Chapitre 14

Méthodes à noyaux

Les méthodes linéaires d'analyse de données et d'apprentissage ont été parmi les premières à être développées. Elles ont également été intensivement étudiées, en particulier parce qu'elles se prêtent bien à l'analyse mathématique. Cependant, de nombreuses applications requièrent des modèles non linéaires pour rendre compte des dépendances et des régularités sous-jacentes dans les données.

Les méthodes à noyaux permettent de trouver des fonctions de décision non linéaires, tout en s'appuyant fondamentalement sur des méthodes linéaires. Une fonction noyau correspond à un produit scalaire dans un espace de redescription des données, souvent de grande dimension. Dans cet espace, qu'il n'est pas nécessaire de manipuler explicitement, les méthodes linéaires peuvent être mises en œuvre pour y trouver des régularités linéaires, correspondant à des régularités non linéaires dans l'espace d'origine.

Grâce à l'utilisation de fonctions noyau, il devient ainsi possible d'avoir le meilleur de deux mondes : utiliser des techniques simples et rigoureusement garanties, et traiter des problèmes non linéaires. C'est pourquoi ces méthodes sont devenues très populaires récemment.

Sommaire

Les méthodes d'apprentissage dites **à noyaux** (*kernel-based methods*) sont actuellement très en vogue. Cela n'est pas le fruit du hasard puisqu'elles sont à la convergence de plusieurs points de vue différents.

1. Trois voies vers les méthodes à noyau

Nous décrivons dans la suite trois points de vue qui conduisent aux méthodes à noyaux. Nous les exposons en ordre d'abstraction croissante. C'est aussi, à peu près, l'ordre dans lequel les méthodes à noyaux en apprentissage ont été considérées dans les travaux de recherche.

1.1 Dans l'esprit des plus proches voisins et des fenêtres de Parzen

Historiquement, il est fait mention des approches à noyaux dès les années soixante. L'idée essentielle est d'utiliser les exemples d'apprentissage $\langle(\mathbf{x}_i, u_i)\rangle_{i=1,m}$ pour réaliser une interpolation sur l'espace des entrées $\mathcal{X}$ grâce à une pondération des sorties associées aux entrées $\big(\mathbf{x}_i\big)_{i=1,m}$. Ainsi, pour une entrée $\mathbf{x}$ inconnue, la réponse $\hat{y}(\mathbf{x})$ est estimée par :

$$\boxed{\hat{y}(\mathbf{x}) \;=\; h_m(\mathbf{x}) \;=\; \frac{1}{\sum_{i=1}^{m} K(\mathbf{x}, \mathbf{x}_i)} \sum_{i=1}^{m} u_i \, K(\mathbf{x}, \mathbf{x}_i)} \tag{14.1}$$

Le poids $K(\mathbf{x}, \mathbf{x}_i)$ qui est associé à chaque sortie u_i dépend de la position relative de $\mathbf{x}_i$ dans l'espace des entrées $\mathcal{X}$ et du point $\mathbf{x}$ considéré. La fonction $K(\mathbf{x}, \mathbf{x}')$ définissant ces poids est appelée, dans ces approches, *fonction noyau* ou *noyaux de Parzen* ou *fenêtres de Parzen*. Le plus souvent, la forme des fonctions noyau est du type :

$$K(\mathbf{x}, \mathbf{x}') \;=\; g\left(\frac{d(\mathbf{x}, \mathbf{x}')}{\sigma}\right)$$

où $d(\mathbf{x}, \mathbf{x}')$ est une « distance » définie sur $\mathcal{X}$, σ est un facteur d'échelle (ou d'adoucissement), et $g(\cdot)$ est une fonction (usuellement monotone) décroissante. Un choix fréquent pour cette fonction est $g(z) = \exp(-z^2/2)$. L'utilisation de ce type de fonction noyau conduit à une estimation $\hat{y}$ comme moyenne pondérée des $(u_i)_{i=1,m}$ avec un poids plus fort pour les exemples pour lesquels la distance $d(\mathbf{x}, \mathbf{x}_i)$ est petite, la notion de « petite » étant déterminée par la valeur de σ.

Ces méthodes, par interpolation, qui sont très liées aux méthodes de plus proches voisins, seront détaillées dans le chapitre 15. On peut cependant d'ores et déjà en évoquer les avantages. Elles sont un approximateur universel : lorsque le nombre d'exemples d'apprentissage croît et devient arbitrairement grand, $m \to \infty$, l'estimation 14.1 approche la fonction de prédiction optimale, $h_m(\mathbf{x}) \to h^*(\mathbf{x})$, si le paramètre d'échelle σ est une fonction de m décroissant vers 0, $\sigma(m) \to 0$, à une vitesse inférieure à $1/m$. Ce résultat est valable pour presque toute fonction de distance (sous de faibles hypothèses telles que la convexité). De plus, ces méthodes d'interpolation ne demandent pas d'apprentissage à proprement parler puisqu'*il n'y a pas d'hypothèse apprise*, mais utilisation directe de l'ensemble d'apprentissage.

1.2 Passage par un espace de redescription et astuce des noyaux

Un autre point de vue sur les méthodes à noyaux introduit la notion d'espace de redescription.

À titre d'exemple, nous considérons ici le **cas de la régression linéaire** déjà étudié dans le chapitre 9. On cherche à trouver une fonction linéaire h définie dans $\mathcal{X} = \mathbb{R}^d$ qui interpole au mieux des données d'apprentissage $\mathcal{S} = \langle(\mathbf{x}_1, u_1), \ldots, (\mathbf{x}_m, u_m)\rangle$ de points $\mathbf{x}_i$ pris dans $\mathcal{X}$ et de valeur associée $u_i \in \mathbb{R}$:

$$\boxed{h(\mathbf{x}) \;=\; w_0 + \sum_{i=1}^{d} w_i \, x^i} \tag{14.2}$$

où les x^i sont les coordonnées d'un point $\mathbf{x}$. Nous écrirons aussi : $h(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle$ en étendant le vecteur d'entrée par l'ajout d'une composante zéro de valeur 1, ou bien $\mathbf{w} \cdot \mathbf{x} + w_0$ quand nous voudrons mettre en exergue le coefficient w_0.

Utilisons le principe inductif de minimisation du risque empirique, en prenant l'écart quadratique comme fonction de perte (voir figure 14.1) :

$$R_{\text{Emp}}(h) = R_{\text{Emp}}(\mathbf{w}) = \frac{1}{m} \sum_{i=1}^{m} l\big(h(\mathbf{x}_i), u_i\big) = \frac{1}{m} \sum_{i=1}^{m} (h(\mathbf{x}_i) - u_i)^2 = \frac{1}{m} \sum_{i=1}^{m} \xi_i^2 \qquad (14.3)$$

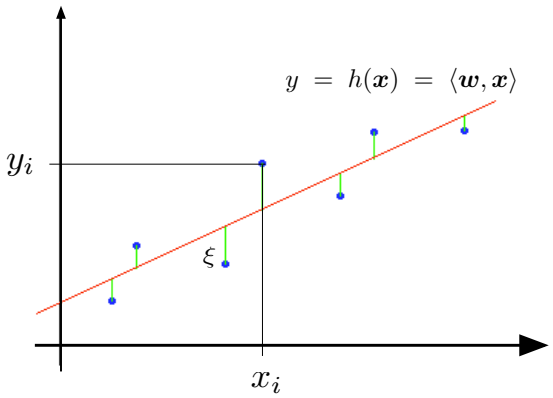


FIG. 14.1: *Régression linéaire.*

On cherche le vecteur de paramètres $\mathbf{w}$ minimisant ce risque empirique. Écrivons ξ le vecteur des écarts ξ_i : $\xi = \mathbf{y} - \mathbf{X}\mathbf{w}$.

On a alors : $R_{\text{Emp}}(\mathbf{w}) = ||\xi||_2^2 = (y - \mathbf{X}\mathbf{w})^\top (y - \mathbf{X}\mathbf{w})$.

On peut chercher la valeur optimale de $\mathbf{w}$ en calculant la dérivée du risque par rapport au vecteur de paramètres $\mathbf{w}$ et en la posant égale au vecteur nul :

$$\frac{\partial R_{\text{Emp}}(\mathbf{w})}{\partial \mathbf{w}} = -2\mathbf{X}^\top y + 2\mathbf{X}^\top \mathbf{X}\mathbf{w}$$

On obtient alors les équations dites « normales » : $\mathbf{X}^\top \mathbf{X}\mathbf{w} = \mathbf{X}^\top y$.

Si l'inverse de $\mathbf{X}\mathbf{X}^\top$ existe, la solution peut s'exprimer sous la forme :

$$\mathbf{w} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y} = \mathbf{X}^\top \mathbf{X} (\mathbf{X}^\top \mathbf{X})^{-2} \mathbf{X}^\top \mathbf{y} = \mathbf{X}^\top \alpha$$

ce qui signifie que $\mathbf{w}$ est une combinaison linéaire des points d'apprentissage : $\mathbf{w} = \sum_{i=1}^{m} \alpha_i \mathbf{x}_i$.

L'hypothèse recherchée a alors la forme, appelé ***représentation duale*** :

$$\boxed{h(\mathbf{x}) = \sum_{i=1}^{m} \alpha_i \langle \mathbf{x}_i, \mathbf{x} \rangle} \qquad (14.4)$$

qui a la particularité de *ne dépendre que de produits scalaires* entre vecteurs de $\mathcal{X}$.

Ce qui est remarquable est que l'on peut retrouver (voir par exemple [STC04], pp.31-32) la même forme pour l'hypothèse h si l'on cherche à minimiser le *risque empirique régularisé* dans

lequel on favorise les fonctions de norme faible :

$$R(h) \;=\; \frac{1}{m}\sum_{i=1}^{m}(u_i - h(\mathbf{x}_i))^2 \;+\; \lambda \min_{\mathbf{w}} ||\mathbf{w}||^2$$

On trouve en effet : $\mathbf{w} = \lambda^{-1}\mathbf{X}^\top(\mathbf{u} - \mathbf{X}\mathbf{w}) = \mathbf{X}^\top\alpha$, avec $\alpha = \lambda^{-1}(\mathbf{u} - \mathbf{X}\mathbf{w})$.
Soit encore : $(\mathbf{X}\mathbf{X}^\top + \lambda\mathbf{I}_m)\alpha = \mathbf{y}$, d'où :

$$\boxed{\alpha = (\mathbf{K} + \lambda\mathbf{I}_m)^{-1}\mathbf{u}} \tag{14.5}$$

où $\mathbf{K}$ est la matrice de Gram (voir plus loin).

Ce **résultat** est **important** car, généralisé (il le sera dans le *théorème de représentation*, section 1.3), il montre que *les problèmes régularisés* de la forme :

$$R(h) \;=\; \frac{1}{m}\sum_{i=1}^{m}(u_i - h(\mathbf{x}_i))^2 \;+\; \lambda\,\mathrm{Reg}(h)$$

ont naturellement des solutions de la forme : $h(\mathbf{x}) \;=\; \sum_{i=1}^{m}\alpha_i \langle \mathbf{x}_i, \mathbf{x}\rangle$, c'est-à-dire d'une combinaison linéaire de produits scalaires.

Supposons que l'on cherche à résoudre cette fois-ci un **problème de régression non linéaire** dans $\mathcal{X}$ (voir figure 14.2), on peut être tenté d'utiliser pour ce faire une fonction de redescription ϕ de $\mathcal{X}$ dans un nouvel espace F (« *feature space* »), de telle manière que le problème devienne un problème de régression linéaire dans ce nouvel espace.

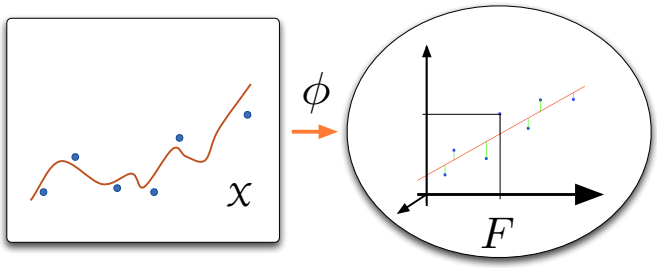


FIG. 14.2: *Régression non linéaire rendue possible par la transformation en un problème de régression linéaire dans l'espace de redescription F.*

Dans ce cas, la solution du problème de régression linéaire, dans l'espace de redescription F, prendra la *forme primale* $h(\mathbf{x}) \;=\; \langle \mathbf{w}, \phi(\mathbf{x})\rangle \;=\; 0$, que l'on peut ré-exprimer en mettant en évidence la base de fonctions non linéaires $\{\phi_1(\mathbf{x}), \phi_2(\mathbf{x}), \ldots\}$, éventuellement infinie, de l'espace F :

$$h(\mathbf{x}) \;=\; \sum_{j=1}^{\infty} w_j\,\phi_j(\mathbf{x}) + w_0 = 0 \tag{14.6}$$

En résolvant comme précédemment, cette fois-ci dans l'espace de redescription, on obtient à nouveau une *expression duale* :

$$\boxed{h(\mathbf{x}) \;=\; \sum_{i=1}^{m}\alpha_i \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x})\rangle} \tag{14.7}$$

Et, en rapprochant les équations (14.6) et (14.7), on tire :

$$\begin{aligned} h(\mathbf{x}) &= \sum_{j=1}^{\infty} w_j \, \phi_j(\mathbf{x}) + w_0 \;=\; \mathbf{w}^\top \cdot \Phi(\mathbf{x}) \\ &= \sum_{i=1}^{m} \alpha_i \, \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}) = 0 \end{aligned}$$

d'où :

$$\boxed{\mathbf{w}^\top = \sum_{i=1}^{m} \alpha_i \, \Phi(\mathbf{x}_i)} \tag{14.8}$$

Un problème est évidemment que la nouvelle formulation implique, d'une part, de trouver une fonction de redescription Φ adéquate, et, d'autre part, de calculer des produits scalaires dans un espace F dont la dimension peut être grande (voire infinie). Cependant, il peut parfois être possible de ne pas avoir à effectuer explicitement ces produits scalaires dans F grâce à l'utilisation de *fonctions noyau*. C'est ce que l'on appelle l'« astuce des noyaux » (*kernel trick*).

Définition 14.1 (Fonction noyau (Kernel function))

Une fonction noyau *est une fonction* $\kappa : \; \mathbf{x}, \mathbf{x}' \in \mathcal{X}^2 \rightarrow \mathbb{R}$ *satisfaisant :*

$$\kappa(\mathbf{x}, \mathbf{x}') \;=\; \langle \Phi(\mathbf{x}), \Phi(\mathbf{x}') \rangle$$

où Φ *est une fonction de* $\mathcal{X}$ *vers un espace de redescription* F *doté d'un produit scalaire :*

$$\Phi : \; \mathbf{x} \mapsto \Phi(\mathbf{x}) \;\in F$$

L'utilisation de fonctions noyau permet ainsi de calculer implicitement un produit scalaire dans un espace de dimension éventuellement infini par un calcul n'impliquant qu'un nombre fini de termes, ce nombre étant le nombre m des exemples d'apprentissage (voire éventuellement moins, comme nous le verrons). Par ailleurs, il apparaît donc que la spécification de la fonction noyau est *suffisante*. Il n'est pas nécessaire de calculer le vecteur poids $\mathbf{w}$ pour spécifier la fonction de décision $h(\mathbf{x}) \;=\; \sum_{i=1}^{m} \alpha_i \, \kappa(\mathbf{x}_i, \mathbf{x})$.

En un sens profond, **une fonction noyau correspond à une mesure de similarité** entre entrées $\mathbf{x}$ et $\mathbf{x}'$, mesure qu'il est naturel d'utiliser en induction puisqu'un *a priori* évident est de supposer que deux entrées similaires doivent être associées à des sorties similaires. De fait, les fonctions noyaux peuvent être considérées comme une *généralisation des fonctions de covariance*. En effet, une fonction noyau peut s'exprimer comme une somme de produits sur un ensemble de fonctions :

$$\kappa(\mathbf{x}, \mathbf{x}') \;=\; \sum_{i=1}^{\infty} \phi_i(\mathbf{x}) \, \phi_i(\mathbf{x}')$$

ce qui suggère de considérer les fonctions noyau comme des fonctions de covariance dépendante d'une mesure de probabilité μ sur une classe de fonctions $\mathcal{F}$:

$$\kappa_\mu(\mathbf{x}, \mathbf{x}') \;=\; \int_{\mathcal{F}} f(\mathbf{x}) f(\mathbf{x}') \mu(f) \mathrm{d}f$$

On appellera ces fonctions noyau des *noyaux de covariance*. On peut montrer que toute fonction noyau peut être obtenue comme noyau de covariance avec une mesure μ particulière sur $\mathcal{F}$.

Étant donné un ensemble de points $\{\mathbf{x}_i | i = 1, \ldots, m\}$, nous pouvons calculer la *matrice de Gram* (ou *matrice noyau*) $\mathbf{K}$ dont les éléments sont : $K_{ij} = \kappa(\mathbf{x}_i, \mathbf{x}_j)$.

