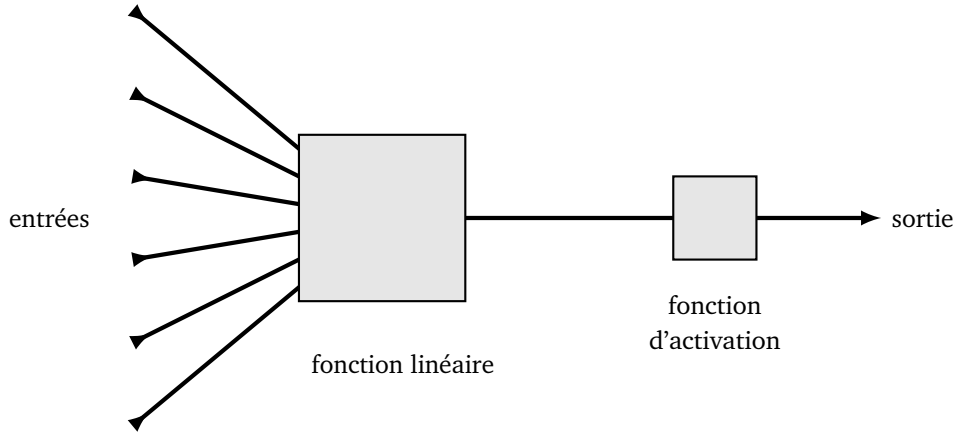
Réseau de neurones

Le cerveau humain est composé de plus de 80 milliards de neurones. Chaque neurone reçoit des signaux électriques d'autres neurones et réagit en envoyant un nouveau signal à ses neurones voisins. Nous allons construire des réseaux de neurones artificiels. Dans ce chapitre, nous ne chercherons pas à expliciter une manière de déterminer dynamiquement les paramètres du réseau de neurones, ceux-ci seront fixés ou bien calculés à la main.

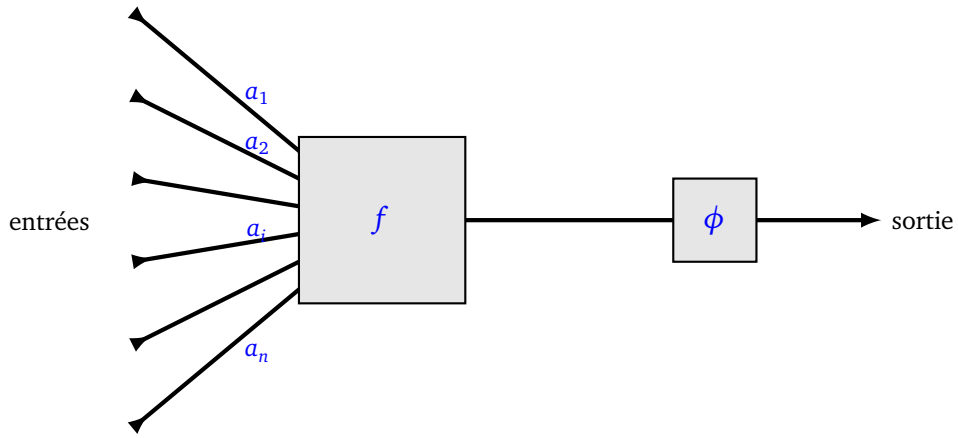
1. Perceptron

1.1. Perceptron linéaire

Le principe du perceptron linéaire est de prendre des valeurs en entrées, de faire un calcul simple et de renvoyer une valeur en sortie. Les calculs dépendent de paramètres propres à chaque perceptron.



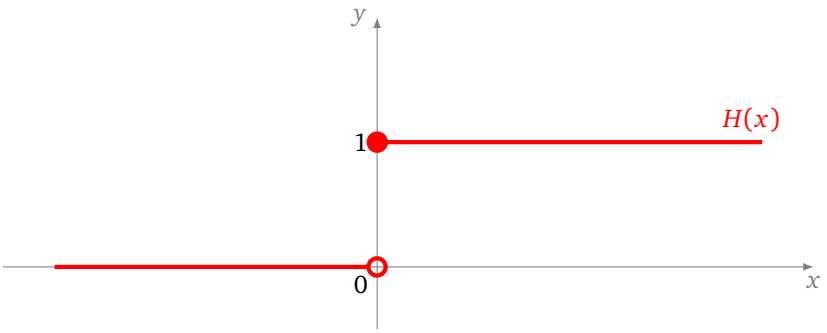
Le calcul effectué par un perceptron se décompose en deux phases : un calcul par une fonction linéaire f, suivi d'une fonction d'activation ϕ.



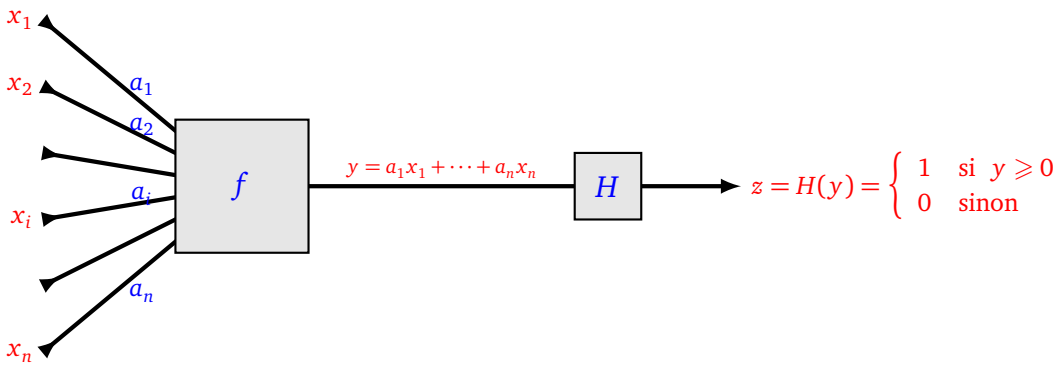
Détaillons chaque phase.

- **Partie linéaire.** Le perceptron est d'abord muni de ***poids*** $a_1, \dots, a_n$ qui déterminent une fonction linéaire
$$f(x_1, \dots, x_n) = a_1x_1 + a_2x_2 + \cdots + a_nx_n.$$
- **Fonction d'activation.** La valeur renvoyée par la fonction linéaire f est ensuite composée par une fonction d'activation ϕ.
- **Sortie.** La valeur de sortie est donc $\phi(a_1x_1 + a_2x_2 + \cdots + a_nx_n)$.

Dans ce chapitre, la fonction d'activation sera (presque) toujours la fonction marche de Heaviside :
$$\begin{cases} H(x) = 1 & \text{si } x \geqslant 0, \\ H(x) = 0 & \text{si } x < 0. \end{cases}$$

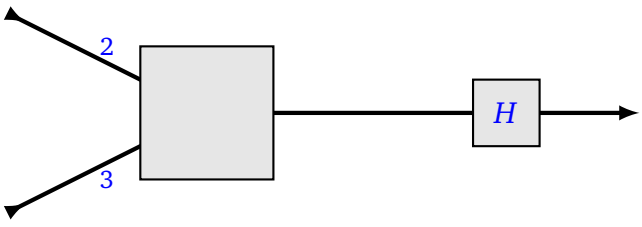


Voici ce que fait un perceptron linéaire de poids $a_1, \dots, a_n$ et de fonction d'activation la fonction marche de Heaviside :

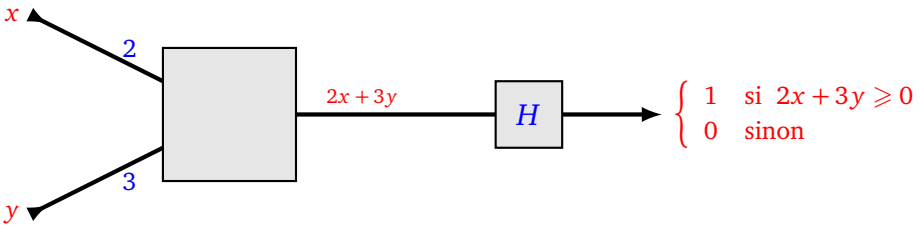


On peut donc définir ce qu'est un perceptron. Un ***perceptron linéaire*** à n variables et de fonction d'activation la fonction marche de Heaviside est la donnée de n coefficients réels $a_1, \dots, a_n$ auxquels sont associés la fonction $F : \mathbb{R} \to \mathbb{R}$ définie par $F = H \circ f$, c'est-à-dire :
$$\begin{cases} F(x_1, \dots, x_n) = 1 & \text{si } a_1x_1 + a_2x_2 + \cdots + a_nx_n \geqslant 0, \\ F(x, \dots, x_n) = 0 & \text{sinon.} \end{cases}$$

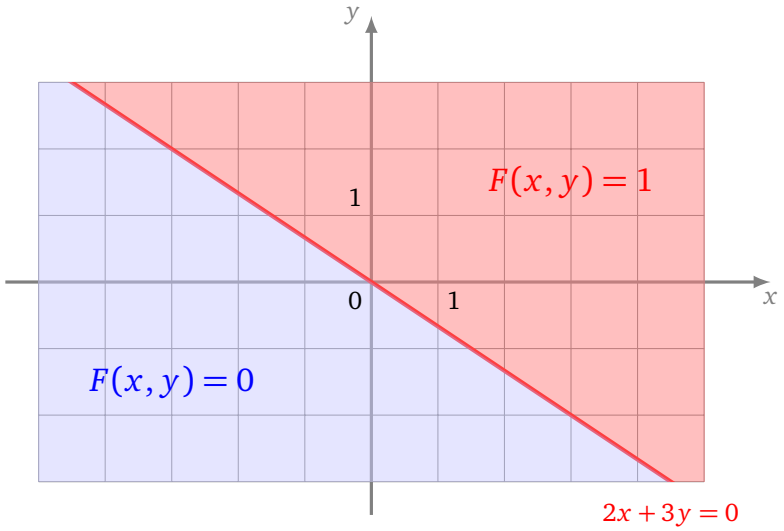
Exemple.
Voici un perceptron à deux entrées. Il est défini par les poids $a = 2$ et $b = 3$.



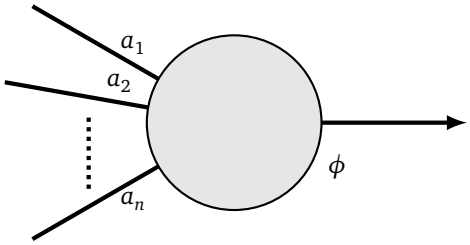
- **Formule.**
  Notons x et y les deux réels en entrée. La fonction linéaire f est donc
  $$f(x,y) = 2x + 3y.$$
  La valeur en sortie est donc :
  $$\begin{cases} F(x,y) = 1 & \text{si } 2x+3y \geqslant 0 \\ F(x,y) = 0 & \text{sinon.} \end{cases}$$



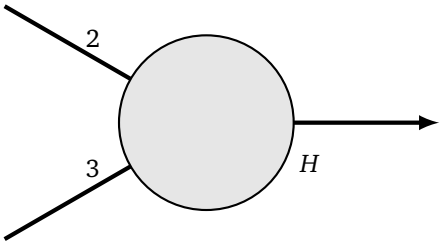
- **Évaluation.** Utilisons ce perceptron comme une fonction. Que renvoie le perceptron pour la valeur d'entrée $(x,y) = (4,-1)$? On calcule $f(x,y) = 2x + 3y = 5$. Comme $f(x,y) \geqslant 0$, alors la valeur de sortie est donc $F(x,y) = 1$.
  Recommençons avec $(x,y) = (-3,1)$. Cette fois $f(x,y) = -3 < 0$ donc $F(x,y) = 0$.
  L'entrée $(x,y) = (6,-4)$ est « à la limite » car $f(x,y) = 0$ (0 est l'abscisse critique pour la fonction marche de Heaviside). On a $F(x,y) = 1$.
- **Valeurs de la fonction.**
  La fonction F prend seulement deux valeurs : 0 ou 1. La frontière correspond aux points (x,y) tels que $f(x,y) = 0$, c'est-à-dire à la droite $2x + 3y = 0$. Pour les points au-dessus de la droite (ou sur la droite) la fonction F prend la valeur 1 ; pour les points en-dessous de la droite, la fonction F vaut 0.



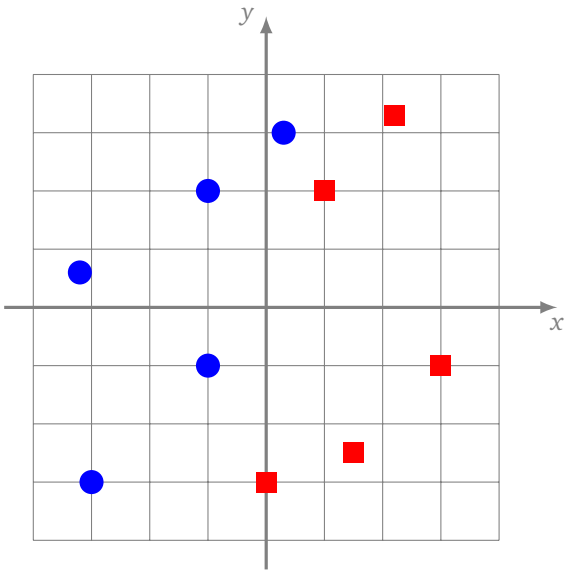
Notation.
Nous représentons un perceptron par une forme plus condensée : sous la forme d'un ***neurone***, avec des poids sur les arêtes d'entrées. Nous précisons en indice la fonction d'activation utilisée ϕ. Si le contexte est clair cette mention est omise.



