

# Neurosciences Computationnelles :

## 5. L'apprentissage et mémoire



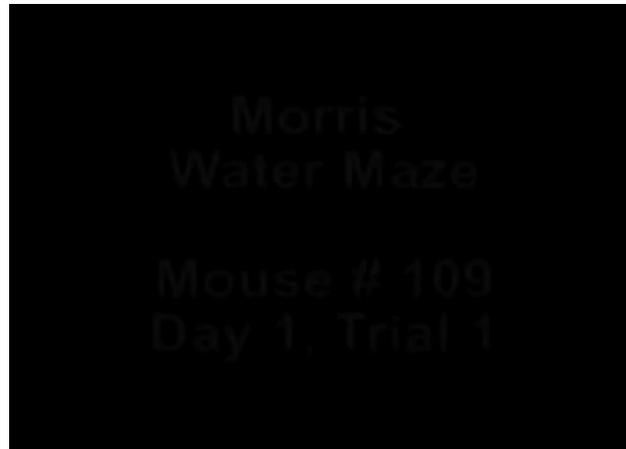
Michael Graupner  
([michael.graupner@parisdescartes.fr](mailto:michael.graupner@parisdescartes.fr))

# Question

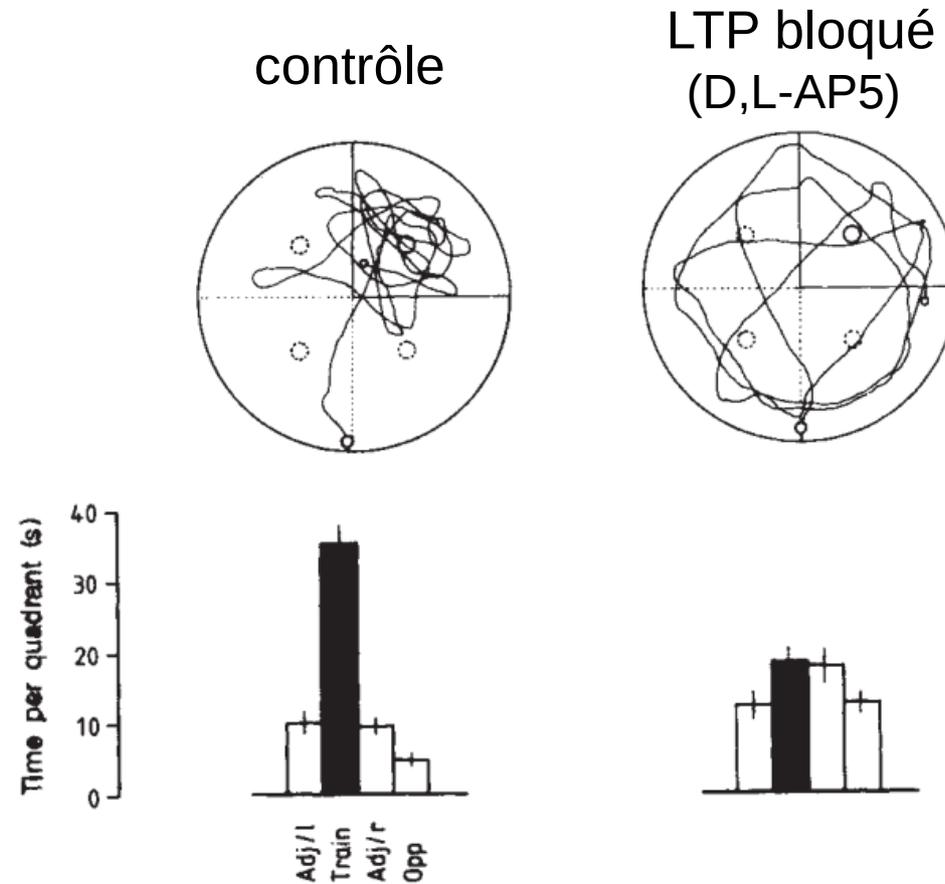
**Sur quelle structure cérébrale suis-je en train de faire des recherches?**

# Pourquoi nous intéressons-nous à la plasticité synaptique?

labyrinthe aquatique de Morris



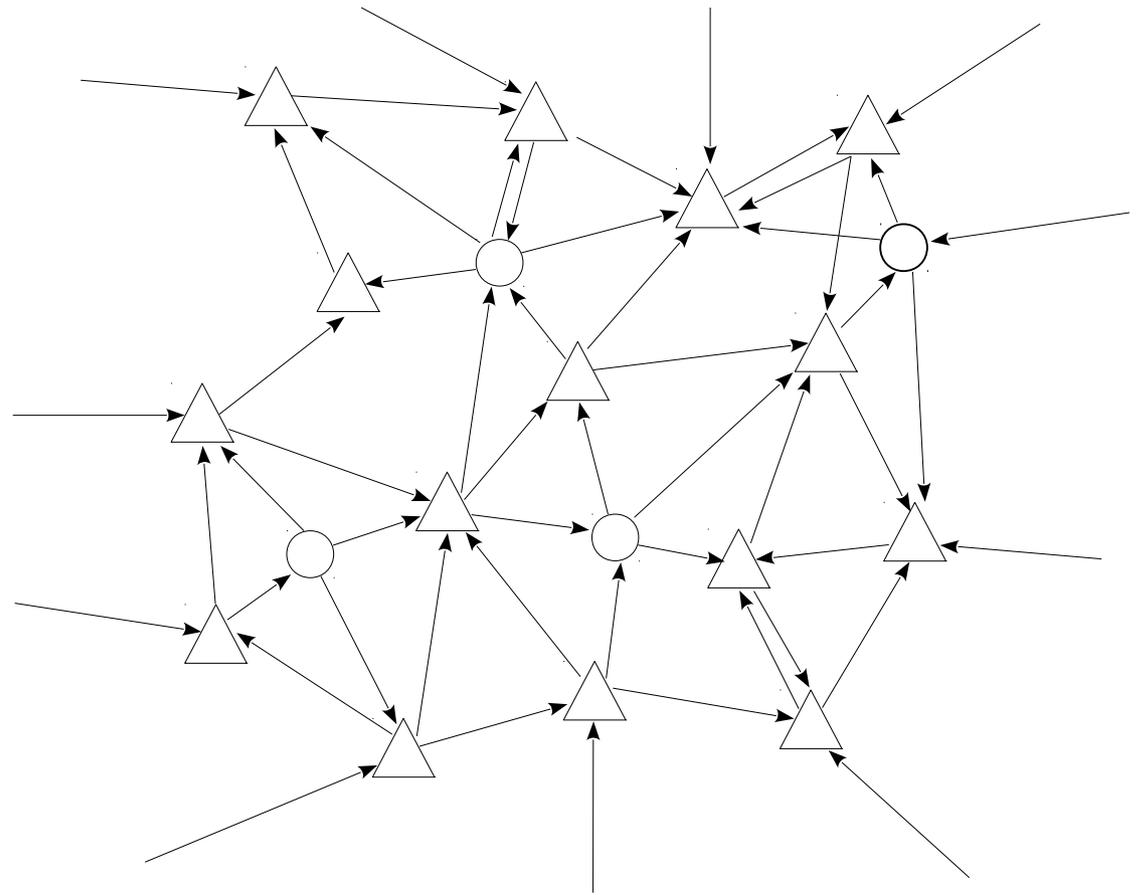
# Relation entre le LTP et l'apprentissage/mémoire



- LTP (récepteur NMDA) nécessaire pour apprendre à localiser la plateforme

[Morris *et al.*, 1986]

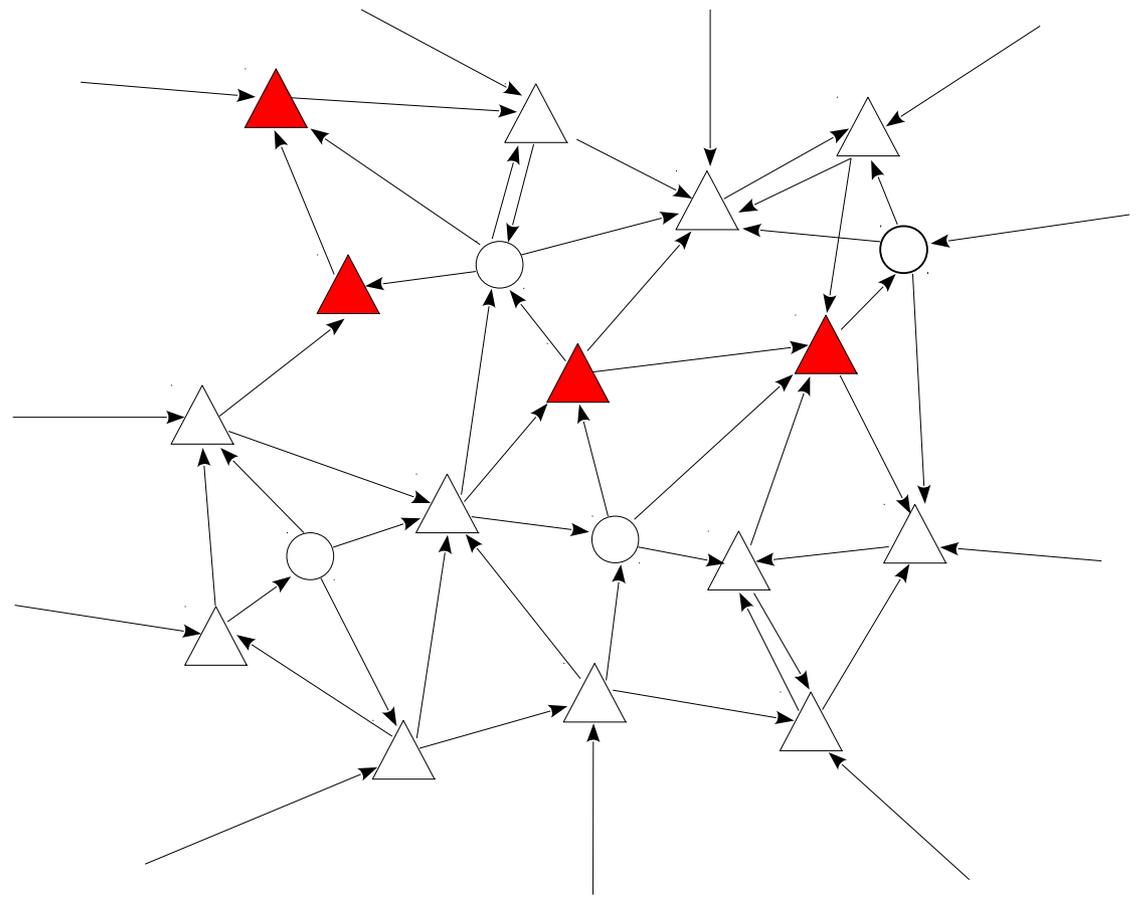
# Mécanismes d'apprentissage et de mémoire



# Mécanismes d'apprentissage et de mémoire

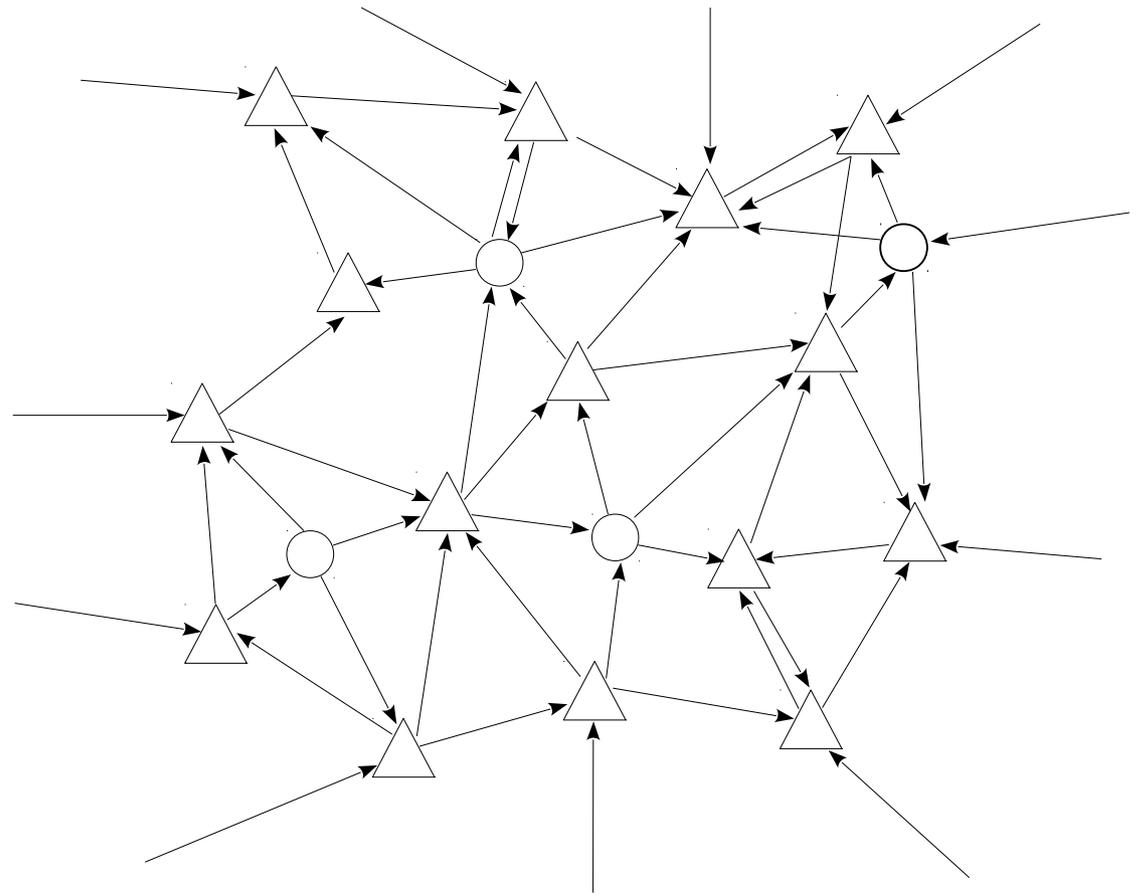
- Les stimuli externes déclenchent des changements d'activité neuronale.

Stimulus A



# Mécanismes d'apprentissage et de mémoire

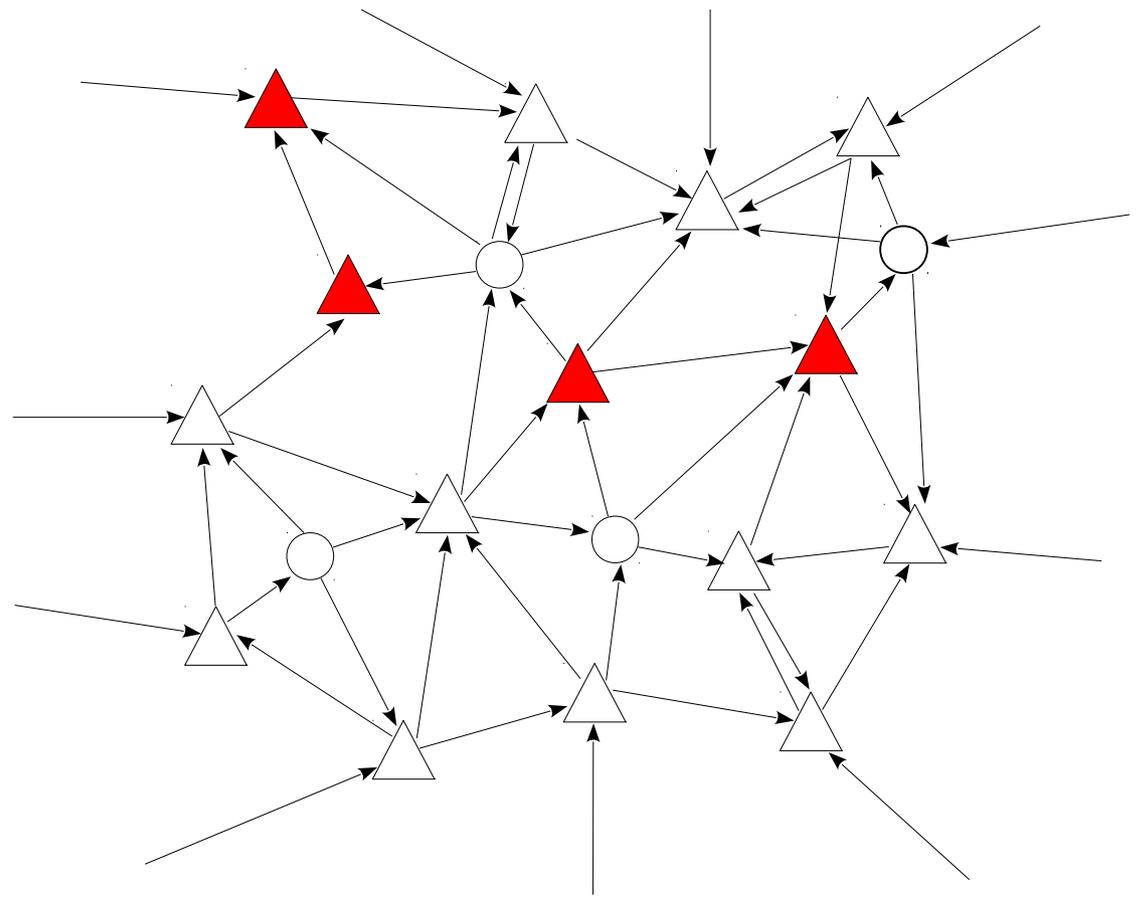
- Si l'activité provoquée par le stimulus ne laisse aucune trace dans le circuit  
→ aucun souvenir du stimulus A



# Mécanismes d'apprentissage et de mémoire

- Les stimuli externes déclenchent des changements d'activité neuronale.

