

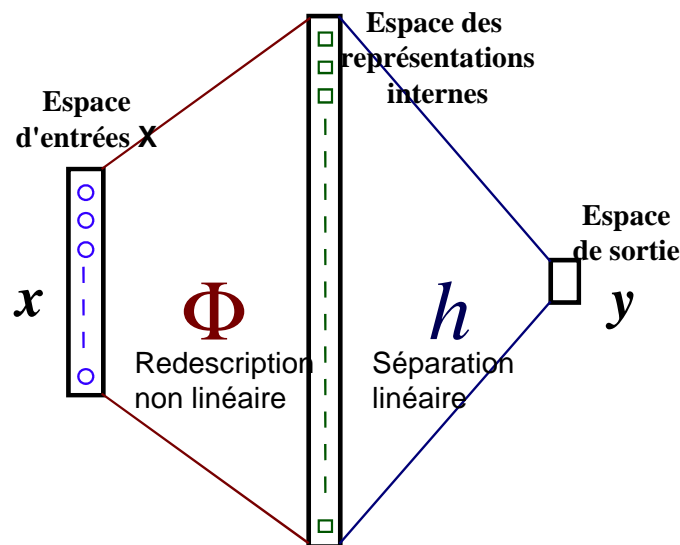
# Brève introduction aux « Support Vector Machines » (SVM), alias « Séparateurs à Vaste Marge »

Fabien Moutarde  
Centre de Robotique (CAOR)  
MINES ParisTech (ENSMP)

[Fabien.Moutarde@mines-paristech.fr](mailto:Fabien.Moutarde@mines-paristech.fr)

<http://perso.mines-paristech.fr/fabien.moutarde>

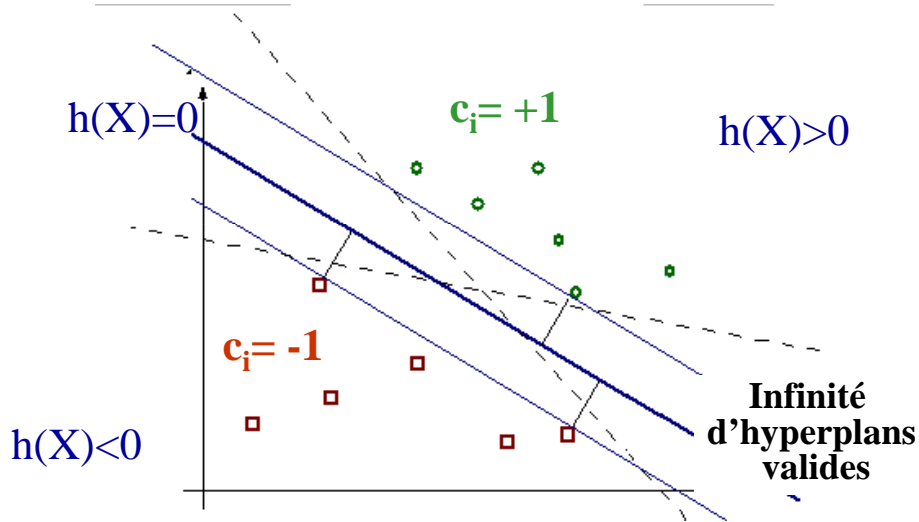
## Principe essentiel des SVM



Passer (*indirectement*) dans un espace  
de plus grande dimension  
pour y faire séparation linéaire *optimale*...

