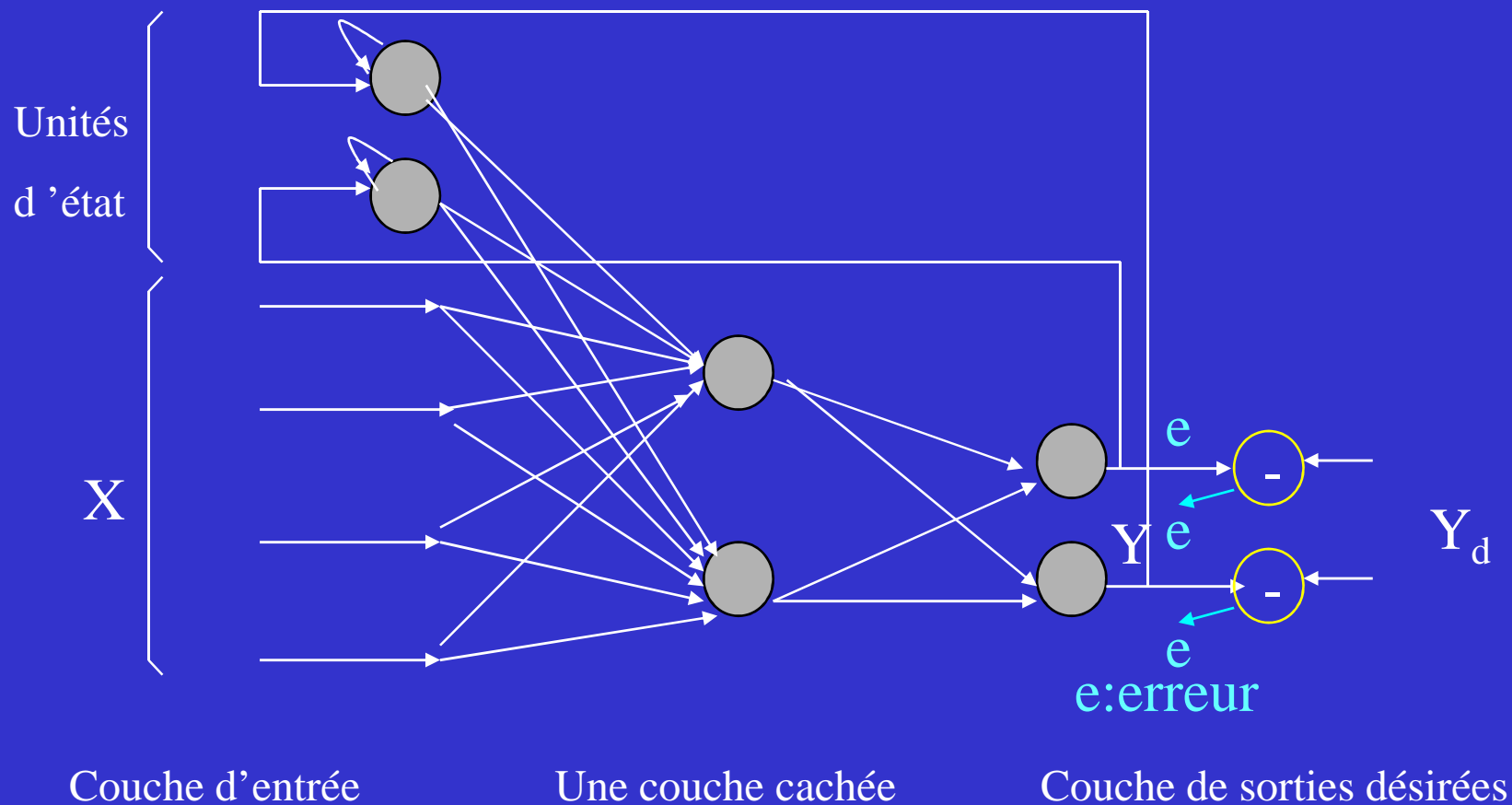
Ce sont des réseaux de type « *feedforward* » sauf qu'une boucle est effectuée entre les couches cachées et la couche de sortie ou entre les couches cachées elles mêmes par l'intermédiaire des couches supplémentaires appelées *couches d'état* ou *couches de contexte*.

Réseaux partiellement récurrents (suite)

Puisque le traitement dans les réseaux récurrents dépend de l'état des réseaux à l'itération précédente, ces réseaux peuvent alors être utilisés pour modéliser des séquences temporelles (système dynamique).