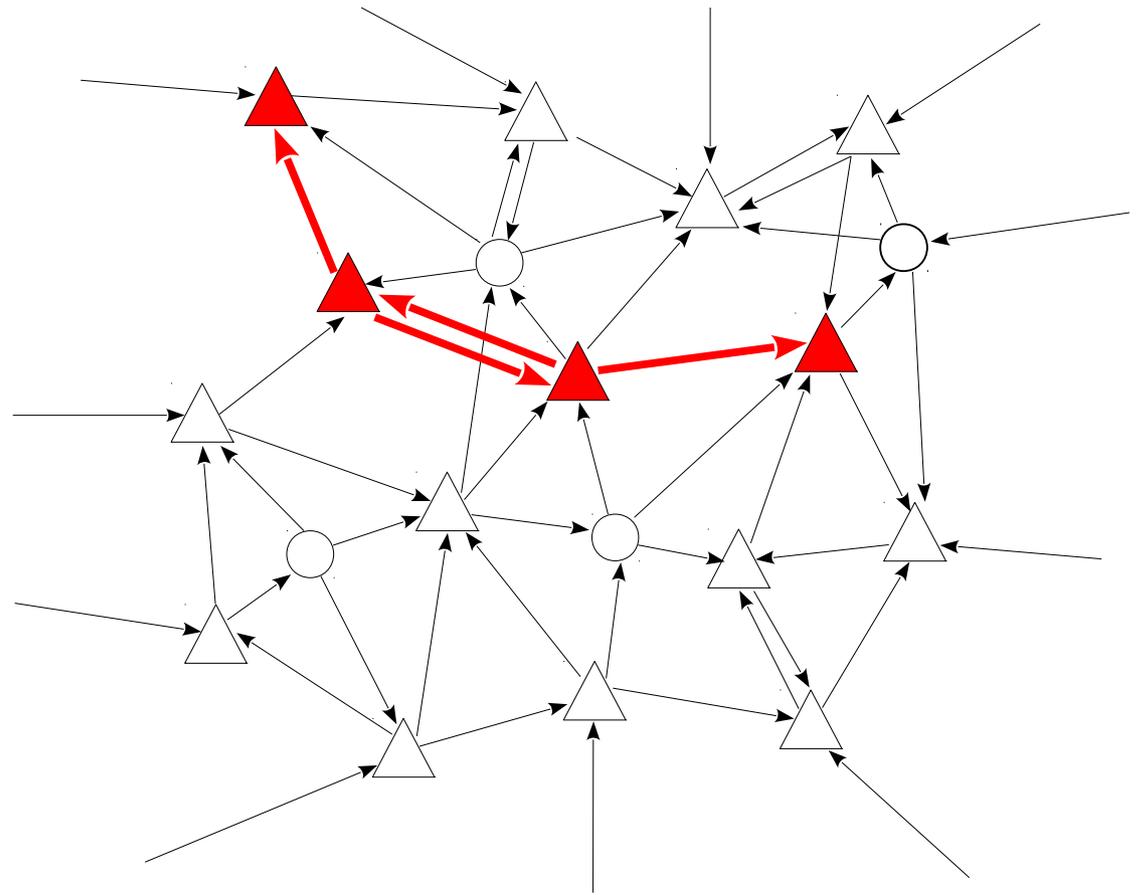
Stimulus A



# Mécanismes d'apprentissage et de mémoire

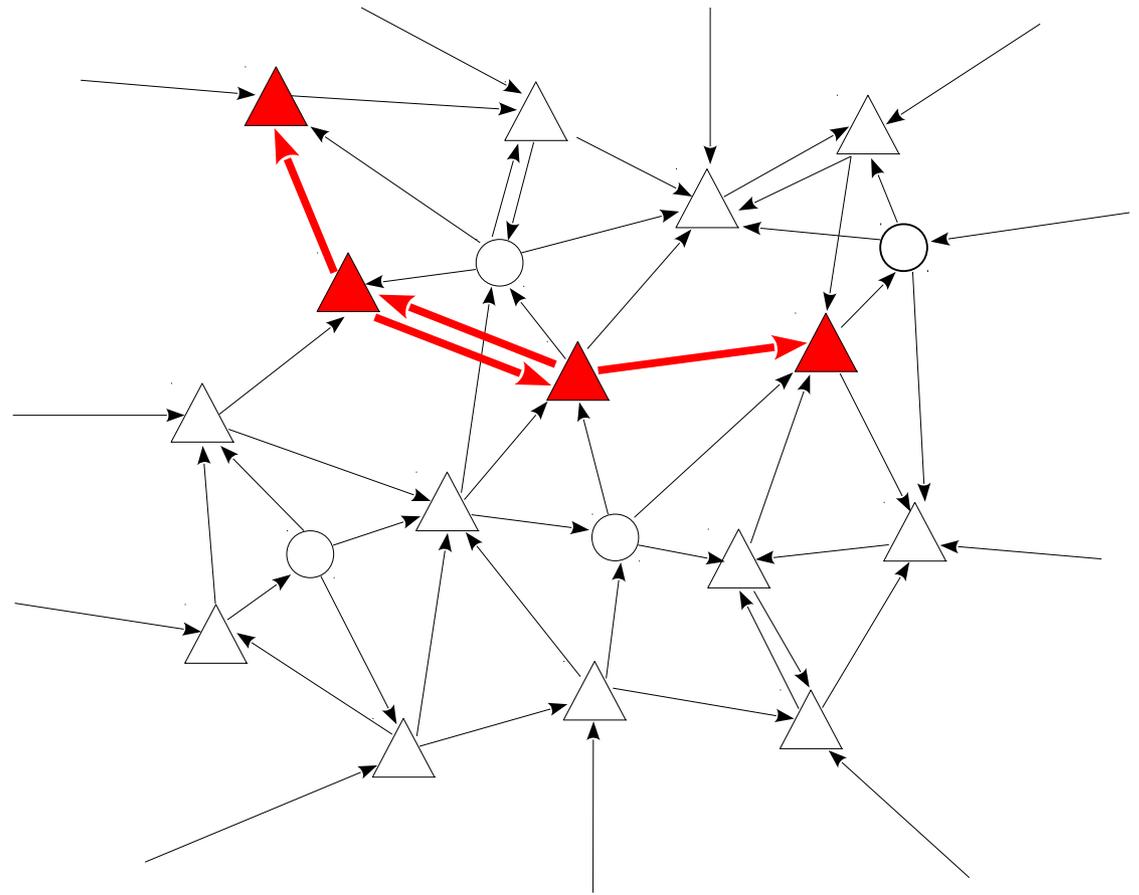
- Changements d'activité → changements dans la connectivité synaptique (plasticité synaptique/structurelle)

Stimulus A



# Mécanismes d'apprentissage et de mémoire

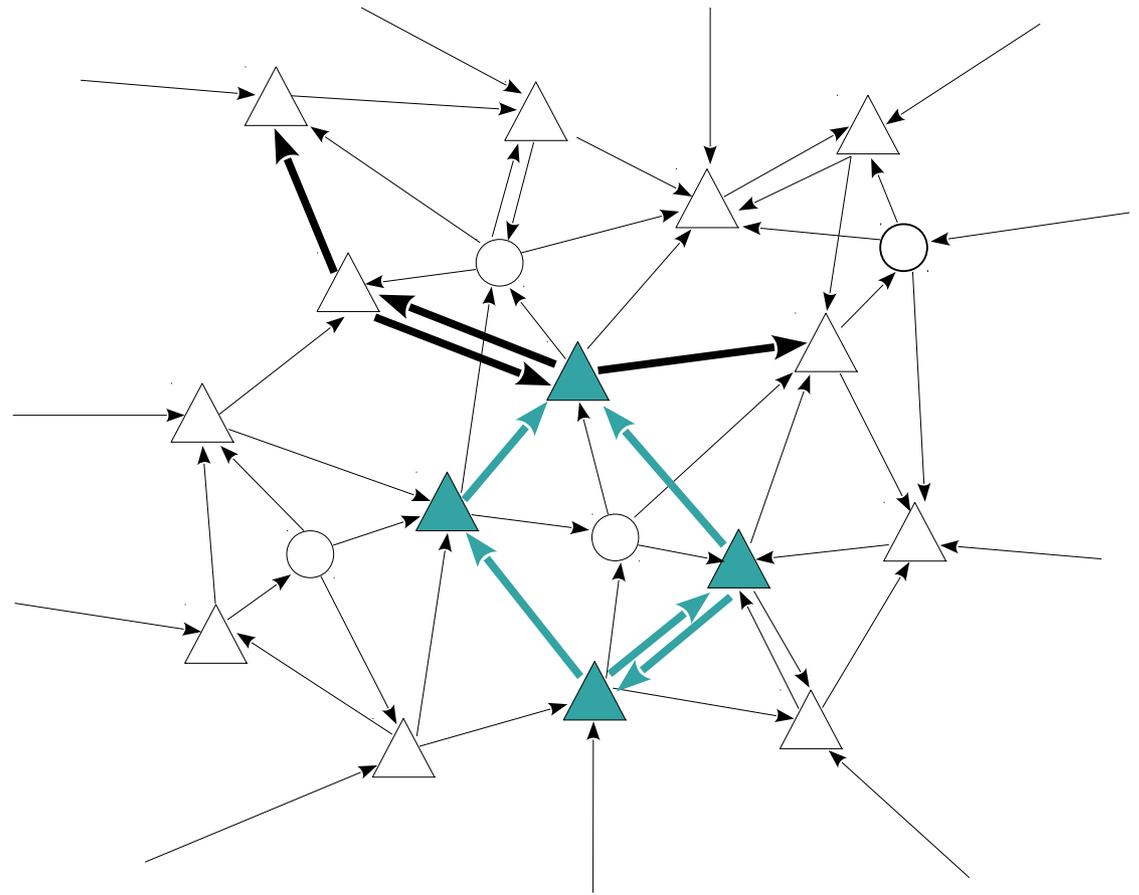
- Changements dans la connectivité synaptique → changements dans l'activité/la dynamique neuronale



# Mécanismes d'apprentissage et de mémoire

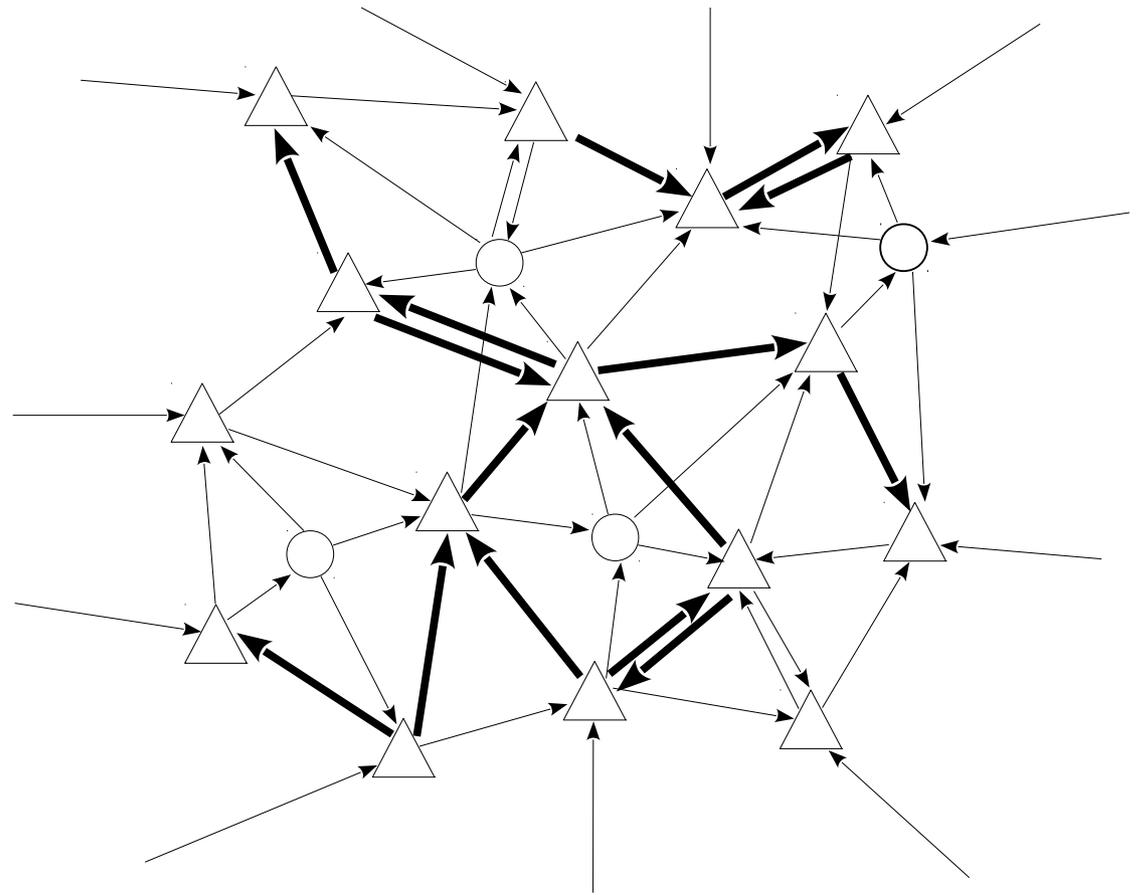
- Un autre stimulus déclenchant une activité dans un sous-ensemble distinct de neurones ...

Stimulus B



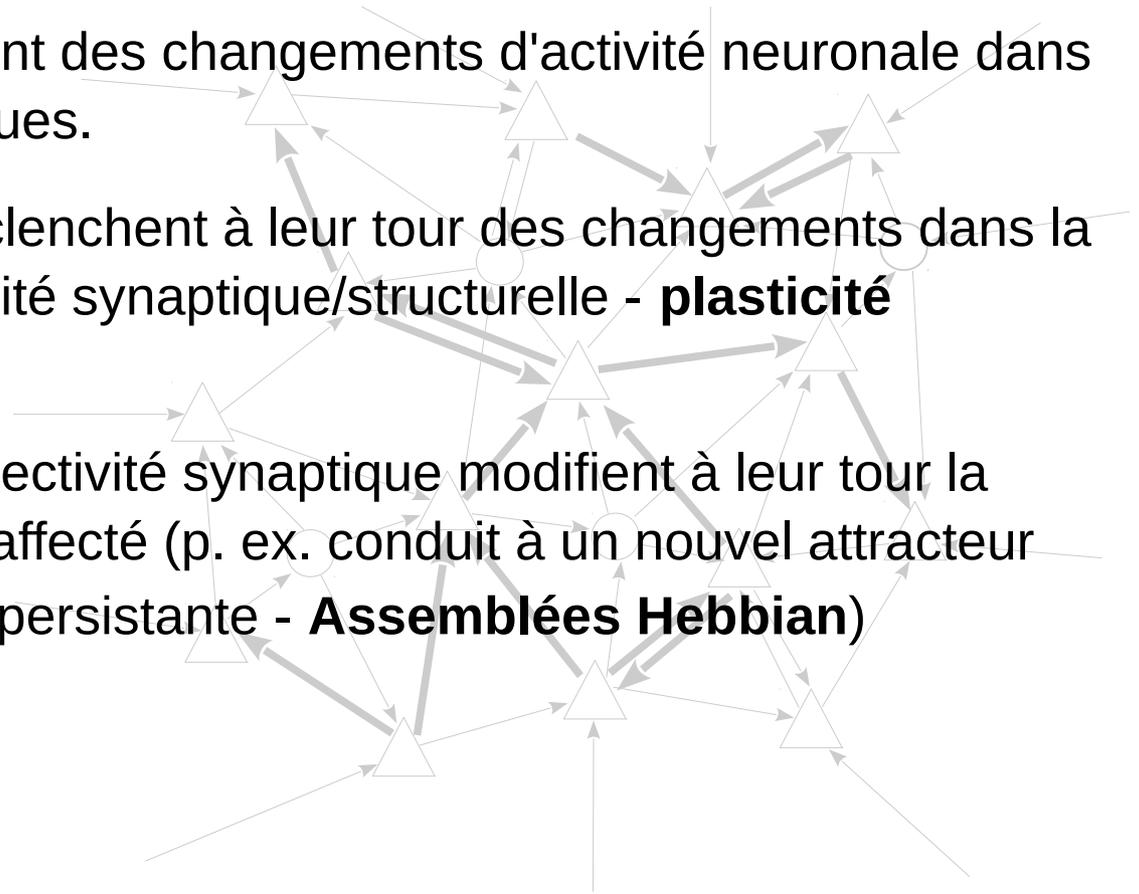
# Mécanismes d'apprentissage et de mémoire

- Connectivité synaptique = superposition des traces laissées par les inputs externes

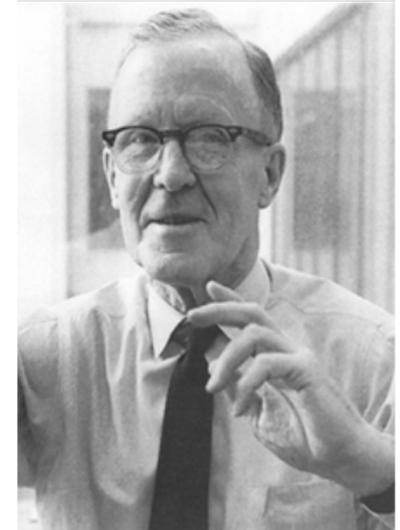
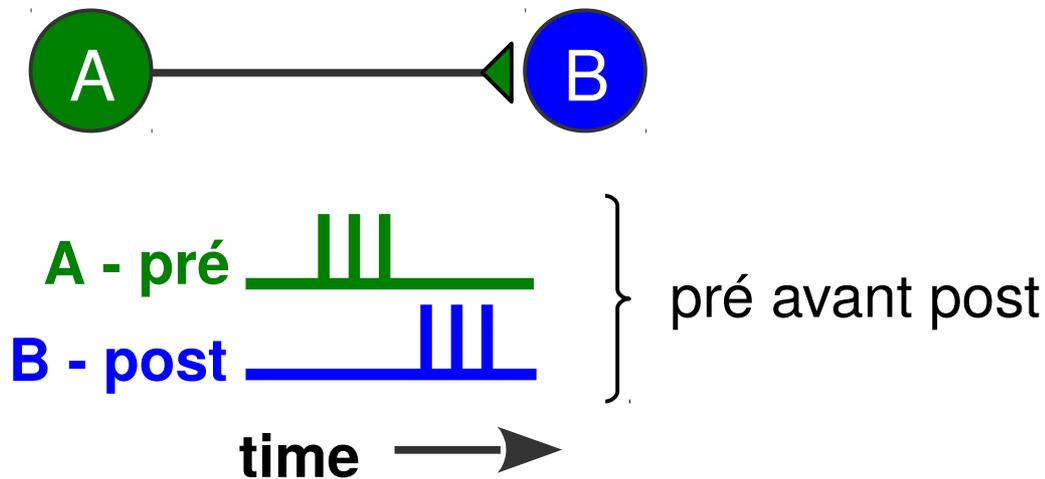


# Apprentissage et mémoire - le scénario "Hebbian"

- Des stimuli externes déclenchent des changements d'activité neuronale dans des circuits neuronaux spécifiques.
- Les changements d'activité déclenchent à leur tour des changements dans la connectivité synaptique (plasticité synaptique/structurelle - **plasticité Hebbian**)
- Les changements dans la connectivité synaptique modifient à leur tour la dynamique du circuit neuronal affecté (p. ex. conduit à un nouvel attracteur → activité de période de retard persistante - **Assemblées Hebbian**)



# Apprentissage et mémoire - le scénario "Hebbian"



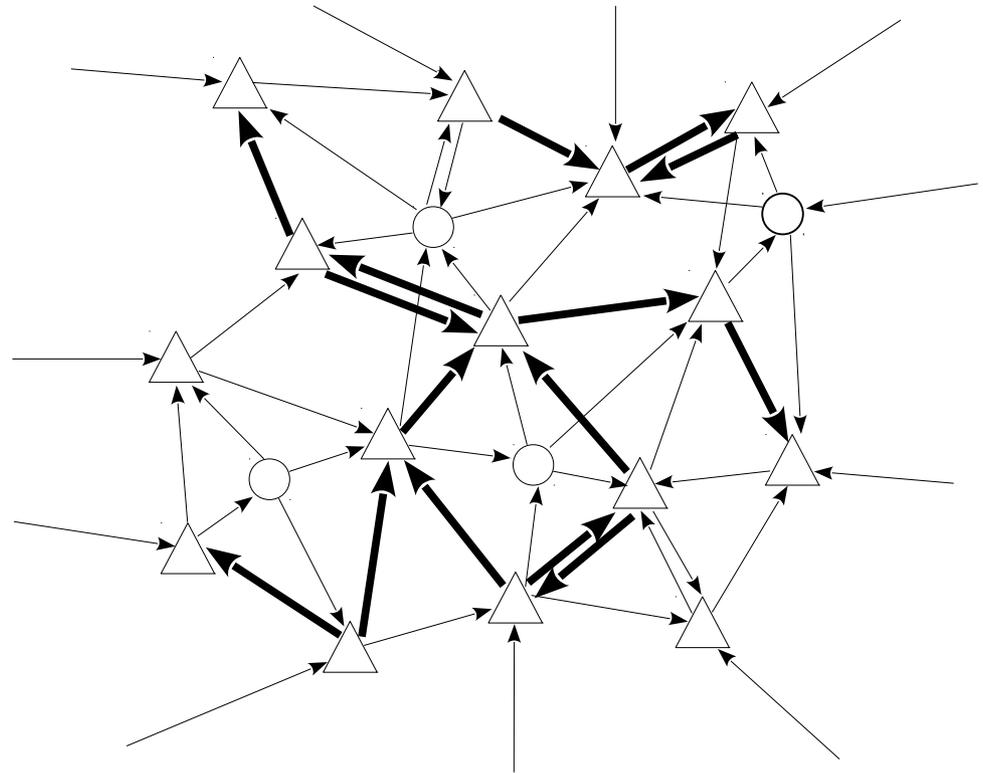
Donal Hebb, 1904 - 1985

“Lorsqu'un axone de la cellule A est assez proche pour exciter une cellule B et qu'il participe de façon *répétée* et *persistante* à sa décharge, une croissance ou des changements métaboliques se produisent dans l'une ou les deux cellules, de sorte que l'efficacité de la cellule A, comme l'une des cellules qui déclanche B, est augmentée.”