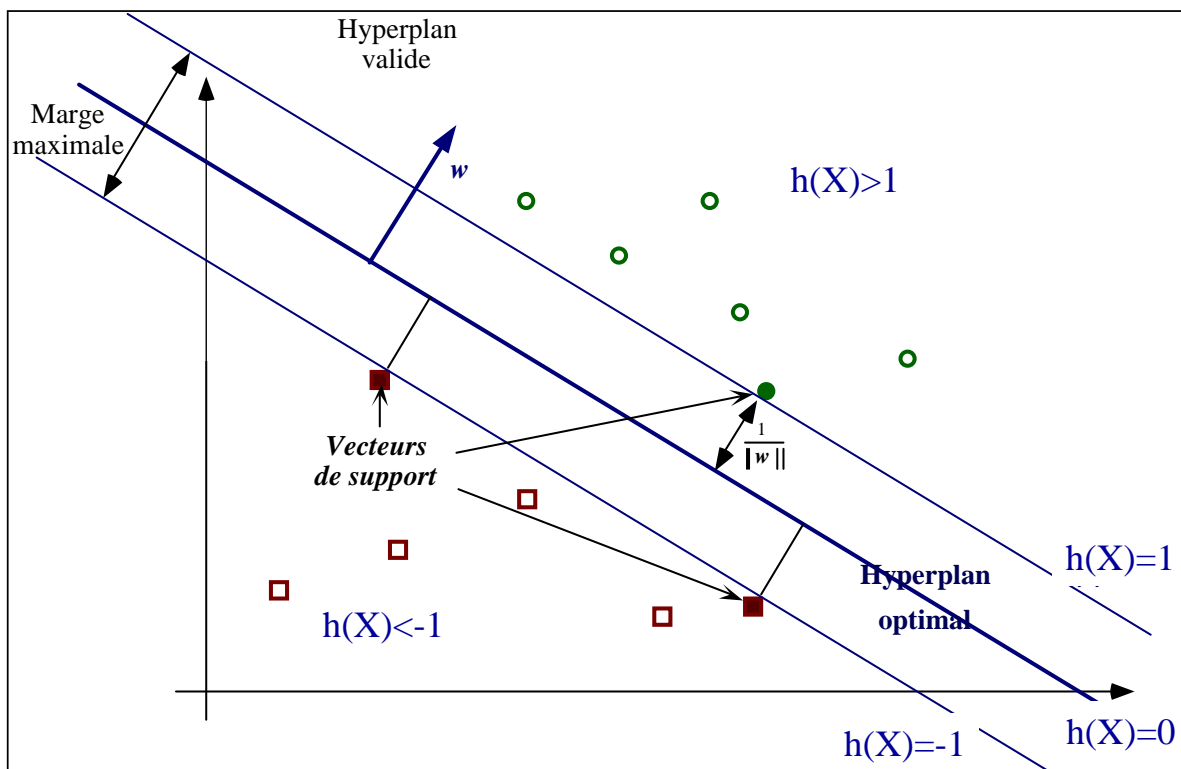
# Séparation linéaire

- Soient les exemples  $\{(X_i, c_i), i=1, \dots, m\}$  d'un problème de classification, avec  $X_i \in X \subset \mathcal{R}^d$ , et  $c_i \in \{-1, +1\}$



- Une séparation linéaire est la partition en 2 par un hyperplan d'équation  $h(X) = w \cdot X + b = 0$

# Hyperplan optimal de séparation



# Maximisation de la « marge »

- La distance d'un point  $p$  à l'hyperplan est  $d(p) = \frac{|w \cdot p + b|}{\|w\|}$
- On veut l'hyperplan tel que sa distance aux points les plus proches (=« marge ») soit maximale.

Maximiser cette marge revient donc à minimiser  $\|w\|$  sous la contrainte que l'hyperplan reste séparateur :

$$\begin{cases} \min \|w\|^2 \\ \forall i \quad c_i (w \cdot X_i + b) \geq 1 \end{cases}$$

C'est un problème convexe d'optimisation

# Résolution de l'optimisation sous contrainte

- Soit par méthode directe d'optimisation quadratique, possible en pratique uniquement si  $d$  « petite »
- Soit (le plus souvent) par méthode des multiplicateurs de Lagrange

$$\begin{cases} L(w, b, \alpha) = \|w\|^2 - \sum_{i=1}^m \alpha_i \{ (X_i \cdot w + b) c_i - 1 \} \\ \forall i \quad \alpha_i \geq 0 \end{cases}$$

Là où  $L$  est minimum,  $\frac{\partial L}{\partial w} = 0 \Rightarrow w^* = \sum_{i=1}^m \alpha_i c_i x_i$