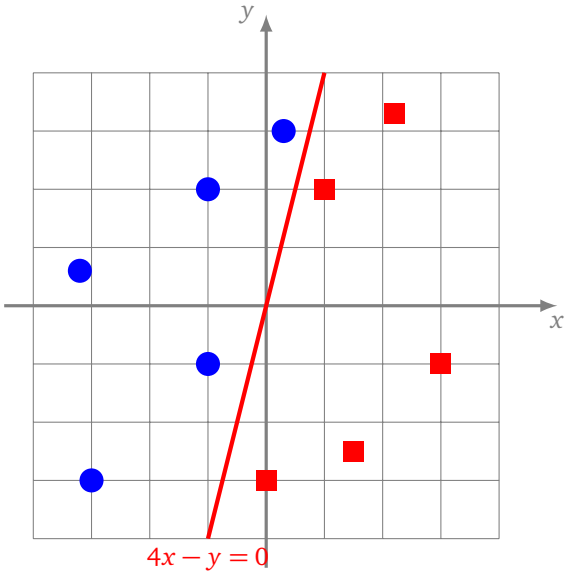
Voici le neurone à deux variables de l'exemple précédent.



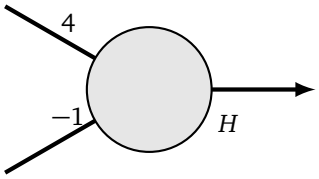
Exemple.
Voici deux catégories de points : des ronds bleus et des carrés rouges. Comme trouver un perceptron qui les sépare ?

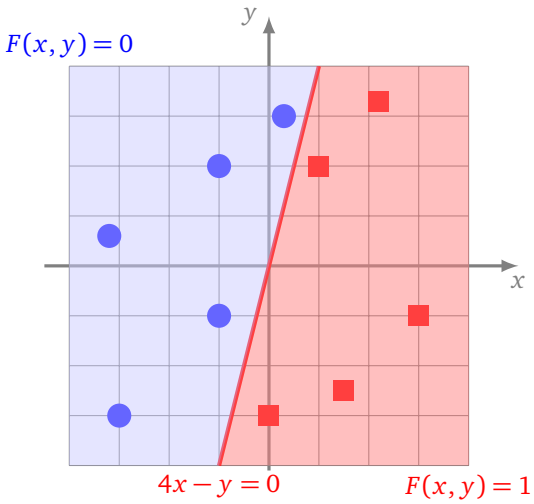


Il s'agit donc de trouver les deux poids a et b d'un perceptron, dont la fonction associée F vérifie $F(x,y)=1$ pour les coordonnées des carrés et $F(x,y)=0$ pour les ronds.



Trouvons un droite qui les sépare. Par exemple, la droite d'équation $4x-y=0$ sépare les ronds des carrés. On définit donc le neurone avec les poids $a=4$ et $b=-1$. Si (x,y) sont les coordonnées d'un carré alors on a bien $F(x,y)=1$ et pour un rond $F(x,y)=0$.





1.2. Biais – Perceptron affine

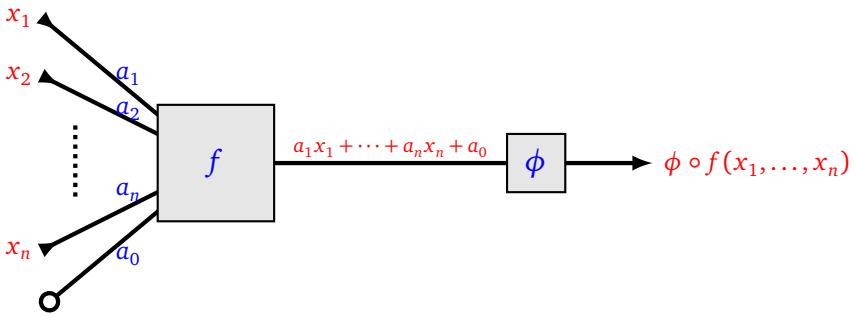
Pour l'instant notre perceptron à deux entrées sépare le plan en deux parties selon une droite passant par l'origine. Nous allons plus loin avec le ***perceptron affine***.
On modifie la définition du perceptron, sans changer le nombre d'entrées, mais en ajoutant un poids supplémentaire. Dans le cas de n entrées il y a donc $n+1$ ***poids*** :

- les ***coefficients*** $a_1, \dots, a_n \in \mathbb{R}$,
- et le ***biais*** $a_0 \in \mathbb{R}$,

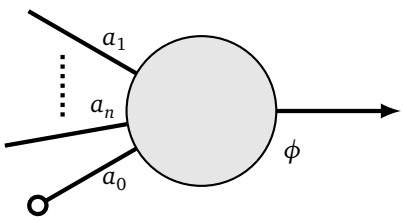
qui définissent la fonction affine :

$$f(x_1, \dots, x_n) = a_1 x_1 + a_2 x_2 + \cdots + a_n x_n + a_0.$$

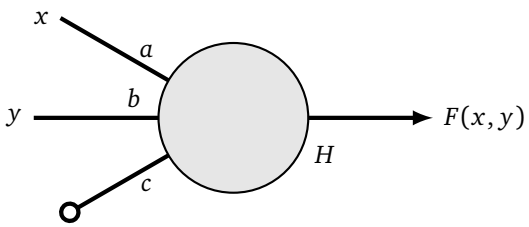
Comme auparavant, ce calcul est suivi de la fonction d'activation.



On représente le neurone avec une nouvelle arête, pondérée par le biais. L'arête est terminée par un rond pour signifier que cela ne correspond pas à une entrée.



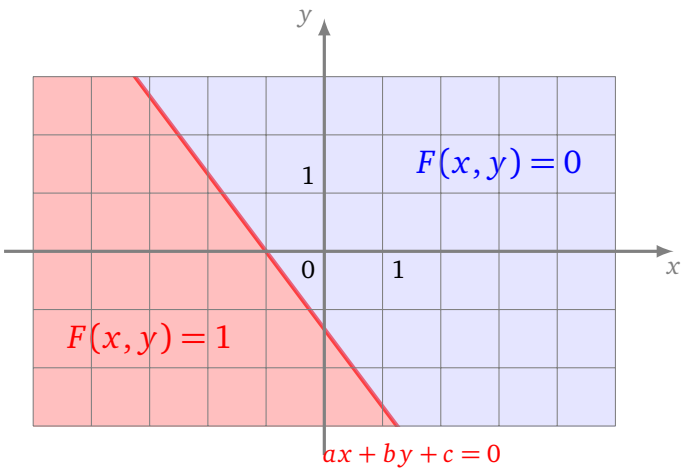
Dans le cas de deux entrées, les poids sont trois réels a, b (les coefficients) et c (le biais).

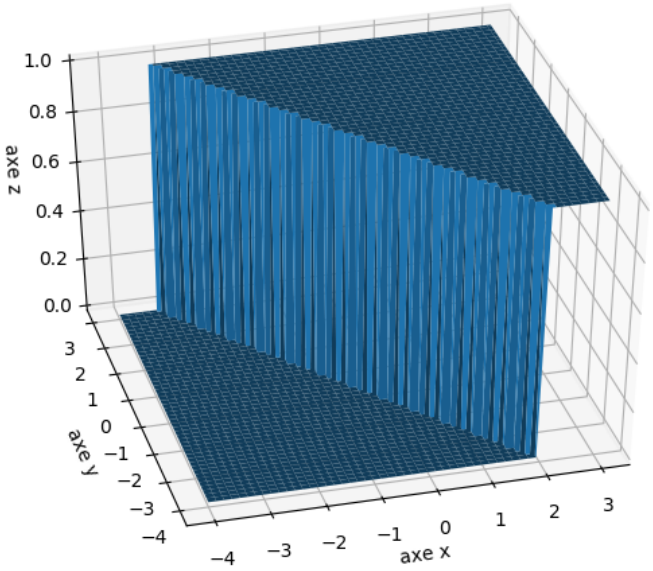


La fonction $f : \mathbb{R}^2 \to \mathbb{R}$ est la fonction affine $f(x,y) = ax + by + c$. Si la fonction d'activation est la fonction marche de Heaviside, alors le perceptron affine définit une fonction $F : \mathbb{R}^2 \to \mathbb{R}$ par la formule :

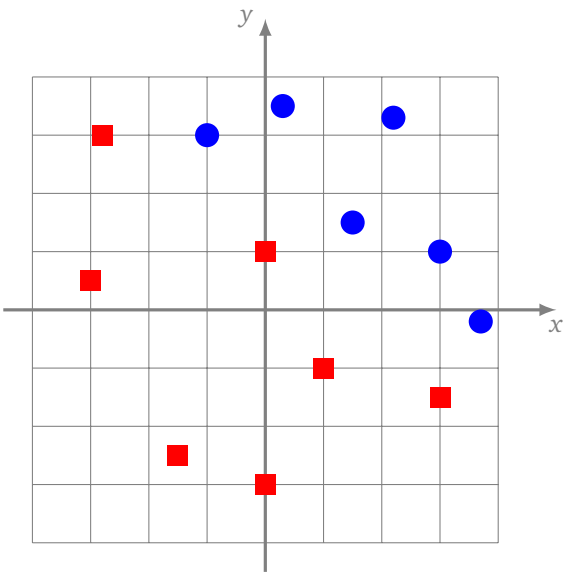
$$\begin{cases} F(x,y) = 1 & \text{si } ax + by + c \geqslant 0 \\ F(x,y) = 0 & \text{sinon.} \end{cases}$$

Un perceptron affine à deux entrées sépare donc le plan en deux parties, la frontière étant la droite d'équation $ax + by + c = 0$. D'un côté de cette droite la fonction F vaut 1, de l'autre elle vaut 0.

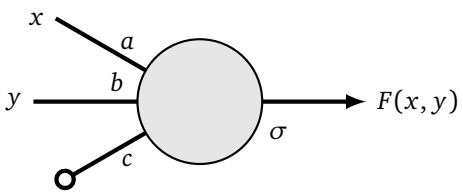


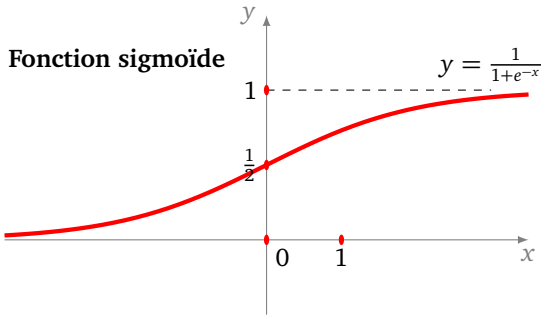


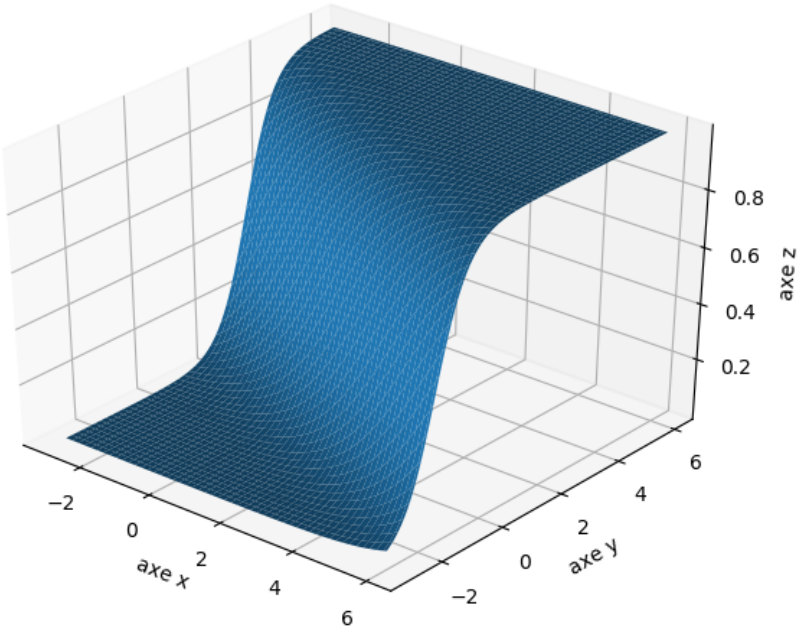
Exercice.
Trouver un perceptron qui distingue les carrés des ronds.



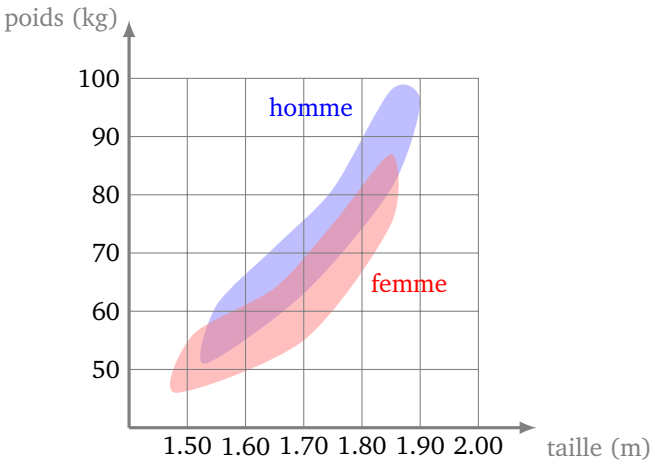
Il n'est pas toujours possible de répondre à une question seulement par « oui » ou « non ». Pour y remédier, on peut changer de fonction d'activation en utilisant par exemple la fonction sigmoïde σ. Dans ce cas, à chaque point du plan, on associe, non plus 0 ou 1, mais une valeur entre 0 et 1.
Ce nombre peut correspondre à un degré de certitude. Par exemple avec une question « Est-ce que cette photo est un chat ? », si la sortie vaut 0.8, cela signifie « c'est bien un chat avec 80% de certitude ». Si la sortie vaut 0.1 alors ce n'est probablement pas un chat.
Voici un neurone à deux entrées muni de la fonction d'activation sigmoïde définie par $\sigma(x) = \frac{1}{1+e^{-x}}$.