Une matrice réelle $\mathbf{K}$ de dimension $m \times m$ vérifiant la forme quadratique $Q(\mathbf{v}) = \mathbf{v}^\top \mathbf{K}\mathbf{v} \geq 0$ pour tout vecteur $\mathbf{v} \in \mathbb{R}^m$ est dite *positive semidéfinie*. Si $Q(\mathbf{v}) = 0$ seulement quand $\mathbf{v} = 0$, alors la matrice $\mathbf{K}$ est dite positive définie. De manière équivalente, une matrice symétrique est positive semidéfinie si et seulement si toutes ses valeurs propres sont positives ou nulles.

Une matrice de Gram correspondant à une fonction de covariance doit être positive semidéfinie.

Une *fonction noyau* est dite *positive semidéfinie* si

$$\int \kappa(\mathbf{x}, \mathbf{x}')\, f(\mathbf{x}) f(\mathbf{x}')\, d\mu(\mathbf{x}) d\mu(\mathbf{x}') \geq 0 \tag{14.9}$$

où μ et μ' dénotent des mesures sur $\mathcal{X}$ et pour toute fonction $f \in L_2(\mathcal{X}, \mu)$[1]

De manière équivalente, une fonction noyau définissant une matrice de Gram positive semidéfinie pour tout ensemble de points d'entrée $\mathcal{S}_m$ et pour tout $m \in \mathbb{N}$ est positive semidéfinie.

Jusqu'à présent, le seul critère que nous ayons énoncé pour savoir si une fonction symétrique $\kappa : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ est une fonction noyau était celui d'exhiber un espace de redescription F et de vérifier que $\kappa(\cdot, \cdot)$ correspondait à un produit scalaire dans F. Plusieurs théorèmes permettent de caractériser les fonctions noyau sans passer explicitement par l'espace de redescription. Ainsi :

Théorème 14.1 (Les fonctions symétriques positives semidéfinies sont des noyaux)

Une fonction $\kappa : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ est une fonction noyau si et seulement si elle est symétrique et positive semidéfinie.

Une démonstration peut être trouvée dans [SC08], p.118.

Un autre théorème, le **théorème de Mercer**, fournit des conditions pour qu'une fonction symétrique $\kappa : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ soit une fonction noyau. Il permet en outre d'identifier une décomposition spectrale des fonctions noyau.

L'appellation *fonction noyau* provient de leur rôle dans des équations très importantes en physique, les *équations intégrales* du type :

$$f(x) = \lambda \int_a^b K(x, y)\, f(y)\, dy + h(x)$$

dans lesquelles l'inconnue est la fonction $f(x)$, et $K(x, y)$ est appelée le noyau de l'équation intégrale. Les fonctions propres de cette équation dépendent des propriétés de la fonction noyau. Si celle-ci est symétrique : $K(x, y) = K(y, x)$ (ce qui correspond en un sens profond à une conservation de l'énergie du système décrit) alors on peut prouver que les valeurs propres cherchées λ_k sont réelles et que les fonctions propres ψ_k associées sont orthogonales. Cela conduit évidemment à *pouvoir représenter chaque fonction par une série de fonctions propres.* (Cette

[1] Une fonction f définie de $\mathbb{R}$ sur $\mathbb{R}$ est dite de carré sommable par rapport à la mesure μ ($f \in L_2(\mathcal{X}, \mu)$)définie sur $\mathbb{R}$ si sa *norme* associée $||f||$ définie par :

$$||f||^2 = \int_a^b [f(x)]^2 \, \mathrm{d}\mu(x)$$

est bornée sur $\mathbb{R}$ tout entier.

propriété permet d'envisager de réaliser une analyse en composantes principales généralisée). On peut aussi réécrire l'équation

$$f(x) \;=\; \lambda \int_a^b K(x,y)\, f(y)\, dy$$

comme

$$f \;=\; \lambda\, A\, f$$

où A est un opérateur linéaire, représentable par une matrice.

Le **théorème de Mercer** permet d'exprimer les fonctions noyau en termes de valeurs propres et de fonctions propres.

Une fonction $\phi(\cdot)$ vérifiant l'équation intégrale

$$\int \kappa(\mathbf{x},\mathbf{x}')\, \phi(\mathbf{x})\, \mathrm{d}\mu(\mathbf{x}) \;=\; \lambda\, \phi(\mathbf{x}')$$

est appelée une *fonction propre* du noyau κ associée à la valeur propre $\lambda \in \mathbb{R}$ en fonction de la mesure μ. Il existe en général une infinité de focntions propres, que nous noterons $\phi_1(\mathbf{x}), \phi_2(\mathbf{x}), \ldots$ en supposant l'ordre $\lambda_1 \geq \lambda_2 \geq \ldots$. Les fonctions propres sont orthogonales en fonction de μ et peuvent être normalisées de telle manière que $\int \phi_i(\mathbf{x})\, \phi_i(\mathbf{x})\, \mathrm{d}\mu(\mathbf{x}) = \delta_{ij}$ où δ_{ij} est le symbole de Kronecker égal à 1 seulement quand $i = j$.

Théorème 14.2 (Théorème de Mercer)

Si $\kappa(\cdot,\cdot)$ est une fonction noyau continue symétrique d'un opérateur intégral

$$g(\mathbf{y}) \;=\; A\, f(\mathbf{y}) \;=\; \int_a^b \kappa(x,y)\, f(y)\, dy \;+\; h(x)$$

vérifiant :

$$\int_{\mathcal{X}\times\mathcal{X}} \kappa(\mathbf{x},\mathbf{x}')\, f(\mathbf{x})\, f(\mathbf{x}')\, d\mathbf{x} d\mathbf{x}' \;\geq 0$$

pour toute fonction $f \in L_2(\mathcal{X})$ (de carré sommable) ($\mathcal{X}$ étant un sous-espace compact de $\mathbb{R}^d$), alors la fonction $\kappa(\cdot,\cdot)$ peut être développée en une série uniformément convergente en fonction des valeurs propres positives λ_i et des fonctions propres ψ_i :

$$\kappa(\mathbf{x},\mathbf{x}') \;=\; \sum_{j=1}^{N} \lambda_j\, \psi_j(\mathbf{x})\, \psi_j(\mathbf{x}') \tag{14.10}$$

où N est le nombre de valeurs propres positives (nombre éventuellement infini).

On peut alors décrire la fonction Φ de redescription des entrées comme :

$$\Phi(\mathbf{x}) \;=\; (\sqrt{\lambda_1}\, \psi_1(\mathbf{x}),\, \sqrt{\lambda_2}\, \psi_2(\mathbf{x}), \ldots) \tag{14.11}$$

Cette décomposition est l'analogue en dimension infinie de la diagonalisation d'une matrice Hermitienne. Le taux de décroissance des valeurs propres λ_i fournit une information sur la régularité du noyau. Plus cette décroissance est rapide, plus régulière est la fonction noyau.

Si les conditions théoriques exposées ci-dessus sont intéressantes, elles ne sont pas faciles à vérifier en pratique. La plupart des praticiens utilisent l'une des fonctions noyau connues ou bien font usage de propriétés particulières de combinaisons de fonctions noyau pour construire de nouvelles fonctions noyau adaptées à leur problème (voir section 5 plus bas).

Il n'y a pas de bijection entre les fonctions noyau et les espaces de redescription pour lesquelles ces fonctions sont analogues à un produit scalaire. Une fonction noyau est ainsi associée à plusieurs espaces de redescription.

—— Exemple **Non bijection entre fonction noyau et espace de redescription** ——————

Soit un espace d'entrée $\mathcal{X} = \mathbb{R}^2$, et la projection $\Phi_1 : (x^1, x^2) \mapsto \big((x^1)^2, (x^2)^2, x^1x^2, x^2x^1\big)$, où x^i dénote la i-ème composante du vecteur $\mathbf{x}$.

Alors le produit scalaire dans F s'écrit :

$$\langle \phi_1(\mathbf{x}, \phi_1(\mathbf{x}')\rangle \;=\; (x^1)^2(x'^1)^2 + (x^2)^2(x'^2)^2 + 2x^1x^2x'^1x'^2 \;=\; \langle \mathbf{x}, \mathbf{x}'\rangle^2$$

La fonction noyau $\kappa(\mathbf{x}, \mathbf{x}') = \langle \mathbf{x}, \mathbf{x}'\rangle^2$ correspond donc à un produit scalaire dans l'espace F à quatre dimensions définit par la projection Φ_1.

Mais il est facile de voir qu'elle correspond aussi à un produit scalaire dans l'espace de redescription F à trois dimensions défini par la projection $\Phi_2 : (x^1, x^2) \mapsto \big((x^1)^2, (x^2)^2, \sqrt{2}x^1x^2\big)$. En effet :

$$\langle \phi_2(\mathbf{x}, \phi_2(\mathbf{x}')\rangle \;=\; (x^1)^2(x'^1)^2 + (x^2)^2(x'^2)^2 + 2x^1x^2x'^1x'^2 \;=\; \langle \mathbf{x}, \mathbf{x}'\rangle^2$$

1.3 Approximation de fonctions dans un espace de Hilbert

Le point de vue dominant en apprentissage considère celui-ci essentiellement comme l'approximation d'une fonction cible inconnue à partir d'informations obtenues par les observations. Il est donc naturel de consulter les idées et résultats de l'analyse fonctionnelle. Celle-ci a en particulier pour but d'étudier comment caractériser ou calculer des fonctions à partir de bases (éventuellement infinies) de fonctions. L'exemple le plus célèbre est celui de l'analyse de Fourier (1772-1837) qui permet d'associer à toute fonction stationnaire continue une série (une somme) de fonctions périodiques (sinus et cosinus). On décompose ainsi une fonction selon ses fréquences, de même qu'un prisme décompose la lumière en couleurs. Le théorème de Mercer, vu à la section précédente, en est une illustration.

Plus généralement, une fonction est définie par l'ensemble des valeurs qu'elle prend, soit une infinité de nombres. Une fonction peut donc être vue comme un point dans un espace de dimension infinie. Mais ce point de vue n'est généralement pas très révélateur sur la nature de la fonction. On cherche alors à caractériser une fonction à l'aide de fonctions analysantes (fonctions trigonométriques, ondelettes, ...) prises dans une base. Chaque fonction analysante détermine un axe de projection de la fonction à analyser, et décomposer une fonction dans une base choisie revient à la représenter par une infinité de coordonnées en la projetant sur une infinité d'axes. Ce langage n'est utilisable que si les notions de longueur et d'angle gardent une signification dans des espaces de dimension élevée (infinie), ce qui n'est pas évident. L'étude des conditions sous lesquelles cela est possible a conduit à la théorie des espaces de Hilbert (1862-1943). Un **espace de Hilbert** est essentiellement une extension algébrique de la notion d'espace euclidien ordinaire dans laquelle l'espace est défini par une liste infinie de fonctions orthogonales (analogues aux axes d'un espace euclidien) et où chaque point de cet espace peut être décrit par une combinaison linéaire de fonctions de bases.

L'exemple le plus classique d'un espace de Hilbert est celui des *séries de Fourier* permettant de représenter n'importe quelle fonction $f(\phi)$ dans le domaine $0 \leq \phi \leq 2\pi$ comme une série :

$$f(\phi) \;=\; \sum_{n=0}^{\infty} a_n \, e^{in\phi}$$

où les fonctions $e^{in\phi}$ servent de fonctions de bases orthogonales (on a $\int_0^{2\pi} e^{in\phi}\, e^{-im\phi} \mathrm{d}\phi \;=\; 2\pi\delta_{nm}$ avec $\delta_{nm} = 1$ si $n = m$ et 0 sinon) et où les a_n sont les coordonnées par rapport à cette base.

Définition 14.1 (Espace de Hilbert)

Un espace de Hilbert *est un espace vectoriel sur* $\mathbb{R}$ *ou* $\mathbb{C}$*, muni d'un produit scalaire dont l'espace normé associé est complet. Dans le cas des espaces fonctionnels de fonctions numériques continues sur un intervalle* $[a, b]$ *de* $\mathbb{R}$*, le produit scalaire utilisé est généralement :*

$$\langle f, g \rangle \;=\; \int_a^b f(x)\, g(x)\, dx$$

La norme associée $||f||$ *est alors définie par :*

$$||f||^2 \;=\; \int_a^b [f(x)]^2\, dx$$

Lorsque la norme est bornée sur $\mathbb{R}$ tout entier, f^2 est intégrable et on parle de *fonctions de carrés intégrables*. Lorsque l'intégrale considérée est celle de Lebesgue (1875-1941) (ce qui permet de traiter des fonctions non continues), cet espace se note $L_2[a, b]$.

Dans notre cas, nous considérerons les produits scalaires et normes par les opérations définies sur $\mathcal{X}$: $\langle f, g \rangle \;=\; \int_{\mathcal{X}} f(\mathbf{x})\, g(\mathbf{x})\, \mathrm{d}\mathbf{x}$ et $||f||^2 \;=\; \int_{\mathcal{X}} [f(\mathbf{x})]^2\, \mathrm{d}\mathbf{x}$. Nous pourrons aussi introduire une mesure μ sur $\mathcal{X}$, ce qui conduit au produit scalaire et à la norme suivants : $\langle f, g \rangle \;=\; \int_{\mathcal{X}} f(\mathbf{x})\, g(\mathbf{x})\, \mathrm{d}\mu\mathbf{x}$ et $||f||^2 \;=\; \int_{\mathcal{X}} [f(\mathbf{x})]^2\, \mathrm{d}\mu\mathbf{x}$.

En plus d'être muni d'un produit scalaire, un espace de Hilbert est séparable et complet[2].

Tout espace de redescription muni d'un produit scalaire et complet et séparable peut être doté d'une base. Dans le cas d'un espace de Hilbert fonctionnel, il s'agit d'une base de fonctions.

Pour récapituler, une fonction noyau est associée à un ou plusieurs espaces de redescription muni(s) d'un produit scalaire. Une condition nécessaire et suffisante pour qu'une fonction $\kappa(\mathbf{x}, \mathbf{x}') : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ soit une fonction noyau est qu'elle soit positive semidéfinie ou qu'elle vérifie la condition de Mercer. Le théorème de Mercer (14.2) fournit en plus une description explicite de l'espace de redescription par une base de fonctions analysantes orthogonales. Ce n'est cependant pas la seule base de fonctions analysantes possibles. Une autre base (qui peut parfois coïncider avec la base de Mercer) est donnée par les *noyaux reproduisants*.

[2] Un espace de fonctions $\mathcal{F}$ est *complet* si toute séquence de Cauchy $\{f_n\}_{n\geq 1}$ d'éléments de $\mathcal{F}$ converge vers un élément $f \in \mathcal{F}$, une séquence de Cauchy satisfaisant la propriété que :

$$\sup_{m>n} ||f_n - f_m|| \rightarrow 0, \text{quand } n \rightarrow \infty$$

Un espace $\mathcal{F}$ est *séparable* si il existe un ensemble énumérable d'éléments de $\mathcal{F}$ $f_1, \ldots, f_i, \ldots$ de $\mathcal{F}$ tel que pour tout élément $f \in \mathcal{F}$ et tout $\varepsilon > 0$ il existe une fonction f_i telle que : $||f_i - f|| < \varepsilon$.

Espace de Hilbert à noyau reproduisant (RKHS ou « *Reproducing Kernel Hilbert Space* »).

Étant donnée une fonction noyau, il est possible de lui associer un espace de Hilbert de fonctions (suffisamment) régulières. En un certain sens, cet espace est l'espace de fonctions minimal associé à la fonction noyau considérée et peut donc servir d'espace de redescription canonique.

Définition 14.2 (Espace de Hilbert à Noyau Reproduisant)

Soit $\mathcal{F}$ une espace de Hilbert de fonctions réelles définies sur un ensemble indexé $\mathcal{X}$:

$$\mathcal{F} = \left\{ \sum_{i=1}^{m} \alpha_i \, \kappa(\mathbf{x}_i, \cdot) : m \in \mathbb{N}, \mathbf{x}_i \in \mathcal{X}, \alpha_i \in \mathbb{R}, i = 1, \ldots, m \right\}$$

$\mathcal{F}$ est appelé Espace de Hilbert à Noyau Reproduisant *doté d'un produit scalaire noté $\langle \cdot, \cdot \rangle_{\mathcal{F}}$ (et de la norme $||f||_{\mathcal{F}} = \sqrt{\langle f, f \rangle_{\mathcal{F}}}$) si il existe une fonction $\kappa : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ ayant les propriétés suivantes :*

1. *pour tout élément $\mathbf{x} \in \mathcal{X}, \kappa(\mathbf{x}, \cdot)$ appartient à $\mathcal{F}$, et*
2. *la fonction κ est une* fonction noyau reproduisante*, c'est-à-dire telle que pour toute fonction $f \in \mathcal{F}$, on a : $\langle f, \kappa(\mathbf{x}, \cdot) \rangle_{\mathcal{F}} = \sum_{i=1}^{m} \alpha_i \kappa(\mathbf{x}_i, \mathbf{x}) = f(\mathbf{x})$.*

Le fait que la fonction noyau soit *reproduisante* signifie que toute fonction $f \in \mathcal{F}$ est égale à un produit scalaire qui est aussi une combinaison linéaire finie de fonctions de base. (Voir par exemple [Aub87], pp.137-141, pour les détails et une démonstration).