De plus, conditions de Karush-Kuhn-Tucker (KKT) :

$$\forall i, \alpha_i [c_i (w x_i + b) - 1] = 0 \Rightarrow \text{seuls les } \alpha_i \text{ sur marge peuvent être non nuls}$$

$$\begin{cases} L(\mathbf{w}, b, \alpha) = \|\mathbf{w}\|^2 - \sum_{i=1}^m \alpha_i \{ (X_i \cdot \mathbf{w} + b) c_i - 1 \} \\ \forall i \alpha_i \geq 0 \end{cases}$$

se ramène au problème dual ci-dessous

$$\begin{cases} \max_{\alpha} \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j c_i c_j (X_i \cdot X_j) \\ \forall i \alpha_i \geq 0 \\ \sum_{i=1}^l \alpha_i c_i = 0 \end{cases}$$

dont résolution est en  $O(m)$  au lieu  $O(d)$

## Solution du problème dual

$$\begin{cases} h(\mathbf{X}) = (\mathbf{w}^* \cdot \mathbf{X} + b^*) \\ \mathbf{w}^* = \sum_{i=1}^m \alpha_i^* c_i X_i \\ b^* = c_s - \sum_{i=1}^m \alpha_i^* c_i (X_i \cdot X_s) \end{cases}$$

- La solution s'exprime explicitement en fonction des exemples de la base d'apprentissage
- Seuls les  $\alpha_i$  des points les plus proches de l'hyperplan sont non-nuls : ce sont les « points de supports ».

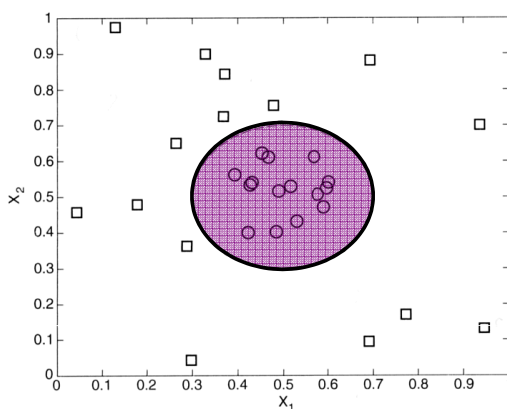
- Pour tolérer quelques exemple non linéairement séparables, on relâche la contrainte de séparation :

$$\forall i \quad c_i (w \cdot X_i + b) \geq 1 - \xi_i$$

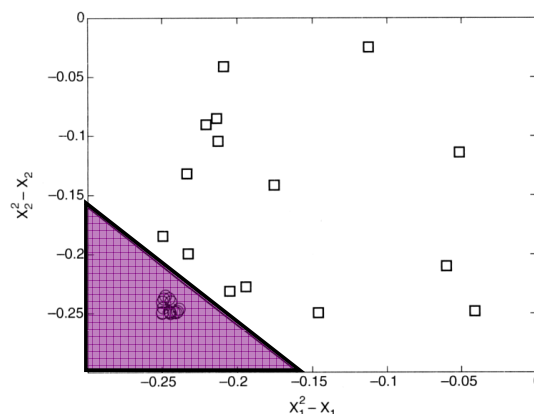
Ceci revient à minimiser  $\|w\|^2 + C \sum_{i=1}^m \xi_i$

On retrouve exactement le même problème dual que dans cas d'une marge « stricte », mais avec contrainte  $0 \leq \alpha_i \leq C$

## Idée du « changement de variable » pour « linéariser » un problème



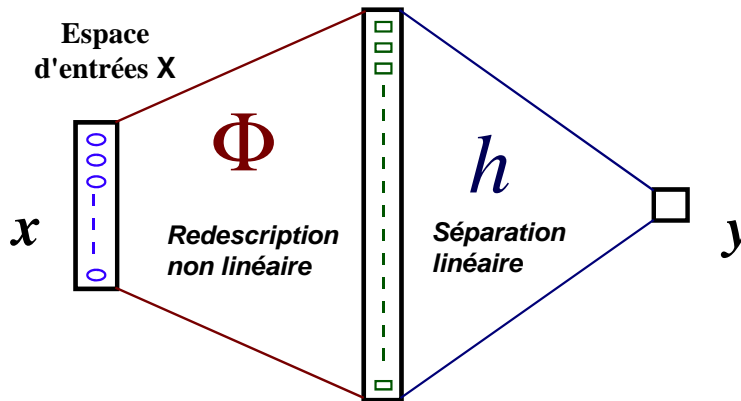
(a) Decision boundary in the original two-dimensional space.