Réseaux partiellement récurrents (suite)

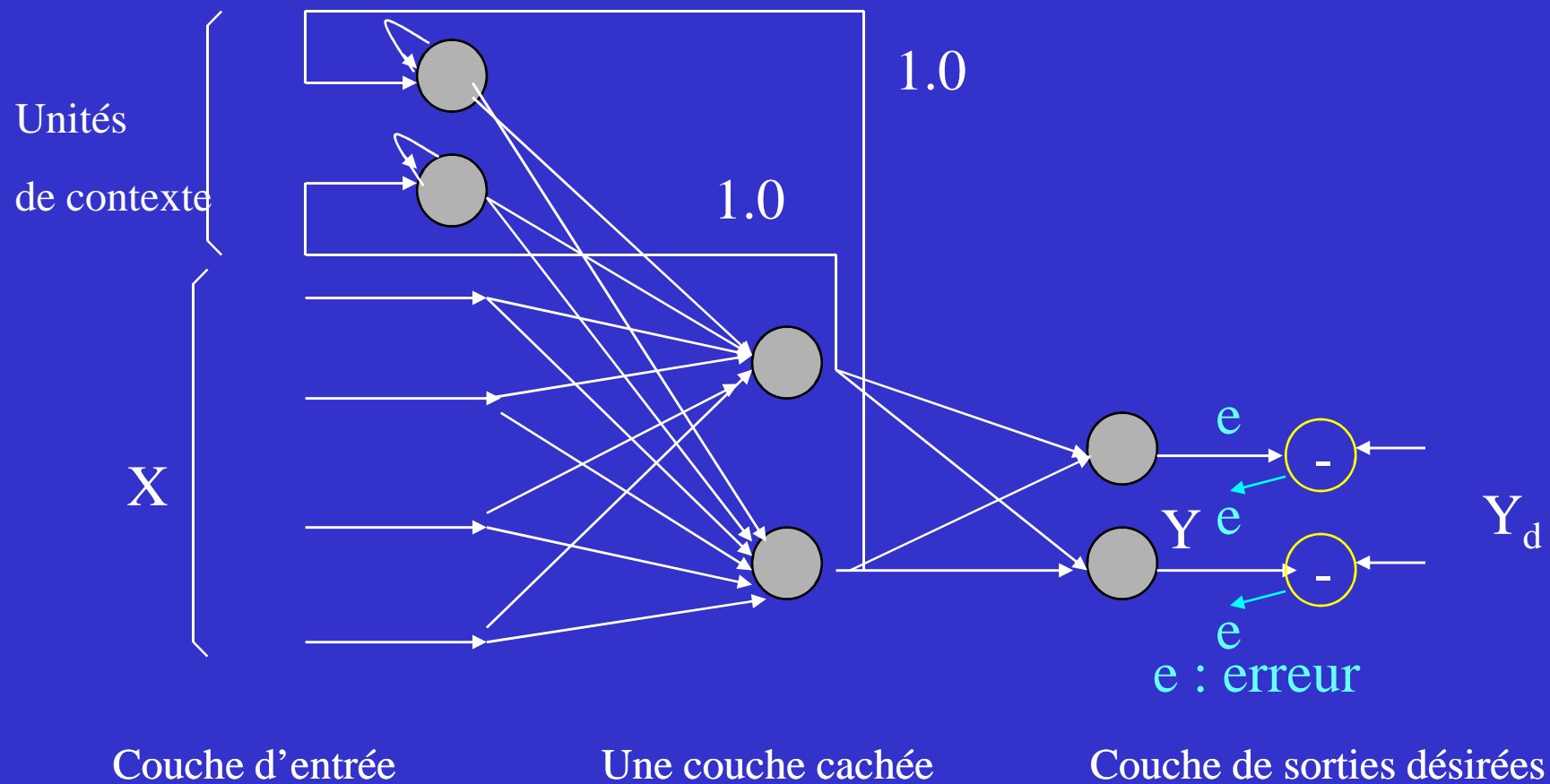
Réseaux de Jordan (Jordan 86a, b)