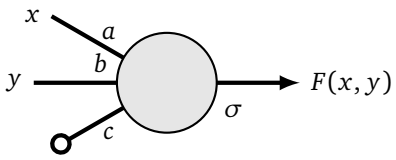
Exemple.
Voici les données (fictives) de la répartition des hommes et des femmes selon leur taille et leur poids.

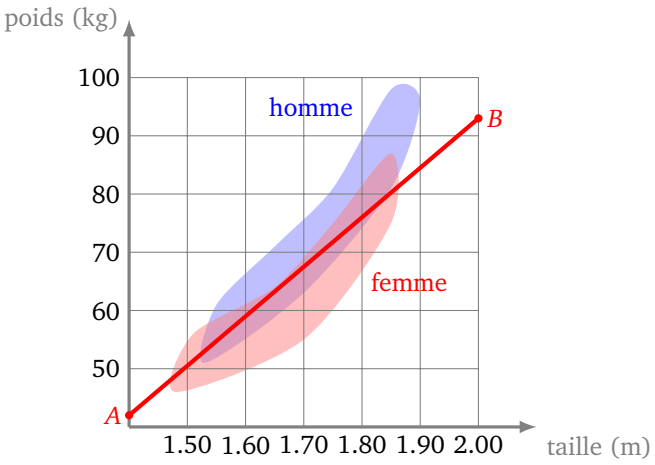


Problème : la taille et le poids d'une personne étant donnés, trouver un perceptron qui réponde à la question « Cette personne est-elle un homme ou une femme ? ».
Au vu de la superposition des zones, il n'est pas possible de répondre avec certitude. On construit donc un perceptron selon les idées suivantes :

- on trace une droite qui sépare au mieux les hommes des femmes, par exemple ici la droite qui passe par les points $A(1.40, 52)$ et $B(2.06, 93)$ d'équation approchée $y = 68x - 44$ où $(x, y) = (t, p)$ représente la taille et le poids ;
- on choisit la fonction d'activation sigmoïde.

Ce qui nous permet de définir le perceptron suivant avec $a = 68$, $b = -1$ et $c = -44$.





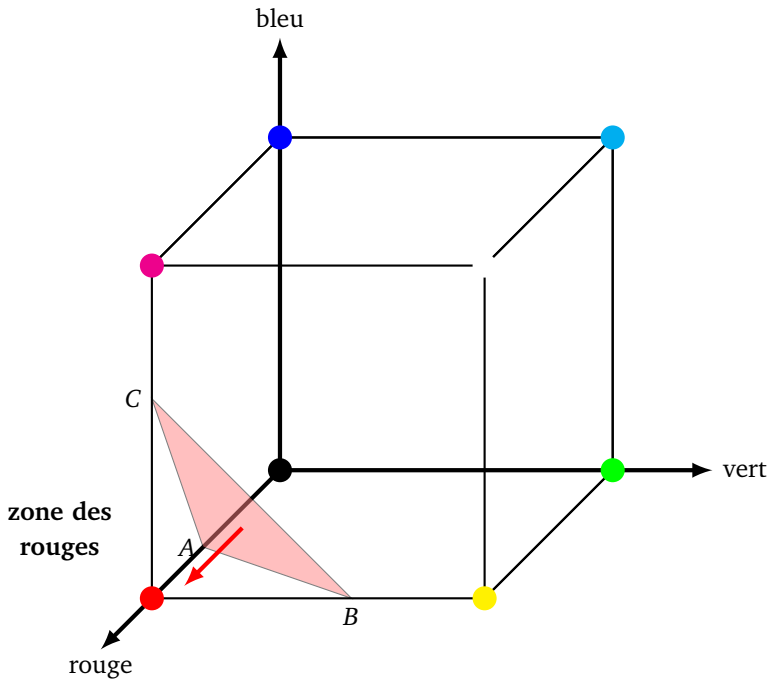
Maintenant pour un couple donné (t, p), le perceptron associe une valeur $F(t, p) \in [0, 1]$. Si $F(t, p)$ est proche de 0 alors la personne est probablement un homme, si $F(t, p)$ est proche de 1 c'est probablement une femme.
Exemple : une personne mesurant 1.77 m et pesant 79 kg est-elle plutôt un homme ou une femme ? On calcule $f(1.77, 79) = -2.64$ où $f(x, y) = ax + by + c$. Puis on calcule $F(1.77, 79) = \sigma(-2.64) \simeq 0.07$. Selon notre modèle cette personne est probablement un homme (car la valeur de F est proche de 0).
Autre exemple : $(t, p) = (1.65, 67)$. On calcule $F(1.65, 67) \simeq 0.76$. Une personne mesurant 1.65m et pesant 67kg est probablement une femme (car la valeur de F est plus proche de 1 que de 0).

Malgré les données fictives, cet exemple met en évidence le problème de la superposition et de l'utilité d'avoir une sortie plus riche que « oui » ou « non ». On peut aussi discuter de la pertinence de la frontière, car la séparation par une droite ne semble pas la mieux adaptée. En fait, le poids varie en fonction du carré de la taille (les zones ont une forme de banane).

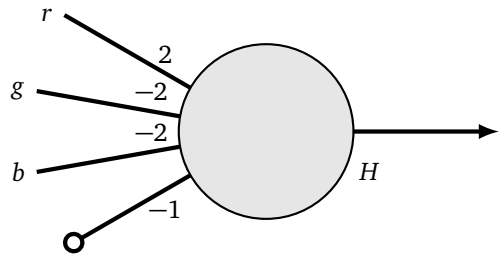
Plus généralement un ***perceptron affine*** à n entrées est la donnée de $n+1$ poids $a_0, a_1, \ldots, a_n \in \mathbb{R}$ et d'une fonction d'activation ϕ qui définissent une fonction $F : \mathbb{R}^n \to \mathbb{R}$ par la formule :

$$F(x_1, \ldots, x_n) = \phi(a_1 x_1 + \cdots + a_n x_n + a_0).$$

Exemple.
Dans le système de couleurs RGB (*red, green, blue*), une couleur est déterminée par trois réels $r, g, b \in [0, 1]$. Le « vrai » rouge est codé par $(1, 0, 0)$, mais bien sûr des couleurs proches sont aussi des nuances de rouge. On représente toutes les couleurs par un « cube de couleurs », chaque point du cube représente le code (r, g, b) d'une couleur.



On décide (un peu arbitrairement) que toute couleur située dans la zone du coin $(1, 0, 0)$ de la figure ci-dessus est une nuance de rouge.
Problème : trouver un perceptron qui répond à la question « Cette couleur est-elle une nuance de rouge ? ».
Solution. On considère que la zone est délimitée par le plan passant par les points $A(\frac{1}{2}, 0, 0)$, $B(1, \frac{1}{2}, 0)$ et $C(1, 0, \frac{1}{2})$ et les plans des faces du cube. Une équation de ce plan est $2x - 2y - 2z - 1 = 0$ où en fait $(x, y, z) = (r, g, b)$. Le perceptron qui répond à la question est donc :

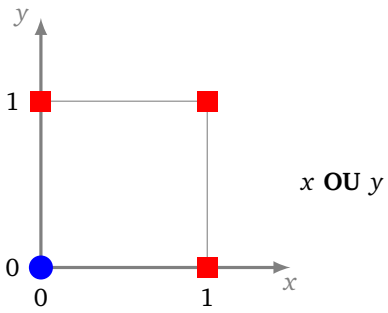


> La fonction F associée, vérifie que $F(r,g,b)=1$ pour les points de la zone des rouges et $F(r,g,b)=0$ sinon.

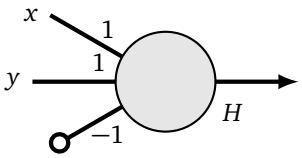
2. Théorie du perceptron

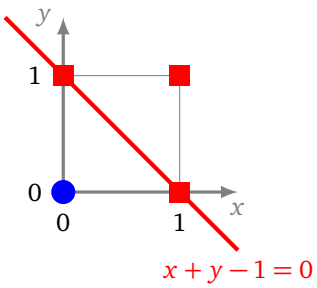
2.1. OU, ET, OU exclusif

OU. Un ***booléen*** est une variable qui peut prendre soit la valeur « vrai », soit la valeur « faux ». Dans la pratique, on associe 1 à « vrai » et 0 à « faux ». À partir de deux booléens x et y, on peut associer un nouveau booléen « x OU y » qui est vrai lorsque x est vrai ou y est vrai.
Graphiquement, on représente toutes les configurations associées à « x OU y » par un diagramme. Un rond bleu en position (x,y) signifie que « x OU y » est faux (le résultat vaut 0), un carré rouge que « x OU y » est vrai (le résultat vaut 1).

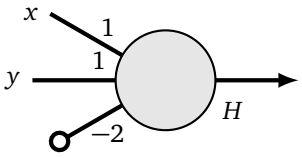


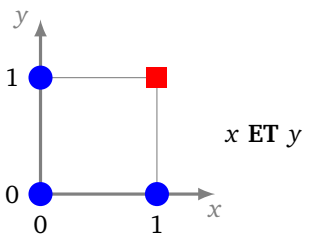
Peut-on réaliser l'opération « x OU y » par un perceptron ? Oui ! Cela revient à séparer les rond bleus des carrés rouges. C'est possible avec le perceptron de poids $a=1$, $b=1$, $c=-1$. Par exemple, si $x=0$ et $y=1$ alors, on calcul $1\cdot 0+1\cdot 1-1=0\geqslant 0$. Composée avec la fonction marche de Heaviside, la fonction $F(x,y)$ définie par le perceptron renvoie dans ce cas 1 (« vrai »). Si $x=0$ et $y=0$, alors $1\cdot 0+1\cdot 0-1=-1<0$ et $F(x,y)=0$ (« faux »).





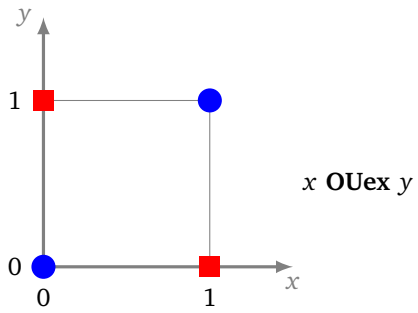
ET. On peut également réaliser l'opération « x ET y » (qui renvoie « vrai » uniquement si x et y sont vrais) en choisissant les poids $a=1$, $b=1$, $c=-2$.





Remarque.
D'un point de vue numérique, si on considère des valeurs réelles pour x et y, notre perceptron « ET » n'est pas numériquement très stable. Par exemple avec $x = 0.9$ et $y = 0.9$, on calcule $x + y - 2 = -0.2$ et la sortie est 0, mais comme x et y sont proches de 1, on aimerait que la sortie soit 1. Il suffit de changer un peu les paramètres : prenons $a = 1$, $b = 1$ et $c = -1.5$, alors $x + y - 1.5 = 0.3$ et cette fois la sortie est 1.

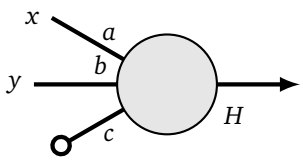
OU exclusif. Le ou exclusif, noté « x OUex y » est vrai lorsque x ou y est vrai, mais pas les deux en même temps. Le ou exclusif est-il réalisable par un perceptron ? Cette fois la réponse est non !

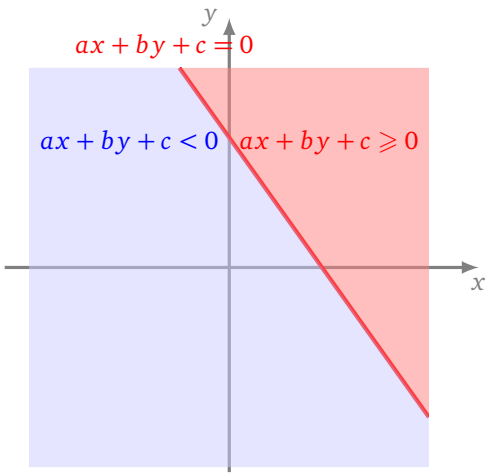


Proposition 1.
Il n'existe pas de perceptron (affine, à deux entrées et de fonction d'activation la fonction marche de Heaviside) qui réalise le « ou exclusif ».

Nous sommes convaincus géométriquement qu'il n'existe pas de droite qui sépare les ronds bleus des carrés rouges. Nous allons le prouver par un petit calcul.

Démonstration. Nous raisonnons par l'absurde en supposant qu'un tel perceptron existe. Nous cherchons une contradiction. Soit $a_1 = a, a_2 = b, a_0 = c$ les coefficients.