[Hebb 1949;  
see also Konorski 1948]

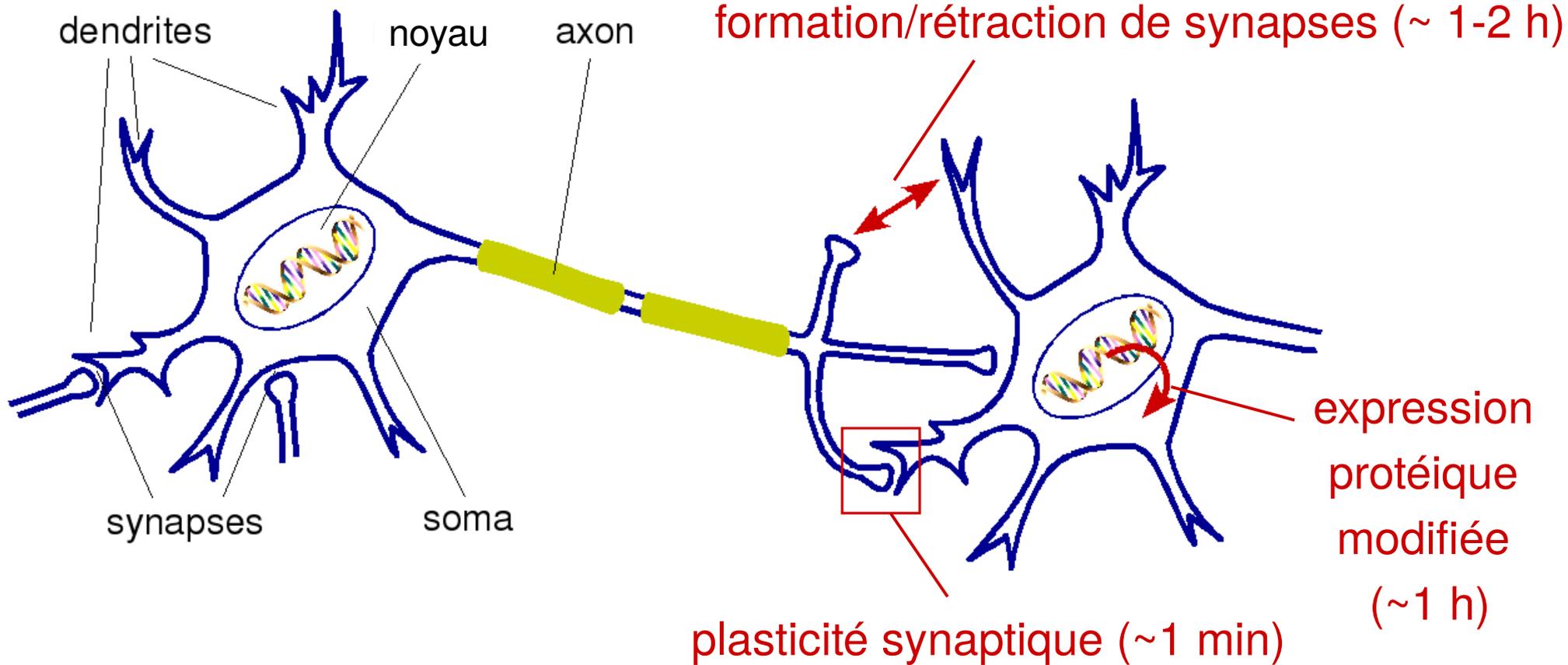
# Questions ouvertes

- Quelles sont les règles de plasticité (apprentissage) des circuits corticaux?
- Comment ces règles de plasticité déterminent-elles la matrice de connectivité de ces circuits?
- Comment ces changements façonnent à leur tour la dynamique des circuits?

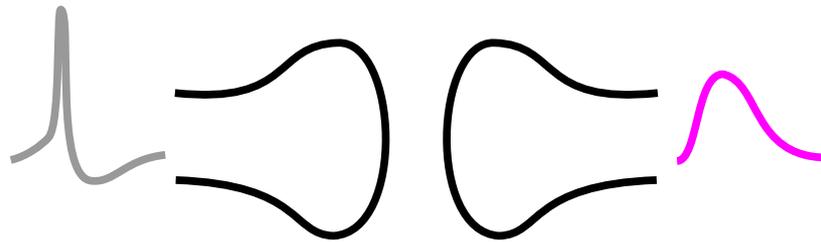


# Différentes formes de plasticité

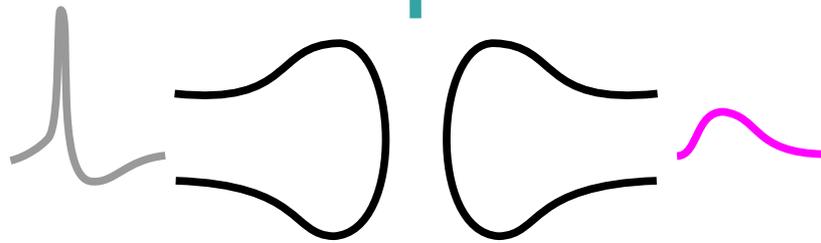
## changements liés à l'activité neurale



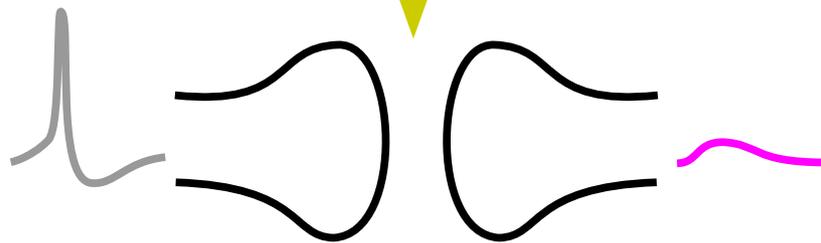
# Plasticité synaptique à long terme



potentialisation à long terme (LTP)

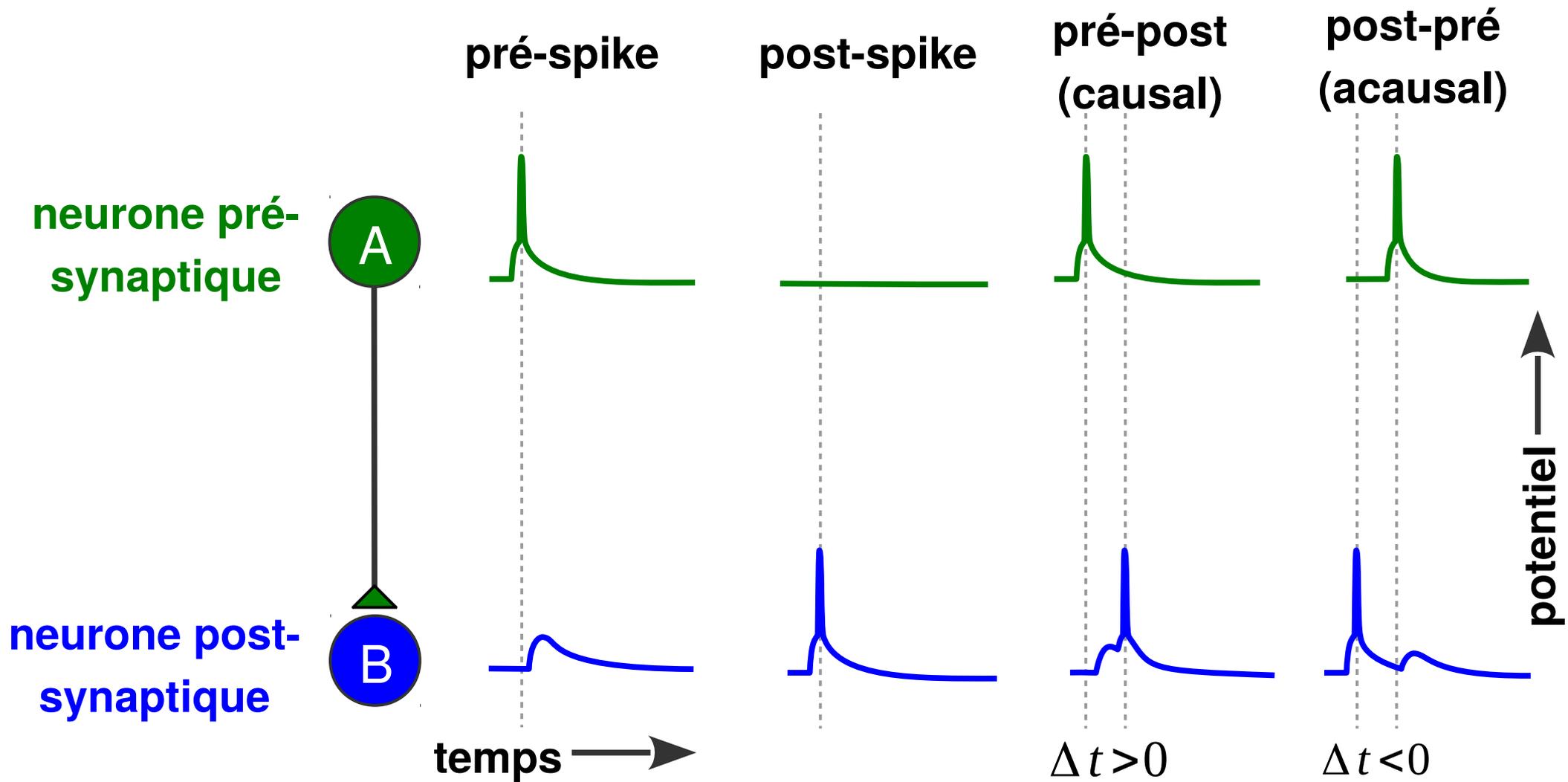


dépression à long terme (LTD)

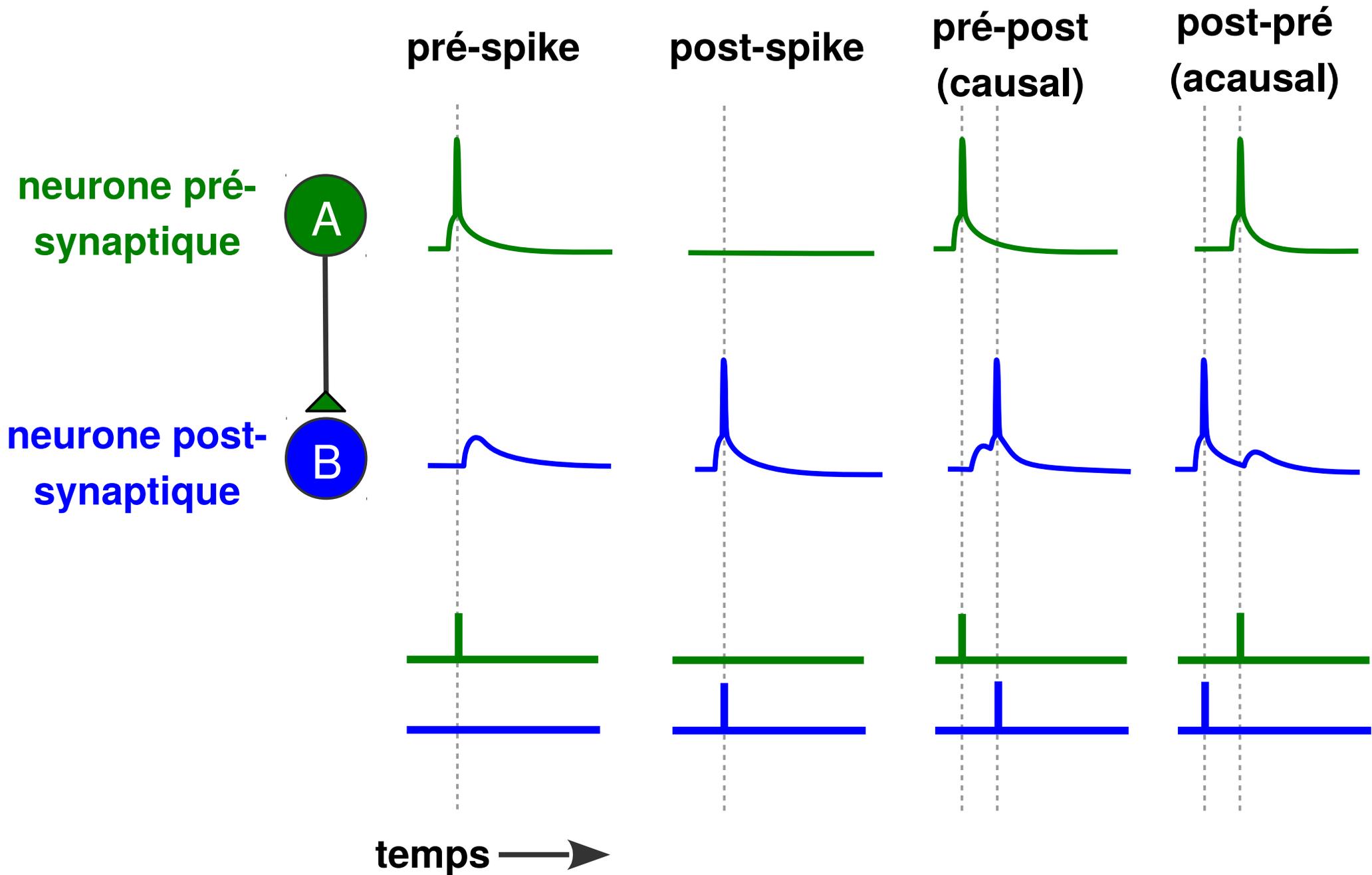


- changement durable (>60 min) en efficacité de transmission
- échelle de temps d'induction ~ 1 min

# Timing du spike : nomenclature

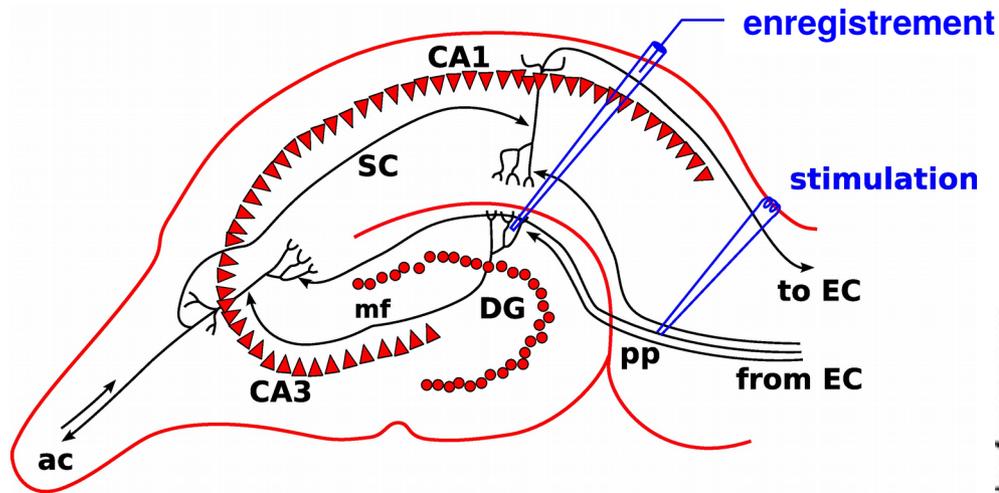


# Timing du spike : nomenclature

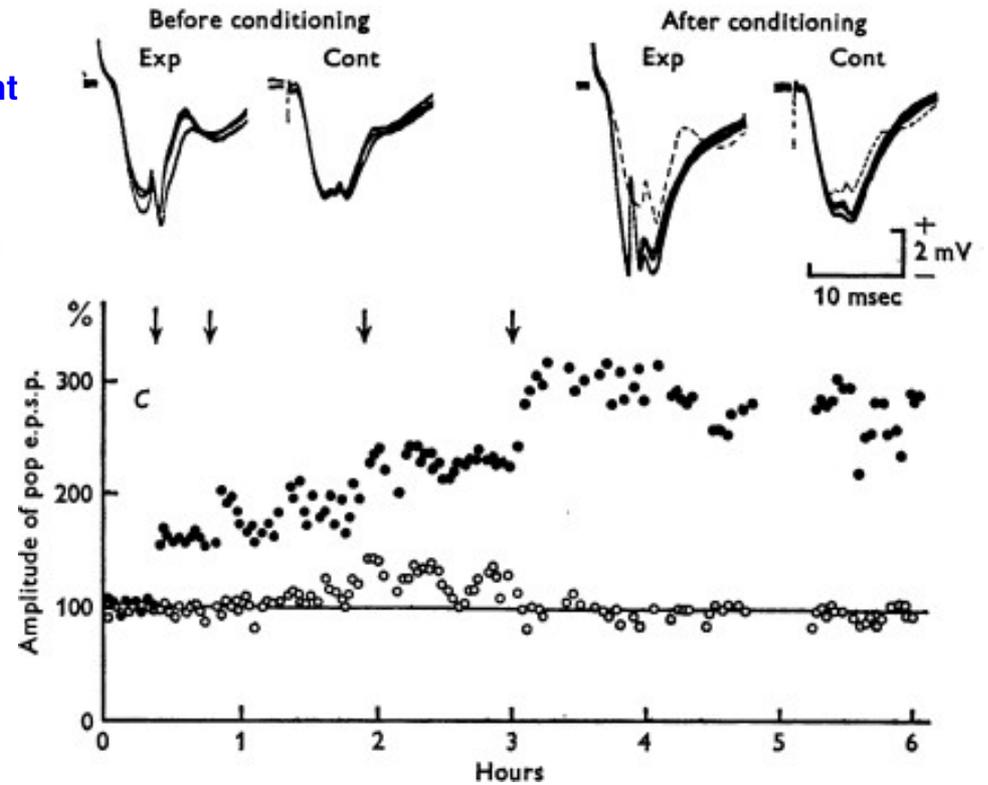


# Induction: LTP par stimulation à haute fréquence

hippocampe (*in vivo*)



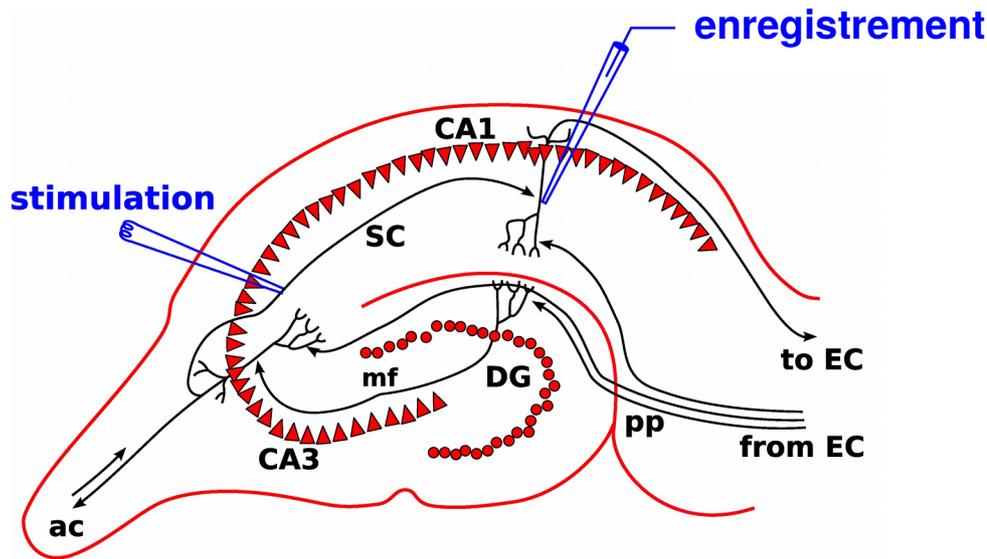
10-20 Hz pendant 10-15 sec  
ou 100 Hz pendant 3-4 sec



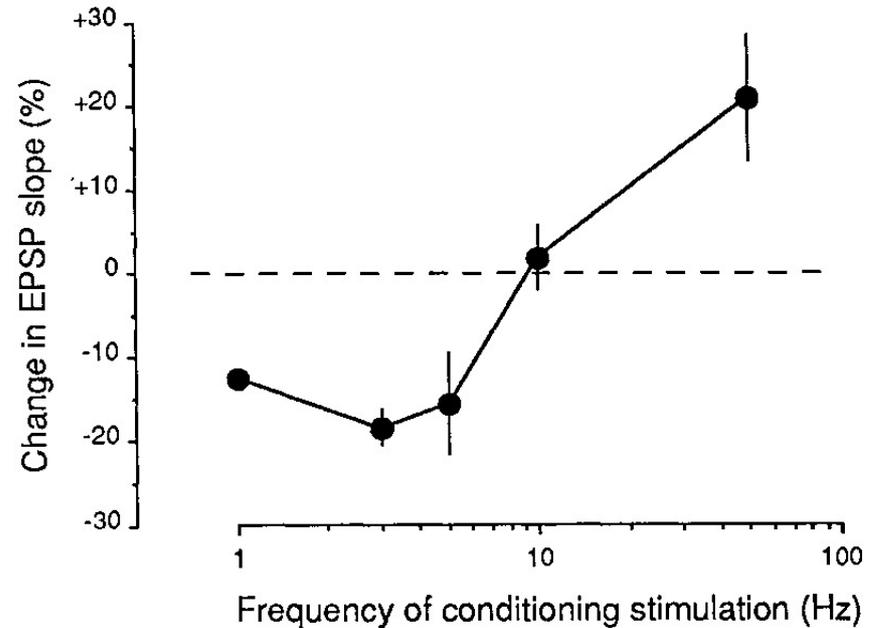
[Bliss and Lømo 1973]

# Induction de plasticité: LTD obtenue à basses fréquences

hippocampe (tranches)