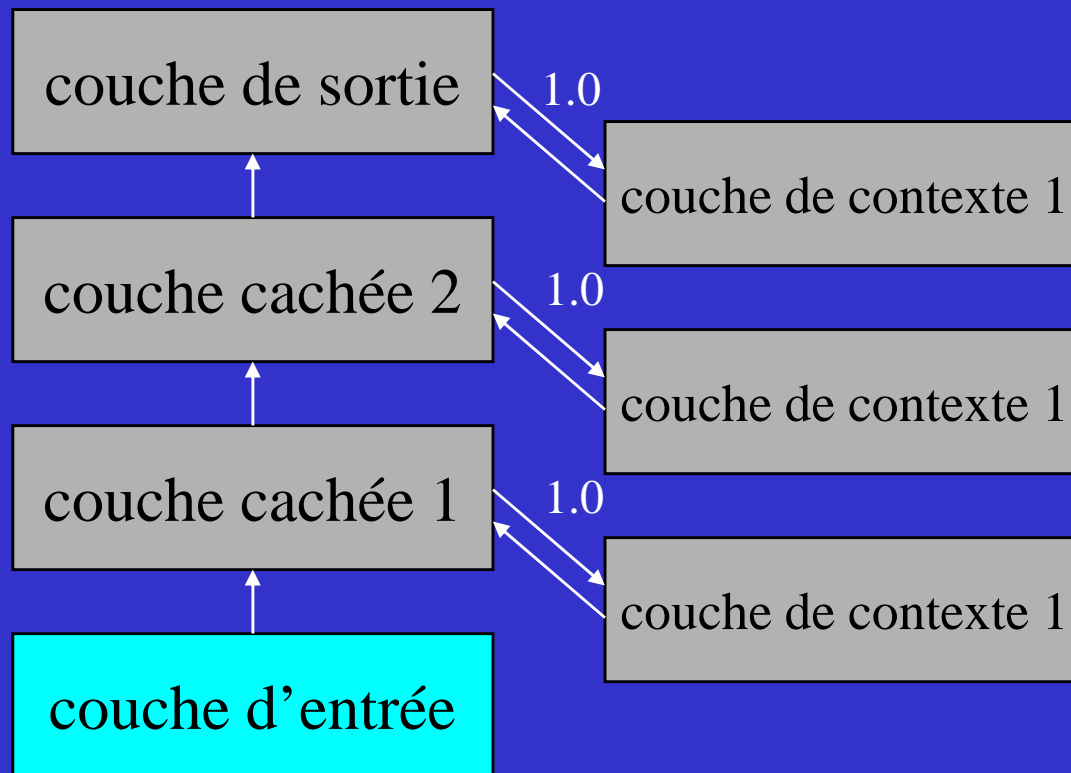
Réseaux de Elman [Elman 1990]

Dans ce cas une couche supplémentaire d'unités contextuelles (context units) est introduite. Les entrées de ces unités sont les activations des unités de la couche cachée.

Réseaux partiellement récurrents - Réseaux de Elman (suite)



Réseaux de Elman étendus



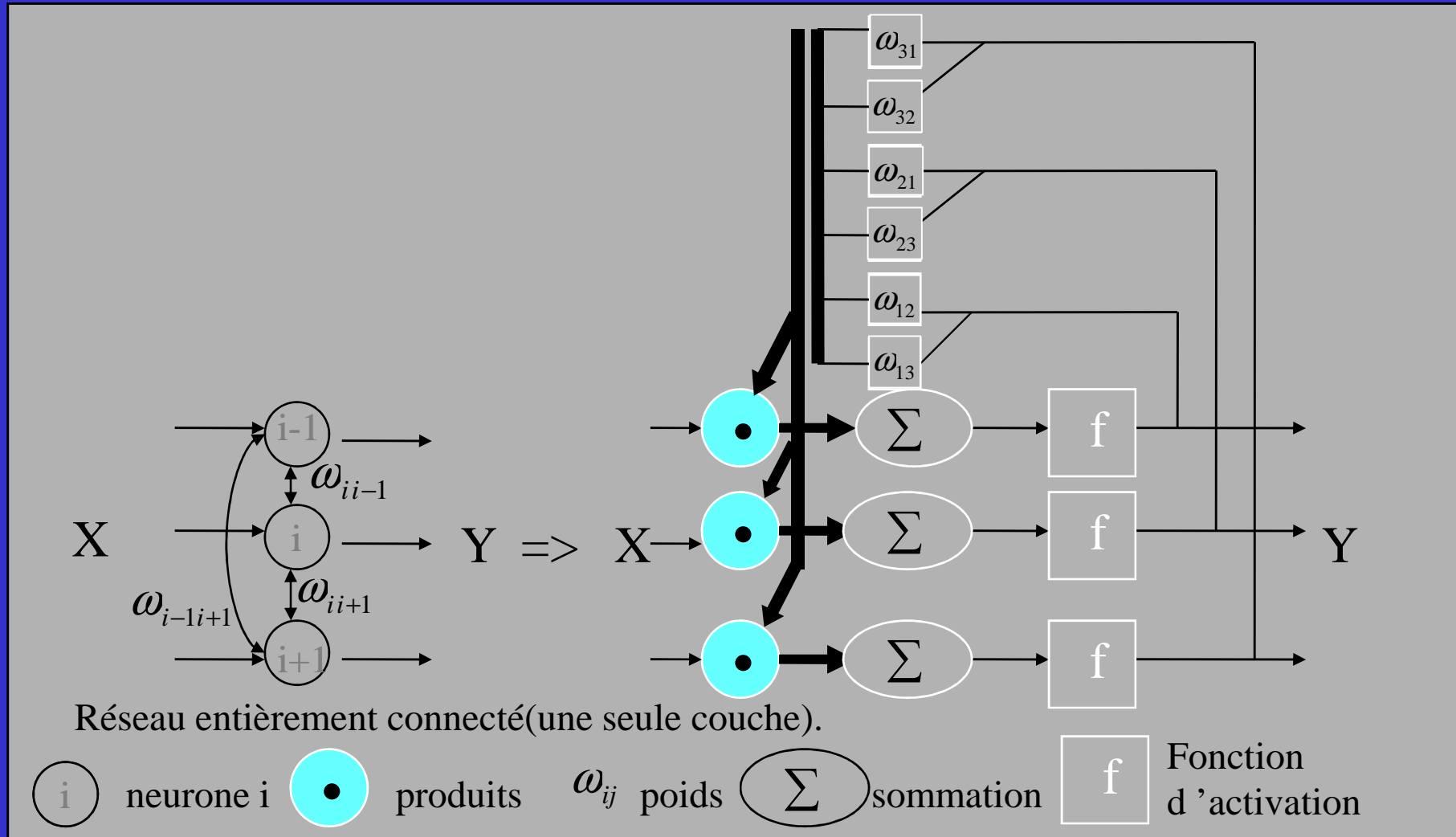
5. Réseaux récurrents mono-couche à connexité totale (Réseaux associatifs)