- Pour $(x, y) = (1, 0)$, on doit avoir $ax + by + c \geqslant 0$ (car après avoir composé avec la fonction d'activation le résultat doit être 1), donc
$$a + c \geqslant 0. \tag{1}$$
- De même pour $(x, y) = (0, 1)$, on doit avoir $ax + by + c \geqslant 0$, ce qui donne :
$$b + c \geqslant 0. \tag{2}$$
- Pour $(0, 0)$ et $(1, 1)$, on a $ax + by + c < 0$, ce qui implique :
$$c < 0, \tag{3}$$
$$a + b + c < 0. \tag{4}$$

Si on additionne les inégalités (1) et (2), on obtient $a+b+2c \geqslant 0$. Par l'inégalité (3) on a $-c > 0$. Donc en ajoutant $-c$ à l'inégalité $a+b+2c \geqslant 0$, on obtient $a+b+c > 0$. Ce qui contredit l'inégalité (4).
Conclusion : il ne peut exister trois réels a, b, c pour définir un perceptron réalisant le « ou exclusif ».

□

2.2. Séparation linéaire

Nous formalisons un peu les idées de la section précédente. Une ***droite*** du plan $\mathbb{R}^2$ est l'ensemble des points (x, y) vérifiant une équation du type $ax + by + c = 0$. Un ***plan*** de l'espace $\mathbb{R}^3$ est l'ensemble des points (x, y, z) vérifiant une équation du type $ax + by + cz + d = 0$. Nous généralisons cette notion en dimension plus grande.

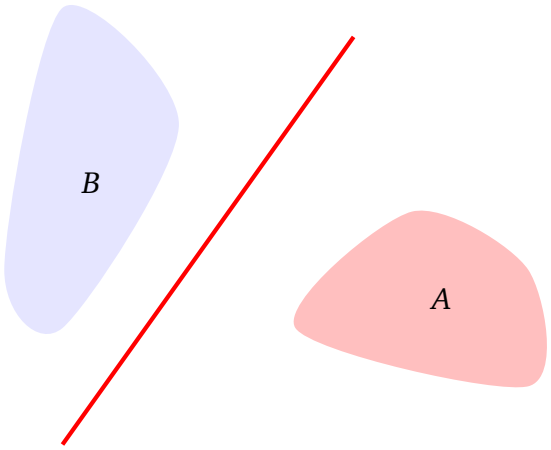
Définition.
Un ***hyperplan (affine)*** de $\mathbb{R}^n$ est l'ensemble des points $(x_1, \ldots, x_n)$ qui vérifient une équation du type :

$$a_1x_1 + a_2x_2 + \cdots + a_nx_n + a_0 = 0$$

où $a_1, \ldots, a_n$ sont des réels (non tous nuls) et a_0 est aussi un réel.

Le but d'un perceptron affine est de séparer deux ensembles. Par exemple deux ensembles A et B sont séparés par une droite, deux ensembles de l'espace sont séparés par un plan.

Définition.
Deux ensembles A et B sont ***linéairement séparables*** s'il existe un hyperplan qui sépare A de B. Plus précisément, s'il existe des réels $a_0, a_1, \ldots, a_n$ tels que $a_1x_1+a_2x_2+\cdots+a_nx_n+a_0 \geqslant 0$ pour tout $(x_1, \ldots, x_n) \in A$ et $a_1x_1 + a_2x_2 + \cdots + a_nx_n + a_0 < 0$ pour tout $(x_1, \ldots, x_n) \in B$.



Nous résumons ce qui précède dans le résultat suivant.

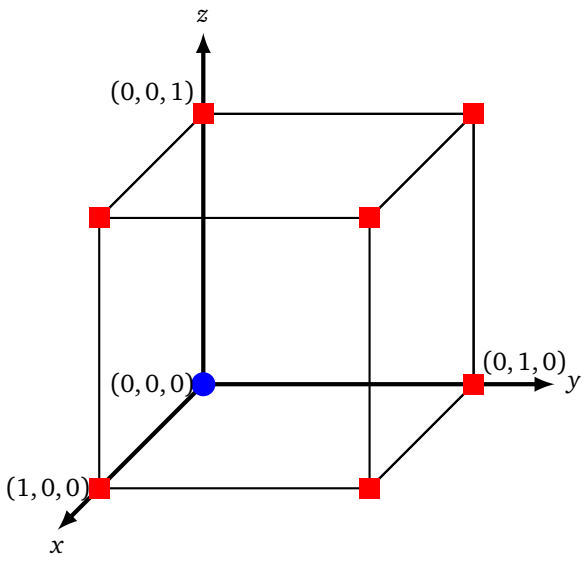
Proposition 2.
Deux ensembles A et B de $\mathbb{R}^n$ sont linéairement séparables si, et seulement si, il existe un perceptron affine dont la fonction F vaut 1 sur A et 0 sur B.

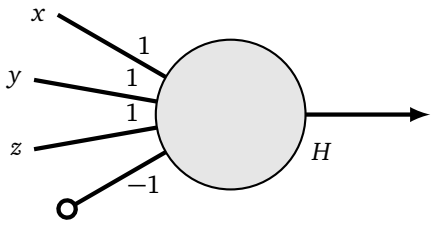
La preuve est simple : la fonction f définie par un perceptron (avant activation) est une fonction affine :

$$f(x_1, \ldots, x_n) = a_1x_1 + a_2x_2 + \cdots + a_nx_n + a_0.$$

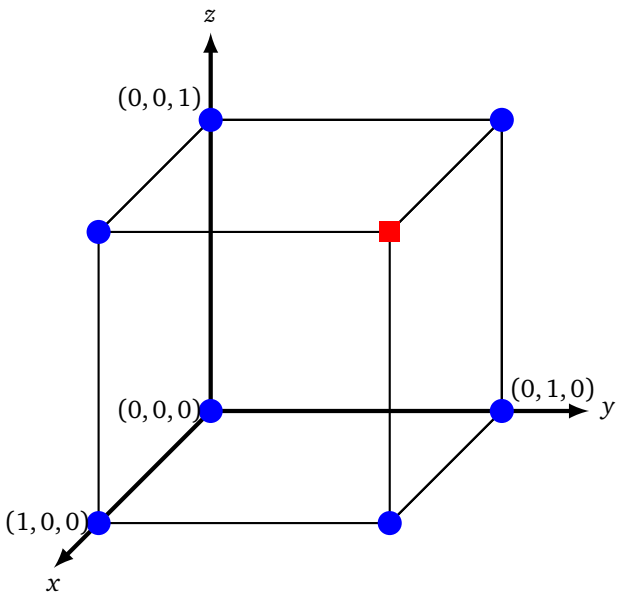
Si $f(x_1, \ldots, x_n) \geqslant 0$, alors après activation $F(x_1, \ldots, x_n) = 1$, sinon $F(x_1, \ldots, x_n) = 0$.

> **Exemple.**
> Réaliser « x OU y OU z » revient à séparer par un plan le rond bleu des carrés rouges du cube suivant. Trouver l'équation d'un de ces plans donne les poids du perceptron qui conviennent.





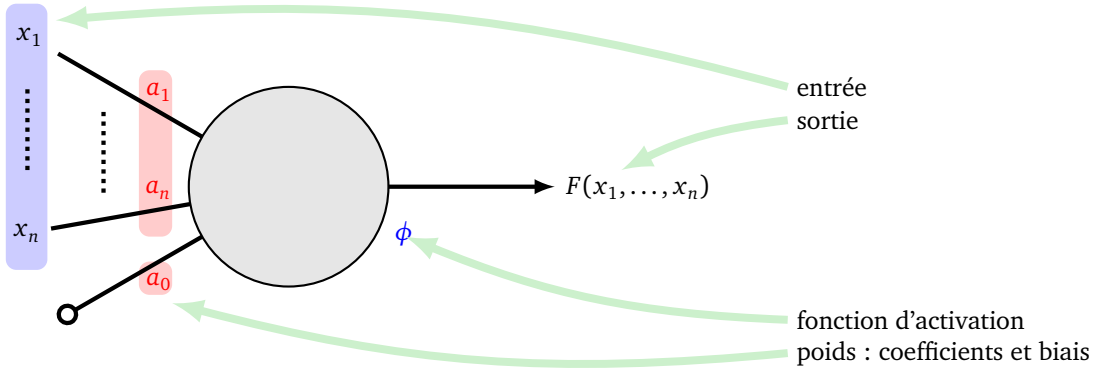
Exercice.
Trouver le perceptron qui réalise « x ET y ET z ».



2.3. Vocabulaire

Résumons le vocabulaire utilisé ci-dessus ainsi que les termes anglais correspondant :

- le **perceptron linéaire** ou **neurone artificiel**, a un biais qui vaut 0, à la différence du **perceptron affine** pour lequel le biais est un réel quelconque ;
- les **poids** (*weights*) ou **paramètres** sont les **coefficients** $a_1, \dots, a_n$ auxquels s'ajoute le **biais** (*bias*) (dans ce livre le biais est l'un des poids, ce qui n'est pas toujours le cas dans la littérature) ;
- un perceptron est la donnée des poids et d'une fonction d'activation ;
- chaque perceptron définit une fonction F qui est la composée d'une **fonction affine** f et d'une **fonction d'activation** ϕ ; la fonction d'activation la plus utilisée dans ce chapitre est la fonction marche de Heaviside (*step function*) ;
- on utilise un perceptron lors d'une **évaluation** : pour une **entrée** (*input*) $(x_1, \dots, x_n) \in \mathbb{R}^n$, on calcule la sortie (*output*) $F(x_1, \dots, x_n) \in \mathbb{R}$ (qui vaut 0 ou 1 dans le cas de la fonction marche de Heaviside).

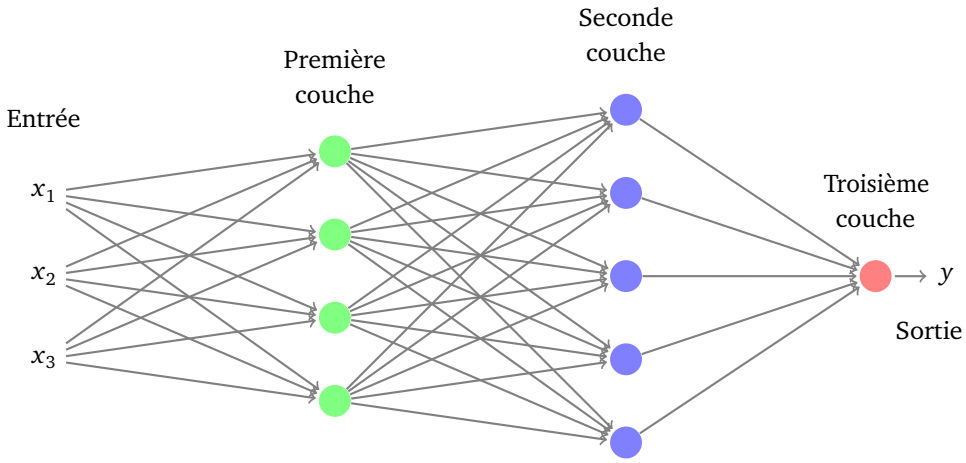


3. Réseau de neurones

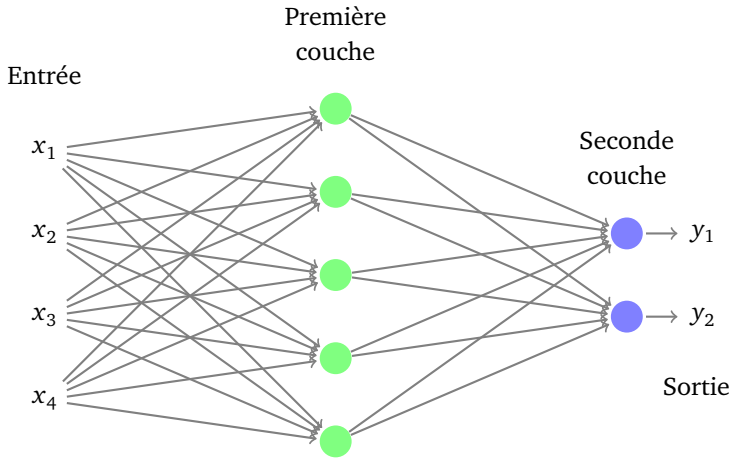
Un neurone permet de séparer l'espace en deux parties, la frontière étant linéaire (avec deux entrées c'est une droite, avec trois entrées c'est un plan…). Un neurone est trop élémentaire pour résoudre nos problèmes, par exemple il ne peut réaliser le « ou exclusif ». Comment faire mieux ? En connectant plusieurs neurones !

3.1. Couches de neurones

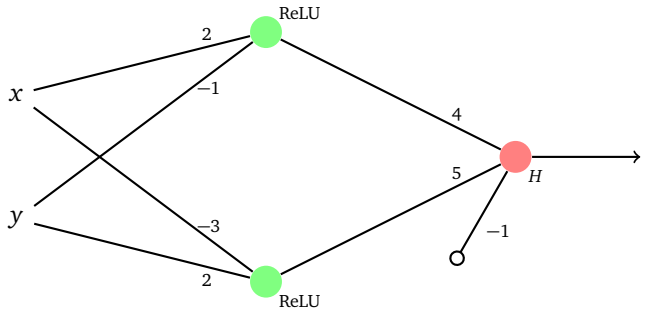
Un ***réseau de neurones*** est la juxtaposition de plusieurs neurones, regroupés par ***couches***.