900 impulsions à 1-50 Hz

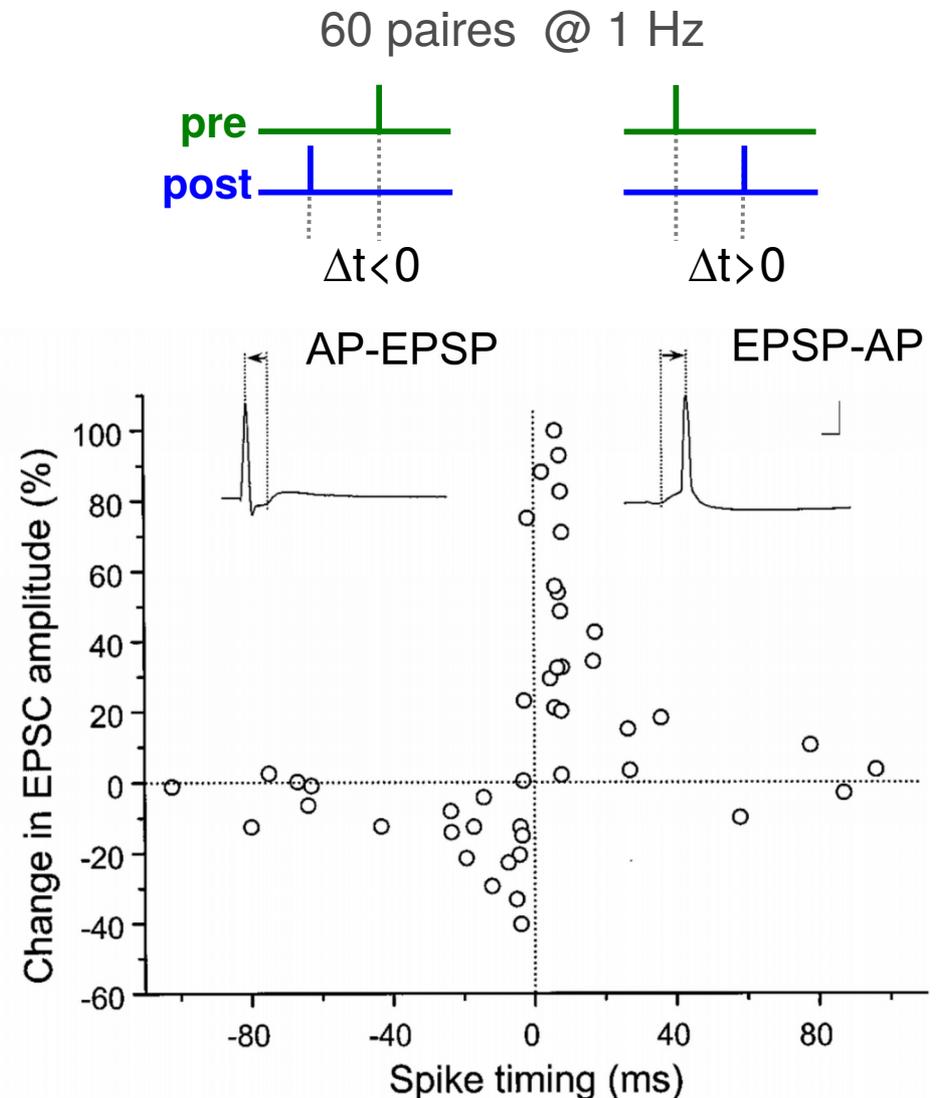
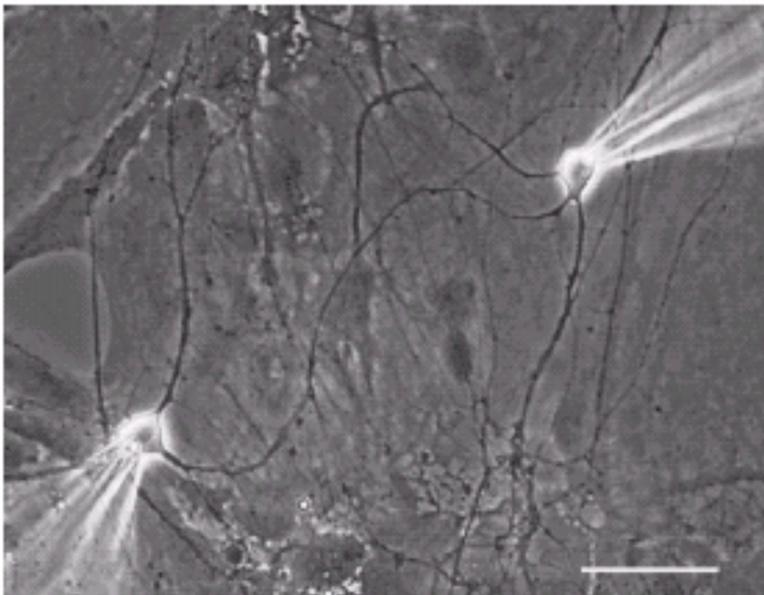


[Dudek and Bear 1992;  
Dunwiddie and Lynch 1978]

# STDP: plasticité à partir de paires de spikes simples

**STDP** : plasticité dépendante du moment du spike

cultures d'hippocampes

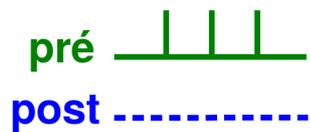


[Bi & Poo, J Neurosci 1998]

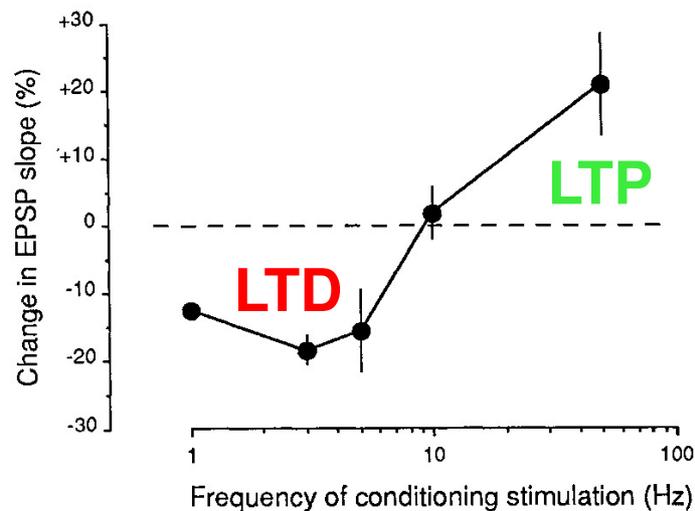
[Magee & Johnston 1997; Zhang et al. 1998; Markram et al. 1997; Sjöström et al. 2001; Feldman 200]

# Plasticité dépend de la fréquence, du moment des spikes et combinaison

dépendante  
de la fréquence



900 impulsions à 1-100 Hz

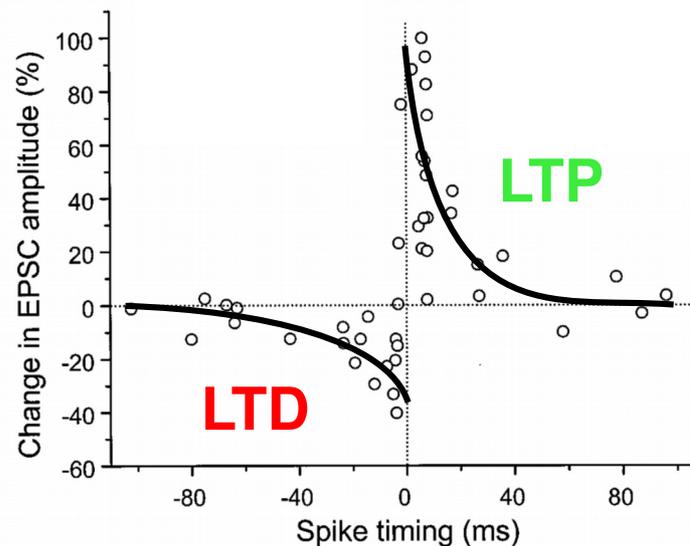


[Dudek and Bear 1992;  
Dunwiddie and Lynch 1978]

dépendante du  
moment des spikes

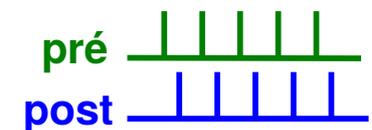


60 paires à 1 Hz

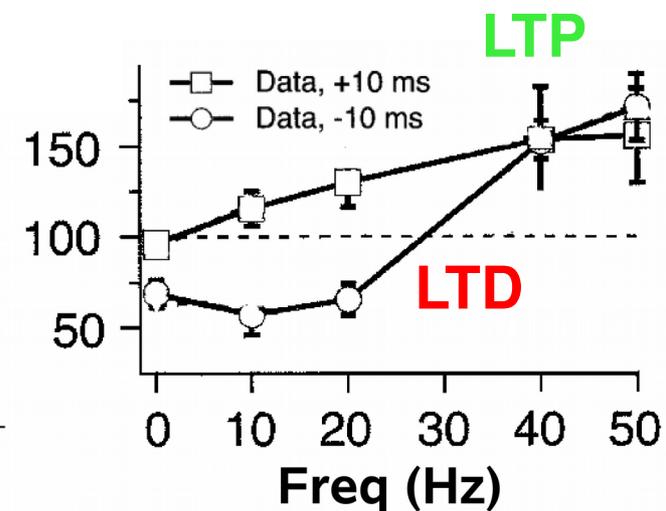


[Bi & Poo *J Neurosci* 1998]

et des combinaisons  
des deux

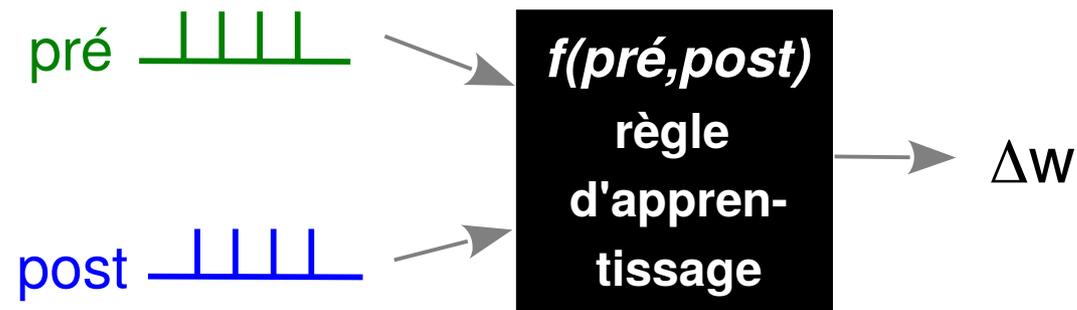
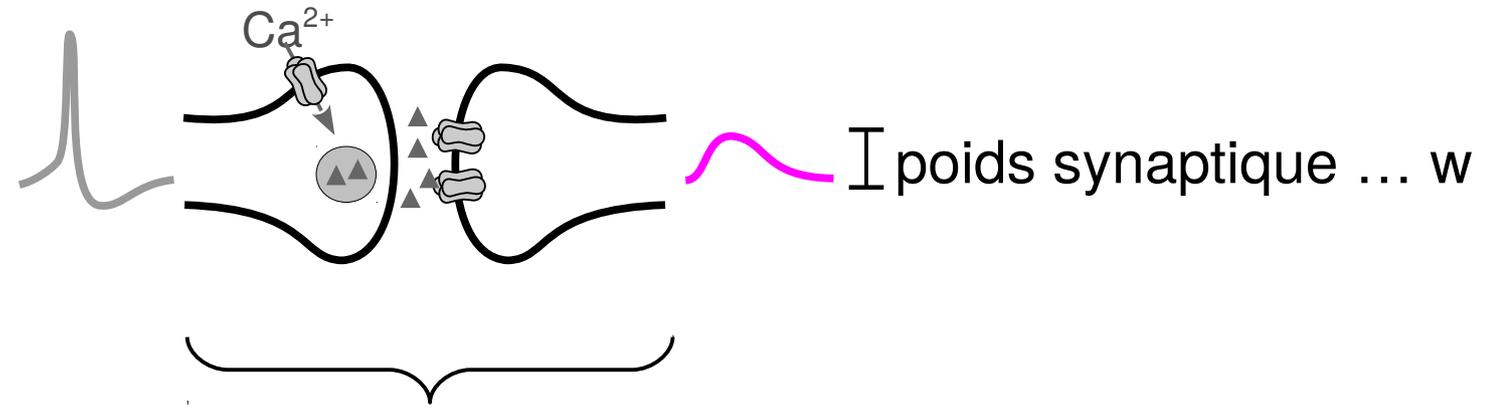


5 paires, 15x à 0.1 Hz



[Sjöström et al. *Neuron* 2001]

# Modélisation: traduction des spikes vers les résultats de plasticité



# Règles d'apprentissage phénoménologique : taux de décharge

- Règles basées sur le taux de décharge :  $\Delta w_{ij} = f(r_i r_j)$

- Pur Hebbian :

$$f(r_i r_j) = r_i r_j$$

- Covariance (Sejnowski 1977) :

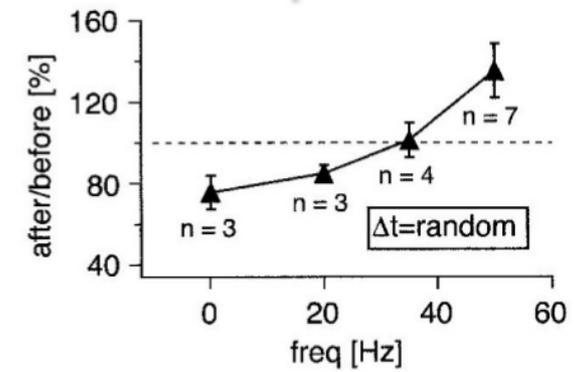
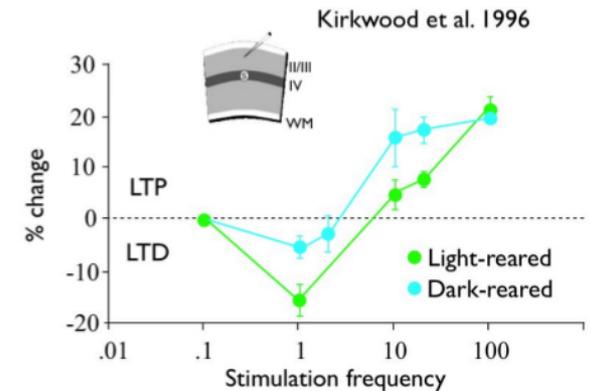
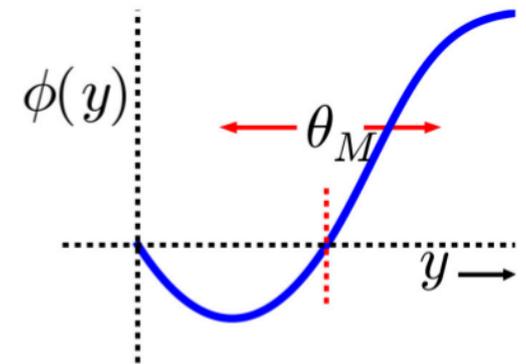
$$f(r_i r_j) = (r_i - \langle r_i \rangle)(r_j - \langle r_j \rangle)$$

- Bienenstock-Cooper-Munro (1982) :

$$f(r_i r_j) = r_i (r_i - \theta) r_j$$

$$\tau_\theta \frac{d\theta}{dt} = r_i^2 - \theta$$

→ seuil adaptatif (méta-plasticité) qui stabilise  
la règle d'apprentissage



# Règles d'apprentissage phénoménologique - STDP

- Règles basées sur le moment du spike :  $\Delta w_{ij} = f(\{t_{ik}\}, \{t_{jk}\})$

- STDP standard :

$$f(\{t_{ik}\}, \{t_{jk}\}) = \sum_{k, k'} F(t_{ik} - t_{jk'})$$

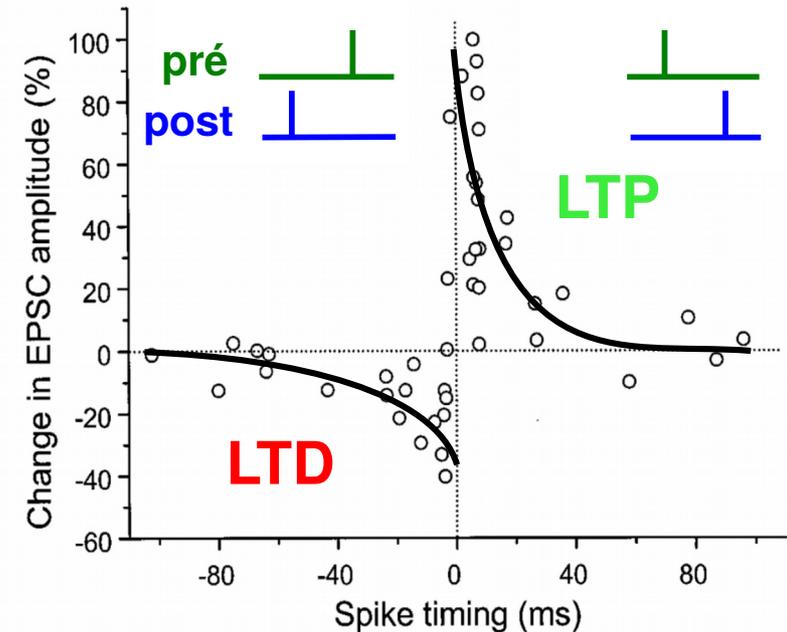
$$F(\Delta t) = \begin{cases} A_+ \exp(-\Delta t / \tau_+) & \Delta t > 0 \\ A_- \exp(-\Delta t / \tau_-) & \Delta t < 0 \end{cases}$$

- Variantes de la règle:

\* Additif/multiplicatif

\* Toutes les paires de spikes / voisins les plus proches

- **Problèmes** : pas dépendant du taux de décharge  
ne résout pas les non-linéarités de plasticité



# Modélisation récente de la plasticité

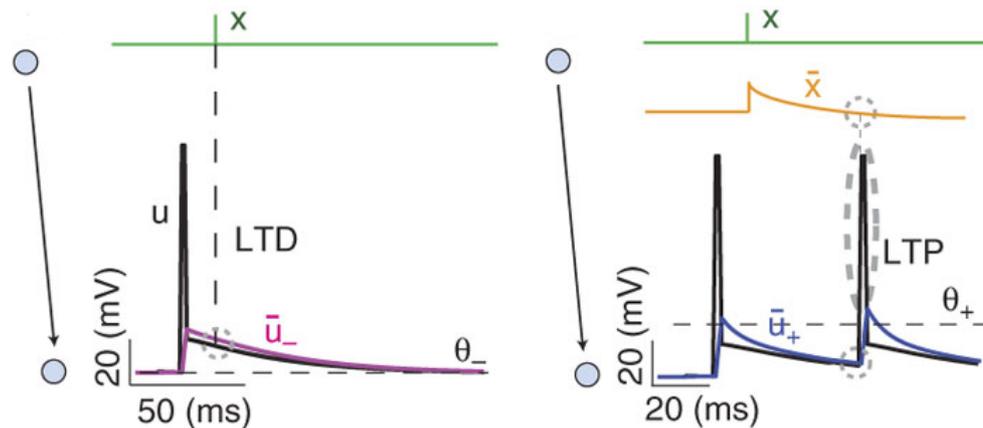
## Modèle basé sur les triplets

[Pfister & Gerstner, 2006;  
Clopath et al., 2010]



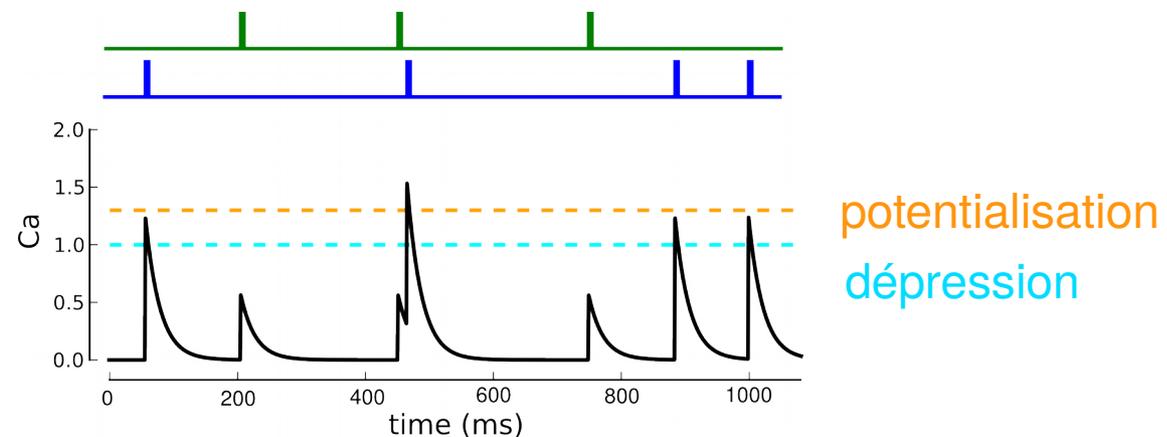
## Modèle basé sur le potentiel

[Clopath et al., 2010]



## Modèle basé sur le calcium

[Shouval et al. 2002, Graupner  
& Brunel 2012]



# Différents types d'apprentissage

- **Apprentissage non supervisé :**

Les stimuli externes déclenchent des changements d'activité neuronale dans des circuits neuronaux spécifiques; Codage - construction d'une représentation neuronale.

- Apprentissage Hebbian

- **Apprentissage supervisé :**

Ensemble de données qui comprend les valeurs cibles (les valeurs que nous voulons prédire). Apprendre une fonction qui prédit correctement les valeurs cibles, qui peut ensuite être utilisée pour faire des prédictions sur d'autres exemples.