(b) Decision boundary in the transformed space.

- En passant de l'espace  $(x_1, x_2)$  à  $(x_1^2, x_2^2, x_1, x_2)$ , on a rendu le problème linéairement séparable !!

- Si  $\{(X_i, c_i), i=1, \dots, m\}$  non linéairement séparable, chercher une transformation  $\Phi : X \rightarrow F$  telle que les exemples transformés  $\{(\Phi(X_i), c_i), i=1, \dots, m\}$  soient linéairement séparables dans  $F$



Pour chaque transformation  $\Phi$ , on sait trouver dans  $F$  le classifieur optimal en terme de « marges »

## Trouver la transformation $\phi$ : l'astuce du noyau

- Pour tout « mapping »  $\phi$  de  $X \rightarrow F$ , on peut poser  $\forall x, z \in X : k(x, z) = \langle \phi(x), \phi(z) \rangle$
- La séparation linéaire optimale dans  $F$  est alors la solution de :

$$\begin{cases} \max_{\alpha} \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j c_i c_j k(X_i, X_j) \\ \forall i \ 0 \leq \alpha_i \leq C \\ \sum_{i=1}^m \alpha_i c_i = 0 \end{cases}$$

qui est :  $h(X) = \sum_{i=1}^{N_s} \alpha_{s(i)} c_{s(i)} k(X_{s(i)}, X) + b$  où  $X_{s(i)}$  sont les  $N_s$  « points de support »

Tout ça peut être calculé en connaissant  $k(X, Z)$  uniquement (i.e. sans connaître  $\phi$  !!)

→ Chercher  $\phi$  revient donc à trouver le bon « noyau »  $k$

## Condition de Mercer :

pour toute fonction  $k(X,Z)$  symétrique ET vérifiant

$\int k(X,Z)f(X)f(Z)dXdZ \geq 0$  pour toute fonction  $f$  L2-intégrable (i.e. tq  $\int f(X)^2dX$  soit finie), il existe une fonction  $\phi$  telle que  $\forall X,Z k(X,Z)=\phi(X) \cdot \phi(Z)$

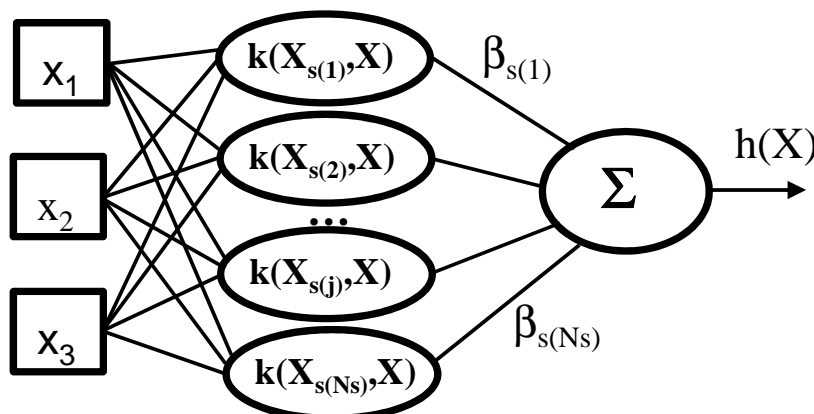
## Noyaux usuels :

- **Polynomiaux**  $k(X,Z) = (X.Z + 1)^q$
- **RBF**  $k(X,Z) = e^{-\frac{\|X-Z\|^2}{2\sigma^2}}$
- **sigmoïde**  $k(X,Z) = \tanh(aX.Z - b)$