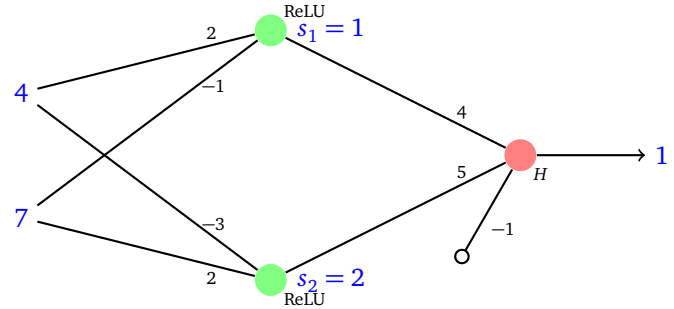
À un réseau de neurones on associe une fonction. Si à la dernière couche la fonction ne contient qu'un seul neurone (voir ci-dessus), cette fonction est $F : \mathbb{R}^n \to \mathbb{R}$, $y = F(x_1,\dots,x_n)$ où $(x_1,\dots,x_n)$ est l'***entrée*** et y est la ***sortie***. Sinon (voir ci-dessous), la fonction est $F : \mathbb{R}^n \to \mathbb{R}^p$, $(y_1,\dots,y_p) = F(x_1,\dots,x_n)$ où $(x_1,\dots,x_n) \in \mathbb{R}^n$ est l'entrée et $(y_1,\dots,y_p)$ est la sortie.



Exemple.
Voici un réseau de neurones à deux couches : 2 neurones (perceptrons linéaires) sur la première couche (ayant pour fonction d'activation la fonction ReLU), 1 neurone (perceptron affine) sur la seconde couche (de fonction d'activation H).



- Si $x = 4$ et $y = 7$ alors on calcule la sortie de chaque neurone de la première couche. Ces sorties sont les entrées du neurone de la seconde couche. Pour le premier neurone, on effectue le calcul $2 \cdot 4 + (-1) \cdot 7 = 1 \geqslant 0$, le réel étant positif la fonction ReLU le garde inchangé : le premier neurone renvoie la valeur $s_1 = 1$. Le second neurone effectue le calcul $(-3) \cdot 4 + 2 \cdot 7 = 2 \geqslant 0$ et renvoie $s_2 = 2$. Le neurone de la couche de sortie reçoit en entrées s_1 et s_2 et effectue le calcul $4 \cdot 1 + 5 \cdot 2 - 1 = 13 \geqslant 0$. La fonction d'activation étant H, ce neurone renvoie 1. Ainsi $F(4,7) = 1$.

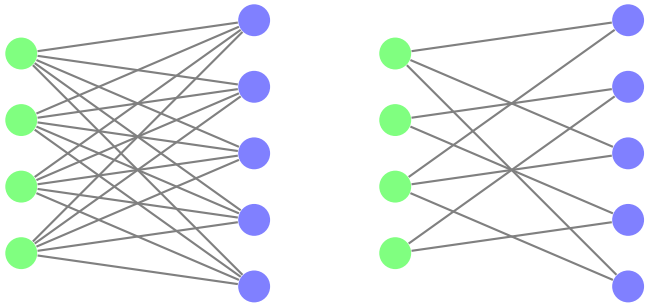


- Si $(x, y) = (3, 2)$ alors $s_1 = 4$, par contre $s_2 = 0$ car $(-3) \cdot 3 + 2 \cdot 2 = -5 < 0$ et la fonction ReLU renvoie 0 (on dit que le neurone ne s'active pas). Les entrées du dernier neurone sont donc $s_1 = 4$ et $s_2 = 0$, ce neurone calcule $4 \cdot 4 + 5 \cdot 0 - 1 = 15 \geqslant 0$ et après la fonction d'activation renvoie 1. Donc $F(3,2) = 1$.
- Vérifier que pour $(x, y) = (\frac{1}{10}, \frac{1}{10})$ le premier neurone s'active, le second ne s'active pas (il renvoie 0) et le dernier neurone renvoie 0. Ainsi $F(\frac{1}{10}, \frac{1}{10}) = 0$.

Remarque.

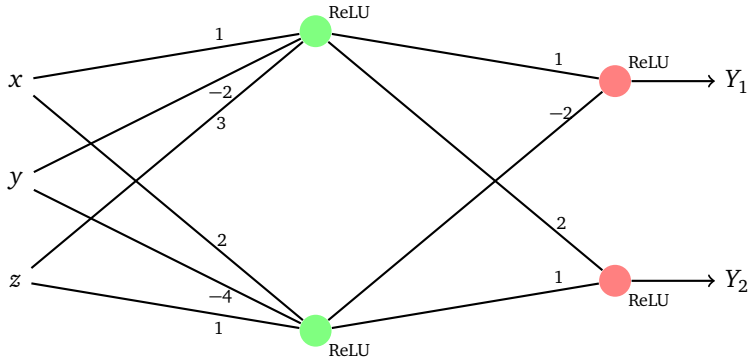
- Chaque neurone est défini par ses poids et une fonction d'activation. Pour une couche donnée, on choisit toujours la même fonction d'activation. En général on choisit la même fonction d'activation pour tout le réseau, sauf peut-être pour la couche de sortie.
- Un neurone prend plusieurs valeurs en entrée mais ne renvoie qu'une seule valeur en sortie ! Si ce neurone est connecté à plusieurs autres neurones, il envoie la même valeur à tous.

- On peut relier un neurone à tous les neurones de la couche suivante (voir ci-dessous figure de gauche). On dit que la seconde couche est ***dense*** ou ***complètement connectée***. Mais on peut aussi choisir de ne relier que certains neurones entre eux (voir ci-dessous figure de droite). S'il n'y a pas d'arêtes du neurone A vers le neurone B, c'est que la sortie de A n'intervient pas comme entrée de B (cela revient à imposer un poids nul entre ces deux neurones).



Exercice.
Pour le réseau de neurones représenté ci-dessous, calculer les valeurs de sortie (Y_1, Y_2) pour chacune des entrées (x, y, z) suivantes :

$$(0,0,0) \quad (1,0,0) \quad (1,-1,1) \quad (3,2,1)$$



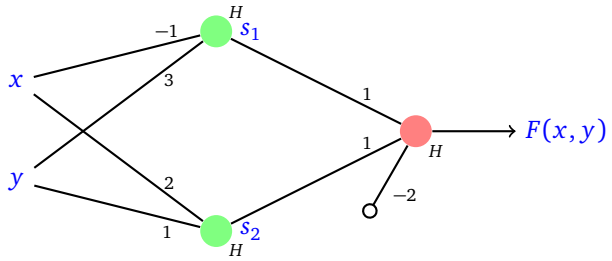
Trouver (x, y, z) tel que Y_2 soit nul, mais pas Y_1.
Trouver (x, y, z) tel que $Y_1 = 1$ et $Y_2 = 7$ (commencer par déterminer les valeurs de sorties s_1 et s_2 de la première couche).

3.2. Exemples

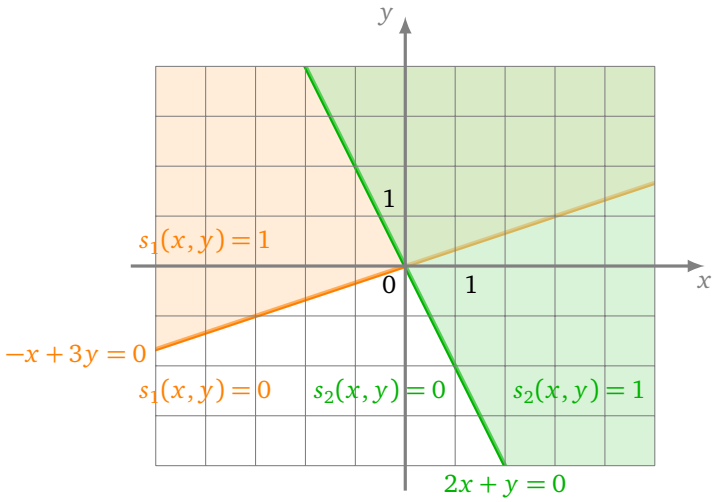
Étudions quelques exemples plus en détails. Au lieu de calculer la sortie $F(x, y)$ pour des valeurs données, nous allons calculer $F(x, y)$ quelque soit (x, y).

> **Exemple.**
> Voici un réseau de 3 neurones : deux sur la première couche et un sur la seconde. La fonction d'activation

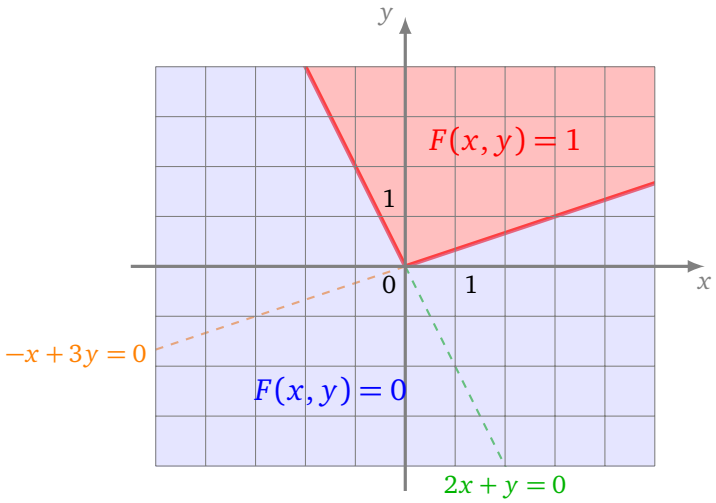
est partout la fonction marche de Heaviside. Combien vaut la fonction associée F selon l'entrée (x, y) ?



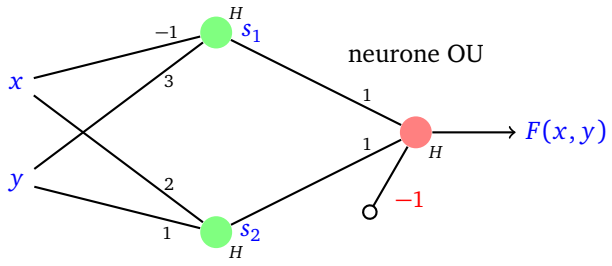
On commence par calculer les sorties des neurones de la première couche. Pour le premier neurone la sortie s_1 dépend du signe de $-x+3y$. Si $-x+3y \geqslant 0$ alors $s_1(x,y)=1$, sinon $s_1(x,y)=0$. Donc pour les points (x,y) situés au-dessus de la droite d'équation $-x+3y=0$, on a $s_1(x,y)=1$. De même pour le second neurone, on a $s_2(x,y)=1$ pour les points situés au dessus de la droite $2x+y=0$. Voir la figure ci-dessous.



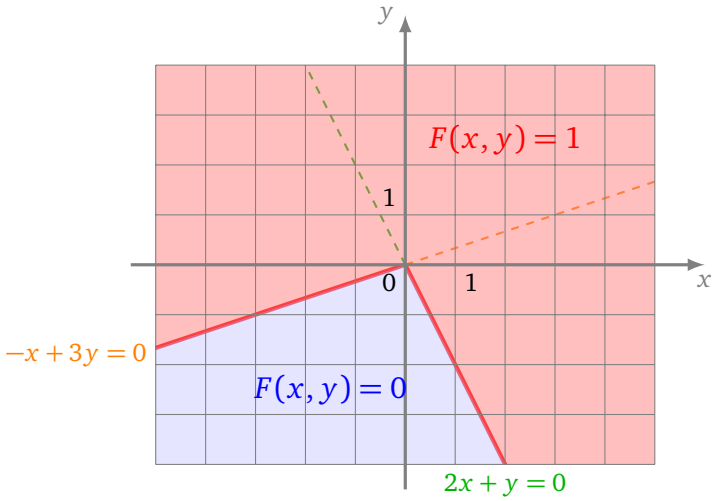
On reconnaît dans le neurone de sortie un neurone qui réalise le « ET ». C'est pourquoi l'ensemble des points pour lesquels F vaut 1 est l'intersection des deux demi-plans en lesquels s_1 et s_2 valent 1. Ainsi $F(x,y)=1$ dans un secteur angulaire et $F(x,y)=0$ ailleurs. Voir la figure ci-dessous.



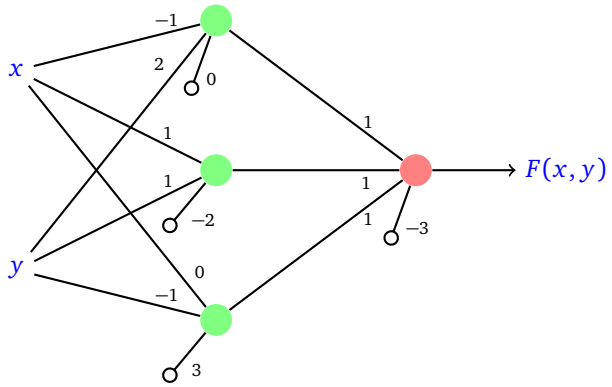
Exemple.
On reprend la même architecture que l'exemple précédent, mais en changeant les poids du neurone de sortie qui réalise cette fois l'opération « OU ».