Type d'apprentissage: *non supervisé*

Dans ces modèles chaque neurone est connecté à tous les autres et théoriquement (mais pas en pratique) possède un retour sur lui-même. Ces modèles ne sont pas motivés par une analogie biologique mais par leurs analogie de comportement avec le verre de spin (mécanique statistique), [Hopfield 79].

Réseaux récurrents (suite)

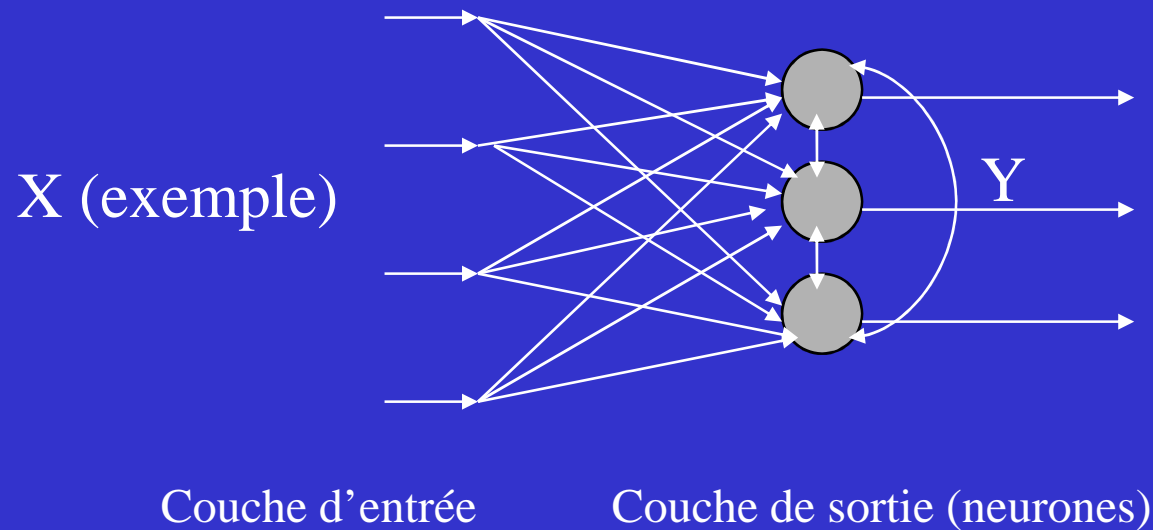
5. Réseaux récurrents mono-couche à connexité totale (Réseaux associatifs)



Réseaux récurrents (suite)

6. Réseaux compétitifs (Self-Organizing and Learning Vector Quantization Nets)

Ces réseaux sont similaires aux réseaux statiques mono-couche sauf qu'il existe des connections, habituellement de signes négatifs, entre les unités de sortie.



Réseaux récurrents- Réseaux compétitifs (suite)

- Un ensemble d'exemples est présenté aux réseaux, un exemple à la fois.
- A chaque exemple présenté, les poids synaptiques sont modifiés.
- Si une version dégradée d'un de ces exemple est présentée aux réseaux, les réseaux permettront alors de reconstruire l'exemple dégradé.

Réseaux récurrents- Réseaux compétitifs (suite)

Grâce à ces connections les unités de sorties tendent à subir une compétition pour représenter l'exemple courant présenté à l'entrée des réseaux.

7. Réseaux dynamiques

Bientôt