Le *produit scalaire* sur $\mathcal{F}$ est alors défini comme suit. Soient les fonctions $f, g \in \mathcal{F}$ définies par :

$$f(\mathbf{x}) = \sum_{i=1}^{m} \alpha_i \, \kappa(\mathbf{x}_i, \mathbf{x}) \quad \text{et} \quad g(\mathbf{x}) = \sum_{i=1}^{n} \beta_j \, \kappa(\mathbf{z}_j, \mathbf{x})$$

alors :

$$\langle f, g \rangle = \sum_{i=1}^{m} \sum_{j=1}^{n} \alpha_i \, \beta_j \, \kappa(\mathbf{x}_i, \mathbf{x}_j) = \sum_{i=1}^{m} \alpha_i \, g(\mathbf{x}_i) = \sum_{j=1}^{n} \beta_j f(\mathbf{x}_j) \tag{14.12}$$

Il est intéressant de noter qu'alors que la base de Mercer dépend de la mesure μ définie sur $\mathcal{X}$, ce n'est pas le cas de la base des noyaux reproduisants qui ne dépend que de la fonction noyau.

Par ailleurs, l'espace de Hilbert des fonctions L_2 (de produit scalaire $\langle f, g \rangle_{L_2} = \int f(\mathbf{x}) g(\mathbf{x}) \mathrm{d}\mathbf{x}$) contient de nombreuses fonctions irrégulières. Dans L_2, la fonction de Dirac δ est la fonction de représentation, i.e. $f(\mathbf{x}) = \int f(\mathbf{x}') \delta(\mathbf{x} - \mathbf{x}') \mathrm{d}\mathbf{x}'$. Les fonctions noyaux jouent un rôle analogue à la fonction δ dans les espaces de Hilbert à Noyau Reproduisant, qui sont de ce fait plus réguliers, leur degré de régularité dépendant de la régularité de la fonction noyau associée. En fait, on peut montrer de manière plus générale que les propriétés de mesurabilité, de continuité et de différentiabilité des fonctions de $\mathcal{F}$ dépendent des propriétés associées de la fonction noyau correspondante. Un résultat fondamental étant effectivement le théorème suivant du à Moore et Aronszajn en 1950.

Théorème 14.3 (Bijection entre noyau et RKHS associé)

Soit $\mathcal{X}$ un ensemble énumérable, alors pour toute fonction positive semidéfinie $\kappa(\cdot, \cdot) : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, il existe un unique espace de Hilbert avec ce noyau reproduisant, et vice-versa.

Une caractéristique cruciale de ces fonctions analysantes $\kappa(\mathbf{x}_i, \cdot)$ est donc qu'elles *ne sont pas*

issues d'un dictionnaire donné a priori, comme c'est le cas par exemple des fonctions trigonométriques ou des ondelettes, mais qu'elles *dépendent directement des points* $\mathbf{x}$ de l'espace $\mathcal{X}$. Cela est capital dans l'estimation de fonctions à partir de points particuliers, car, dans ce cas, les fonctions analysantes utilisées pour estimer la fonction vont dépendre des points connus $\{\mathbf{x}_i\}_{1\leq i\leq m}$.

La fonction estimée prendra donc la forme :

$$\boxed{h(\mathbf{x}) \;=\; \sum_{i=1}^{m} \alpha_i \, \kappa(\mathbf{x}, \mathbf{x}_i)} \tag{14.13}$$

Il devient alors naturel de se demander quelle est la qualité de l'approximation fournie par h vis-à-vis d'une fonction cible f en fonction du nombre m de données et de la « représentativité » de l'échantillon $\{(\mathbf{x}_i)\}_{i=1,m}$ (on notera ici l'absence de référence explicite aux réponses mesurées u_i. Celles-ci interviendront naturellement dans l'expression des coefficients α_i).

Il est évidemment impossible de répondre à cette question, mal posée, sans hypothèses additionnelles. Cette connaissance *a priori* prend souvent la forme d'une préférence pour des fonctions hypothèse régulières ou issues d'un espace d'hypothèses de capacité limitée, comme nous l'avons vu au chapitre 2. Le théorème de représentation établit un lien fondamental entre les critères inductifs régularisés, réglant le compromis biais-variance, et les espaces hilbertiens à noyau reproduisant.

Critère inductif régularisé et espace d'hypothèses

On utilise généralement une version régularisée du risque empirique, de la forme :

$$\hat{h}(\mathbf{x}) \;=\; \underset{h\in\mathcal{H}}{\text{ArgMin}} \left\{ R_{\text{Emp}}(h, \mathcal{S}) + \lambda \, \text{Capacité}(\mathcal{H}) \right\} \tag{14.14}$$

où *Capacité* est un estimateur de la capacité de l'espace d'hypothèses à offrir des hypothèses de faible risque empirique quelque soit l'échantillon d'apprentissage,

ou de la forme :

$$\hat{h}(\mathbf{x}) \;=\; \underset{h\in\mathcal{H}}{\text{ArgMin}} \left\{ R_{\text{Emp}}(h, \mathcal{S}) + \lambda \, \text{Reg}(h) \right\} \tag{14.15}$$

où *Reg* est une fonction mesurant la « régularité » de l'hypothèse. (Par exemple $\text{Reg}(h) = ||h||_{\mathcal{H}_\kappa}$, avec $||h||^2_{\mathcal{H}_\kappa} := \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j \, \kappa(\mathbf{x}_i, \mathbf{x}_j)$).

Un théorème essentiel affirme que **toute fonction $\hat{h}$ minimisant un risque empirique régularisé admet une représentation de la forme** :

$$\hat{h}(\cdot) \;=\; \sum_{i=1}^{m} \alpha_i \, \kappa(\mathbf{x}_i, \cdot) \tag{14.16}$$

Plus précisément (voir, par exemple, [Her02a] pp.48-49 et 257-258 pour la preuve, ou [STV04], pp.47-48) :

Théorème 14.4 (Théorème de représentation (representer theorem))

Soit un noyau reproduisant $\kappa : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, un échantillon d'apprentissage $S_m \in (\mathcal{X} \times \mathcal{Y})^m$, et un risque empirique quelconque R_{Emp} (muni d'une fonction de perte ℓ). Soit Reg $: \mathbb{R} \to [0, \infty[$ *une fonction croissante strictement monotone. Soit $\mathcal{H}_\kappa$ l'espace hilbertien induit par le noyau reproduisant κ. Alors, toute fonction $\hat{h} \in \mathcal{H}_\kappa$ minimisant le risque régularisé :*

$$\hat{h}(\mathbf{x}) \;=\; \underset{h\in\mathcal{H}}{\text{ArgMin}} \left\{ R_{\text{Emp}}(h, \mathcal{S}) + \lambda \, \text{Reg}(h) \right\}$$

admet une représentation de la forme : $\hat{h}(\cdot) \;=\; \sum_{i=1}^{m} \alpha_i \, \kappa(\mathbf{x}_i, \cdot) \quad \alpha_i \in \mathbb{R}, \forall i.$

Cela signifie que la ou les hypothèses minimisant un risque empirique régularisé se trouve(nt) dans le sous-espace vectoriel des fonctions engendrées par les combinaisons linéaires de fonctions noyau (analysantes) dépendantes des données d'apprentissage : $\kappa(\mathbf{x}_i, \cdot)$. Il s'ensuit en particulier que la solution $\hat{h}$ est prise dans un espace de dimension m au maximum, même si l'espace $\mathcal{H}_\kappa$ est lui-même de dimension éventuellement infinie. De plus, bien entendu, il y aura un lien fort entre la forme des fonctions noyau κ et le type de régularisation imposée dans le critère inductif.

Pour résumer, trois motivations différentes :
- l'adaptation de la méthode des plus proches voisins,
- le passage par un espace de redescription et la dépendance des résultats sur des produits scalaires, et donc *in fine* sur des fonctions noyau,
- l'approche de l'analyse fonctionnelle et le théorème de représentation

conduisent à considérer l'espace des hypothèses $\mathcal{H}$ de la forme : $h(\cdot) = \sum_{i=1}^{m} \alpha_i \kappa(\mathbf{x}_i, \cdot)$.

2. Philosophie des méthodes à noyaux

2.1 Les méthodes linéaires remises au goût du jour

La section 1.2 a fourni une première illustration de l'utilisation de fonctions noyau comme moyen de transformer la recherche de régularités non-linéaires en recherche de régularités linéaires grâce au passage par un espace de redescription virtuel F (voir la figure 14.1). **Quatre grandes idées** sous-tendent l'approche générale :

1. Les données décrites dans l'espace d'entrée $\mathcal{X}$ sont projetées dans un espace vectoriel de redescription F
2. Des régularités linéaires sont cherchées dans cet espace F
3. Les algorithmes de recherche n'ont pas besoin de connaître les coordonnées des projections des données dans F, mais seulement leurs produits scalaires
4. Ces produits scalaires peuvent être calculés efficacement grâce à l'utilisation de fonctions noyau.

Cette approche permet ainsi d'employer tout l'arsenal des méthodes linéaires développées jusque dans les années soixante pour découvrir des relations non-linéaires dans les données. Au fil des années récentes ont été ainsi revisitées l'analyse en composantes principales (ACP), la méthode des filtres de Kalman, des méthodes linéaires de clustering, la discrimination linéaire de Fisher, etc.

La figure 14.3 résume les étapes génériques de la méthode. Il est crucial de réaliser que, dans cette approche, l'essentiel des informations sur les données, en dehors de leurs étiquettes, s'exprime dans la *matrice noyau* K, ou encore *matrice de Gram*, qui encode les produits scalaires entre les projections des données d'apprentissage $\mathbf{G} = \langle \Phi(\mathbf{X}), \Phi(\mathbf{X}^\top) \rangle$.

2.2 L'importance de la matrice de Gram

La matrice de Gram (ou matrice noyau) contient toute l'information utilisée par les méthodes à noyaux sur les données. Elle est symétrique : $\mathbf{G}_{ij} = \mathbf{G}_{ji}$. L'utilisation exclusive des informations contenues dans cette matrice à propos des données d'apprentissage a pour conséquence qu'une partie des informations sur ces données est perdue. Par exemple, cette matrice est invariante par

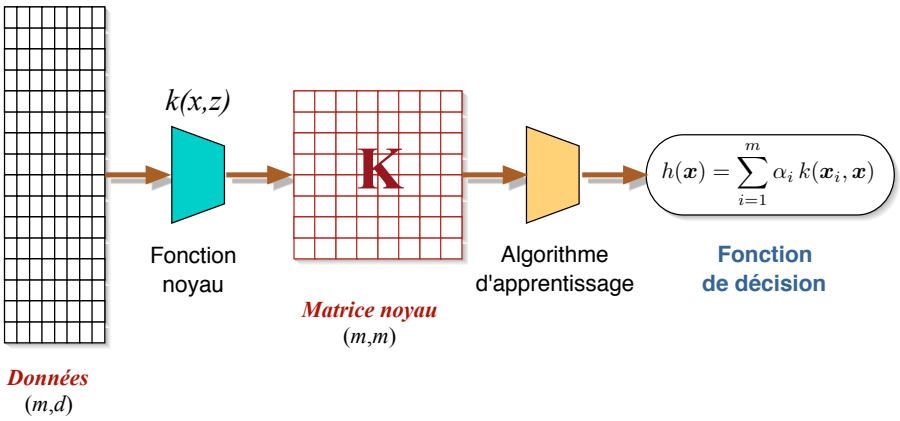


FIG. 14.3: *Chaîne de traitements générique des méthodes à noyaux.*

rotation des points dans l'espace d'entrée $\mathcal{X}$. La position angulaire des données par rapport à un référentiel est donc ignorée, de même que les alignements éventuels des données sur les axes.

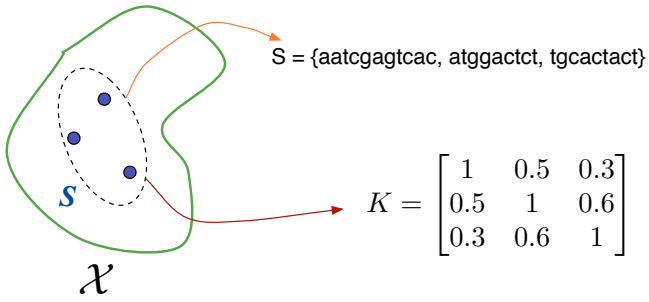


FIG. 14.4: *Les méthodes à noyau remplacent les informations sur les objets comparés par la matrice de leurs distances : la matrice de Gram. C'est un changement de perspective radical.*

Étant donné le caractère fondamental de la matrice de Gram dans la traduction des données, il n'est pas surprenant que l'on puisse estimer certaines propriétés de l'apprentissage à partir de l'examen des caractéristiques de cette matrice. De même, il est possible de modifier l'apprentissage en effectuant certaines opérations sur la matrice de Gram préalablement à l'apprentissage.

Ainsi, si il existe une structure dans les données, elle doit se refléter dans la matrice de Gram, si, du moins, la fonction noyau choisie est appropriée pour détecter les similarités sous-jacentes. Cette fonction noyau agit de fait comme un filtre sensible à certaines « fréquences » et pas à d'autres. Par exemple, si la fonction noyau est mal choisie, ou si les données ne présentent pas de structures, les éléments de la matrice de Gram auront des valeurs indifférenciées. Toute entrée est alors proche de n'importe quel autre point, et tous les éléments sont classés dans une seule classe. C'est du sous-apprentissage sévère. Dans les cas où les éléments hors diagonale sont de valeur proche de 0, seul un apprentissage par cœur devient possible.

Inversement, un moyen d'influencer l'apprentissage est de modifier la matrice de Gram. Ainsi, par exemple, en vertu de l'équation (14.5), l'ajout d'une constante sur les termes diagonaux de la matrice de gram $\mathbf{G}$ revient à accentuer la régularisation.

3. Les Séparatrices à Vaste Marge (SVM)

Nous nous intéressons ici à la *classification binaire* c'est-à-dire à l'approximation de fonctions de décision permettant de distinguer entre deux classes, et ceci par des hyperplans $h : \mathcal{X} \to \mathcal{Y}$, d'équation $h(\mathbf{x}) = \langle \mathbf{w}\,,\,\mathbf{x}\rangle + w_0 = \sum_{i=0}^{d} w_i x_i = 0$, en supposant que $\mathcal{X}$ soit de dimension d (par exemple, $\mathcal{X} = \mathbb{R}^d$). On utilisera alors la fonction de décision $g(\mathbf{x}) = \text{signe}\big(\langle \mathbf{w}\,,\,\mathbf{x}\rangle + w_0\big)$ pour prédire la classe ($+$ ou $-$) de l'entrée $\mathbf{x}$.

Selon le principe de minimisation du risque empirique classique, un tel « meilleur » hyperplan sera celui qui minimise le nombre d'exemples mal classés par cette fonction de décision dans l'échantillon d'apprentissage $\mathcal{S} = \langle(\mathbf{x}_1, u_1), \ldots, (\mathbf{x}_m, u_m)\rangle$. Ce critère conduit par exemple à l'algorithme du perceptron (voir section 3.3).

Vapnik et ses collègues ([BGV92]) ont proposé de s'intéresser plutôt à un critère de *confiance* ou encore de *robustesse* de la fonction de décision obtenue. Ce critère découle de l'étude de la théorie statistique de l'apprentissage conduite en particulier par Vapnik et Chervonenkis depuis les années soixante ([VC71]). Un moyen de traduire la robustesse de la fonction de décision, ou encore du classifieur, est de considérer la *marge* séparant les exemples de la classe '+' des exemples de la classes '−'.

La distance d'un point $\mathbf{x}'$ à l'hyperplan d'équation $h(\mathbf{x}) = \mathbf{w}^\top\mathbf{x} + w_0$ est égale à : $\langle \mathbf{w}, \mathbf{x}'\rangle$ puisque $\mathbf{w}$ est un vecteur orthogonal à $h(\mathbf{x})$. Afin de pouvoir comparer plusieurs hyperplans séparateurs de vecteurs directeurs $\mathbf{w}$ différents, on normalise la distance en la divisant par la norme du vecteur $\mathbf{w}$: $\langle \mathbf{w}\,,\,\mathbf{x}\rangle / \| \mathbf{w} \|$

Lorsqu'il existe une séparatrice linéaire entre les points d'apprentissage, il en existe en général une infinité. On peut alors chercher parmi ces séparatrices celle qui sépare « au mieux » les deux nuages de points exemples et contre-exemples. Cet hyperplan optimal est défini par :

$$\underset{\mathbf{w},w_0}{\text{Argmax}} \;\; \min\{\| \mathbf{x} - \mathbf{x}_i \| \;:\; \mathbf{x} \in \mathbb{R}^d \,,\; (\mathbf{w}^\top\mathbf{x} + w_0) = 0 \,,\; i = 1, \ldots, m\}$$

c'est-à-dire l'hyperplan dont la distance minimale aux exemples d'apprentissage (voir figure 14.5) est la plus grande possible. Dans ce cas, la marge normalisée, appelée aussi *marge géométrique* vaut : $2/\| \mathbf{w} \|$.

Chercher un hyperplan séparateur de marge maximale présente plusieurs avantages. D'abord, même si les arguments théoriques sont délicats (voir chapitre 21), il est montré que *la capacité de l'espace d'hypothèses constitué par les hyperplans séparateurs diminue lorsque la marge augmente*. Ainsi, on peut espérer que l'hyperplan de marge maximale séparant les exemples des deux classes est l'hypothèse satisfaisant au mieux un risque empirique régularisé, favorisant donc la performance en généralisation. Ensuite, il existe un unique hyperplan séparateur de marge maximale et, le *problème* étant *convexe*, il peut être résolu par des méthodes de programmation quadratique, donc de manière efficace.