- Exemple : classification, régression, ...

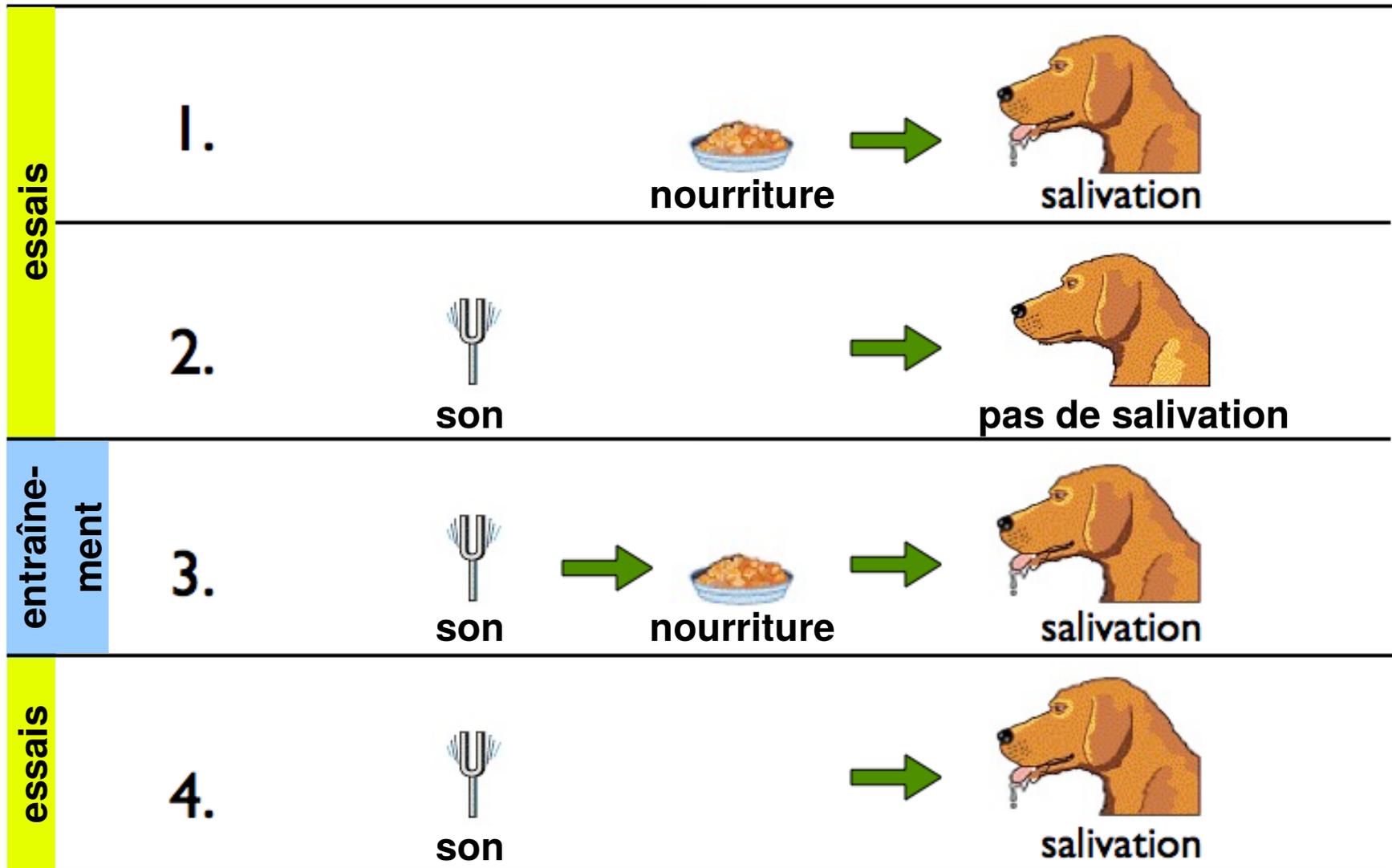
- Le perceptron

- **Apprentissage par renforcement :**

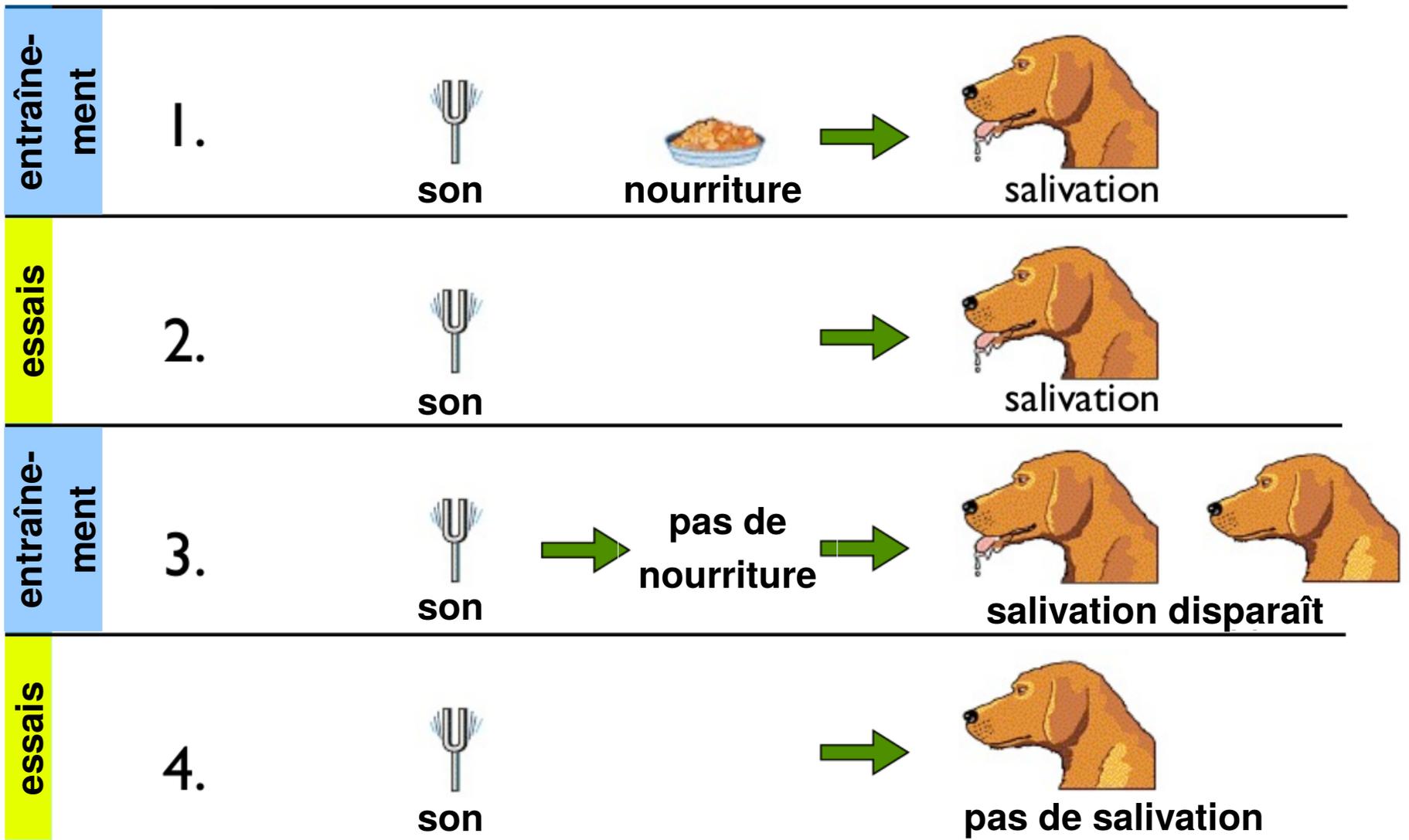
Apprendre à connaître les stimuli uniquement sur la base des récompenses et des punitions qui leur sont associées.

- Conditionnement classique

# Apprentissage par renforcement : conditionnement classique



# Apprentissage par renforcement : extinction



# Interprétation : conditionnement classique et extinction

## Pourquoi est-ce que ça arrive?

- comportement involontaire
- apprentissage des associations
- proximité temporelle des stimuli



Ivan Pavlov  
(1849-1936)  
travaux: 1890s



John B Watson  
(1878-1958)  
travaux: 1920s

## Pourquoi est-ce que ça arrive?

- apprendre à prévoir

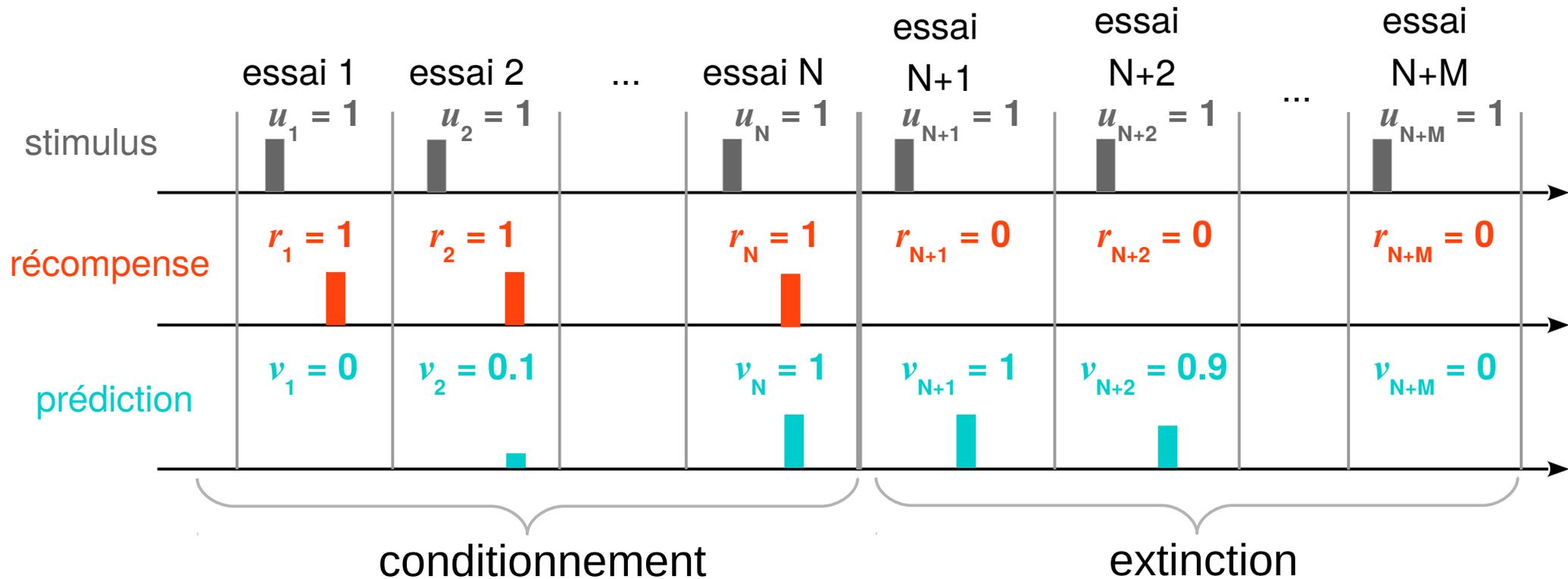


Robert Rescorla



Allan R Wagner

# Modèle Rescorla-Wagner

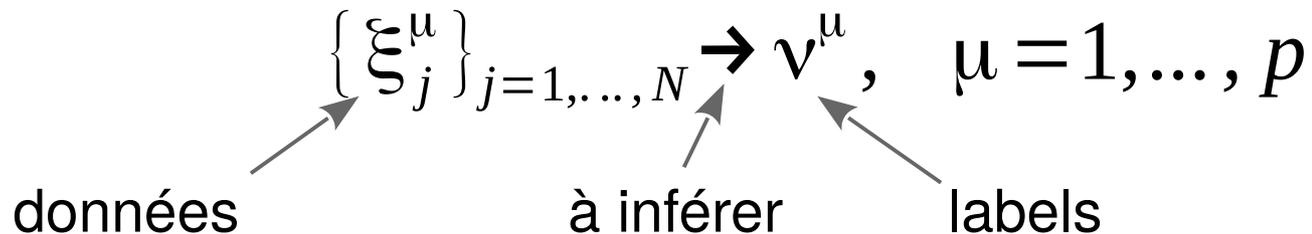


**Suppose** : Le chien veut pouvoir prédire la récompense.  
→ veut créer une association entre stimulus et récompense

**Modèle** : Peut expliquer le conditionnement et l'extinction.

# Apprentissage supervisé

- **But** : inférer une fonction  $F$  à partir d'un ensemble
- **Ensemble d'apprentissage** :



- **Intérêt**:  
On peut ensuite utiliser cette fonction sur des données ne faisant pas partie de l'ensemble d'apprentissage.
- **Condition** :  
Pour inférer la fonction  $F$  « correctement », il faut un ensemble d'apprentissage important.

# Apprentissage supervisé : exemple

**Ensemble d'apprentissage :**

$$\left\{ \begin{matrix} \xi^\mu \\ \zeta_j \end{matrix} \right\}_{j=1, \dots, N} \rightarrow v^\mu, \quad \mu = 1, \dots, p$$

données

labels



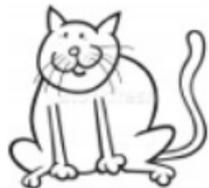
→ chat



→ chien



→ chien



→ chat

...

à inférer

**Test :**

une fois la fonction apprise



→ ?

# Apprentissage supervisé : Le Perceptron

- **Le perceptron (1958) :**

## The New York Times

### *NEW NAVY DEVICE LEARNS BY DOING*

*July 8, 1958*

“The Navy revealed the embryo of an electronic computer today that it expects will be able to walk, talk, see, write, reproduce itself and be conscious of its existence... Dr. Frank Rosenblatt, a research psychologist at the Cornell Aeronautical Laboratory, Buffalo, said Perceptrons might be fired to the planets as mechanical space explorers”



Rosenblatt 1958

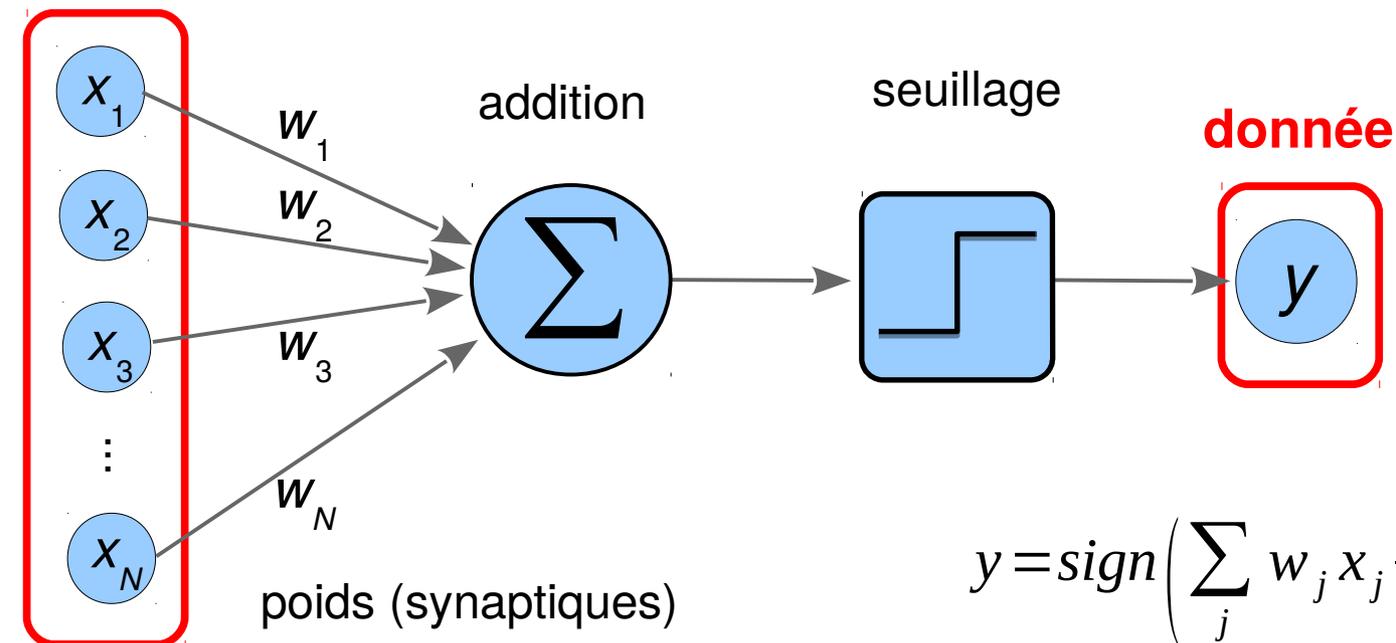
# Apprentissage supervisé : Le Perceptron

- **Le perceptron :**

Une machine qui permet de faire des classifications binaires

*Couche d'entrée*

*Couche de sortie*



ajuster/apprendre

$$y = \text{sign} \left( \sum_j w_j x_j - \theta \right)$$

$$\text{sign}(s) = \begin{cases} 1 & \text{si } s > 0 \\ -1 & \text{sinon} \end{cases}$$

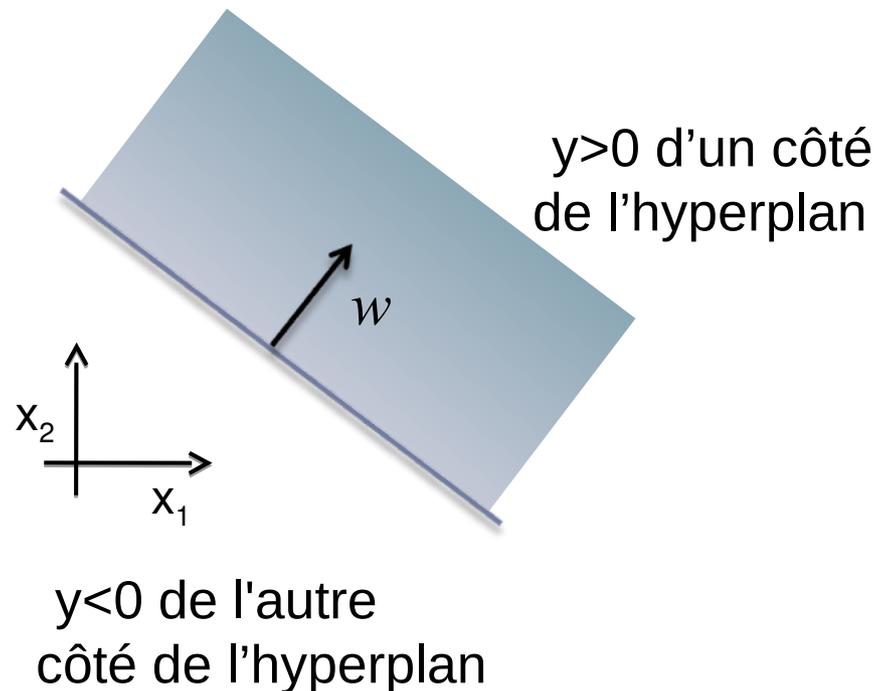
# Apprentissage supervisé : Le Perceptron

- **Le perceptron :**

- Une machine qui permet de faire des classifications binaires
- Les perceptrons à une seule couche sont seulement capables d'apprendre des motifs linéaires séparables.
- le réseau neuronal feedforward à deux ou plusieurs couches (également appelé un perceptron multicouche) a une puissance de traitement beaucoup plus grande que les perceptrons à une seule couche
- Outil pour étudier les problèmes de capacité de stockage.

# Le Perceptron : Interprétation géométrique

- Équation du perceptron :  $y = \text{sign}\left(\sum_j w_j x_j - \theta\right)$  souvent :  $y = \text{sign}\left(\sum_{j=0} w_j x_j\right)$   
avec  $x_0 = 1$
- Équation d'un hyperplan :  $\vec{w} \cdot \vec{x} + b = 0$

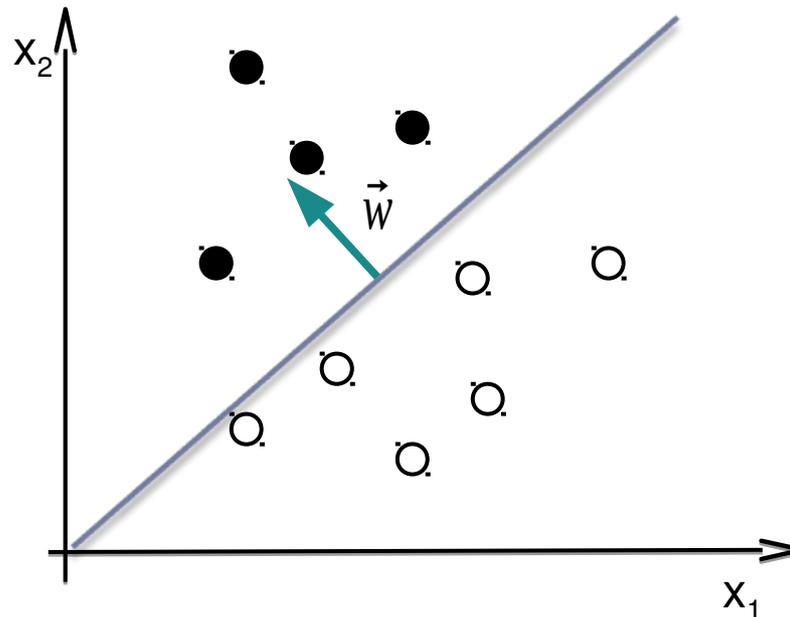


# Classification par un perceptron

- Quelles tâches de classification peut effectuer un perceptron?