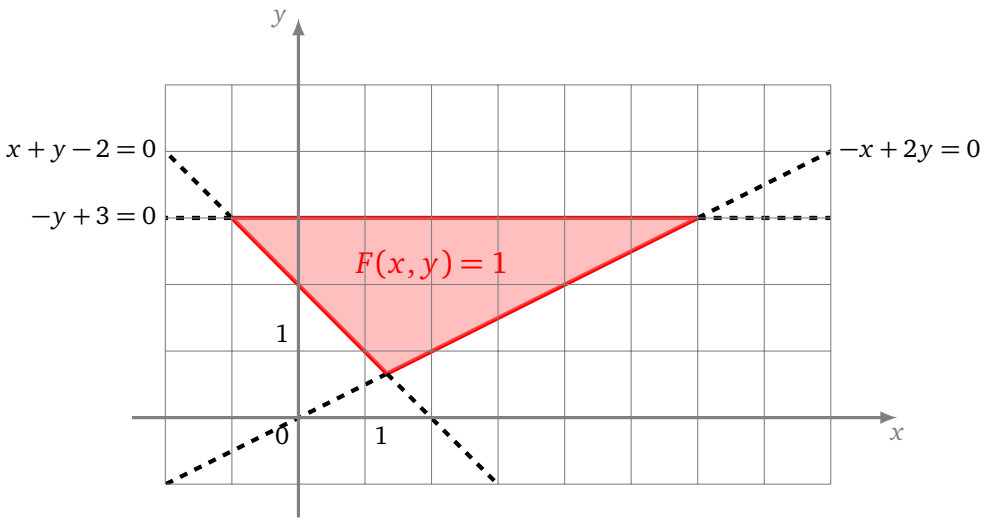
L'ensemble des points pour lesquels F vaut 1 est maintenant l'union des deux demi-plans en lesquels s_1 et s_2 valent 1.



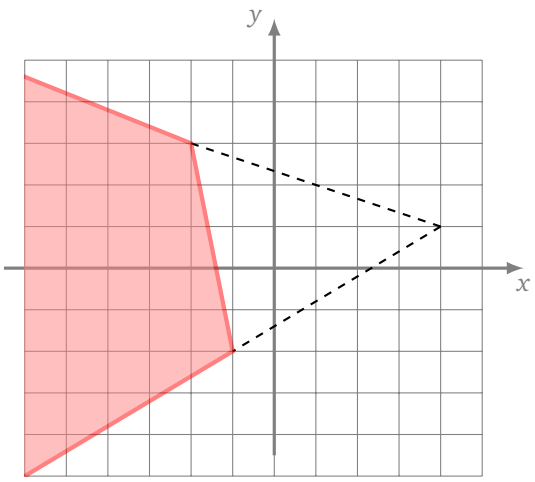
Exemple.
Voici un réseau de neurones un peu plus compliqué (la fonction d'activation est H partout).



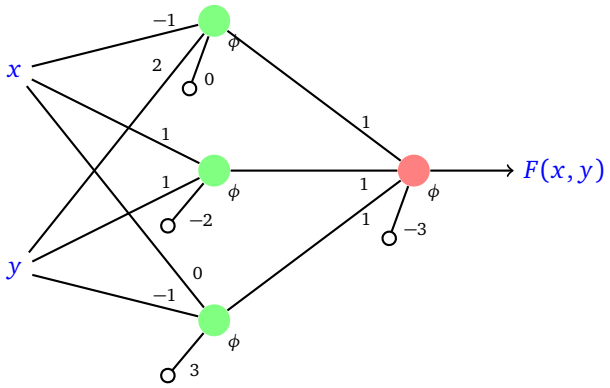
Chaque neurone de la première couche délimite un demi-plan. Ce sont les demi-plans $-x+2y \geqslant 0$, $x+y-2 \geqslant 0$ et $-y+3 \geqslant 0$.
Le neurone de sortie est un neurone qui réalise l'opération « ET » : il s'active uniquement si les trois précédents neurones sont activés. Ainsi la sortie finale $F(x,y)$ vaut 1 si et seulement si (x,y) appartient simultanément aux trois demi-plans.



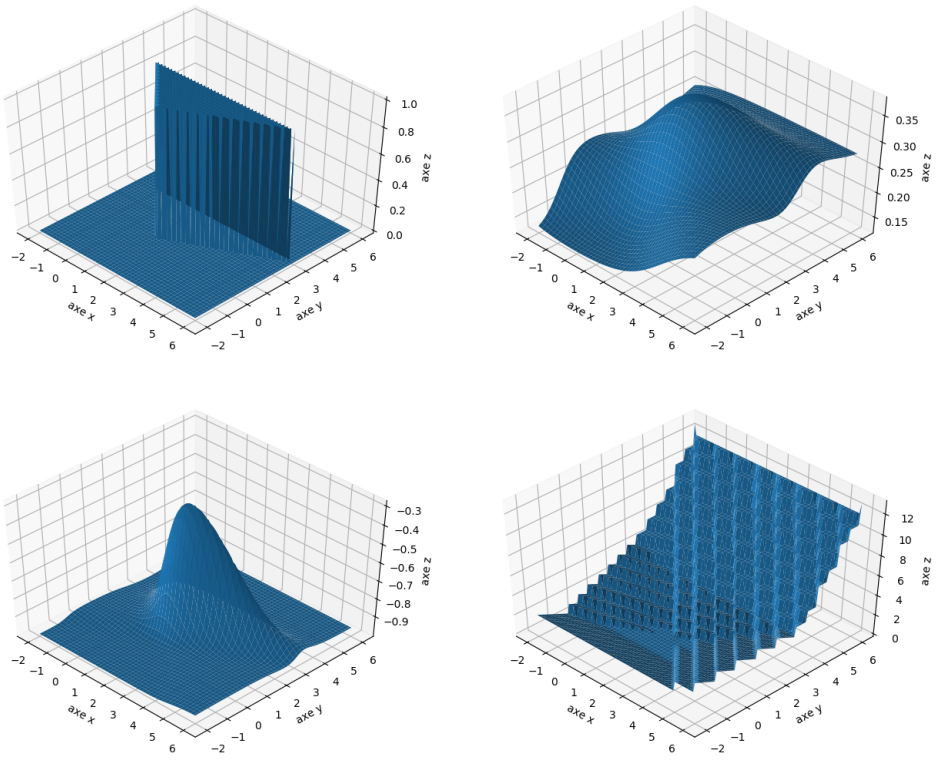
Exercice.
En utilisant les idées de l'exemple précédent et pour le dessin ci-dessous, trouver un réseau de neurones dont la fonction F vaut 1 pour la zone colorée et 0 ailleurs.



Exemple.
On termine en comparant les fonctions produites par des réseaux ayant la même architecture, les mêmes poids mais de fonction d'activation ϕ différente : H, σ, tanh et ReLU.



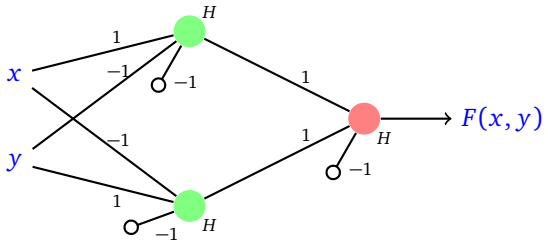
Voici les graphiques 3D obtenus pour les fonctions d'activation H (comme dans l'exemple vu auparavant), σ, tanh et ReLU. Les fonctions F obtenues dépendent fortement du choix de la fonction d'activation.

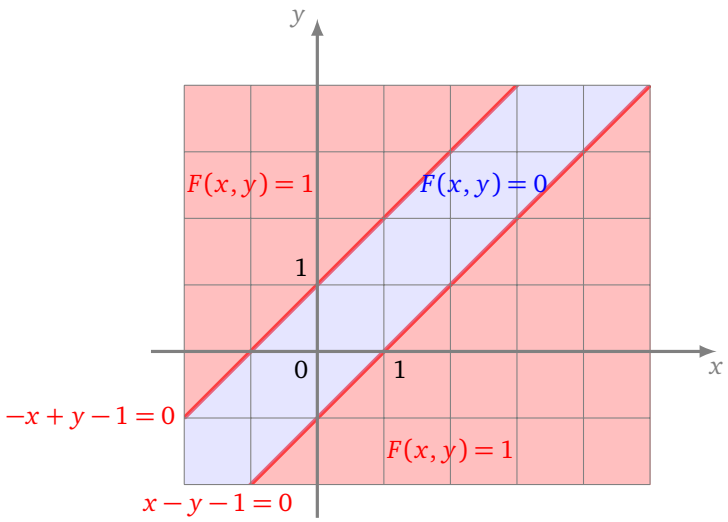


4. Théorie des réseaux de neurones

4.1. OU exclusif

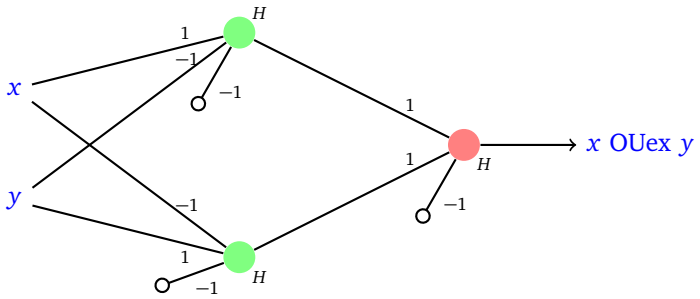
Nous avons vu qu'un seul neurone ne permet pas de réaliser la fonction associée au « OU exclusif ». Avec plusieurs neurones c'est facile ! Voici un réseau de neurones qui sépare le plan en trois parties, le secteur central en lequel la fonction associée au réseau vaut 0, alors que la fonction vaut 1 partout ailleurs (y compris sur la frontière rouge).

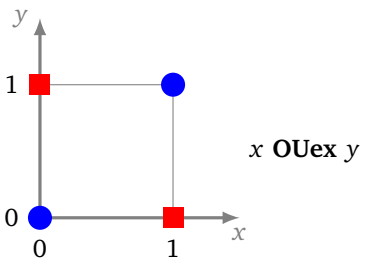




Ce réseau permet d'obtenir une valeur $F(x,y)=1$ en $(1,0)$ et $(0,1)$ et une valeur $F(x,y)=0$ en $(0,0)$ et $(1,1)$.
L'idée de ce réseau vient du fait que l'opération « OU exclusif » est une combinaison de « OU » et de « ET » :

$$x \text{ OUex } y = (x \text{ ET non} y) \text{ OU } (\text{non} x \text{ ET } y).$$



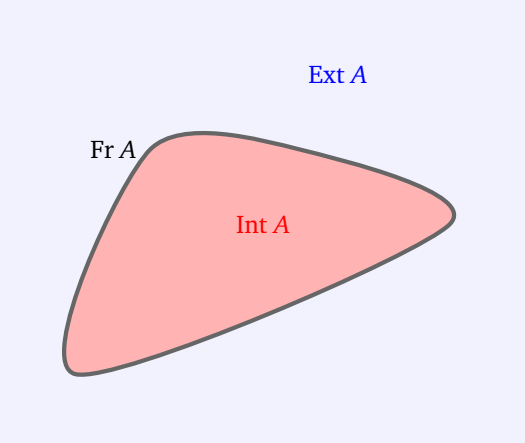


Le neurone du haut réalise « x ET non$y$ », le neurone du bas « non$x$ ET y » et celui de sortie l'opération « OU ».

4.2. Ensemble réalisable

Nous aimerions savoir quels ensembles peuvent être décrits par un réseau de neurones. On rappelle qu'un ensemble $A \subset \mathbb{R}^2$ découpe le plan en trois parties disjointes :

$$\text{Int}(A) \qquad \text{Fr}(A) \qquad \text{Ext}(A).$$

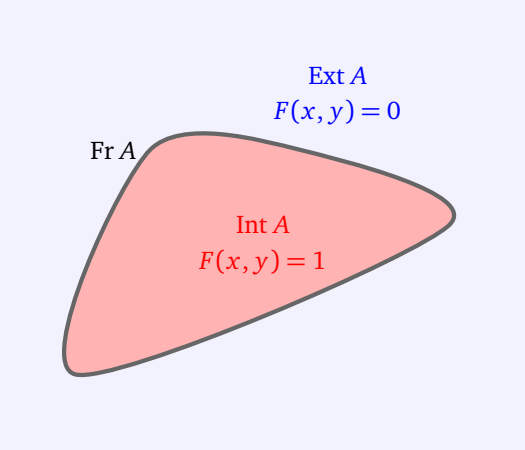


Définition.
Un ensemble A est dit ***réalisable par un réseau de neurones*** s'il existe un réseau de neurones $\mathcal{R}$ (d'unique fonction d'activation la fonction marche de Heaviside) tel que la fonction $F : \mathbb{R}^2 \to \mathbb{R}$ associé à $\mathcal{R}$ vérifie :

$$F(x, y) = 1 \quad \text{pour tout } (x, y) \in \text{Int}(A)$$

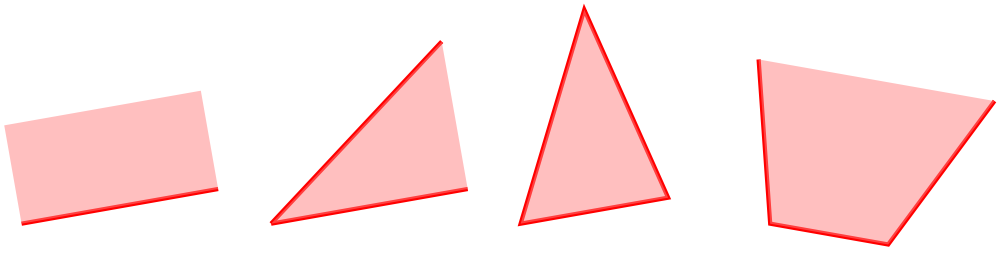
et

$$F(x, y) = 0 \quad \text{pour tout } (x, y) \in \text{Ext}(A).$$



Remarque : on n'exige rien sur la frontière Fr(A), F peut y prendre la valeur 0 ou la valeur 1.
Voici les types d'ensembles que nous avons déjà réalisés :

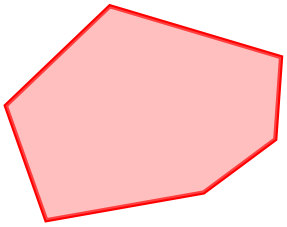
- les demi-plans,
- les « quart de plans »,
- les zones triangulaires,
- les triangles avec un sommet « à l'infini ».