3.1 La résolution du problème d'optimisation

3.1.1 Formulation primale du problème

La recherche de l'hyperplan optimal revient donc à résoudre le problème d'optimisation suivant qui porte sur les paramètres $\mathbf{w}$ et w_0 :

$$\begin{cases} \text{Minimiser} & \frac{1}{2}\| \mathbf{w} \|^2 \\ \text{sous les contraintes} & u_i\,(\mathbf{w}^\top\mathbf{x}_i + w_0) \geq 1 \quad i = 1, \ldots, m \end{cases} \tag{14.17}$$

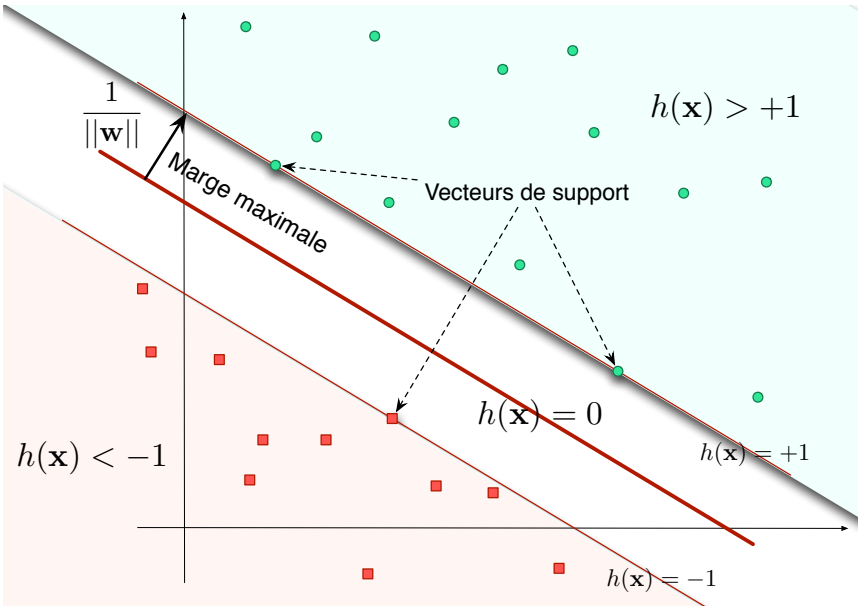


FIG. 14.5: *L'hyperplan optimal est perpendiculaire au segment de droite le plus court joignant un exemple d'apprentissage à l'hyperplan. Ce segment a pour longueur* $\frac{1}{\|\mathbf{w}\|}$ *lorsqu'on normalise convenablement les paramètres* $\mathbf{w}$ *et* w_0.

Cette écriture du problème, appelée *formulation primale*, implique le réglage de $d+1$ paramètres, d étant la dimension de l'espace des entrées $\mathcal{X}$. Cela est possible avec des méthodes de programmation quadratique[3] pour des valeurs de d assez petites, mais devient inenvisageable pour des valeurs de d dépassant quelques centaines. En effet, les calculs sont de complexité en $\mathcal{O}(d^3)$. Heureusement, il existe une transformation de ce problème dans une formulation duale que l'on peut résoudre en pratique.

3.1.2 Formulation duale du problème

D'après la théorie de l'optimisation, un problème d'optimisation possède une forme duale dans le cas où la fonction objectif et les contraintes sont strictement convexes. Dans ce cas, la résolution de l'expression duale du problème est équivalente à la solution du problème original. Ces critères de convexité sont réalisés dans le problème (14.17). Pour résoudre ces types de problèmes, on utilise une fonction que l'on appelle *lagrangien* qui incorpore des informations sur la fonction objectif et sur les contraintes et dont le caractère stationnaire peut être utilisé pour détecter des solutions. Plus précisément, le lagrangien est défini comme étant la somme de la fonction objectif et d'une combinaison linéaire des contraintes dont les coefficients $\alpha_i \geq 0$ sont appelés *multiplicateurs de Lagrange* ou encore *variables duales*.

$$L(\mathbf{w}, w_0, \alpha) \;=\; \frac{1}{2}\| \mathbf{w} \|^2 \;-\; \sum_{i=1}^{m} \alpha_i \,(u_i.((\mathbf{w}\mathbf{x}_i^\top) \;+\; w_0) \;-\; 1) \tag{14.18}$$

Un théorème de Kuhn-Tucker, couronnant des travaux commencés par Fermat (1601-1665), puis poursuivis par Lagrange (1736-1813), démontre que le problème primal et sa formulation

[3] Les problèmes d'optimisation pour lesquels la fonction et les contraintes sont linéaires ressortent des techniques de programmation linéaire, tandis que les problèmes d'optimisation dans lesquels la fonction est quadratique et les contraintes linéaires ressort des techniques de la programmation quadratique : voir le chapitre 3.

duale ont la même solution. Celle-ci correspond à un *point-selle* du lagrangien (il faut le minimiser par rapport aux variables primaires $\mathbf{w}$ et w_0 et le maximiser par rapport aux variables duales α_i).

Au point-selle, la dérivée du lagrangien par rapport aux variables primaires doit s'annuler :

$$\frac{\partial L}{\partial \mathbf{w}}(\mathbf{w}, w_0, \alpha) \;=\; 0, \quad \frac{\partial L}{\partial w_0}(\mathbf{w}, w_0, \alpha) \;=\; 0 \tag{14.19}$$

d'où :

$$\sum_{i=1}^{m} \alpha_i \, u_i \;=\; 0 \tag{14.20}$$

et :

$$\mathbf{w}^* \;=\; \sum_{i=1}^{m} \alpha_i \, u_i \, \mathbf{x}_i \tag{14.21}$$

Nous vérifions ici le théorème de représentation selon lequel le vecteur paramètres $\mathbf{w}^*$ de la fonction optimisant le critère régularisé est combinaison linéaire impliquant les exemples d'apprentissage $\mathbf{x}_1, \ldots, \mathbf{x}_m$.

Plus précisément, il est montré (*conditions de Karush-Kuhn-Tucker*) que seuls les points qui sont sur les hyperplans frontière $(\mathbf{w}^* \mathbf{x}_i^\top) \;+\; w_0^* \;=\; \pm 1$ jouent une rôle. Ces points pour lesquels les multiplicateurs de Lagrange sont non nuls sont appelés *vecteurs de support* par Vapnik. Nous utiliserons aussi le terme plus imagé d'*exemples critiques* puisque ce sont eux qui déterminent l'hyperplan optimal, tandis que les autres exemples ne jouent pas de rôle dans cette analyse[4].

L'hypothèse optimale présente ainsi deux **propriétés remarquables de *parcimonie***. D'une part, elle ne s'exprime qu'à l'aide d'un sous-ensemble (éventuellement très restreint) des exemples d'apprentissage, les exemples critiques. D'autre part, parce qu'elle correspond à la solution la plus « robuste », elle tolère d'être spécifiée avec moins de précision, donc avec moins de bits. On retrouve là un lien avec les principes inductifs par compression d'information (par exemple le « principe de minimisation de la taille de description » ou MDLP) (voir chapitre 21).

En substituant (14.20) et (14.21) dans (14.19), on élimine les variables primaires et l'on obtient la *forme duale* du problème d'optimisation, dans laquelle on cherche les multiplicateurs de Lagrange tels que :

$$\left\{ \begin{array}{l} \mathrm{Max}_\alpha \left[\sum_{i=1}^m \alpha_i \;-\; \frac{1}{2} \; \sum_{i,j=1}^m \alpha_i \alpha_j u_i u_j \langle \mathbf{x}_i \, , \, \mathbf{x}_j \rangle \right] \\ \\ \text{avec} \quad \left\{ \begin{array}{l} \alpha_i \geq 0, \quad i = 1, \ldots, m \\ \sum_{i=1}^m \alpha_i u_i \;=\; 0 \end{array} \right. \end{array} \right. \tag{14.22}$$

L'**hyperplan solution** correspondant peut alors être écrit :

$$\boxed{h^*(\mathbf{x}) \;=\; (\mathbf{w}^\star \; \mathbf{x}) \;+\; w_0^\star \;=\; \sum_{i=1}^{m} \alpha_i^\star \, u_i \, . \, \langle \mathbf{x}_i \, , \, \mathbf{x} \rangle \;+\; w_0^\star} \tag{14.23}$$

où les $\alpha_i^\star$ sont solution de (14.22) et w_0 est obtenue en utilisant n'importe quel exemple critique $(\mathbf{x}_c, u_c)$ dans l'équation :

$$w_0 \;=\; u_c - \mathbf{w}^\top \mathbf{x}_c \;=\; u_c \;-\; \sum_{i=1}^{m_c} u_i \alpha_i \mathbf{x}_i^\top \mathbf{x}_c \tag{14.24}$$

[4] On pourrait au contraire estimer que la distance des autres exemples à l'hyperplan séparateur doit participer aussi à l'évaluation de la qualité de l'hyperplan. C'est le cas dans l'analyse PaC-bayésienne des SVM.

où m_c est le nombre d'exemples critiques, c'est-à-dire d'exemples situés sur la marge. Pour des raisons de stabilité numérique, on fait en général ce calcul pour tous les exemples critiques et on calcule w_0 en prenant la moyenne des résultats.

Deux choses sont remarquables. D'abord, *l'hyperplan solution ne requiert que le calcul des produits scalaires* $\langle \mathbf{x}_i \,.\, \mathbf{x} \rangle$ entre des vecteurs de l'espace d'entrée $\mathcal{X}$. C'est ce qui va permettre d'utiliser l'astuce des noyaux. Ensuite, *la solution ne dépend plus de la dimension d de l'espace d'entrée*, mais dépend de la taille m de l'échantillon de données et même, plus précisément, du nombre m_c d'exemples critiques qui est généralement bien inférieur à m. Les méthodes d'optimisation quadratique standards suffisent donc pour de nombreux problèmes pratiques.

3.2 Le cas d'un échantillon non linéairement séparable dans $\mathcal{X}$

L'analyse précédente suppose que les classes d'exemples sont linéairement séparables dans l'espace d'entrée $\mathcal{X}$. Ce cas est rare dans la pratique et c'est ce qui a motivé en particulier le développement des perceptrons multi-couche (chapitre 10). En un sens, les couches cachées de ces réseaux connexionnistes calculent de nouvelles corrélations et, partant, de nouveaux descripteurs des formes d'entrée, ce qui permet à la couche de sortie de trouver une séparatrice simple, éventuellement linéaire, entre les entrées ainsi re-décrites. La question est naturellement d'identifier les bons descripteurs. Dans le cas des réseaux connexionnistes, une manière de faire est d'introduire des couches cachées complètement connectées entre elles et avec les couches d'entrée et de sortie et de faire confiance à l'algorithme de calcul des poids des connexions pour découvrir les corrélations pertinentes. Une autre manière de faire est de réfléchir à la nature du problème et d'introduire les bons blocs de neurones, comme par exemple dans le réseau à convolution `LeNet` développé pour reconnaître les chiffres manuscrits [MBB+92].

L'approche par fonctions noyau offre une autre possibilité. Nous l'avons vu en section 1.2, les fonctions noyau sont associées à un produit scalaire dans un espace de redescription, or un produit scalaire est un type de corrélation particulier entre les points de l'espace de redescription F, c'est-à-dire entre les projections des formes d'entrée $\Phi(\mathbf{x})$. L'utilisation de fonctions noyau va donc permettre d'exploiter automatiquement un certain type de corrélation entre les formes d'entrée, et, si possible, d'identifier les exemples critiques par rapport auxquels une frontière de décision pourra être trouvée. Le choix de la (ou des) fonction noyau va naturellement implicitement déterminer le type de corrélations considérées (voir section 5). Ainsi, par exemple, si l'on estime que ce sont des corrélations mettant en jeu des sous-régions de 10 pixels dans une image qui permettent de distinguer des classes d'objets, on choisira une fonction noyau permettant de tester toutes les corrélations de 10 pixels au moins.

Soit donc Φ une transformation non linéaire de l'espace d'entrée $\mathcal{X}$ en un *espace de redescription* $\Phi(\mathcal{X})$:

$$\mathbf{x} \;=\; (x_1, \ldots, x_d)^\top \;\;\mapsto\;\; \Phi(\mathbf{x})^\top \;=\; (\phi_1(\mathbf{x}), \ldots, \phi_d(\mathbf{x}), \ldots)^\top \tag{14.25}$$

Le problème d'optimisation se transcrit dans ce cas par :

$$\left\{ \begin{array}{l} \mathrm{Max}_\alpha \left\{ \sum_{i=1}^m \alpha_i \;-\; \frac{1}{2} \;\sum_{i,j=1}^m \alpha_i \alpha_j u_i u_j \langle \Phi(\mathbf{x}_i) \,,\, \Phi(\mathbf{x}_j) \rangle \right\} \\ \alpha_i \geq 0, \quad i = 1, \ldots, m \\ \sum_{i=1}^m \alpha_i u_i \;=\; 0 \end{array} \right. \tag{14.26}$$

L'équation de l'hyperplan séparateur dans le nouvel espace devient :

$$h(\mathbf{x}) \;\;=\;\; \sum_{i=1}^{m} \alpha_i^\star \, u_i \, \langle \Phi(\mathbf{x}) \,.\, \Phi(\mathbf{x}_i) \rangle \;+\; w_0^\star \tag{14.27}$$

où les coefficients $\alpha_i^\star$ et $w_0^\star$ sont obtenus comme précédemment par résolution de (14.26).

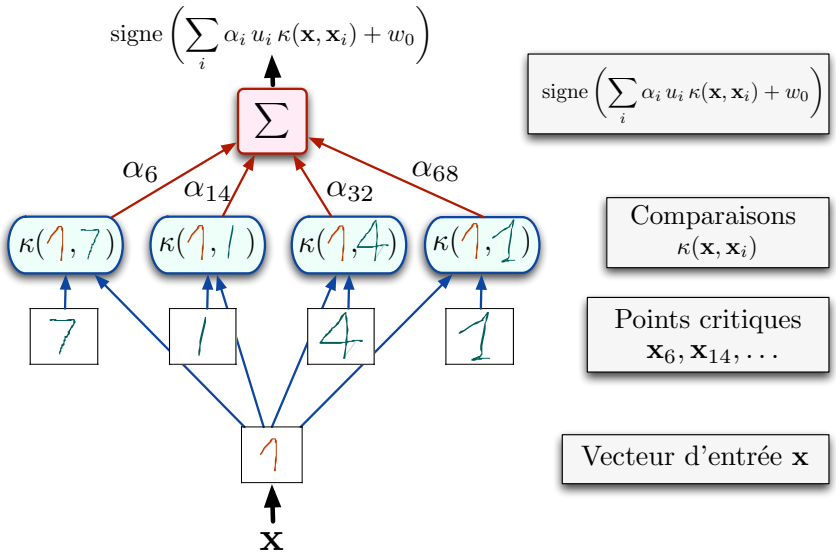


FIG. 14.6: *Cette figure résume le fonctionnement des séparateurs à vastes marges et montre le rôle des fonctions noyau. Lors de l'apprentissage, ici de chiffres manuscrits, un certain nombre d'exemples critiques sont retenus pour définir la fonction de décision (ici deux exemples positifs de '1' et deux exemples négatifs, un '4' et un '7'). Lorsqu'une nouvelle entrée est présentée au système, elle est comparée aux exemples critiques à l'aide des fonctions noyau qui réalisent un produit scalaire dans l'espace de redescription $\Phi(\mathcal{X})$. La sortie est calculée en faisant une somme pondérée (une combinaison linéaire) de ces comparaisons.*

Comme nous l'avons vu, les fonctions noyau correspondent à un (ou des) produits scalaires dans un espace de redescription F. On peut donc remplacer le problème précédent dans lequel il fallait déterminer la bonne transformation non linéaire Φ, par celui du choix d'une bonne fonction noyau $\kappa(\cdot, \cdot)$. Le problème d'optimisation associé devient alors :

$$\left\{ \begin{array}{l} \mathrm{Max}_\alpha \left\{ \sum_{i=1}^m \alpha_i \;-\; \frac{1}{2} \sum_{i,j=1}^m \alpha_i \alpha_j u_i u_j \kappa(\mathbf{x}_i, \mathbf{x}_j) \right\} \\ \alpha_i \geq 0, \quad i = 1, \ldots, m \\ \sum_{i=1}^m \alpha_i u_i \;=\; 0 \end{array} \right. \tag{14.28}$$

dont la solution est l'hyperplan séparateur donnant la fonction de décision :

$$\boxed{h(\mathbf{x}) \;=\; \mathrm{signe}\left\{ \sum_{i=1}^m \alpha_i^\star\, u_i\, \kappa(\mathbf{x}, \mathbf{x}_i) \;+\; w_0^\star \right\}} \tag{14.29}$$

où les coefficients $\alpha_i^\star$ et $w_0^\star$ sont obtenus comme précédemment par résolution de (14.28). Le processus total est illustré par la figure 14.6.