## Tache :

Peut-on trouver des poids  $(w_1, w_2)$  tels que  $y=1$  pour les points noirs et  $y=0$  pour les points blancs?



**oui:**

$$w_1 = -1$$

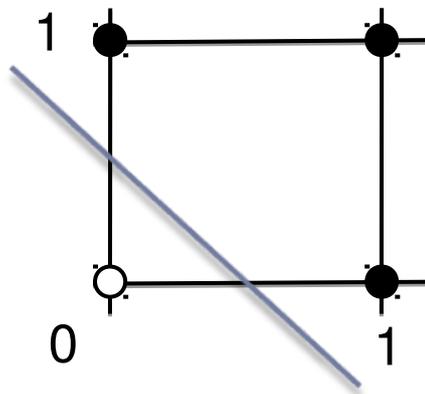
$$w_2 = 1$$

$$b = 0$$

# Exemples

- On regarde le cas binaire:  $x_i = 0, 1$

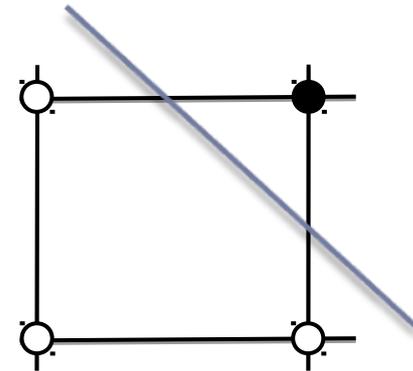
OU( $x_1, x_2$ )



○ = Faux  
● = Vrai

$$\begin{aligned}w_1 &= 1 \\w_2 &= 1 \\b &= -1/2\end{aligned}$$

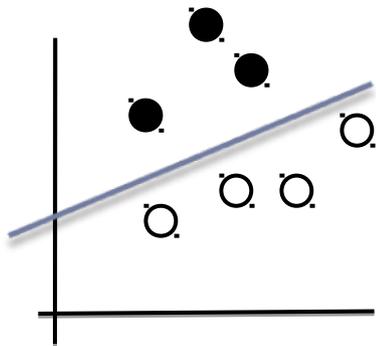
ET( $x_1, x_2$ )



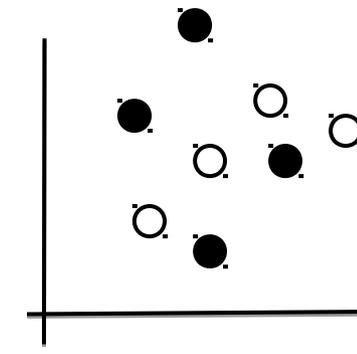
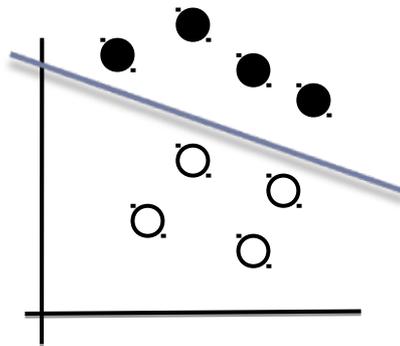
$$\begin{aligned}w_1 &= 1 \\w_2 &= 1 \\b &= -3/2\end{aligned}$$

# Séparabilité linéaire

- Deux ensembles de points linéairement séparables = sont de deux côtés d'un hyperplan



linéairement séparable

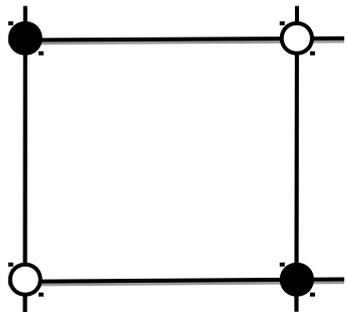


non linéairement  
séparable

- Le perceptron peut distinguer des ensembles de points linéairement séparables.

# Le ou exclusif

- $XOR(x_1, x_2)$  :  $XOR(0,1) = XOR(1,0) = 1$   
 $XOR(0,0) = XOR(1,1) = 0$



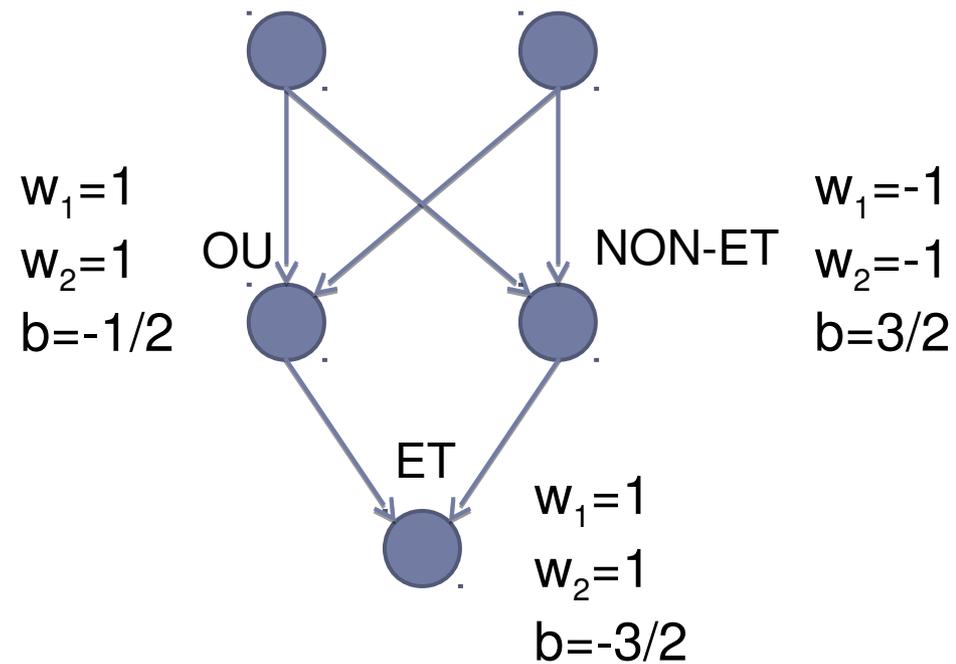
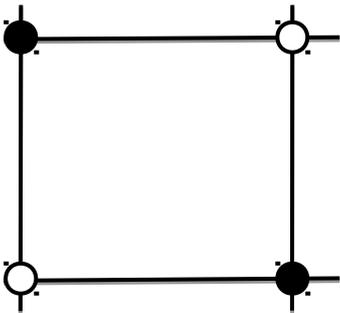
○ =  $XOR^{-1}(0)$   
● =  $XOR^{-1}(1)$

non linéairement séparable

- Un perceptron ne peut pas exécuter un ou exclusif.

# Le ou exclusif : possible avec plusieurs couches

- $XOR(x_1, x_2)$  :  $XOR(0,1) = XOR(1,0) = 1$   
 $XOR(0,0) = XOR(1,1) = 0$



# Algorithme d'apprentissage Perceptron

Sortie désirée

$$y(n) = \begin{cases} 1 & \text{si } x(n) \in A \\ -1 & \text{si } x(n) \in B \end{cases}$$

1. Initialiser poids et seuils : mettre  $w_i$  et  $b$  à petits nombre alleatoire dans l'intervalle  $[-1, +1]$ .
2. Sélectionner un échantillon aléatoire à partir de l'ensemble d'entraînement comme entrée.
3. Si la classification est correcte, ne rien faire
4. Si la classification est incorrecte, modifier le vecteur de poids  $w$  en utilisant :

$$w_i = w_i + \eta y(n) x_i(n)$$

Répétez cette procédure jusqu' à ce que l'ensemble du entrainement soit correctement classé.

# Perceptron : exemple d'apprentissage

l'ensemble du entrainement :

classe A (étoiles)  
sortie désirée +1

classe B (cercles)  
sortie désirée -1

$$x(1) = (-2, 1)$$

$$x(4) = (-2, -1)$$

$$x(2) = (1, 1)$$

$$x(5) = (-1, -1.5)$$

$$x(3) = (1.5, -0.5)$$

$$x(6) = (2, -2)$$

valeurs initiales :

$$\eta = 0.2$$

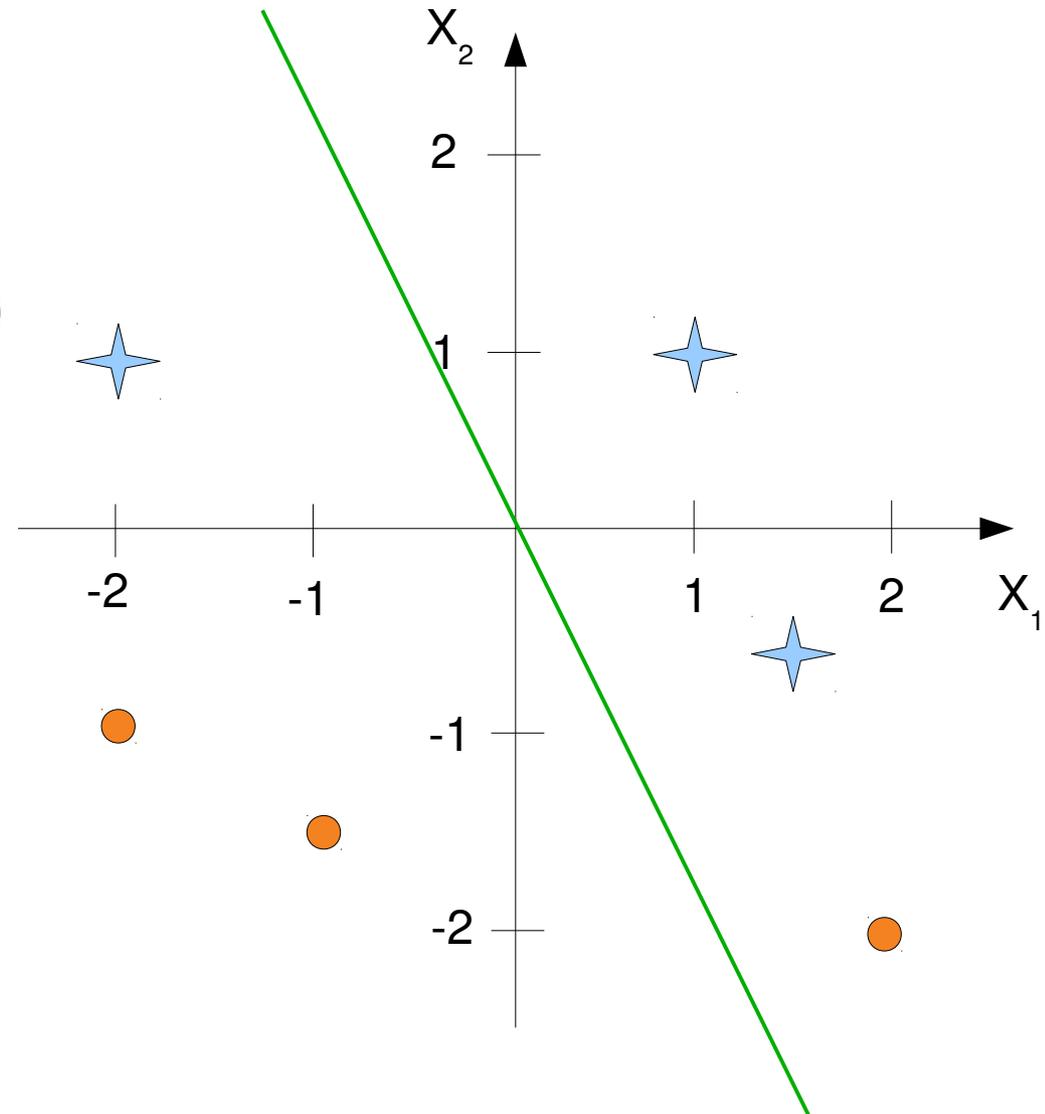
$$w = \begin{pmatrix} 0 \\ 1 \\ 0.5 \end{pmatrix}$$

l'hyperplan :

$$0 = w_0 + w_1 x_1 + w_2 x_2$$

$$0 = 0 + x_1 + 0.5 x_2$$

$$\Rightarrow x_2 = -2 x_1$$



# Perceptron : exemple d'apprentissage

$$\eta = 0.2$$

$$w = \begin{pmatrix} 0 \\ 1 \\ 0.5 \end{pmatrix}$$

$$x_1 = 1, x_2 = 1$$

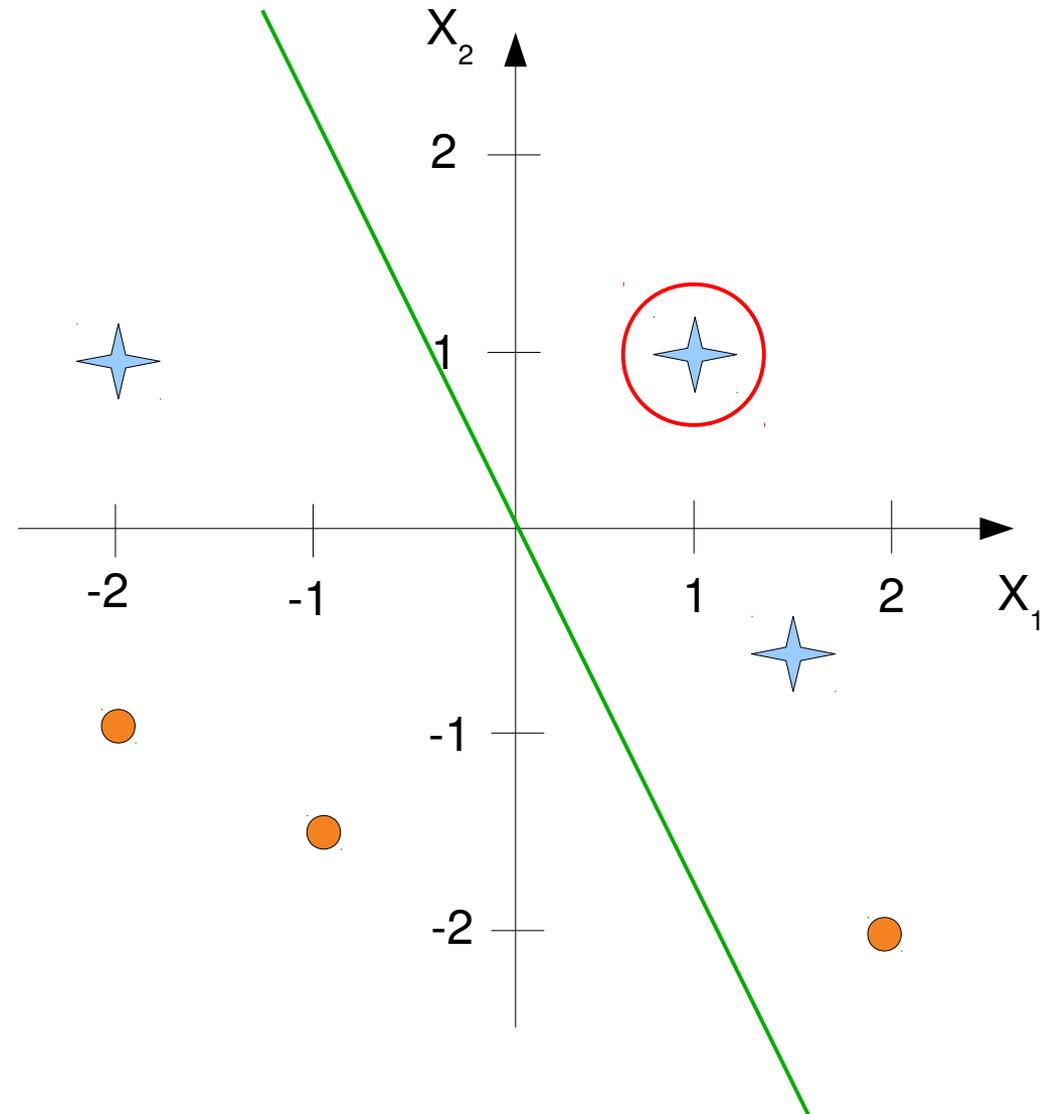
$$w^T x = ???$$

$$y = ???$$

$$w^T x = 1.5$$

$$y = +1$$

Classification correcte,  
aucune mesure



# Perceptron : exemple d'apprentissage

$$\eta = 0.2$$

$$w = \begin{pmatrix} 0 \\ 1 \\ 0.5 \end{pmatrix}$$

$$x_1 = 2, x_2 = -2$$

$$w^T x = \mathcal{P}??$$

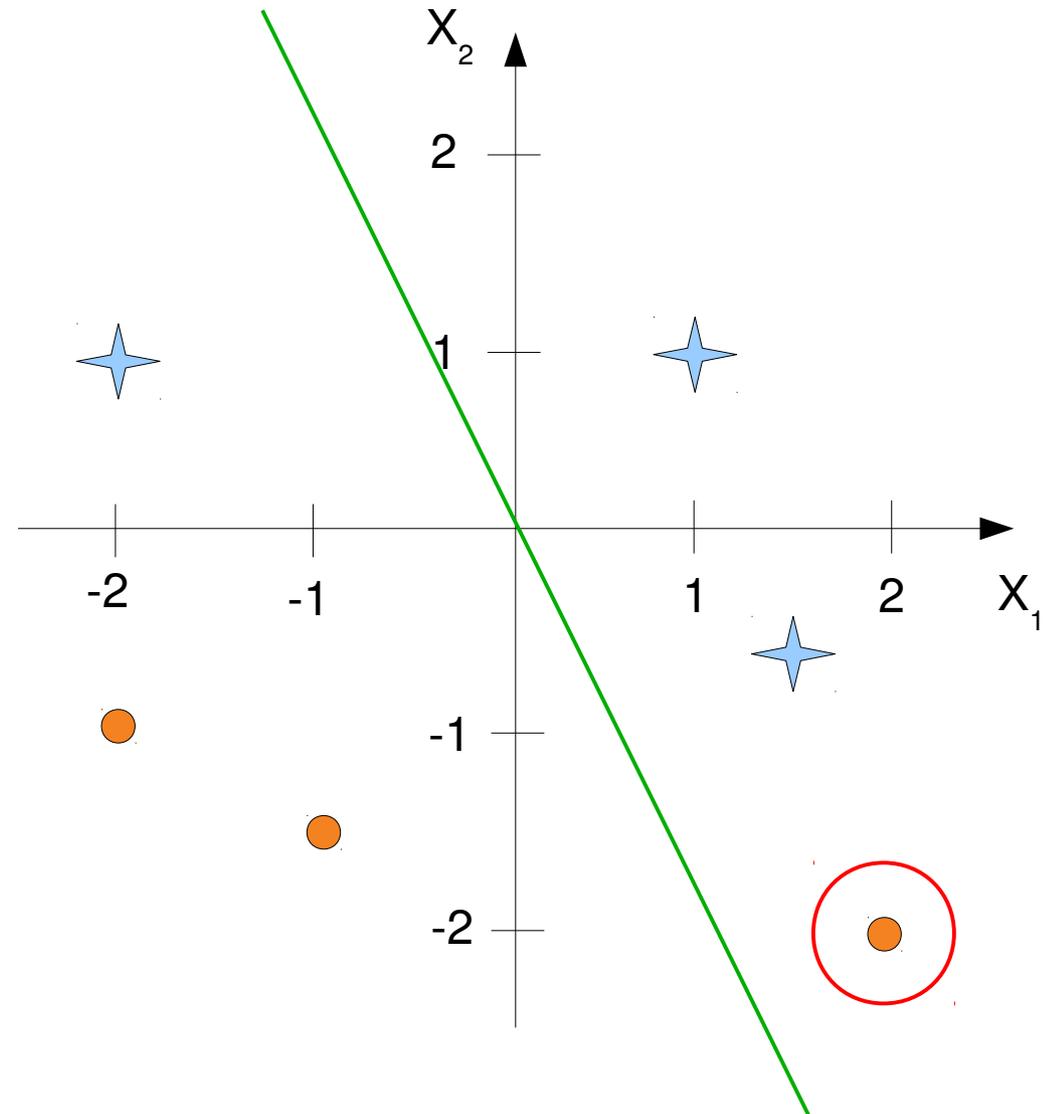
$$y = +1??$$

Classification incorrecte :