Demi-plan **Quart de plan** **Triangle** **Triangle infini**

En augmentant le nombre de neurones, on peut réaliser les quadrilatères convexes et plus généralement n'importe quelle zone polygonale convexe.

Proposition 3.
Tout zone polygonale convexe à n côtés est réalisable par un réseau de $n + 1$ neurones.



Polygone convexe

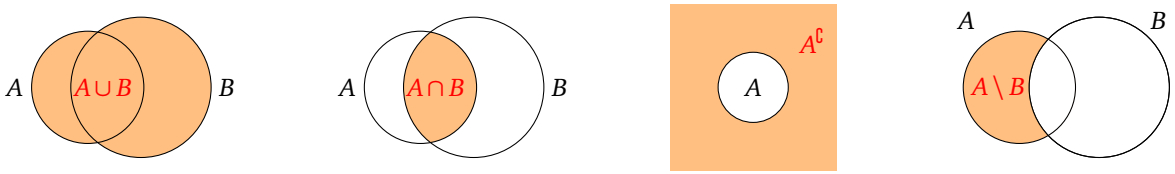
Démonstration. Un polygone convexe à n côtés est l'intersection de n demi-plans. Chaque demi-plan, bordé par une droite d'équation du type $ax+by+c=0$, correspond à un neurone de la première couche dont les coefficients sont (a,b) et le biais est c (ou alors $(-a,-b)$ et $-c$). Sur la seconde couche, on place un neurone qui réalise l'opération « ET » sur les n entrées : tous ses coefficients sont 1 et le biais est $-n$. □

Continuons avec des opérations élémentaires sur les ensembles réalisables.

Proposition 4.
Si A et B sont deux ensembles du plan réalisables par des réseaux de neurones alors :

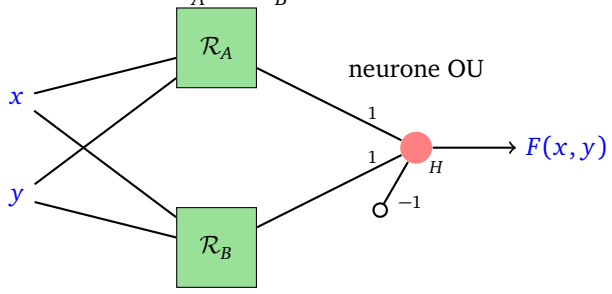
$$A\cup B \qquad A\cap B \qquad A^{\complement} \qquad A\setminus B$$

sont aussi des ensembles réalisables.

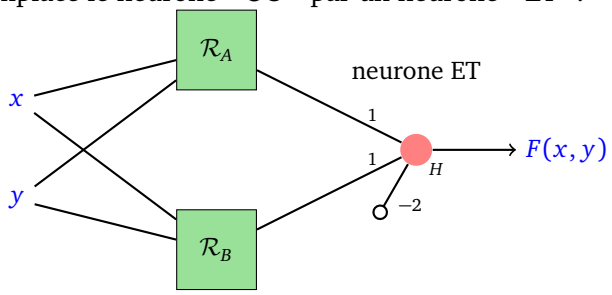


Démonstration.

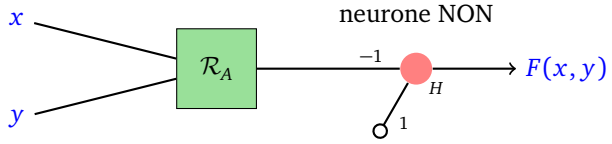
- Si A est réalisé par un réseau $\mathcal{R}_A$ et B par un réseau $\mathcal{R}_B$ alors on crée un nouveau réseau $\mathcal{R}$ en superposant $\mathcal{R}_A$ et $\mathcal{R}_B$ et en ajoutant un neurone « OU » à partir des sorties de $\mathcal{R}_A$ et $\mathcal{R}_B$. Ainsi si (x,y) est dans $A\cup B$, il est dans A ou dans B, une des sorties $\mathcal{R}_A$ ou $\mathcal{R}_B$ vaut alors 1 et le neurone sortant de $\mathcal{R}$ s'active.



- Pour réaliser $A\cap B$, on remplace le neurone « OU » par un neurone « ET ».



- Pour réaliser le complément d'un ensemble A, on utilise l'opération « non » : 0 s'envoie sur 1 et 1 sur 0. Ceci se fait par l'application $s\mapsto 1-s$. Il suffit juste de rajouter un neurone « NON » à $\mathcal{R}_A$.

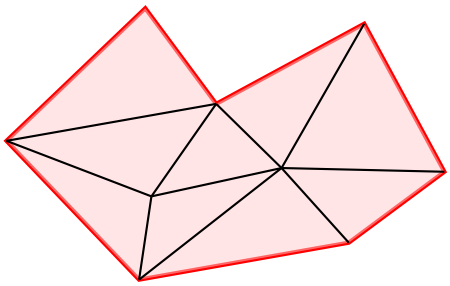


- Comme $A\setminus B = A\cap\left((A\cap B)^{\complement}\right)$ alors il est possible de réaliser $A\setminus B$ comme succession d'opérations élémentaires $\cap$, $\complement$ et $\cup$.

□

Proposition 5.
Tout polygone (convexe ou non) est réalisable par un réseau de neurones.

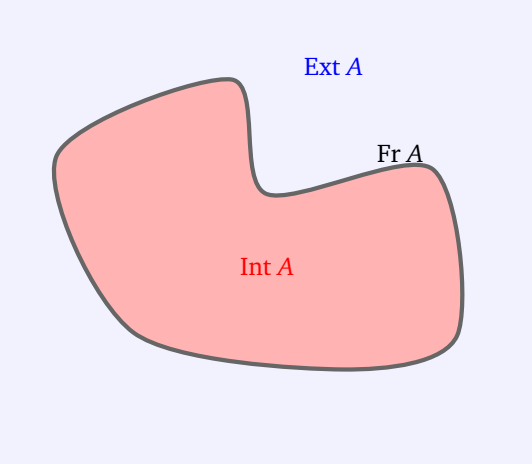
Démonstration. Un polygone peut être découpé en triangles. Chaque triangle est réalisable, donc l'union des triangles l'est aussi.



□

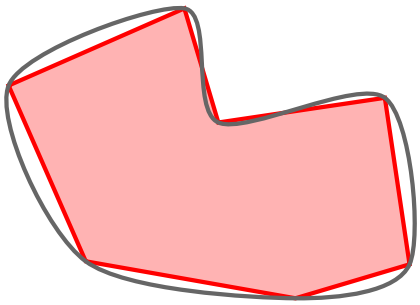
4.3. Approximation d'ensembles

On rappelle qu'une ***courbe de Jordan*** C est une courbe fermée simple (c'est l'image d'un cercle par une application continue injective). Le théorème suivant est une sorte de théorème d'approximation universelle géométrique.



Théorème 1.
Un ensemble A délimité par une courbe de Jordan peut être approché d'aussi près que l'on veut par un ensemble A' réalisable par un réseau de neurones.

Il s'agit juste du fait qu'une courbe de Jordan peut être approchée par un polygone. C'est un résultat théorique qui ne dit en rien comment choisir la structure du réseau ou les poids.



5. Théorème d'approximation universel

Nous allons maintenant prouver qu'un réseau de neurones bien construit peut approcher n'importe quelle fonction.
Dans cette section nous partirons d'une seule entrée $x \in \mathbb{R}$, avec une seule sortie $y = F(x) \in \mathbb{R}$. Les réseaux de neurones de cette section produisent donc des fonctions $F : \mathbb{R} \to \mathbb{R}$.
L'objectif est donc le suivant : on nous donne une fonction $f : [a, b] \to \mathbb{R}$ et nous devons trouver un réseau, tel que la fonction F associée à ce réseau soit proche de f :

$$F(x) \simeq f(x) \qquad \text{pour tout } x \in [a, b].$$

Pour paramétrer le réseau nous allons bien sûr fixer des poids, mais avant cela choisissons les fonctions d'activation :

- pour le neurone de sortie, on choisit la fonction identité $\phi(x) = x$,
- pour tous les autres neurones, on choisit la fonction marche de Heaviside.

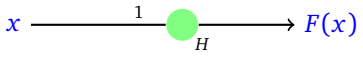
Remarque.

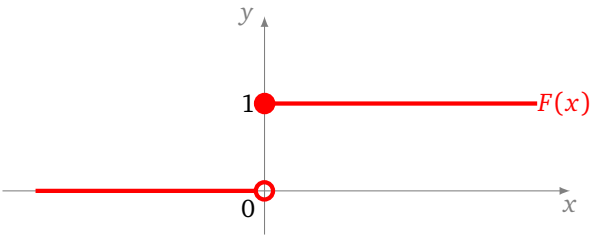
- Si on choisissait aussi la fonction d'activation marche de Heaviside pour le neurone de sortie, alors F ne pourrait prendre que deux valeurs, 0 ou 1, ce qui empêcherait de réaliser la plupart des fonctions.
- Par contre, si on choisissait la fonction identité pour tous les neurones alors on ne réaliserait que des fonctions affines $F(x) = ax + b$ (en effet la composition de plusieurs fonctions affines reste une fonction affine).

5.1. Fonctions marches

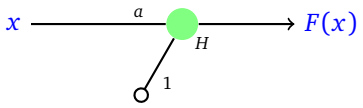
Nous allons réaliser des fonctions de plus en plus compliquées à partir d'éléments très simples.
Commençons par étudier le comportement d'un seul neurone avec la fonction d'activation marche de Heaviside (on rajoutera le neurone de sortie plus tard).
Voici différents neurones et les fonctions qu'ils réalisent :

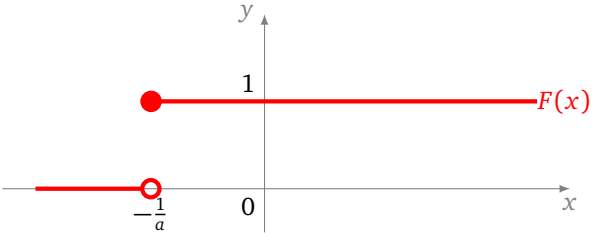
- La fonction marche de Heaviside.



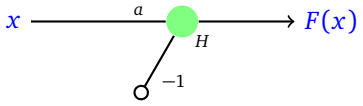


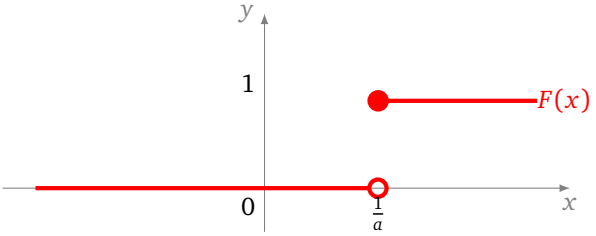
- La fonction marche décalée vers la gauche, avec $a > 0$.
  Preuve : $F(x) = 1 \iff ax + 1 \geqslant 0 \iff x \geqslant -\frac{1}{a}$.



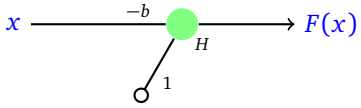


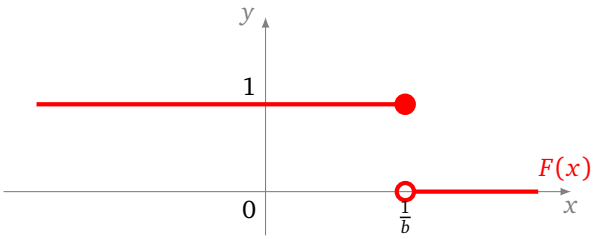
- La fonction marche décalée vers la droite, avec $a > 0$.



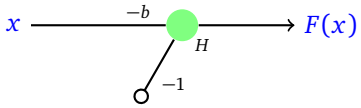


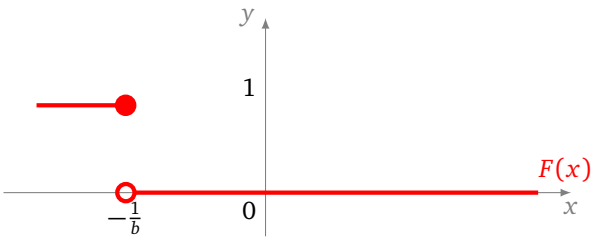
- La fonction marche à l'envers décalée vers la droite, avec $b > 0$.
  Preuve : $F(x) = 1 \iff -bx + 1 \geqslant 0 \iff bx \leqslant 1 \iff x \leqslant \frac{1}{b}$.



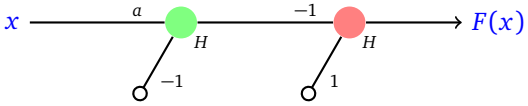


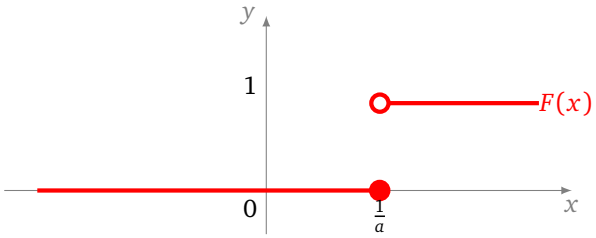
- La fonction marche à l'envers décalée vers la gauche, avec $b > 0$.