3.3 Échantillon difficilement séparable et marge douce (ou poreuse)

Nous avons supposé jusqu'ici qu'il existait un hyperplan (éventuellement dans l'espace de redescription) permettant de séparer les exemples des deux classes. Or, d'une part, il n'est pas

nécessairement souhaitable de rechercher absolument un tel hyperplan, cela peut en effet conduire à une suradaptation aux données, d'autre part, il se peut que du bruit ou des erreurs dans les données ne permette tout simplement pas de trouver un tel hyperplan. Pour ces raisons, une version moins contrainte du problème de la recherche d'une séparatrice à vastes marges est le plus souvent considérée.

L'idée est de pénaliser les séparatrices admettant des exemples qui ne sont pas du bon côté des marges, sans cependant interdire une telle possibilité. On définit pour ce faire une fonction de coût particulière introduisant une pénalité pour tous les exemples mal classés et qui sont à une distance ξ de la marge qu'ils devraient respecter. On considère généralement des fonctions de coût qui soient compatibles avec la fonction de perte traditionnelle $\{0,1\}$-perte qui compte un coût de 1 pour chaque exemple mal classé. En particulier, on cherche des fonctions de coût qui conduisent à un critère inductif compatible avec le critère classique de minimisation du nombre d'exemples mal classés et donc à des solutions optimales compatibles. On parle de *fonctions de coût de substitution* (« *surrogate loss functions* ») Toute une gamme de fonctions sont possibles. Les deux plus étudiées sont la « fonction coude » (*hinge loss*) et la *fonction quadratique assymétrique* (voir figure 14.7). On parle alors de « marges douces » (*soft margin*)[5].

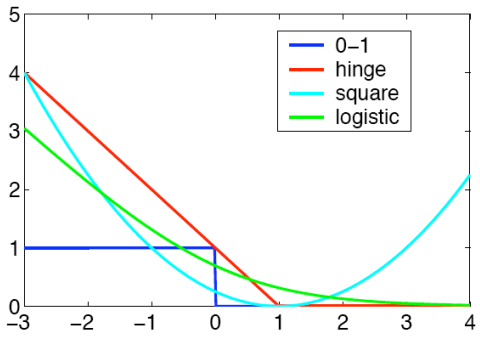


FIG. 14.7: *Approximations de la fonction de perte* $\{0,1\}$ *(ligne faisant une marche) par les fonction de perte « coude » (ligne), « quadratique » (courbe symétrique en 1), et « logistique » (courbe en descente douce). L'axe des abscisses correspond à la quantité* $y\,h(\mathbf{x})$ *qui est négative si l'exemple* $\mathbf{x}$ *est mal classé par* h*.*

On a alors :

$$\ell_{0-1}(h(\mathbf{x}),y) = \mathbf{I}_{-y\,h(\mathbf{x})>0} \;\leq\; \max\{1-y\,h(\mathbf{x}),0\} = \ell_{\text{lin}}(h(\mathbf{x}),y) \tag{14.30}$$

$$\ell_{0-1}(h(\mathbf{x}),y) = \mathbf{I}_{-y\,h(\mathbf{x})>0} \;\leq\; \max\{1-y\,h(\mathbf{x}),0\}^2 = \ell_{\text{quad}}(h(\mathbf{x}),y) \tag{14.31}$$

Approximation linéaire (fonction coude)

On cherche maintenant à minimiser un risque régularisé avec un taux de compromis λ :

$$R_{\text{reg}}(h) \;=\; \frac{1}{m}\sum_{i=1}^{m} \ell_{\text{lin}}(h(\mathbf{x}),u_i) \;+\; \lambda \parallel h \parallel^2 \tag{14.32}$$

[5] En raison du manque d'espace, nous ne discutons pas plus en détail les *fonctions de coût de substitution* ici. Il faut cependant être conscient que pour être valables, ces fonctions doivent vérifier un certain nombre de conditions. Leur choix doit se faire en fonction de la tâche : classification, régression, estimation de densité. De même, il est parfois utile d'avoir une fonction de classification *auto-calibrée*, c'est-à-dire pouvant être interprétée en terme de probabilité d'appartenance à la classe calculée. Ici encore, le choix de la fonction de coût de substitution est important. (voir par exemple [SC08] pour une étude détaillée et théorique.

soit encore :

$$\begin{cases} \text{Maximiser} & \sum_{i=1}^{m} \xi_i \, + \, \lambda \, m \parallel \mathbf{w} \parallel^2 \\ \text{avec} & \begin{cases} u_i \left\langle \mathbf{x}_i, \mathbf{w} \right\rangle \geq 1 - \xi_i \qquad i = 1, \ldots, m \\ \xi \geq 0 \end{cases} \end{cases} \tag{14.33}$$

Le passage par la formulation duale conduit à :

$$\begin{cases} \text{Max}_{\alpha} \left[\sum_{i=1}^{m} \alpha_i \; - \; \frac{1}{2} \; \sum_{i=1}^{m} \; \sum_{j=1}^{m} \alpha_i \alpha_j u_i u_j \kappa(\mathbf{x}_i \, , \, \mathbf{x}_j) \right] \\ \forall i, \quad 0 \leq \alpha_i \leq C = \frac{1}{2\lambda m} \\ \sum_{i=1}^{m} \alpha_i u_i \; = \; 0 \end{cases} \tag{14.34}$$

Le coefficient C règle le compromis entre la marge possible entre les exemples et le nombre d'erreurs admissibles. Il doit être choisi par l'utilisateur. Une valeur de C grande correspond à une valeur de λ petite, c'est-à-dire à une grande pénalisation associée à chaque exemple mal classé. Il est à noter que puisque $C = \frac{1}{2\lambda m}$, l'intervalle des valeurs possibles pour les α_i décroît lorsque le nombre d'exemples m augmente.

Pour le cas d'une ***fonction de coût quadratique***, moins utilisée, nous reportons le lecteur par exemple à [Her02a], pp.55-56.

3.4 Utilisation de fonctions noyau : illustrations

—— Exemple **avec $\mathcal{X} = \mathbb{R}$** ——

Afin d'illustrer le fonctionnement des méthodes à noyaux, nous allons considérer un des problèmes de discrimination les plus simples imaginables. On suppose que l'espace des entrées est réduit à $\mathbb{R}$, c'est-à-dire à une dimension. On suppose que l'on a cinq points d'apprentissage associés à deux classes '+1' et '−1' :

$$S \; = \; \{(\mathbf{x}_1 = 1, u_1 = 1), (\mathbf{x}_2 = 1, u_2 = 1), (\mathbf{x}_3 = 4, u_3 = -1), (\mathbf{x}_4 = 5, u_4 = -1), (\mathbf{x}_5 = 6, u_5 = 1)\}$$

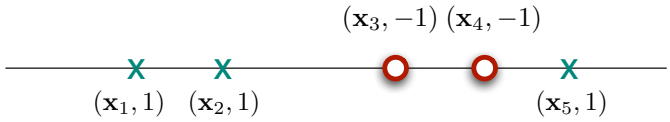


Fig. 14.8: *Un problème de discrimination à une dimension.*

Il n'est pas possible de trouver une séparatrice linéaire (un seul seuil sur l'axe $\mathbb{R}$) permettant de distinguer les exemples '1' des exemples '−1'.

Supposons que l'on choisisse d'utiliser la fonction noyau : $\kappa(\mathbf{x}, \mathbf{x}') \; = \; (\mathbf{x} \cdot \mathbf{x}' + 1)^2$. Cela signifie que l'on est disposé à considérer les corrélations au deuxième ordre entre les positions des exemples.

En choisissant $C = 100$, le problème d'optimisation associé est :

$$\begin{cases} \max\limits_{\alpha} \sum\limits_{i=1}^{5} \alpha_i - \frac{1}{2} \sum\limits_{i,j=1}^{5} \alpha_i \alpha_j u_i u_j \, (\mathbf{x}_i \cdot \mathbf{x}_j + 1)^2 \\ \sum\limits_{i,j=1}^{5} \alpha_i u_i \; = \; 0 \\ 0 \leq \alpha_i \leq 100 \quad (\forall i) \end{cases}$$

Un programme de résolution de problème quadratique de ce type retourne la solution :

$$\alpha_1 = 0 \ ; \ \alpha_2 = 2.5 \ ; \ \alpha_3 = 0 \ ; \ \alpha_4 = 7.333 \ ; \ \alpha_5 = 4.833$$

La frontière de décision correspondante, de la forme $D(\mathbf{x}) = \sum_{i=1}^{m} \alpha_i^\star \, u_i \, \kappa(\mathbf{x}, \mathbf{x}_i) + w_0^\star$, est ainsi :

$$\begin{aligned} D(\mathbf{x}) &= 2.5\,(1)\,(2x+1)^2 + 7.333\,(1)\,(5x+1)^2 + 4.833\,(1)\,(6x+1)^2 + w_0 \\ &= 0.6667\,x^2 - 5.333\,x + w_0 \end{aligned}$$

où w_0 est obtenue en résolvant $h(2) = 1$, ou $h(5) = -1$ ou $h(6) = 1$, puisque $\mathbf{x}_2$, $\mathbf{x}_4$ et $\mathbf{x}_5$ sont sur la droite $u_i(\mathbf{w}^\top \Phi(\mathbf{x}) + w_0 = 1$ pour $i \in \{2, 4, 5\}$. Ce qui donne $w_0 = 9$, d'où :

$$h(\mathbf{x}) = \text{signe}\{0.6667\,x^2 - 5.333\,x + 9\}$$

C'est une frontière quadratique qui se traduit par deux seuils dans l'espace d'origine $\mathcal{X} = \mathbb{R}$. Les exemples $\mathbf{x}_2, \mathbf{x}_4, \mathbf{x}_5$ sont les exemples critiques.

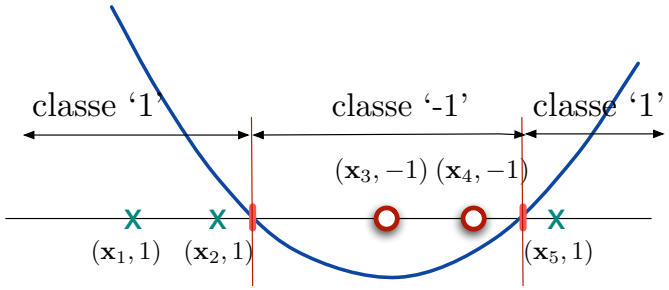
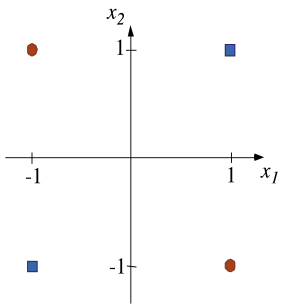


FIG. 14.9: *Un problème de discrimination à une dimension. La séparatrice est issue d'une fonction à deux dimensions qui se traduit par deux seuils sur* $\mathcal{X} = \mathbb{R}$.

EXEMPLE **avec $\mathcal{X} = \mathbb{R}^2$: la fonction XOR**

Soient de données décrites en deux dimensions : $\mathcal{X} = \mathbb{R}^2$, et soient les exemples d'apprentissage décrits dans la table suivante :

Indice i	$\mathbf{x}_i$	u_i
1	$(1, 1)$	1
2	$(1, -1)$	-1
3	$(-1, -1)$	1
4	$(-1, 1)$	-1



Supposons que l'on choisisse la fonction noyau polynomiale de degré 2 :

$$\begin{aligned} \kappa(\mathbf{x}, \mathbf{x}') &= \big(1 + (\mathbf{x} \cdot \mathbf{x}')\big)^2 \\ &= \big(1 + (x_1 x_1' + x_2 x_2')\big)^2 \\ &= 1 + 2(x_1 x_1' + x_2 x_2') + (x_1 x_1' + x_2 x_2')^2 \\ &= 1 + 2\,x_1 x_1' + 2\,x_2 x_2' + x_1^2 {x_1'}^2 + x_2^2 {x'}_2^2 + 2\,x_1 x_1' x_2 x_2' \end{aligned}$$

qui correspond à la projection Φ :

$$(x_1, x_2)^\top \quad \stackrel{\Phi}{\mapsto} \quad (1, \sqrt{2}\,x_1,\ \sqrt{2}\,x_2,\ x_1^2,\ x_2^2,\ \sqrt{2}\,x_1x_2)^\top$$

On doit alors résoudre le problème :

$$\left\{ \begin{array}{l} \max\limits_{\alpha} \sum\limits_{i=1}^{4} \alpha_i - \frac{1}{2} \sum\limits_{i,j=1}^{4} \alpha_i\alpha_j u_i u_j \,(\mathbf{x}_i \cdot \mathbf{x}_j + 1)^2 \\ \sum\limits_{i,j=1}^{4} \alpha_i u_i \ = \ 0 \\ 0 \leq \alpha_i \leq C \quad (\forall i) \end{array} \right.$$

où l'on doit maximiser :

$$\begin{aligned} &\alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 \\ &\quad -\frac{1}{2}(9\alpha_1^2 - 2\alpha_1\alpha_2 - 2\alpha_1\alpha_3 + 2\alpha_1\alpha_4 + \\ &\qquad 9\alpha_2^2 + 2\alpha_2\alpha_3 - 2\alpha_2\alpha_4 + 9\alpha_3^2 - 2\alpha_3\alpha_4 + 9\alpha_4^2) \end{aligned}$$

Soit, en dérivant par rapport à chaque multiplicateur de Lagrange α_i :

$$\left\{ \begin{array}{lcl} 9\alpha_1 - \alpha 2 - \alpha_3 + \alpha_4 &=& 1 \\ \alpha_1 - 9\alpha 2 - \alpha_3 + \alpha_4 &=& 1 \\ \alpha_1 - \alpha 2 - 9\alpha_3 + \alpha_4 &=& 1 \\ \alpha_1 - \alpha 2 - \alpha_3 + 9\alpha_4 &=& 1 \end{array} \right.$$

On trouve alors la valeur optimale des multiplicateurs de Lagrange :

$$\alpha_1^\star \ = \ \alpha_2^\star \ = \ \alpha_3^\star \ = \ \alpha_4^\star \ = \ \frac{1}{8}$$

Les quatre exemples sont donc tous des exemples critiques, et la fonction de décision est alors :

$$h(\mathbf{x}) \ = \ \text{signe}\left\{ \sum_{i=1}^{m_s} \alpha_i^\star \, u_i \, \kappa(\mathbf{x}, \mathbf{x}_i) + w_0 \right\} \ = \ \text{signe}\left\{ \frac{1}{8} \sum_{i=1}^{4} u_i \, [(\mathbf{x}^\top \cdot \mathbf{x}_i) + 1]^2 \right\}$$

soit :

$$h(\mathbf{x}) \ = \ \text{signe}\left\{ \frac{1}{8} \sum_{i=1}^{4} u_i \, [1 + (\mathbf{x} \cdot \mathbf{x}_i)]^2 \right\}$$

Tous calculs faits, on arrive à l'*équation de la séparatrice dans l'espace d'entrée* $\mathcal{X}$:

$$h(\mathbf{x}) \ = \ \text{signe}\{-x_1x_2\}$$

qui est une fonction non linéaire (voir figure 14.10, à gauche).
Quelle est l'équation de la séparatrice dans l'espace de redescription F ? On a :

$$D(\mathbf{x}) = \sum_{i=1}^{4} \alpha_i^\star u_i \kappa(\mathbf{x}_i, \mathbf{x}) + w_0 = \sum_{i=1}^{4} \alpha_i^\star u_i \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}) \rangle + w_0 = {\mathbf{w}^\star}^\top \Phi(\mathbf{x}) + w_0$$

avec : $\mathbf{w}^\star = \sum_{i=1}^{4} \alpha_i^\star u_i \Phi(\mathbf{x}_i)$, et $w_0 = 0$ dans ce problème symétrique par rapport à l'origine.
On a alors : $\mathbf{w}^\star = \frac{1}{8}[-\Phi(\mathbf{x}_1) + \Phi(\mathbf{x}_2) + \Phi(\mathbf{x}_3) - \Phi(\mathbf{x}_4)]$.

En calculant les projections $\Phi(\mathbf{x}_i)$ des quatre points, on trouve :

$$\mathbf{w}^\star = \frac{1}{8}\left\{ -\begin{pmatrix} 1 \\ \sqrt{2} \\ \sqrt{2} \\ 1 \\ 1 \\ \sqrt{2} \end{pmatrix} + \begin{pmatrix} 1 \\ \sqrt{2} \\ -\sqrt{2} \\ 1 \\ 1 \\ -\sqrt{2} \end{pmatrix} + \begin{pmatrix} 1 \\ -\sqrt{2} \\ -\sqrt{2} \\ 1 \\ 1 \\ \sqrt{2} \end{pmatrix} - \begin{pmatrix} 1 \\ -\sqrt{2} \\ \sqrt{2} \\ 1 \\ 1 \\ -\sqrt{2} \end{pmatrix} \right\} = \begin{pmatrix} 0 \\ 0 \\ -1/\sqrt{2} \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

D'où l'équation de la frontière de décision dans l'espace de redescription F :

$$\begin{aligned} D_F(\mathbf{x}) &= \mathbf{w}^{\star\top}\, \Phi(\mathbf{x}) \\ &= (0, 0, -1/\sqrt{2}, 0, 0, 0)\ (1, \sqrt{2}\,x_1,\ \sqrt{2}\,x_2,\ x_1^2,\ x_2^2,\ \sqrt{2}\,x_1 x_2)^\top \\ &= -\sqrt{2}\,x_1 x_2 = 0 \end{aligned}$$

qui est donc une équation linéaire dans l'espace F (sixième coordonnée constante) (voir figure 14.10 à droite).