$$w_0 = w_0 - 0.2 * 1$$

$$w_1 = w_1 - 0.2 * 2$$

$$w_2 = w_2 - 0.2 * (-2)$$

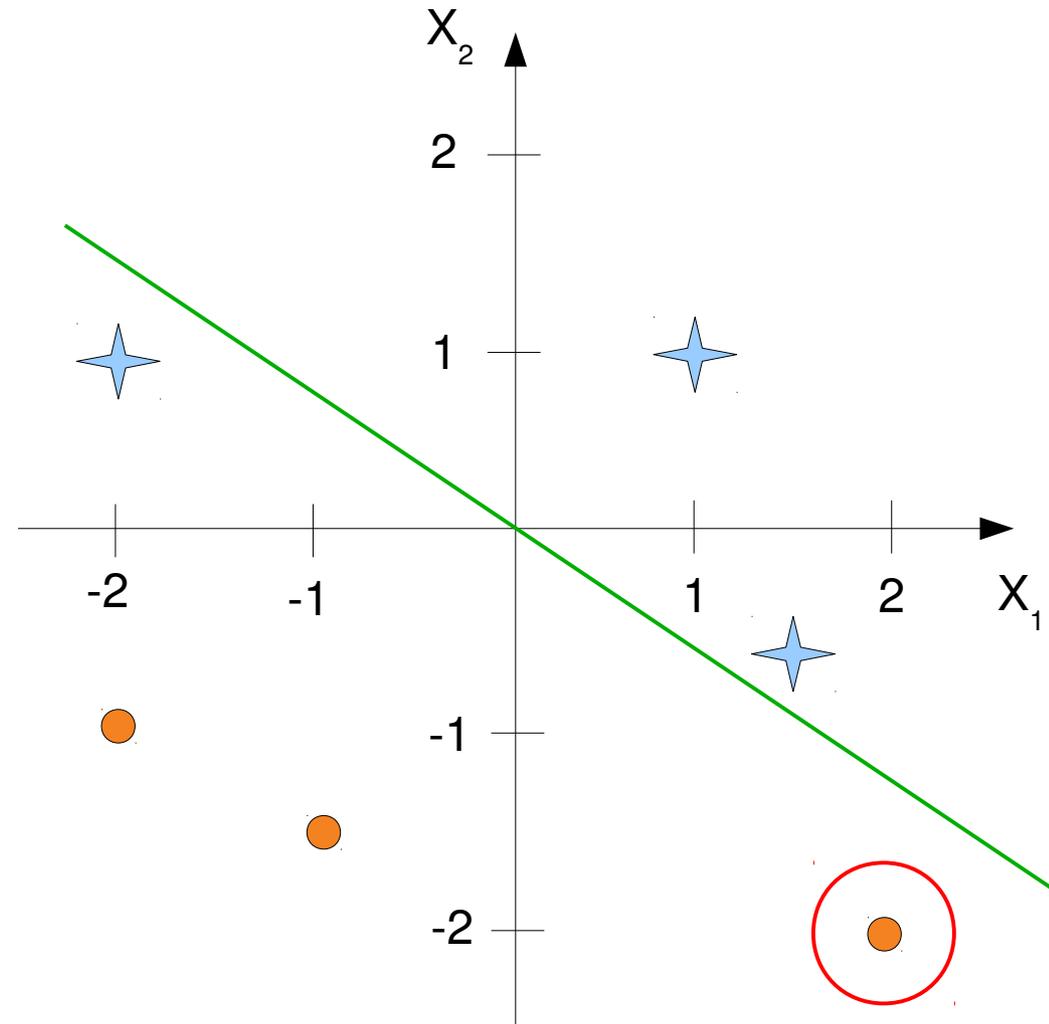


# Perceptron : exemple d'apprentissage

$$\eta = 0.2$$

$$w = \begin{pmatrix} -0.2 \\ 0.6 \\ 0.9 \end{pmatrix}$$

$$x_1 = 2, x_2 = -2$$



# Perceptron : exemple d'apprentissage

$$\eta = 0.2$$

$$w = \begin{pmatrix} -0.2 \\ 0.6 \\ 0.9 \end{pmatrix}$$

$$x_1 = -1, x_2 = -1.5$$

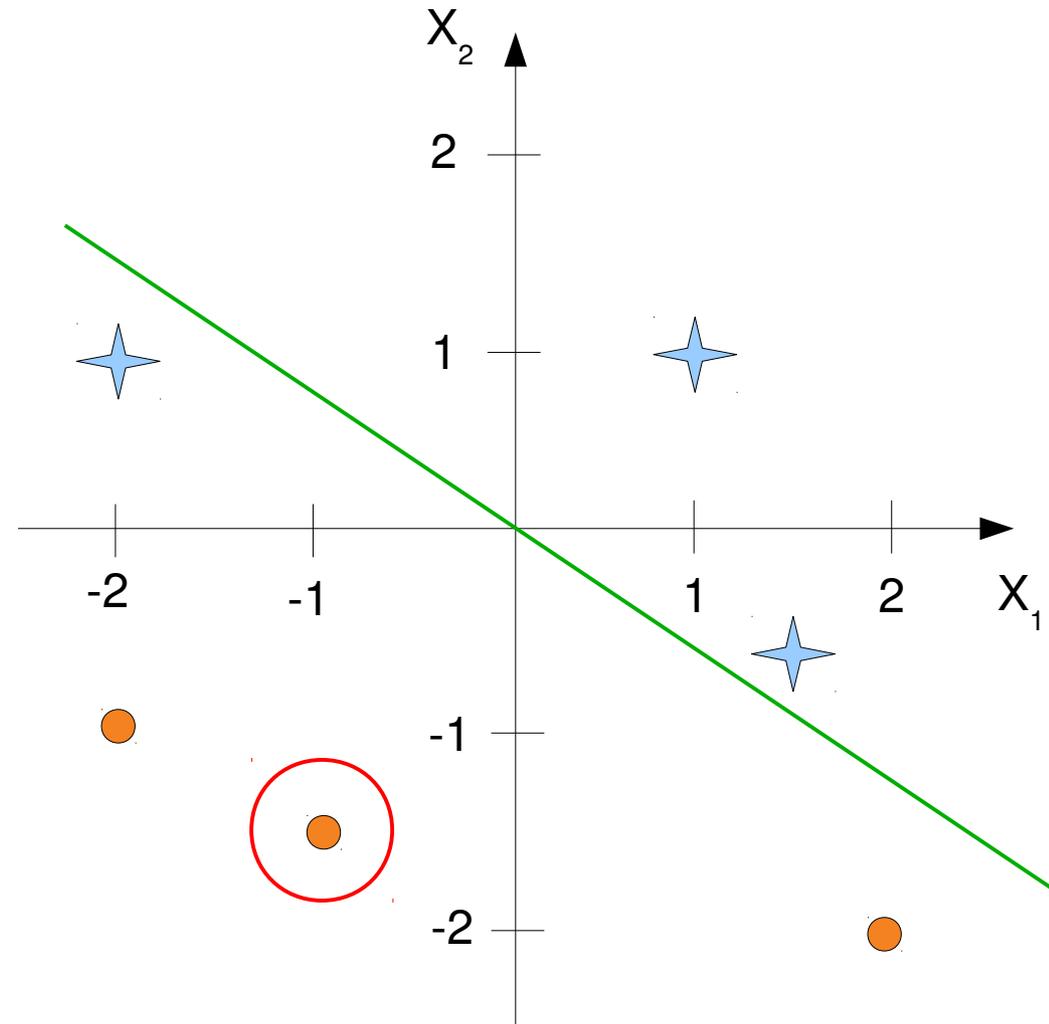
$$w^T x = ???$$

$$y = ???$$

$$w^T x = -2.15$$

$$y = -1$$

Classification correcte,  
aucune mesure



# Perceptron : exemple d'apprentissage

$$\eta = 0.2$$

$$w = \begin{pmatrix} -0.2 \\ 0.6 \\ 0.9 \end{pmatrix}$$

$$x_1 = -2, x_2 = -1$$

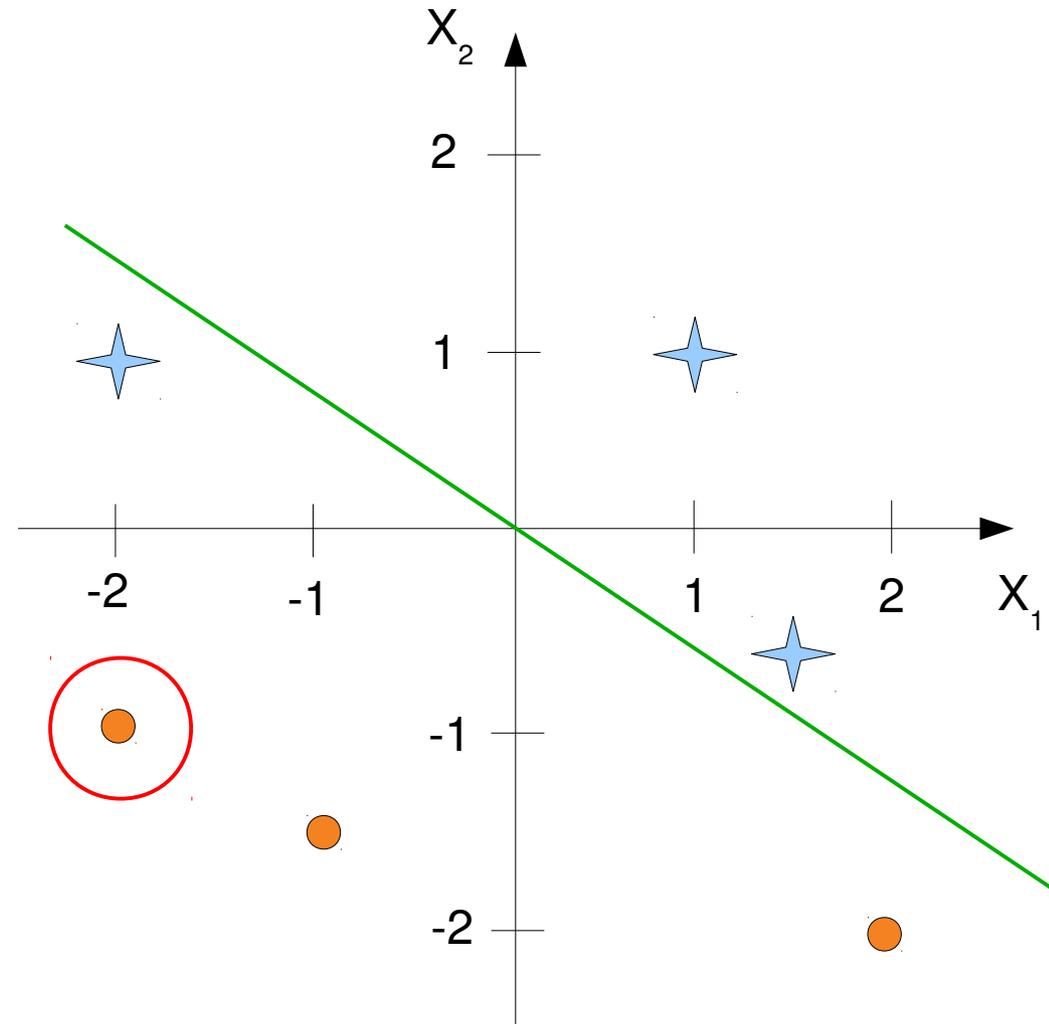
$$w^T x = ???$$

$$y = ???$$

$$w^T x = -2.3$$

$$y = -1$$

Classification correcte,  
aucune mesure



# Perceptron : exemple d'apprentissage

$$\eta = 0.2$$

$$w = \begin{pmatrix} -0.2 \\ 0.6 \\ 0.9 \end{pmatrix}$$

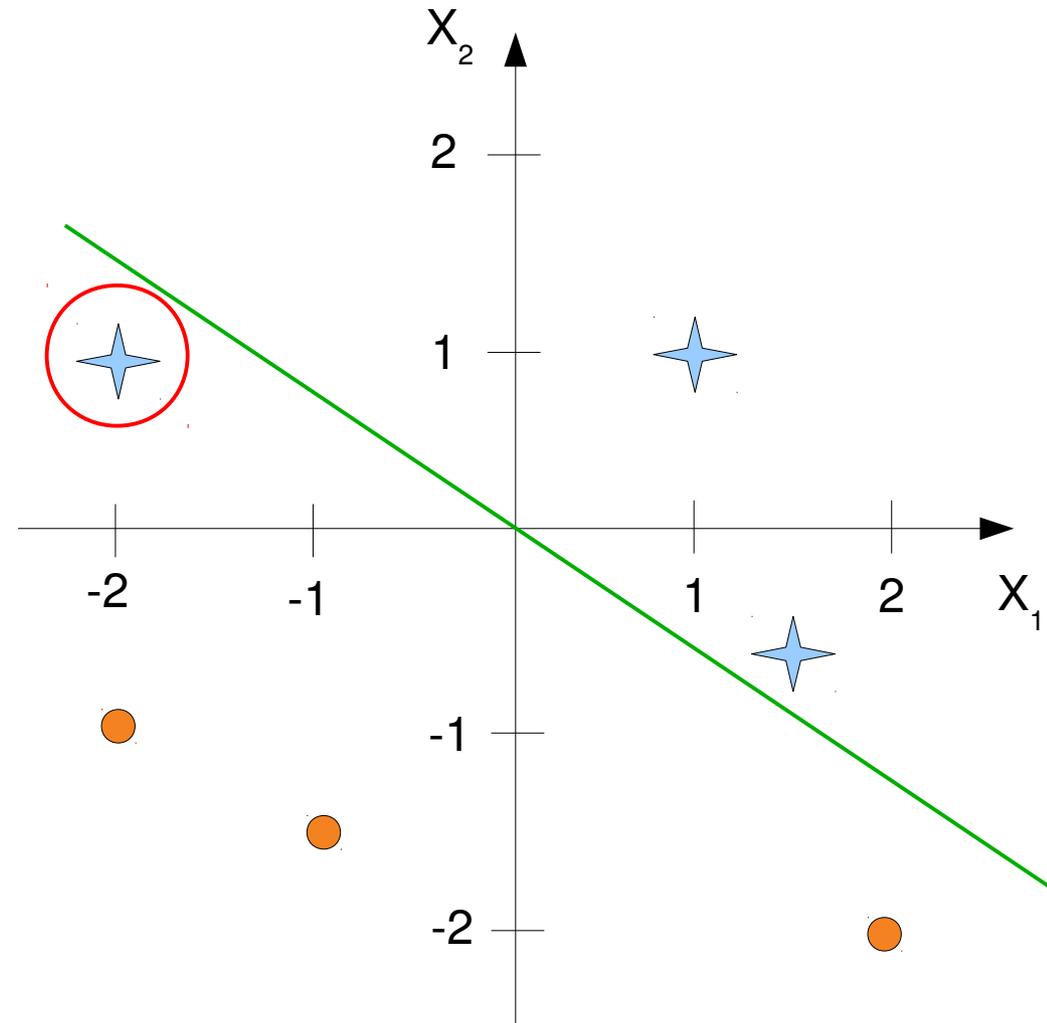
$$x_1 = -2, x_2 = 1$$

Classification incorrecte :

$$w_0 = w_0 + 0.2 * 1$$

$$w_1 = w_1 + 0.2 * (-2)$$

$$w_2 = w_2 + 0.2 * 1$$



# Perceptron : exemple d'apprentissage

$$\eta = 0.2$$

$$w = \begin{pmatrix} 0 \\ 0.2 \\ 1.1 \end{pmatrix}$$

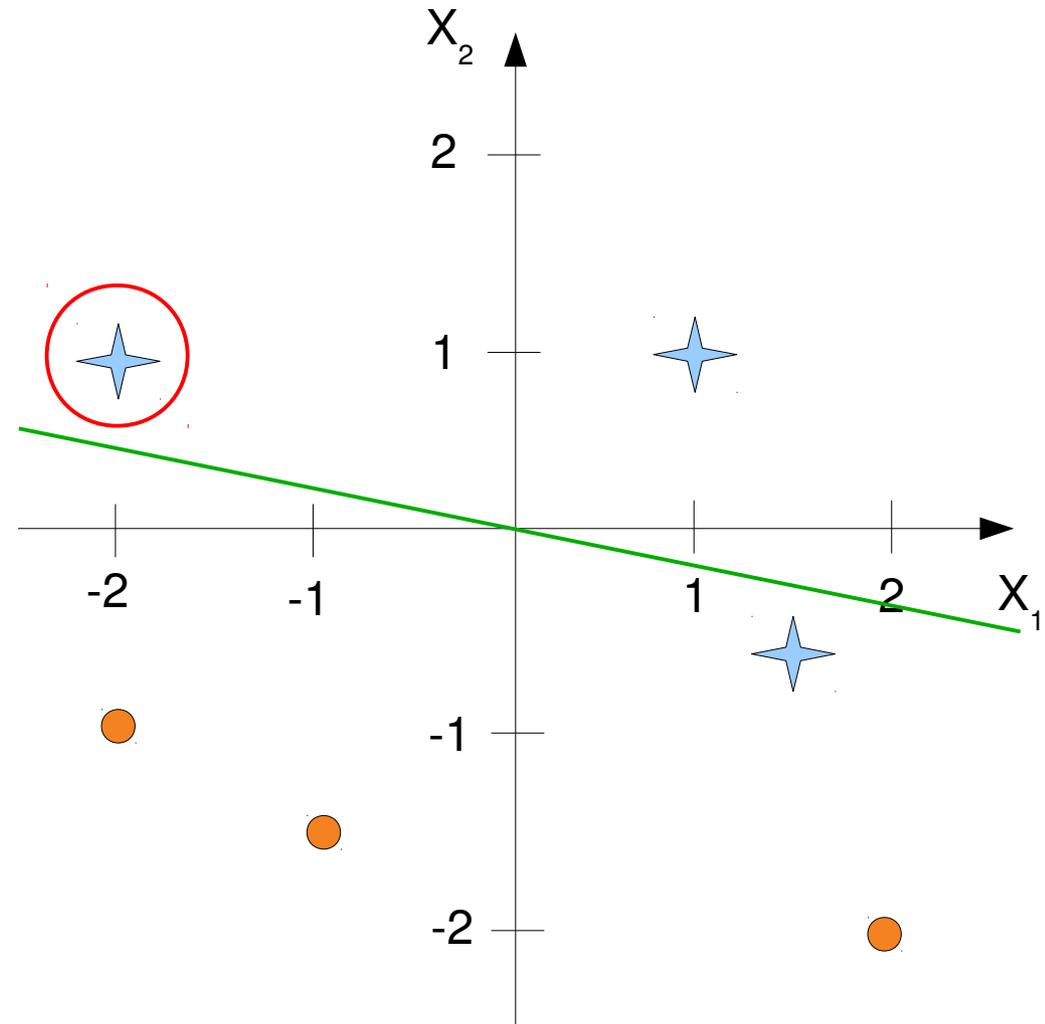
$$x_1 = -2, x_2 = 1$$

Classification incorrecte :