## Forme « neuronale » d'un classifieur SVM

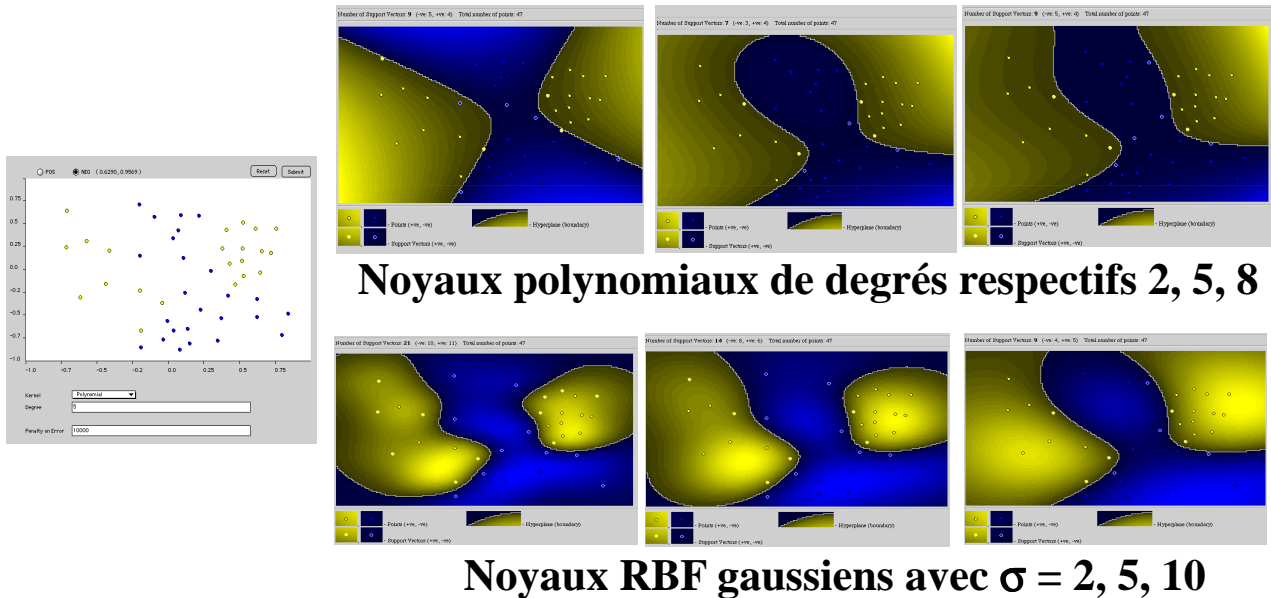
$$h(X) = \sum_{i=1}^{N_s} \beta_{s(i)} k(X_{s(i)}, X) + b \quad (\text{avec } \beta_{s(i)} = \alpha_{s(i)} c_{s(i)})$$

Peut se mettre sous forme d'un réseau neuronal avec une couche cachée de  $N_s$  neurones type « noyau » (avec poids respectifs = vecteurs-support) et une sortie linéaire :



$N_s = \text{nb de « vecteurs-support » } X_{s(j)}$

1. Choisir type et paramètres du noyau k
2. Choisir éventuellement paramètre de tolérance C (cas d'une « marge douce »)



- **Le choix des paramètres doit généralement se faire par méthode empirique rigoureuse type « validation croisée »**
  - **Validation croisée :**
    - Découper ensemble d'exemples en k sous-ensembles disjoints de taille m/k
    - Apprendre sur union de k-1 des parties
    - Calculer erreur sur k-ième partie
    - Erreur finale = moyenne des k erreurs
- Comparaison erreurs moyennes obtenues avec divers noyaux et paramètres → choix meilleure combinaison**

*[Démonstration SVM]*



- *Learning with kernels: support vector machines, regularization, optimization, and beyond*, Bernard Scholkopf & Alexander J. Smola, The MIT Press, 2002.
- *Support Vector Machines and other kernel-based learning methods*, John Shawe-Taylor & Nello Cristianini, Cambridge University Press, 2000

<http://www.kernel-machines.org/>