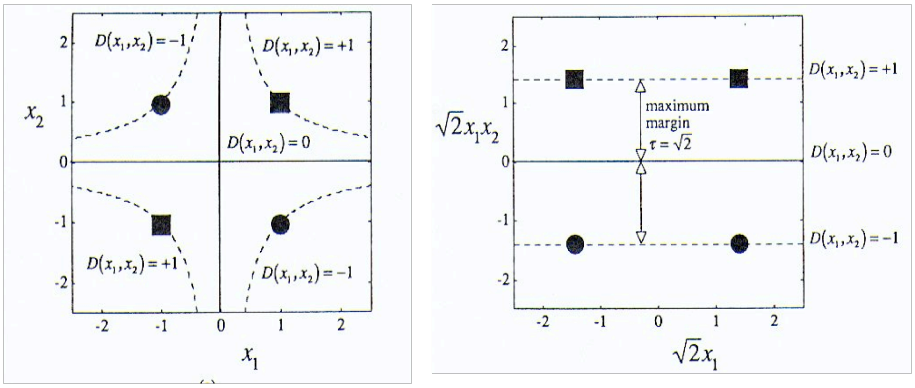


FIG. 14.10: *Les frontières de décision pour le problème du XOR. À gauche dans l'espace d'entrée $\mathcal{X}$, à droite dans l'espace de redescription F en projetant sur deux coordonnées parmi les 6.*

3.5 Séparateurs à Vastes Marges et généralisation

Selon la théorie statistique de l'apprentissage, la dimension de Vapnik-Chervonenkis de la classe des séparateurs linéaires de marge γ est égale à R^2/γ^2, où R est le rayon de la plus petite sphère englobant tous les points d'apprentissage.

Il a été montré plus récemment [SSTSW99, WSS01] qu'il est possible de borner l'erreur en généralisation en utilisant le spectre d'un opérateur intégral associé et le spectre[6] de la matrice de Gram $\kappa(\mathbf{x}_i, \mathbf{x}_j)$. En effet, ces nouvelles analyses tirent avantage du fait que, pour la plupart des applications, la distribution des données n'est pas isotrope dans l'espace, et, de ce fait, les bornes utilisant la mesure de sphère sont moins précises.

[6] Le spectre d'une matrice est l'ensemble de ses valeurs propres.

4. Autres types d'induction avec fonctions noyau

4.1 La régression

Nous avons vu dans la section 1.2 qu'il est possible d'utiliser l'approche par fonction noyau pour résoudre des tâches de régression non linéaire, c'est-à-dire de prédiction d'une valeur réelle à partir d'une entrée prise dans un espace, éventuellement multidimensionnel, typiquement $\mathbb{R}^d$.

Le problème inductif revient à chercher une fonction $h(\mathbf{x}) = y \in \mathbb{R}$ telle que, pour tous les points d'apprentissage $\{(\mathbf{x}_i, u_i)\}_{1 \leq i \leq m}$, $h(\mathbf{x}_i)$ soit le plus « proche » possible de u_i, tout en se plaçant dans un espace d'hypothèses contrôlé pour éviter le risque de sur-apprentissage.

Une analyse de la convergence du risque empirique sur le risque réel, mesuré en terme d'écart quadratique, ainsi que la recherche d'une solution parcimonieuse en terme d'exemples support, a conduit Vapnik à proposer d'utiliser la fonction de perte suivante :

$$|y - h(\mathbf{x})|_\varepsilon \;=\; \max\{0, |y - h(\mathbf{x})| \;-\; \varepsilon\} \tag{14.35}$$

Alors, afin d'estimer la régression linéaire :

$$h(\mathbf{x}) \;=\; \langle \mathbf{w} \cdot \mathbf{x} \rangle \;+\; w_0 \tag{14.36}$$

avec une précision de ε, on minimise :

$$\frac{1}{2} \| \mathbf{w} \|^2 \;+\; C \sum_{i=1}^{m} |u_i - f(\mathbf{x}_i)|_\varepsilon \tag{14.37}$$

On peut réexprimer ce problème sous forme d'un problème d'optimisation sous contraintes (voir [Vap95]) :

$$\begin{cases} \text{Minimiser} & \tau(\mathbf{w}, \xi, \xi^\star) \;=\; \dfrac{1}{2} \| \mathbf{w} \|^2 \\ \text{sous les contraintes} & \begin{cases} ((\mathbf{w} \cdot \mathbf{x}_i) \;+\; w_0) \;-\; u_i \;\leq\; \varepsilon \;+\; \xi_i \\ u_i \;-\; ((\mathbf{w} \cdot \mathbf{x}_i) \;+\; w_0) \;\leq\; \varepsilon \;+\; \xi_i \\ \xi_i, \xi_i^\star \;\geq\; 0 \end{cases} \end{cases} \tag{14.38}$$

pour tous les $i \;=\; 1, \ldots, m$. On peut noter à nouveau que toute erreur plus petite que ε ne requiert pas une valeur non nulle de ξ ou de ξ_i et donc ne doit pas être prise en compte par la fonction objectif (14.38). La figure 14.11 illustre le rôle des contraintes.

Comme dans le cas de la classification, on peut généraliser à la régression non linéaire en passant par un espace de redescription des entrées grâce à l'utilisation de fonctions noyau. L'introduction de multiplicateurs de Lagrange conduit au problème d'optimisation suivant, dans lequel les constantes $C > 0$ et $\xi \geq 0$ sont choisies *a priori* :

$$\begin{cases} \text{Maximiser} & \begin{aligned} W(\alpha, \alpha) \;=\; -\varepsilon \sum_{i=1}^{m} (\alpha_i^\star \;+\; \alpha_i) + \sum_{i=1}^{m} (\alpha_i^\star \;-\; \alpha_i) u_i \\ -\frac{1}{2} \sum_{i,j=1}^{m} (\alpha_i^\star \;-\; \alpha_i)(\alpha_i^\star \;-\; \alpha_j) k(\mathbf{x}_i, \mathbf{x}_i) \end{aligned} \\ \text{sous les contraintes} & \begin{cases} 0 \;\leq\; \alpha_i, \alpha_i^\star \;\leq\; C \quad i = 1, \ldots, m \\ \sum_{i=1}^{m} (\alpha_i - \alpha_i^\star) \;=\; 0 \end{cases} \end{cases} \tag{14.39}$$

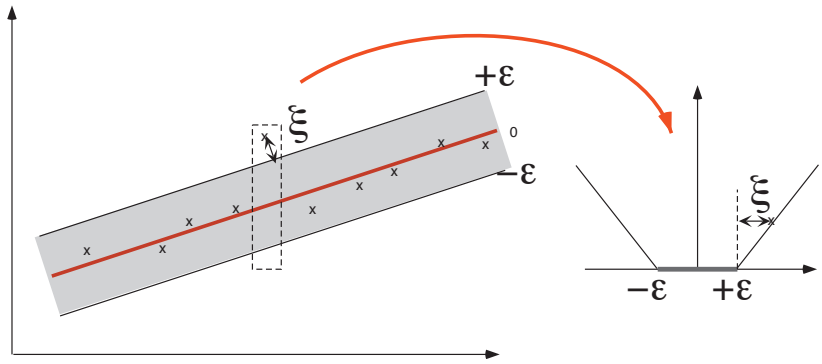


FIG. 14.11: *Dans la régression par SVM, au lieu d'imposer une marge entre les points des deux classes, on impose une sorte de « chaussette » autour des points grâce au paramètre ε. Le compromis entre la complexité du modèle (la fonction de régression) et la fidélité aux points d'apprentissage est règlée par la variable ressort ξ.*

L'estimation de la régression prend alors la forme :

$$\boxed{h(\mathbf{x}) \;=\; \sum_{i=1}^{m}(\alpha_i^\star - \alpha_i)\, k(\mathbf{x}_i, \mathbf{x}) \;+\; w_0} \tag{14.40}$$

où b est calculé en utilisant le fait que la contrainte $((\mathbf{w}\cdot\mathbf{x}_i) + w_0) - u_i \leq \varepsilon + \xi_i$ devient une égalité avec $\xi_i = 0$ si $0 < \alpha_i < C$, et la contrainte $u_i - ((\mathbf{w}\cdot\mathbf{x}_i) + w_0) \leq \varepsilon + \xi_i$ devient une égalité avec $\xi_i^\star = 0$ si $0 < \alpha_i^\star < C$.

4.2 Induction avec une seule classe ou non supervisée

À partir du moment où l'on considère l'astuce des noyaux (utilisation des fonctions noyau pour changer d'espace de description dans lequel des méthodes linéaires s'appuyant sur des produits scalaires peuvent être employées), il est tentant de chercher à l'utiliser dans tous les domaines de l'apprentissage. C'est ce qui a été également fait en apprentissage non supervisé. Contrairement aux problèmes d'apprentissage supervisé dans lesquels la tâche fondamentale est de prédire l'étiquette d'une nouvelle forme d'entrée, et pour lesquels le critère de performance est assez aisé à définir, ces problèmes sont moins faciles à préciser. Il s'agit essentiellement de trouver une caractérisation des données permettant de trouver des régularités intéressantes sur leur structure ou sur la distribution de probabilité sous-jacente (voir chapitre 18).

Les méthodes d'apprentissage non supervisé incluent :

- La recherche d'un « repère » ou référentiel permettant de décrire au mieux les données en comprimant leur description. Les techniques d'*Analyse en Composantes Principales* par noyau (« *Kernel PCA* ») répondent en partie à ce problème. (Voir [SS02] pour une description générale).
- La *recherche de sous-variétés* dans les données préservant leur topologie. (Voir [LV07]).
- La recherche de nuages de points (« *clustering* ») ou de catégories dans les données.
- L'estimation de densité sous-jacente aux données.

Dans cette section, nous étudions un problème plus simple, binaire : celui de trouver une région de l'espace d'entrée $\mathcal{X}$ contenant une grande partie des données (proportion à préciser). Comme dans tout problème d'induction, le principe est d'identifier cette région sur la base d'un

échantillon d'apprentissage jugé représentatif. Une tâche souvent associée à ce problème est celui de l'identification de formes exceptionnelles ou anormales (« *outliers* »).

Suivant en cela le principe philosophique de Vapnik selon lequel il ne faut pas résoudre un problème plus difficile ou plus général que celui considéré, il n'est pas nécessaire d'induire la densité de probabilité génératrice des données pour résoudre ce problème. Il suffit de trouver une frontière de décision entre la région considérée, couvrant une proportion minimale des données, et le reste de l'espace $\mathcal{X}$. C'est l'objet des SVM mono-classe (« *one-classe SVM* »).

À l'instar des autres problèmes inductifs, il est nécessaire de résoudre un compromis entre disposer d'une classe de fonctions de décision suffisamment riches pour bien approcher la région recherchée et éviter cependant de risquer d'être victime de sur-adaptation. La méthode des SVM permet de résoudre ce problème de régularisation.

En fait, l'origine de la méthode proposée réside dans une question théorique : peut-on déterminer avec précision le centre de masse de données générées à partir d'une distribution génératrice inconnue sur la base d'un échantillon de données tirées aléatoirement et indépendamment ? Soit $\mathbb{E}_{\mathbf{P}_\mathcal{X}}[\Phi(\mathbf{x})] = \int_\mathcal{X} \Phi(\mathbf{x}) d\mathbf{P}_\mathcal{X}(\mathbf{x})$ le « vrai » centre de masse. Une analyse statistique permet de borner la valeur de $||\frac{1}{m}\sum_{i=1}^m \Phi(\mathbf{x}_i) - \mathbb{E}_{\mathbf{P}_\mathcal{X}}[\Phi(\mathbf{x})]||$ et de montrer que cette borne ne dépend pas de la dimension de l'espace, mais est de l'ordre de $\sqrt{1/m}$ comme attendu, ainsi que du rayon de l'hypersphère englobant les projections de données dans $\Phi(\mathcal{X})$.

Cette analyse suggère un algorithme de détection de nouveauté ou d'anomalie. Considérons en effet la boule ou hypersphère minimale englobant les m points d'apprentissage dans l'espace $\Phi(\mathcal{X})$, la probabilité qu'un nouveau point ne soit pas englobé dans une boule minimale *centrée en* $\frac{1}{m}\sum_{i=1}^m \Phi(\mathbf{x}_i)$ est $\leq \frac{1}{m+1}$ (par un raisonnement simple de symétrie entre les données et l'emploi d'une inégalité triangulaire) (voir par exemple [STC04], pp.116-117). On pourra ainsi considérer que tout point tombant en dehors de cette boule, représentant les exemples « normaux », est anormal.

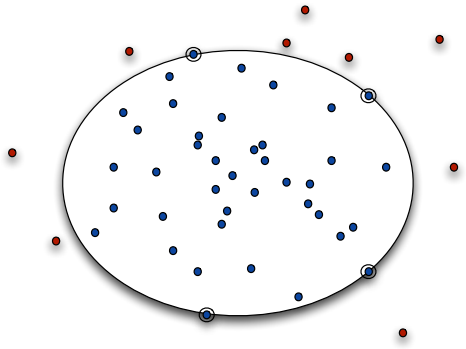


FIG. 14.12: *Exemple d'une « hypersphère » définie sur un ensemble de points d'apprentissage. Les points entourés sont les exemples support. Les points bleus, extérieurs à l'hypersphère, sont jugés « nouveaux ».*

Une analyse plus fine[7] prenant en compte le compromis entre le rayon de l'hypersphère et la sensibilité à des points normaux mais peu représentatifs de la distribution de probabilité considérée conduit au problème suivant. Étant donné un échantillon d'apprentissage $S = \langle \mathbf{x}_1, \ldots, \mathbf{x}_m \rangle$,

[7] Voir par exemple [STC04], pp.196-205.

trouver le vecteur $\alpha^\star$ optimisant l'expression :

$$\left\{\begin{array}{ll} \text{Maximiser} & \sum_{i=1}^m \alpha_i \kappa(\mathbf{x}_i,\mathbf{x}_i) \; - \; \sum_{i,j=1}^m \alpha_i \alpha_j \kappa(\mathbf{x}_i,\mathbf{x}_j) \\ \text{sous contraintes de :} & \left\{\begin{array}{l} \sum_{i=1}^m \alpha_i = 1 \\ 0 \leq \alpha_i \leq C, \quad i = 1,\ldots,m. \end{array}\right. \end{array}\right. \tag{14.41}$$

En choisissant alors i tel que $0 < \alpha_i^\star < C$, on calcule :

$$\begin{aligned} r^\star &= \sqrt{\kappa(\mathbf{x}_i,\mathbf{x}_i) - 2\sum_{j=1}^m \alpha_j^\star \kappa(\mathbf{x}_j,\mathbf{x}_i) + \sum_{i,j=1}^m \alpha_i^\star \alpha_j^\star \kappa(\mathbf{x}_i,\mathbf{x}_j)} \\ D &= \sum_{i,j=1}^m \alpha_i^\star \alpha_j^\star \kappa(\mathbf{x}_i,\mathbf{x}_j) - (r^\star)^2 - \gamma \end{aligned}$$

Et la fonction de décision :

$$\boxed{h(\mathbf{x}) \; = \; H\big[\kappa(\mathbf{x},\mathbf{x}) - 2\sum_{i=1}^m \alpha_i^\star \kappa(\mathbf{x}_i,\mathbf{x}) + D\big]} \tag{14.42}$$

où H est la fonction de Heaviside :

$$\forall x \in \mathbb{R}, \qquad H(x) = \begin{cases} 0 & \text{si} \quad x < 0 \\ 1 & \text{si} \quad x \geq 0. \end{cases} \tag{14.43}$$

La fonction h retourne la valeur 1 pour une entrée tombant en dehors de l'hypersphère, et devant donc être interprétée comme une forme nouvelle.

Encore une fois, la résolution du problème n'implique généralement qu'une partie réduite des exemples d'apprentissage qui se trouvent être sur la frontière de l'hypersphère calculée.

Dans le cas où les données sont normalisées, on peut montrer que le problème peut également être résolu en cherchant un hyperplan dans $\Phi(\mathcal{X})$ séparant au maximum les données de l'origine.

5. Ingénierie des fonctions noyau

5.1 Les fonctions noyau : signification et capacité d'approximation

5.1.1 Signification des fonctions noyau

La section 1 a montré que l'utilisation de fonctions noyau était liée à des raisons fondamentales diverses. En résumant, on peut dire que le choix d'une fonction noyau correspond implicitement au choix :

- d'une *mesure de similarité* entre éléments de l'espace d'entrée $\mathcal{X}$,
- d'une *projection* des éléments de $\mathcal{X}$ dans un espace vectoriel F doté d'un produit scalaire,
- d'un *espace fonctionnel* doté d'une base de fonctions analysantes (e.g. base de Mercer ou base d'un espace de Hilbert à noyau reproduisant (RKHS)) dans laquelle est cherchée une fonction hypothèse,

- d'un *critère inductif régularisé* puisque le théorème de représentation montre que chaque noyau correspond à un terme de régularisation,
- d'une *fonction de covariance* définissant comment les éléments de $\mathcal{X}$ sont corrélés,
- d'une *mesure de probabilité sur un ensemble de fonctions* comme le montre l'interprétation des fonctions noyau comme fonction de covariance.