$$w_0 = w_0 + 0.2 * 1$$

$$w_1 = w_1 + 0.2 * (-2)$$

$$w_2 = w_2 + 0.2 * 1$$



# Perceptron : exemple d'apprentissage

$$\eta = 0.2$$

$$w = \begin{pmatrix} 0 \\ 0.2 \\ 1.1 \end{pmatrix}$$

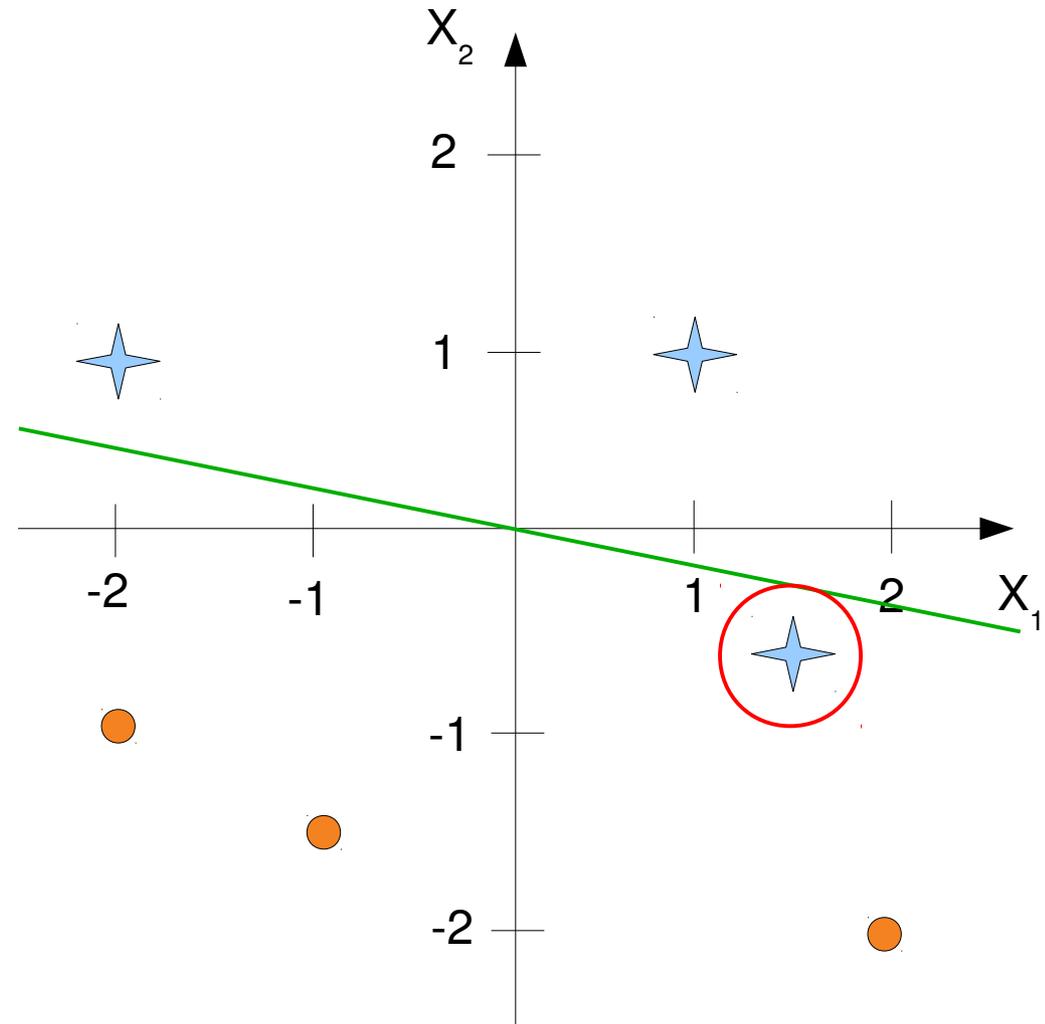
$$x_1 = 1.5, x_2 = -0.5$$

Classification incorrecte :

$$w_0 = w_0 + 0.2 * 1$$

$$w_1 = w_1 + 0.2 * 1.5$$

$$w_2 = w_2 + 0.2 * (-0.5)$$



# Perceptron : exemple d'apprentissage

$$\eta = 0.2$$

$$w = \begin{pmatrix} 0.2 \\ 0.5 \\ 1 \end{pmatrix}$$

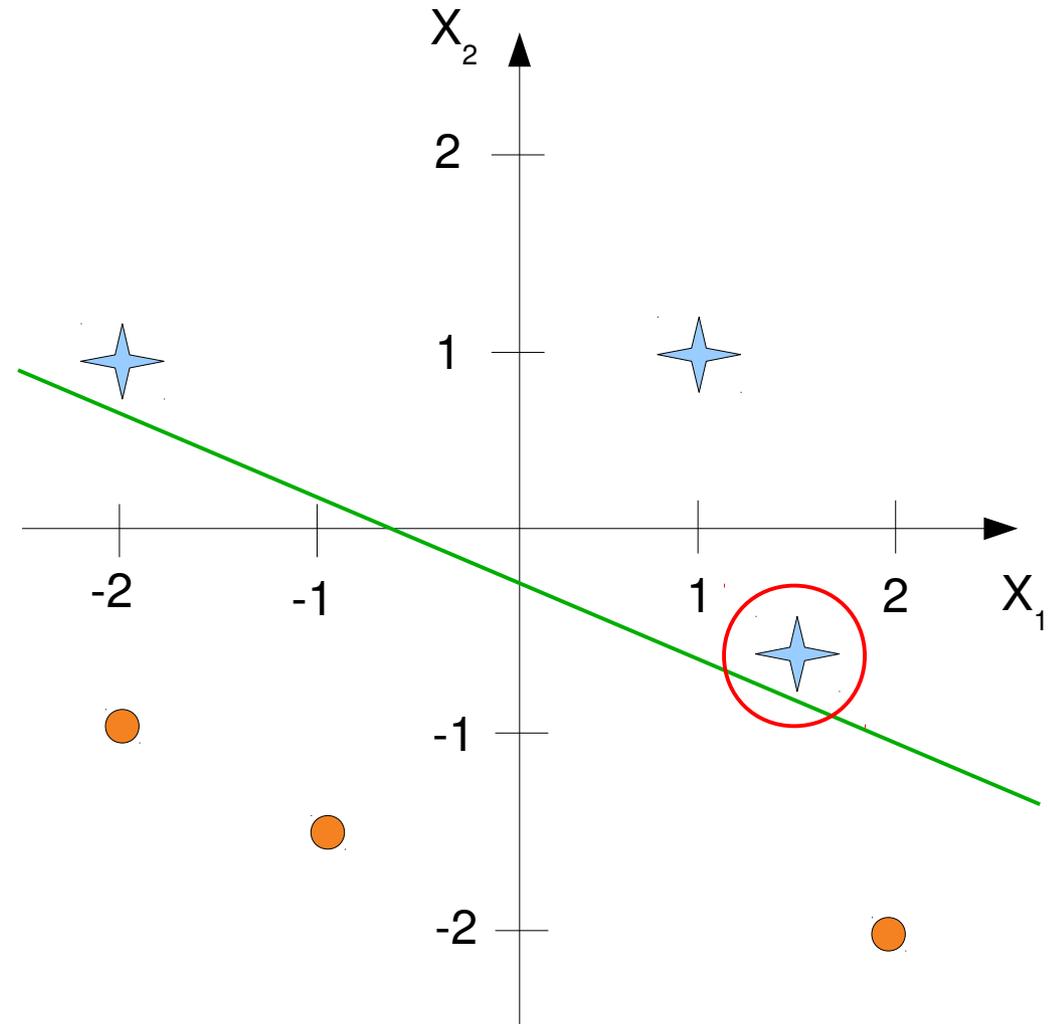
$$x_1 = 1.5, x_2 = -0.5$$

Classification incorrecte :

$$w_0 = w_0 + 0.2 * 1$$

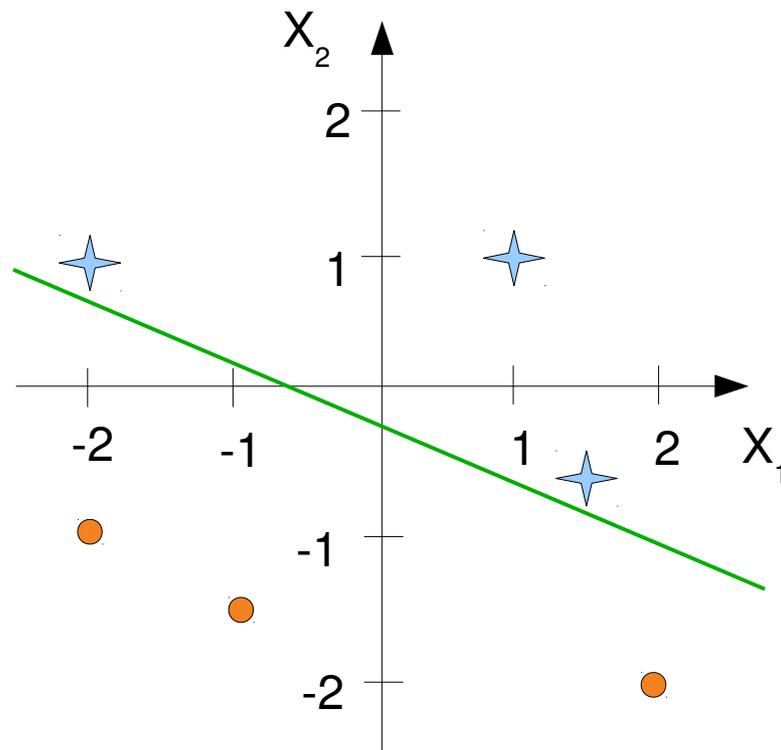
$$w_1 = w_1 + 0.2 * 1.5$$

$$w_2 = w_2 + 0.2 * (-0.5)$$



# Théorème de convergence Perceptron

Le théorème stipule que pour tout ensemble de données linéairement séparable, la règle d'apprentissage du perceptron est garantie de trouver une solution dans un nombre fini d'itérations.



# Projet de fin de cours

Ecrire un programme mettant en œuvre un Perceptron qui apprend à reconnaître les chiffres manuscrits.



# Projet de fin de cours : les entrées

La base de données MNIST : <http://yann.lecun.com/exdb/mnist/>

60,000 chiffres pour l'entrainement

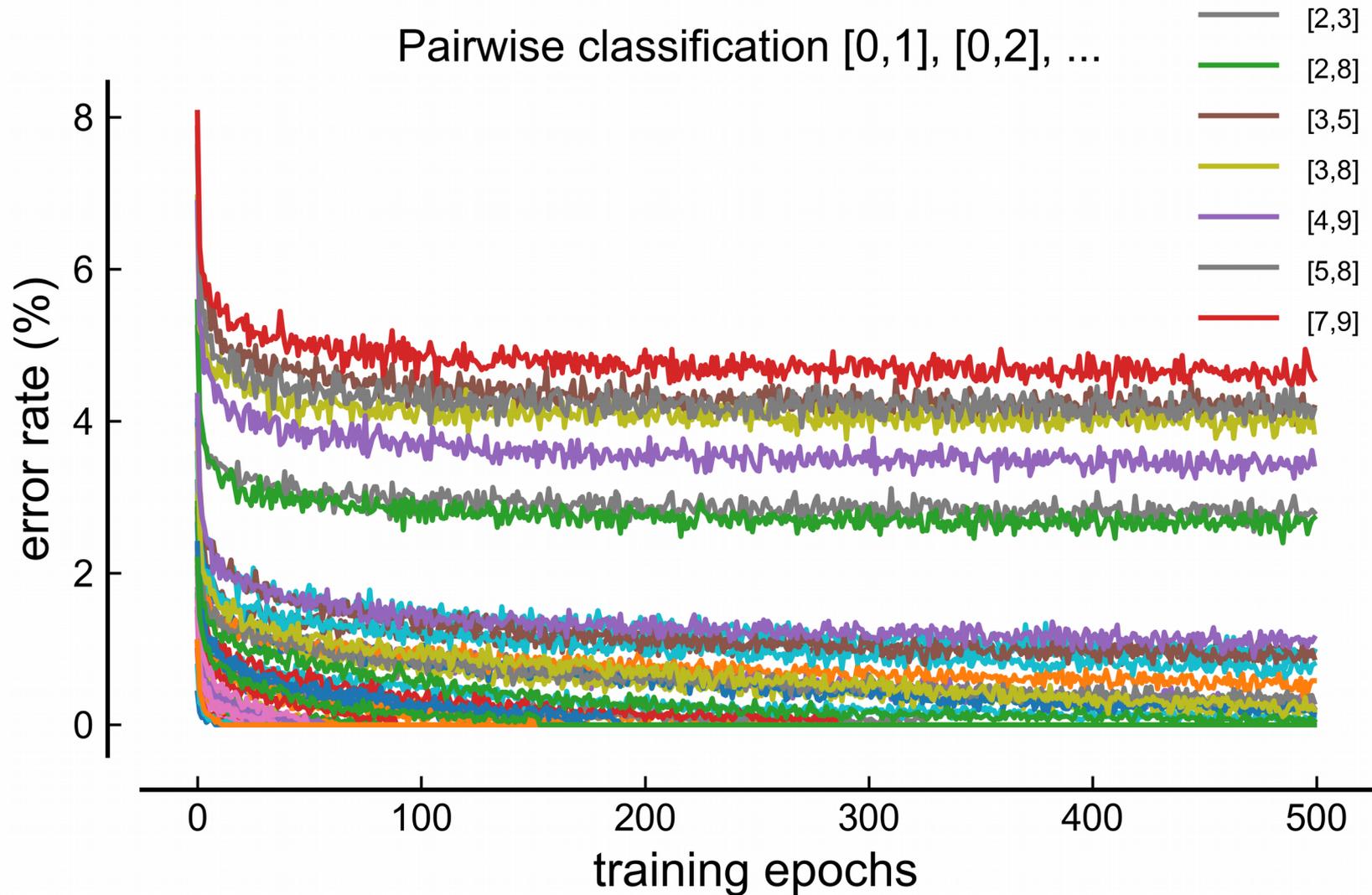
10,000 chiffres pour le test



# Projet de fin de cours : conseils

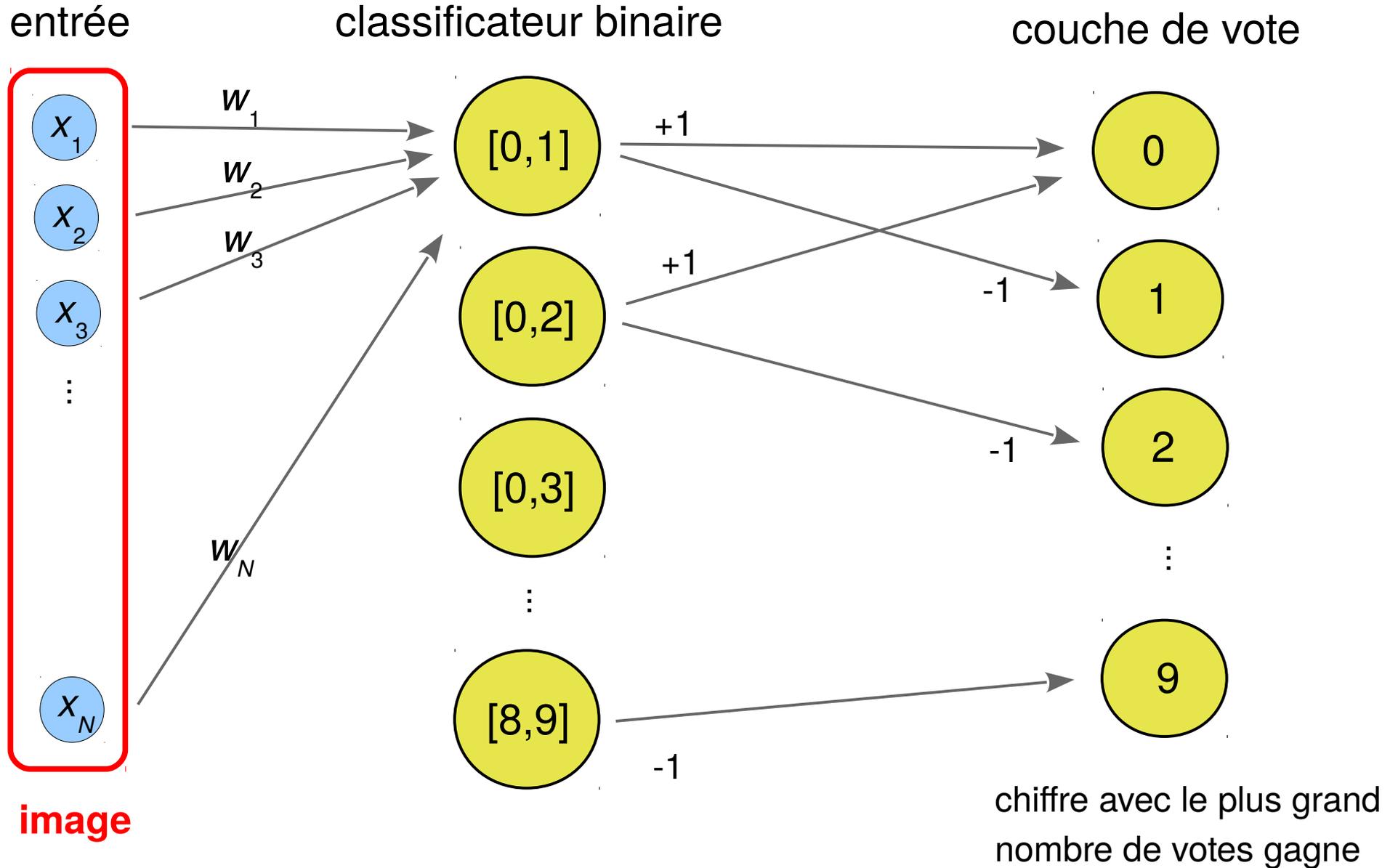
- Le but est d'entraîner le Perceptron avec les 60,000 chiffres et ensuite de tester la reconnaissance des 10,000 chiffres.
- Pour ce faire, entraînez le perceptron à différencier d'abord deux chiffres.
- Utilisez un Perceptron d'une seule couche où l'image est traitée comme entrée et l'unité de sortie doit décider si l'un ou l'autre chiffre a été présenté.
- Suivez la performance de la classification pendant l'entraînement et déterminez la performance avec les chiffres de test.
- Important : L'entraînement nécessitera plusieurs boucles de présentation des 60,000 chiffres.
- Réfléchir à la façon de généraliser le perceptron pour reconnaître tous les nombres et pas seulement deux.

# Perceptron : performance de classification binaire

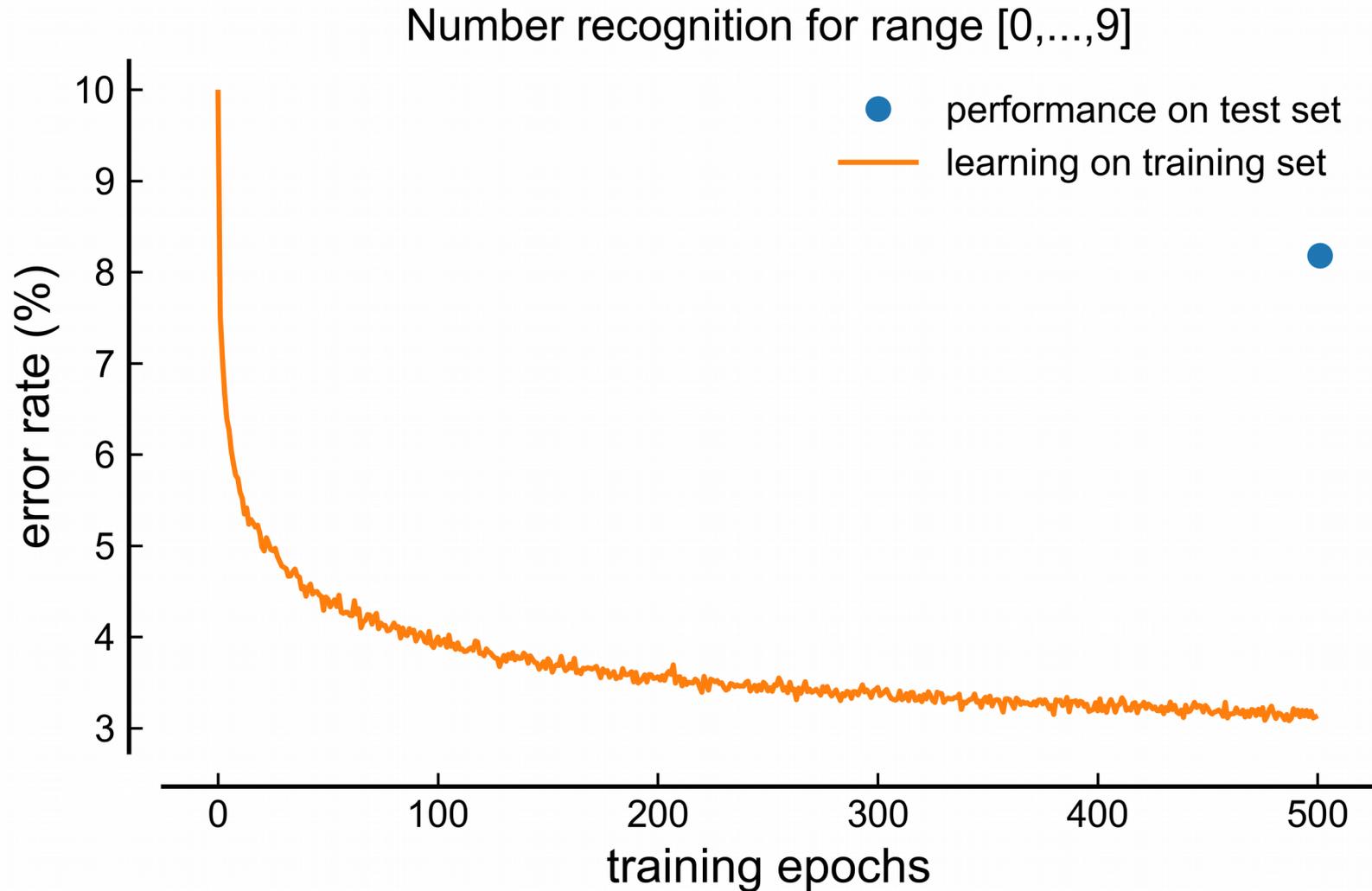




# Perceptron : reconnaître 10 chiffres



# Perceptron : reconnaître 10 chiffres



# Perceptron : reconnaître 10 chiffres