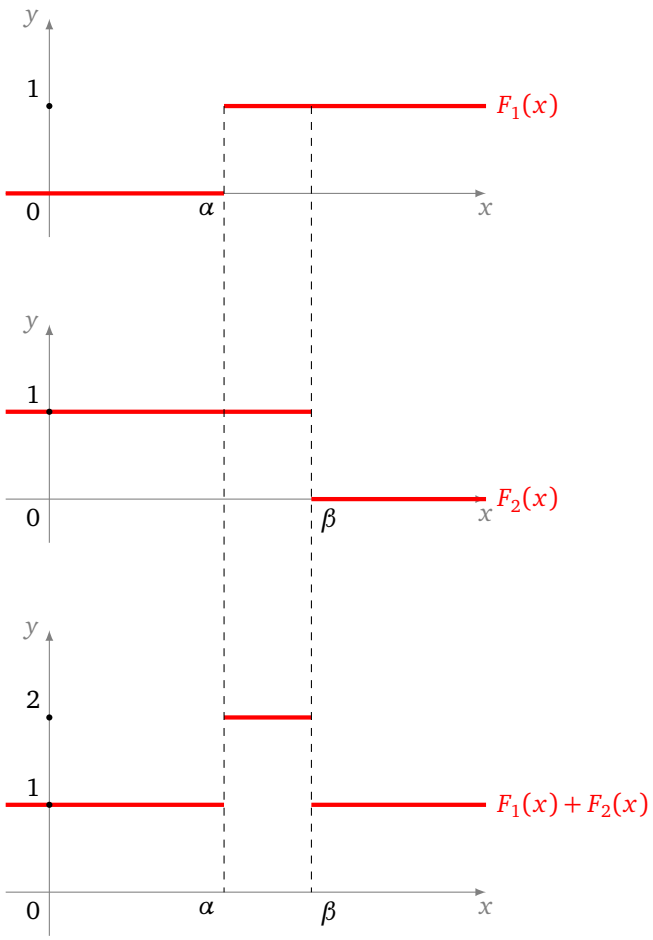
Selon les cas, la valeur au niveau de la marche est 0 ou 1. On peut obtenir l'autre situation en rajoutant un neurone de type « NON ».



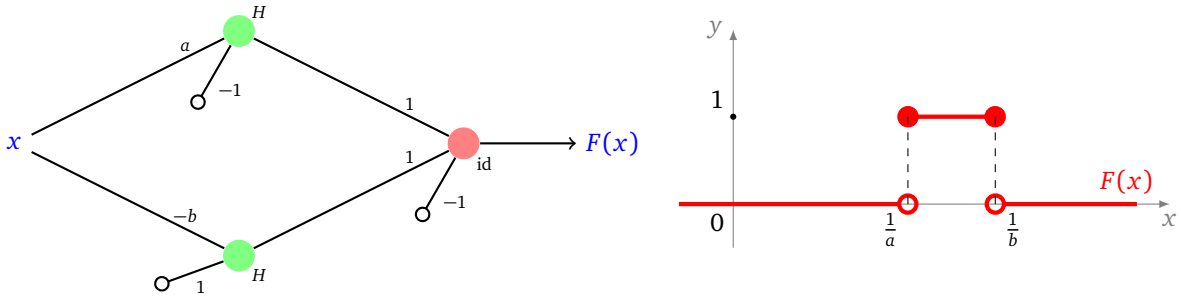


5.2. Fonctions créneaux

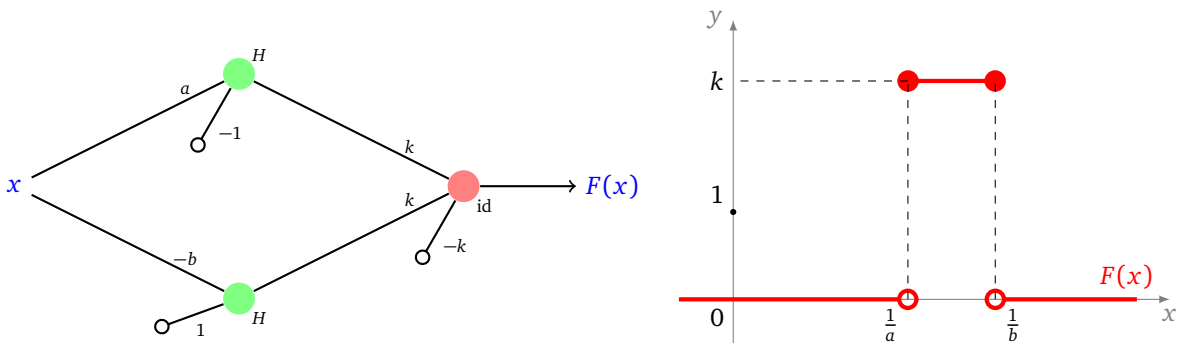
Pour réaliser un « créneau », l'idée est d'additionner une marche à l'endroit et une marche à l'envers.



Pour effectuer cette opération, nous allons construire un réseau avec deux neurones sur la première couche (de fonction d'activation H) et un neurone sur la seconde couche (de fonction d'activation identité) qui additionne les deux sorties précédentes et soustrait 1 (afin de ramener la ligne de base à 0).

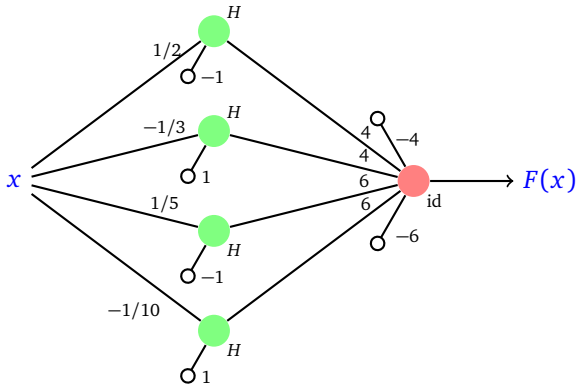


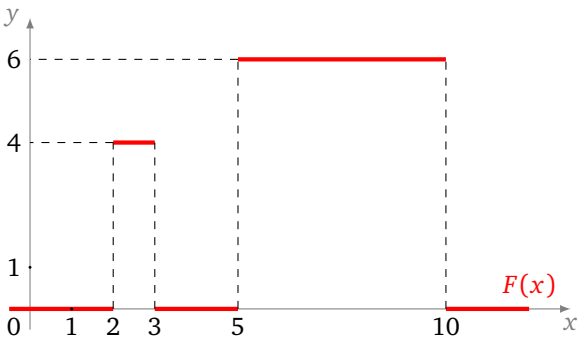
Si on veut une marche plus haute il suffit de changer les poids du neurone de sortie d'un facteur k.



5.3. Fonctions en escalier

On réalise des doubles créneaux en superposant les premières couches de chaque créneau et en réunissant les deux neurones de sortie. Voici un exemple avec un créneau de hauteur 4 sur $[2,3]$ et un créneau de hauteur 6 sur $[5,10]$.



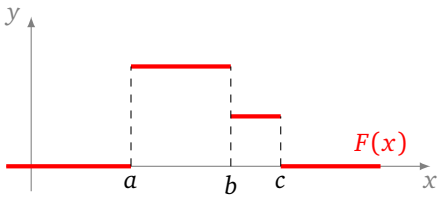


On peut aussi calculer la valeur de la fonction F de la façon suivante (s_i représente la sortie du neurone numéro i de la première couche) :

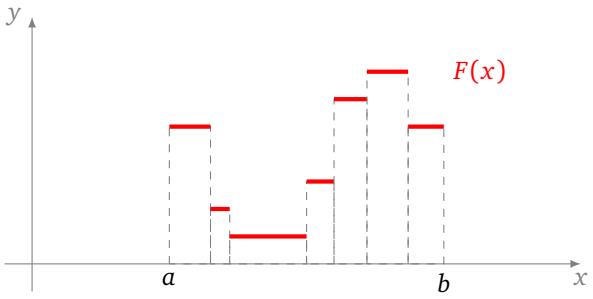
$$F(x) = \underbrace{4s_1 + 4s_2 - 4}_{\text{vaut 4 ou 0}} + \underbrace{6s_3 + 6s_4 - 6}_{\text{vaut 6 ou 0}} = \begin{cases} 4 & \text{si } x \in [2,3[\\ 6 & \text{si } x \in [5,10[\\ 0 & \text{sinon} \end{cases}$$

Remarque.

- Noter l'écriture avec deux biais -4 et -6 pour le neurone de sortie. C'est juste une écriture pour décomposer et expliquer le « vrai » biais qui est $-4-6=-10$.
- Il peut y avoir un problème pour réaliser deux créneaux contigus. Si on n'y prend pas garde, la valeur est mauvaise à la jonction (c'est la somme des deux valeurs). Pour corriger le problème, il faut utiliser une marche où on a changé la valeur au bord, voir la fin de la section 5.1.



Une ***fonction en escalier*** est une fonction qui est constante sur un nombre fini d'intervalles bornés.



Proposition 6.
Toute fonction en escalier est réalisable par un réseau de neurones.

Démonstration. Soit n le nombre de marches de l'escalier. On construit un réseau de $2n+1$ neurones. La première couche est constituée de n paires de neurones, chaque paire réalise une marche de l'escalier.

La seconde couche contient uniquement le neurone de sortie, les coefficients sont choisis pour ajuster la hauteur de la marche et le biais assure que la fonction vaut 0 en dehors de l'escalier.
Si on veut les valeurs exactes aux bornes des marches, il faut éventuellement ajouter des neurones de type « NON » entre la première couche et le neurone de sortie. □

5.4. Théorème d'approximation universel (une variable)

Nous pouvons maintenant énoncer le résultat théorique le plus important de ce chapitre.

Théorème 2 (Théorème d'approximation universel)**.**
Toute fonction continue $f : [a,b] \to \mathbb{R}$ *peut être approchée d'aussi près que l'on veut par une fonction* $F : [a,b] \to \mathbb{R}$ *réalisée par un réseau de neurones.*

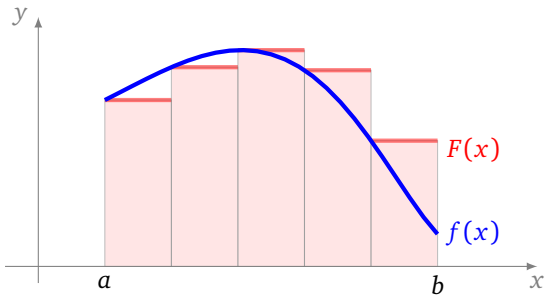
Plusieurs commentaires importants.

- Tout d'abord rappelons que nous réalisons nos neurones avec des fonctions d'activation H (marche de Heaviside) sauf pour le neurone de sortie qui a pour fonction d'activation l'identité.
- Les hypothèses sont importantes : f est continue et définie sur un intervalle borné. Si une des hypothèses venait à manquer l'énoncé serait faux.
- Bien que l'énoncé ne le dise pas, on peut concrètement réaliser à la main un réseau qui approche la fonction f (voir le chapitre suivant). Cependant, ce n'est pas l'esprit des réseaux de neurones.
- Que signifie « approcher d'aussi près que l'on veut la fonction f »? C'est dire que pour chaque $\epsilon > 0$, il existe une fonction en escalier F, telle que :

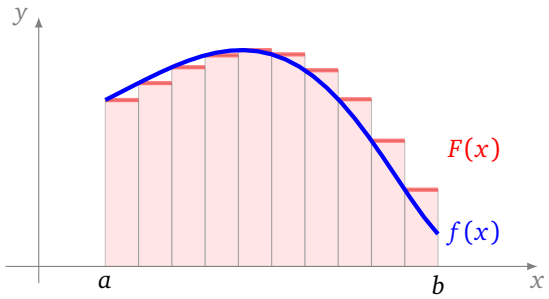
$$\text{pour tout } x \in [a,b] \qquad |f(x) - F(x)| < \epsilon.$$

C'est l'***approximation uniforme*** des fonctions.

Démonstration. La preuve est simple : toute fonction continue sur un intervalle $[a,b]$ peut être approchée d'aussi près que l'on veut par une fonction en escalier. Par exemple, on subdivise l'intervalle $[a,b]$ en n sous-intervalles $[x_i, x_{i+1}]$ et on prend une marche de hauteur $f(x_i)$ sur cet intervalle. Comme nous avons prouvé que l'on sait réaliser toutes les fonctions en escalier, la preuve est terminée.



$n = 5$



$n = 10$

□

Remarque : la preuve se rapproche de la construction de l'intégrale. Pour calculer l'intégrale, on calcule en fait l'aire de rectangles. Ces rectangles correspondent à nos fonctions en escalier.

5.5. Théorème d'approximation universel (deux variables et plus)

Ce que nous avons fait pour une variable, nous pouvons le faire pour deux variables (ou plus).

Théorème 3 (Théorème d'approximation universel)**.**
Toute fonction continue $f : [a,b] \times [a,b] \to \mathbb{R}$ *peut être approchée d'aussi près que l'on veut par une fonction* $F : [a,b] \times [a,b] \to \mathbb{R}$ *réalisée par un réseau de neurones.*

Il suffit là encore de réaliser des fonctions marches élémentaires. Nous ne donnons pas de détails mais seulement l'exemple d'un réseau qui réalise la fonction $F : \mathbb{R}^2 \to \mathbb{R}$ qui vaut 1 sur $[0,1] \times [0,1]$ et 0 partout ailleurs.