Le choix d'une fonction noyau est donc essentiel pour l'apprentissage et doit refléter au mieux toute connaissance *a priori* sur le domaine étudié.

5.1.2 Les fonctions noyau et la capacité du RKHS associé

Nous avons vu qu'une fonction noyau induisait l'espace de fonctions dans lequel était cherché une fonction hypothèse satisfaisant au mieux un critère inductif régularisé. Le chapitre 2 a par ailleurs amplement insisté sur l'importance du contrôle de la capacité de l'espace des hypothèses pour garantir, en probabilité, un lien entre le risque empirique et le risque réel. Il semble donc essentiel de contrôler le choix de la fonction noyau afin d'éviter le risque de sous-apprentissage (espace de fonctions trop pauvre) ou de sur-apprentissage (espace de fonctions trop riche). En fait, de nombreuses fonctions noyau sont associées à des espaces fonctionnels universels, c'est-à-dire permettant d'approcher n'importe quelle fonction sur les points d'apprentissage (en d'autres termes des espaces fonctionnels de dimension de Vapnik-Cervonenkis infinie).

Définition 14.3 (Noyau universel)

Une fonction noyau continue sur un espace compact métrique $\mathcal{X}$ est dit universel *si l'espace de Hilbert à Noyau Reproduisant (RKHS) $\mathcal{F}$ associé est dense dans l'espace $\mathcal{C}(\mathcal{X})$ des fonctions continues de $\mathcal{X}$ dans $\mathbb{R}$, c'est-à-dire si pour toute fonction $g \in \mathcal{C}(\mathcal{X})$ et pour tout $\varepsilon > 0$, il existe une fonction $f \in \mathcal{F}$ telle que :*

$$||f - g||_\infty \leq \varepsilon$$

où $||\cdot||_\infty$ indique la norme supremum $||x||_\infty := \sup_{s\in[0,1]} x(s)$.

Une fonction noyau universelle permet en particulier de séparer tout sous-ensemble de points de $\mathcal{X}$ en deux sous-ensembles quelconques disjoints, ce qui correspond à une dimension de Vapnik-Cervonenkis infinie.

Il peut être montré que les fonctions noyau strictement positives peuvent séparer ainsi tout sous-ensemble fini de points. De nombreuses fonctions noyau standard possèdent cette propriété :

- les *noyaux exponentiels* : $\kappa(\mathbf{x}, \mathbf{x}') = \exp\left(\langle \mathbf{x}, \mathbf{x}' \rangle\right)$
- les *noyaux gaussiens (RBF)* : $\kappa(\mathbf{x}, \mathbf{x}') = \exp -\frac{||\mathbf{x}-\mathbf{x}'||}{\gamma^2}$
- les *noyaux binomiaux* : $\kappa(\mathbf{x}, \mathbf{x}') = (1 - \langle \mathbf{x}, \mathbf{x}' \rangle)^{-\alpha}$

Si l'espace fonctionnel correspondant à l'espace des hypothèses est de capacité d'approximation universelle, l'induction n'est possible que si le critère inductif est proprement régularisé. C'est ce que réalise en particulier la technique des Séparateurs à Vastes Marges (SVM).

5.2 L'espace des fonctions noyau : règles simples de construction

Les fonctions noyau ont pour propriété d'être positives définies, c'est-à-dire d'être associées à des matrices de Gram : $\mathbf{K} = (k(\mathbf{x}_i, \mathbf{x}_j))_{i,j}$ positives semi-définies. La construction de fonctions noyau à partir de fonctions noyau connues utilise des opérations qui préservent cette propriété.

Théorème 14.5 (Règles de construction)

Soient κ_1 et κ_2 deux fonctions noyau sur $\mathcal{X} \times \mathcal{X}$, $\mathcal{X} \in \mathbb{R}^d$, $c \in \mathbb{R}^+$, $f(\cdot)$ une fonction réelle sur $\mathcal{X}$, $\text{poly}(\cdot)$ un polynome avec des coefficients positifs ou nuls, $\Phi(\cdot)$ une fonction de $\mathcal{X}$ sur $\mathbb{R}^D$, κ_3 une fonction noyau définie sur $\Phi(\mathcal{X}) \times \Phi(\mathcal{X})$, $\mathbf{A}$ une matrice positive semi-définie, $\mathbf{x}_a$ et $\mathbf{x}_b$ des variables avec $\mathbf{x} = (\mathbf{x}_a, \mathbf{x}_b)$ et κ_a et κ_b des fonctions noyau dans leur espace respectif. Alors les fonctions suivantes sont des fonctions noyau :

$$\begin{array}{lrcl}
\text{(i)} & \kappa(\mathbf{x}, \mathbf{x}') & = & c\,\kappa_1(\mathbf{x}, \mathbf{x}') \\
\text{(ii)} & \kappa(\mathbf{x}, \mathbf{x}') & = & f(\mathbf{x})\,\kappa_1(\mathbf{x}, \mathbf{x}')f(\mathbf{x}') \\
\text{(iii)} & \kappa(\mathbf{x}, \mathbf{x}') & = & \text{poly}\big(\kappa_1(\mathbf{x}, \mathbf{x}')\big) \\
\text{(iv)} & \kappa(\mathbf{x}, \mathbf{x}') & = & \exp\big(\kappa_1(\mathbf{x}, \mathbf{x}')\big) \\
\text{(v)} & \kappa(\mathbf{x}, \mathbf{x}') & = & \kappa_1(\mathbf{x}, \mathbf{x}') + \kappa_2(\mathbf{x}, \mathbf{x}') \\
\text{(vi)} & \kappa(\mathbf{x}, \mathbf{x}') & = & \kappa_1(\mathbf{x}, \mathbf{x}')\,\kappa_2(\mathbf{x}, \mathbf{x}') \\
\text{(vii)} & \kappa(\mathbf{x}, \mathbf{x}') & = & \kappa_3\big(\Phi(\mathbf{x}), \Phi(\mathbf{x}')\big) \\
\text{(viii)} & \kappa(\mathbf{x}, \mathbf{x}') & = & \mathbf{x}^\top \mathbf{A}\, \mathbf{x}' \\
\text{(ix)} & \kappa(\mathbf{x}, \mathbf{x}') & = & \kappa_a(\mathbf{x}_a, \mathbf{x}'_a) + \kappa_b(\mathbf{x}_b, \mathbf{x}'_b) \\
\text{(x)} & \kappa(\mathbf{x}, \mathbf{x}') & = & \kappa_a(\mathbf{x}_a, \mathbf{x}'_a)\,\kappa_b(\mathbf{x}_b, \mathbf{x}'_b)
\end{array} \tag{14.44}$$

La règle de construction (v) correspond à une simple « addition » de deux espaces. En effet :

$$\begin{aligned}
\kappa(\mathbf{x}, \mathbf{x}') &= \kappa_1(\mathbf{x}, \mathbf{x}') + \kappa_2(\mathbf{x}, \mathbf{x}') \\
&= \langle \Phi_1(\mathbf{x}), \Phi_1(\mathbf{x}') \rangle + \langle \Phi_2(\mathbf{x}), \Phi_2(\mathbf{x}') \rangle \\
&= \langle [\Phi_1(\mathbf{x}), \Phi_2(\mathbf{x}')], [\Phi_1(\mathbf{x}), \Phi_2(\mathbf{x}')] \rangle
\end{aligned} \tag{14.45}$$

On peut facilement généraliser cette règle de construction à la somme tensorielle de noyaux.

Théorème 14.6 (Noyau *somme directe*)

Soient κ_1 et κ_2 deux fonctions noyau définies respectivement sur $\mathcal{X}_1 \times \mathcal{X}_1$ et sur $\mathcal{X}_2 \times \mathcal{X}_2$, alors leur somme directe

$$(\kappa_1 \oplus \kappa_2)(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}'_1, \mathbf{x}'_2) \;=\; \kappa_1(\mathbf{x}_1, \mathbf{x}'_1) + \kappa_2(\mathbf{x}_2, \mathbf{x}'_2) \tag{14.46}$$

est une fonction noyau sur $(\mathcal{X}_1 \times \mathcal{X}_2) \times (\mathcal{X}_1 \times \mathcal{X}_2)$. Ici, $\mathbf{x}_1, \mathbf{x}'_1 \in \mathcal{X}_1$ et $\mathbf{x}_2, \mathbf{x}'_2 \in \mathcal{X}_2$.

La règle de construction (vi) est particulièrement importante. Elle permet de combiner des caractéristiques ou attributs des deux espaces de redescription associés à κ_1 et à κ_2 entre eux. On obtient ainsi le *produit tensoriel de noyaux*. Par exemple, ici, les attributs de l'espace de redescription sont les produits de toutes les paires d'attributs du 1$^{\text{er}}$ et du 2$^{\text{nd}}$ espaces. En effet :

$$\begin{aligned}\kappa(\mathbf{x},\mathbf{x}') &= \kappa_1(\mathbf{x},\mathbf{x}')\,\kappa_2(\mathbf{x},\mathbf{x}') \\ &= \sum_{i=1}^{D_1}\Phi_1(\mathbf{x})_i\Phi_1(\mathbf{x}')_i \sum_{j=1}^{D_2}\Phi_2(\mathbf{x})_j\Phi_2(\mathbf{x}')_j \\ &= \sum_{i=1}^{D_1}\sum_{j=1}^{D_2}\Phi_1(\mathbf{x})_i\,\Phi_1(\mathbf{x}')_i\,\Phi_2(\mathbf{x})_j\,\Phi_2(\mathbf{x}')_j \\ &= \sum_{i=1}^{D_1}\sum_{j=1}^{D_2}\Phi(\mathbf{x})_{ij}\,\Phi(\mathbf{x}')_{ij}\end{aligned} \tag{14.47}$$

en posant $\Phi(\mathbf{x})_{ij} = \Phi_1(\mathbf{x})_i\,\Phi_2(\mathbf{x})_j$.

—— Exemple **Fonction noyau à exposant simple** ——

Soit la fonction noyau : $\kappa(\mathbf{x},\mathbf{x}') = \kappa_1(\mathbf{x},\mathbf{x}')^s$, on obtient par induction l'espace associé qui est constitué de tous les monômes de degré s :

$$\Phi_i(\mathbf{x}) \;=\; \Phi_1(\mathbf{x})_1^{i_1}\,\Phi_1(\mathbf{x})_2^{i_2}\ldots\Phi_1(\mathbf{x})_N^{i_N} \tag{14.48}$$

avec i satisfaisant $\sum_{j=1}^{N} i_j = s$.

Par exemple, pour $s = 5$, et $d = 3$ la dimension de $\mathcal{X}$, on pourrait avoir $\Phi_i(\mathbf{x}) = x_1^2x_2^1x_3^2$ où les x_i sont les coordonnées de $\mathbf{x}$ dans $\mathcal{X}$.

Chacune des composantes $\Phi_i(\mathbf{x})$ est un monôme de degré s. Il peut en effet être intéressant de considérer les combinaisons de s attributs de base des objets dans $\mathcal{X}$.

Par exemple, en reconnaissance de scènes visuelles, lorsque les images sont décrites à l'aide de pixels, ne prendre que les produits scalaires directs entre les images correspond à multiplier les intensités des pixels entre elles. Cela signifie aussi que l'*on pourrait redistribuer les pixels arbitrairement dans l'image sans changer le résultat*. Mais la reconnaissance d'objets peut impliquer la présence de plusieurs caractéristiques structurelles ou relationnelles à l'intérieur de l'image. Ainsi, un '8' implique la présence d'un arc de cercle convexe dans le sommet de l'image et d'un arc de cercle concave en bas de l'image. L'absence d'une seule de ces caractéristiques suffit à éliminer la possibilité d'un '8'. Il s'agit donc d'un critère très non linéaire au niveau des pixels. Pour parvenir à le prendre en compte, il faut considérer des régions entières de l'image, c'est-à-dire des combinaisons de pixels.

Il faut noter que l'on arrive très vite à des espaces de redescription de très grande dimension. Ainsi, la taille de l'espace de redescription pour $|\mathcal{X}| = d$ et pour un degré s d'un noyau à exposant simple est de :

$$|\Phi(\mathcal{X})| = \left(\begin{array}{c} s+d-1 \\ s \end{array}\right) \tag{14.49}$$

où l'on utilise la notation anglo-saxonne pour dénoter le coefficient binomial C^s_{s+d-1}.

Ce qui donne par exemple pour des images en 16×16 pixels, et des combinaisons de 5 pixels ($s = 5$) : $|\Phi(\mathcal{X})| \approx 10^{10}$. Il est heureux que la fonction noyau à exposant simple nous permette de ne pas manipuler directement ce genre d'espace.

Plus généralement, les fonctions noyau *polynomial*, noyau de *tous les sous-ensembles* et noyau *ANOVA* utilisent les règles de construction (14.44) et correspondent à un produit scalaire dans un espace de redescription construit à partir de certaines combinaisons des attributs de l'espace $\mathcal{X}$ (voir figure 14.13).

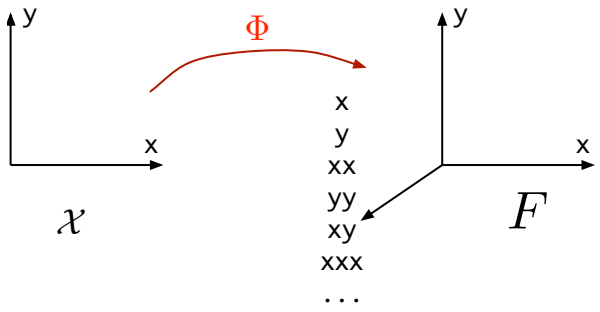


FIG. 14.13: *Construction d'un espace de redescription à partir de combinaisons des attributs de l'espace d'origine $\mathcal{X}$.*

EXEMPLE **Caractéristique associée à un noyau polynomial**

$$\Phi_p(\mathbf{x}) = x_1^3\, x_2^4\, x_4\, x_5^3 \dots x_d^5 \tag{14.50}$$

Les noyaux polynomiaux

L'application de la règle (iii) permet de construire des noyaux polynomiaux définis comme $\kappa(\mathbf{x}, \mathbf{x}') = \text{poly}\big(\kappa_1(\mathbf{x}, \mathbf{x}')\big)$ où poly est un polynôme à coefficients positifs.

Définition 14.2 (Noyau polynomial)

Un cas particulier est appelé noyau polynomial *:*

$$\kappa_s(\mathbf{x}, \mathbf{x}') = (\langle \mathbf{x}, \mathbf{x}' \rangle + R)^p = \sum_{s=0}^{p} \begin{pmatrix} p \\ s \end{pmatrix} R^{p-s} \langle \mathbf{x}, \mathbf{x}' \rangle^s \tag{14.51}$$

Contrairement au noyau à exposant simple qui ne considère que les monômes d'un degré donné p, le noyau polynomial *permet de prendre en compte tous les monômes de degré* $\leq p$.

La dimension de l'espace de redescription $\Phi(\mathcal{X})$ pour le noyau polynomial $\kappa(\mathbf{x}, \mathbf{x}') = (\langle \mathbf{x}, \mathbf{x}' \rangle + R)^p$ est :

$$\begin{pmatrix} d+p-1 \\ p-1 \end{pmatrix} + \begin{pmatrix} d+p-1 \\ p \end{pmatrix} = \begin{pmatrix} d+p \\ p \end{pmatrix}$$

Il faut noter qu'augmenter le paramètre p peut conduire à du surapprentissage. Le paramètre de pondération R permet de contrôler le poids de chaque monôme. Ainsi, le poids des monômes d'ordre s est $\begin{pmatrix} p \\ s \end{pmatrix} R^{p-s}$. De ce fait, augmenter R diminue les poids des monômes d'ordre élevés.

Les noyaux « tous sous-ensembles »

Un autre type de combinaison des attributs d'origine consiste à considérer les attributs Φ_i de l'espace de redescription $\Phi(\mathcal{X})$ constitués de tous les sous-ensembles d'attributs d'origine, y compris le sous-ensemble vide.

Comme pour les noyaux polynomiaux, on considère ainsi toutes les combinaisons d'attributs jusqu'à un ordre donné p. Mais, contrairement aux noyaux polynomiaux, l'exposant de chaque attribut retenu vaut 1.

Soit A un sous-ensemble des d attributs d'origine : $A \subseteq \{1, 2, \ldots, d\}$. On a :

$$\Phi_A(\mathbf{x}) = x_1^{i_1} x_2^{i_2} \ldots x_d^{i_d}$$

avec $\mathbf{i} = (i_1, i_2, \ldots, i_d) \in \{0, 1\}^d$.

On alors :

$$\kappa_{\subseteq}(\mathbf{x}, \mathbf{x}') \;=\; \langle \Phi(\mathbf{x}), \Phi(\mathbf{x}') \rangle \;=\; \sum_{A \subseteq \{1,\ldots,d\}} \prod_{i \in A} x_i x'_i \;=\; \prod_{i=1}^{d} (1 + x_i x'_i) \tag{14.52}$$

où la dernière équation découle d'une application de la loi distributive. On définit ainsi le noyau « tous sous-ensembles ».

Définition 14.3 (Noyau « tous sous-ensemble »)

Le noyau « tous sous-ensemble *» est défini par :*

$$\kappa_{\subseteq}(\mathbf{x}, \mathbf{x}') \;=\; \prod_{i=1}^{d} (1 + x_i x'_i) \tag{14.53}$$

On remarquera qu'il s'agit d'un noyau défini comme une somme de produits dans l'espace des caractéristiques (les dimensions de l'espace de redescription), mais qui se calcule, dans $\mathcal{X}$, comme le produit de sommes.

— Exemple **Caractéristique associée à un noyau « tous sous-ensemble »** —

$$\Phi_p(\mathbf{x}) = x_1\, x_2\, x_4\, x_5 \ldots x_d \tag{14.54}$$

Les noyaux ANOVA

Une autre méthode pour prendre en compte des combinaisons d'attributs de base fait appel aux fonctions noyau ANOVA. Similaires aux fonctions noyau « tous sous-ensembles », elles permettent de ne prendre en compte que des sous-ensembles de variables de cardinal donné p.

Définition 14.4 (Noyau ANOVA)

Le noyau ANOVA *est défini par :*

$$\kappa_p(\mathbf{x}, \mathbf{x}') \;=\; \langle \Phi_p(\mathbf{x}), \Phi_p(\mathbf{x}') \rangle \;=\; \sum_{|A|=p} \Phi_A(\mathbf{x})\, \Phi_A(\mathbf{x}') \;=\; \sum_{1 \le i_1 < i_2 < \ldots < i_p \le d} \prod_{j=1}^{p} x_{i_j} x'_{i_j} \tag{14.55}$$

— Exemple **Caractéristique associée à un noyau ANOVA** —

$$\Phi_p(\mathbf{x}) = \underbrace{x_1\, x_2\, x_4\, x_5 \ldots x_d}_{p} \tag{14.56}$$

Il est possible de généraliser les noyaux ANOVA. On peut en effet remarquer qu'il est tentant de remplacer les produits scalaires de base $x_i x'_i$ par des noyaux de base $\kappa_i(\mathbf{x}, \mathbf{x}') = a_i\, x_i\, x'_i$. On peut même aller plus loin et remplacer ces fonctions noyau par des fonctions noyau générales permettant de comparer les objets $\mathbf{x}$ et $\mathbf{x}'$, et définies éventuellement sur des sous-ensembles non nécessairement disjoints des d attributs de base. On a alors une version plus générale des fonctions noyau ANOVA :

$$\kappa_p(\mathbf{x}, \mathbf{x}') \;=\; \sum_{1 \le i_1 < i_2 < \ldots < i_p \le d} \;\prod_{j=1}^{p} \kappa_{ij}(\mathbf{x}, \mathbf{x}') \tag{14.57}$$

Ce point de vue mène alors assez naturellement à l'idée de comparer des objets structurés ou composites entre eux. En effet, la comparaison peut se faire par une sorte de calcul récursif dans lequel la comparaison au plus haut niveau résulte de comparaisons faites à des niveaux inférieurs, elles-mêmes pouvant éventuellement être définies récursivement, jusqu'à un niveau de description élémentaire.

Avant de passer à la description de fonctions noyau définies sur des structures, il est intéressant de noter que les noyaux polynomiaux, « tous sous-ensembles » ou ANOVA peuvent *se prêter facilement à des procédés de calcul récursifs*. Nous reportons le lecteur à [STC04] par exemple, pour des détails sur ces approches. Mais, ces procédés de calcul, d'abord poursuivis pour des raisons de complexité calculatoire, ont également inspiré les chercheurs à définir des fonctions noyaux applicables sur des objets structurés. C'est ainsi par deux chemins, *a priori* indépendants, que l'on aboutit à la même idée.

Les noyaux gaussiens

La fonction noyau Gaussienne est très employée.

Définition 14.5 (Noyau Gaussien)

Le noyau Gaussien *est défini par :*

$$\kappa_p(\mathbf{x}, \mathbf{x}') \;=\; e^{(-||\mathbf{x}-\mathbf{x}'||^2/2\sigma^2)} \tag{14.58}$$

Il s'agit bien d'une fonction noyau car :

$$||\mathbf{x} - \mathbf{x}'||^2 \;=\; \mathbf{x}^\top \mathbf{x} \,+\, (\mathbf{x}')^\top \mathbf{x}' \,-\, 2\mathbf{x}^\top \mathbf{x}'$$

ce qui donne :

$$\kappa_p(\mathbf{x}, \mathbf{x}') \;=\; e^{(-\mathbf{x}^\top \mathbf{x}/2\sigma^2)}\, e^{(\mathbf{x}^\top \mathbf{x}'/\sigma^2)}\, e^{(-(\mathbf{x}')^\top \mathbf{x}'/2\sigma^2)}$$

et en vertu de (ii) et (iv) et du fait que $\kappa(\mathbf{x}, \mathbf{x}') \;=\; \mathbf{x}^\top \mathbf{x}'$ est une fonction noyau, il s'agit bien d'une fonction noyau.

L'espace de redescription associé est de dimension infinie.

Il est important de noter que le noyau gaussien n'est pas restreint à l'utilisation de la distance enclidienne. On peut remplacer $\mathbf{x}^\top \mathbf{x}'$ par un noyau non linéaire $\kappa(\mathbf{x}, \mathbf{x}')$, ce qui donne :

$$\kappa_p(\mathbf{x}, \mathbf{x}') \;=\; \exp\!\left\{ -\frac{1}{2\sigma^2} \big(\kappa(\mathbf{x}, \mathbf{x}) \,+\, \kappa(\mathbf{x}', \mathbf{x}') \,-\, 2\kappa(\mathbf{x}, \mathbf{x}') \big) \right\} \tag{14.59}$$

Le paramètre σ permet de contrôler la flexibilité de la fonction noyau comme le degré d d'une fonction noyau polynomiale. De petites valeurs de σ correspondent à de grandes valeurs de d,

permettant de s'adapter aisément à tout échantillon d'apprentissage et donc à être sujet au sur-apprentissage. Dans ce cas, la matrice de Gram devient proche de la matrice identité. Au contraire, de grandes valeurs de σ réduisent la fonction noyau à une fonction constante, donc à une grande difficulté à s'adapter à autre chose qu'à des classifications triviales.

On peut aussi se représenter chaque point $\mathbf{x}$ comme associé à une fonction dans un espace de Hilbert :

$$\mathbf{x} \longmapsto \Phi(\mathbf{x}) \;=\; \kappa(\mathbf{x}, \cdot) \;=\; \exp\Big(-\frac{||\mathbf{x} - \cdot||^2}{2\sigma^2}\Big) \tag{14.60}$$

Chaque point d'apprentissage représente ainsi une fonction de base potentiellement dans une direction orthogonale aux autres fonctions. Mais plus proches sont deux points $\mathbf{x}$ et $\mathbf{x}'$, et plus les directions de leurs fonctions associées sont alignées.

5.3 Construction de fonctions noyau sur des espaces non vectoriels

Connaissant à la fois la signification des fonctions noyau et leurs propriétés, il est possible de chercher à construire des fonctions noyau plus complexes, adaptées à des domaines d'application particuliers. Il est nécessaire que la fonction noyau $\kappa(\mathbf{x}, \mathbf{x}')$ soit symétrique et définie semi-positive, et qu'elle exprime la forme appropriée de similarité entre les objets manipulés. Dans la suite, nous examinons un certain nombre de fonctions noyau qui ont été développées pour traduire des distances dans des espaces particuliers.

L'idée générale est de remplacer les n coordonnées réelles de l'espace de redescription $\mathcal{R}$ par n fonctions noyau de base servant à comparer des aspects des objets considérés.

5.3.1 Noyaux sur des ensembles

L'un des exemples les plus simples de fonction noyau s'appliquant à des données symboliques est celui du *noyau ensemble*. Soit un espace d'entrée constitué de tous les sous-ensembles possibles d'un ensemble A, par exemple les lettres d'un alphabet, ou les pixels d'une image. Si A_1 et A_2 sont deux sous-ensembles de A (e.g. deux ensembles de lettres, ou deux sous-images particulières), une fonction noyau simple permettant de mesurer leur similarité est :

$$\boxed{\kappa(A_1, A_2) \;=\; 2^{|A_1 \cap A_2|}}$$

où $A_1 \cap A_2$ est l'intersection de A_1 et de A_2 et $|A|$ est le cardinal de l'ensemble A.

On mesure donc la taille de l'ensemble des parties définissables sur l'intersection entre les ensembles A_1 et A_2 (la taille de l'ensemble des parties d'un ensemble de taille n étant 2^n).

Cette mesure est très utilisé dans la classification de textes, après que des prétraitements aient ramenés ces textes à des « sacs de mots » (*bags of words*, aussi appelée « modèle de l'espace vecteur », *vector space model* (VSM)).

Il est également possible de pondérer les sous-ensembles de A par une distribution de probabilité μ sur ses éléments. On peut alors définir le **noyau intersection** par :

$$\boxed{\kappa_\cap(A_1, A_2) \;=\; \mu(A_1 \cap A_2)}$$

c'est-à-dire la probabilité de l'intersection des deux sous-ensembles.

On peut montrer que ces fonctions sont bien des fonctions noyau car elles correspondent à des produits scalaires dans un espace de redescription.

5.3.2 Noyaux sur des textes

L'approche la plus rudimentaire pour représenter des textes est d'utiliser la représentation en sacs de mots. Si cette approche a le mérite de la simplicité et, en général, de l'efficacité, en revanche elle ignore des aspects importants des textes, en particulier leur sémantique. Ainsi, des mots synonymes ne sont pas pris en compte dans la similarité entre textes, tandis, qu'au contraire, des mots à plusieurs significations vont compter dans la similarité alors même qu'ils sont peut-être employés dans des sens différents. De même, la position des mots n'est pas prise en compte.

Pour tenir compte de la sémantique des mots, ou du moins de leur proximité sémantique, une manière simple consiste à utiliser une matrice de similarité ou de sémantique $\mathbf{S}$ qui sera utilisée dans la définition du noyau :

$$\boxed{\kappa(d_1, d_2) \;=\; \phi(d_1)\,\mathbf{S}\mathbf{S}^\top\,\phi(d_2)^\top}$$

où d_1 et d_2 sont des textes ou des documents, et $\phi(d)$ est la projection de d dans un espace de redescription.

La matrice $\mathbf{S}$ peut résulter de la composition de plusieurs étapes de traitement. Par exemple, une opération de pondération des termes, qui peut utiliser l'approche *tf-idf*[8], et une opération mesurant la proximité entre les mots :

$$\mathbf{S} \;=\; \mathbf{RP}$$

où $\mathbf{R}$ est une matrice diagonale spécifiant le poids des termes, et $\mathbf{R}$ est la matrice de proximité, dans laquelle $\mathbf{P}_{ij} > 0$ quand les termes t_i et t_j sont sémantiquement liés.

Une manière de calculer automatiquement cette matrice de proximité est de calculer les co-occurences des termes dans les documents (des termes en co-occurrences fréquentes étant jugés sémantiquement liés). Une méthode particulière pour ce faire est l'*analyse sémantique latente* (LSA) (voir chapitre 18). Le lecteur intéressé pourra trouver des détails sur ces techniques en particulier dans [STC04].

5.3.3 Fonctions noyau sur des séquences et des chaînes de caractères

Les fonctions noyau sur les séquences et les chaînes[9] s'appuient généralement sur une projection explicite dans un espace de redescription choisi pour que les distances ou similarités d'intérêt se calculent efficacement. Il s'agit donc de déterminer quelle notion de similarité devrait être calculable dans l'espace de redescription. Par exemple, souhaitons-nous regrouper les chaînes ou séquences par longueur, par composition similaire en sous-séquences, ou toute autre propriété ?

La plupart des fonctions noyau proposées jusqu'ici reposent sur le comptage de sous-chaînes ou de sous-séquences communes aux objets comparés. En génomique, par exemple, cela a un sens car on suppose que l'évolution a opéré par mutations, délétions et insertions successives et, par conséquent, le comptage de sous-séquences ou de sous-chaînes en commun permet d'estimer une distance évolutive entre génomes, et éventuellement, une similarité de fonction.

[8] La représentation *tf-idf* prend en compte la fréquence du terme dans le document étudié (*tf*) et la fréquence du terme dans l'ensemble des documents (*idf*). Plus précisément, soit le document d_j et le terme t_i, alors la fréquence du terme dans le document est : $\mathrm{df}_{ij} = \frac{n_{i,j}}{\sum_k n_{k,j}}$, où $n_{i,j}$ est le nombre d'occurrences du terme t_i dans d_j. Le dénominateur compte le nombre d'occurrences de tous les termes dans le document. La fréquence inverse du document (*idf*) est défini par : $\mathrm{idf}_i = \log \frac{|D|}{|\{d_j : t_i \in d_j\}|}$, où $|D|$ est le nombre total de documents, et $|\{d_j : t_i \in d_j\}|$ est le nombre de documents où le terme t_j apparaît. Alors $\text{tf-idf}_{i,j} = \mathrm{tf}_{i,j} \cdot \mathrm{idf}_i$.

[9] Conformément à l'usage en informatique, nous appellerons *chaîne* une suite consécutive d'éléments de base, tandis que le terme *séquence* sera utilisé pour une suite ordonnée mais non nécessairement consécutive d'éléments.

Soit s une séquence définie sur un alphabet fini de symboles $|\Sigma|$. La séquence vide est notée ϵ. On note Σ^n l'ensemble de toutes les chaînes de taille n, et $\Sigma^\star$ l'ensemble de toutes les chaînes :

$$\Sigma^\star \;=\; \bigcup_{n=0}^{\infty} \Sigma^n$$

Les fonctions noyau considérées sont définies par une projection explicite de l'espace $\mathcal{X}$ de toutes les séquences finies définies sur l'alphabet Σ sur un espace vectoriel $\Phi(\mathcal{X})$. Le plus souvent, les coordonnées de $\Phi(\mathcal{X})$ correspondent à un sous-ensemble I de séquences de l'espace d'entrée. Par exemple, cela pourrait être l'espace de toutes les séquences de longueur p, c'est-à-dire un espace de redescription de dimension $|\Sigma|^p$. Les projections Φ sont définies comme des applications :

$$\Phi : s \rightarrow (\phi_u(s))_{u \in I} \in \Phi(\mathcal{X})$$

où I désigne l'ensemble des coordonnées de $\Phi(\mathcal{X})$.

Pour chaque espace de redescription, il peut y avoir plusieurs projections possibles. Par exemple, pour un même espace de redescription, une projection peut consister à calculer chaque coordonnée $\phi_u(s)$ comme étant le nombre de fois où la *chaîne* u est présente dans s, tandis qu'une autre projection pourrait consister à calculer chaque coordonnée comme étant le nombre de fois où la *séquence* u est présente dans s. Dans ce dernier cas, on peut envisager de pondérer la somme des séquences présentes par le nombre de trous dans les séquences considérées.

Comme exemples de fonctions noyau envisagées dans la littérature, nous noterons les suivantes, souvent utilisées, mais n'épuisant pas l'ensemble des fonctions proposées (voir [STC04] pour une description plus complète de ces fonctions noyau).

Fonction noyau spectre

Le spectre d'ordre p d'une séquence s est l'histogramme des fréquences de toutes ses sous-chaînes de longueur p. La comparaison des p_spectres de deux séquences permet de mesurer un certain type de similarité entre ces séquences. Le produit interne entre p_spectres est une fonction noyau, appelée *fonction noyau spectre* (« spectrum kernel function »).

Définition 14.6 (Fonction noyau spectre)

À une fonction noyau à p_spectre est associé un espace de redescription indicé par $I = \Sigma^p$, et la projection définie par :

$$\phi_u^p(s) \;=\; |\{(v_1, v_2) : s = v_1 u v_2\}|, \quad u \in \Sigma^p.$$

La fonction noyau spectre *correspondante est définie par :*

$$\kappa_p(s,t) \;=\; \langle \Phi^p(s), \Phi^p(t) \rangle \;=\; \sum_{u \in \Sigma^p} \phi_u^p(s) \phi_u^p(t) \tag{14.61}$$

— Exemple **Fonction noyau 3_spectre** —

Soient deux chaînes construites sur l'alphabet {A,C,G,T} :

$s =$ GAGTTCTAAT

$t =$ GGATCACTAA

Les sous-chaînes de longueur 3 présentes dans ces deux chaînes sont :

GAG, AGT, GTT, TTC, TCT, CTA, TAA, AAT

GGA, GAT, ATC, TAC, CAC, ACT, CTA, TAA

Les sous-chaînes en commun sont : CTA et TAA, ce qui donne un produit interne $\kappa(s,t) = 2$.

Des méthodes de calcul récursives permettent de calculer efficacement cette fonction noyau.

Fonction noyau de séquences avec pondération en fonction des trous

Cette fonction noyau prend en compte les trous dans les sous-séquences prises comme coordonnées de l'espace de redescription. Ainsi, comme dans la fonction noyau p_spectre, la ressemblance entre deux séquences est d'autant plus importante que le nombre de sous-séquences communes est grand, mais le poids de ses sous-séquences prend en compte leur dispersion dans les séquences comparées.

— Exemple **Pondération des sous-séquences (1)** —

Par exemple, la séquence ACG est présente dans les séquences ACGT, ACTTGA et AGGCATGA, mais la première occurrence est plus significative car elle apparaît comme une sous-séquence consécutive de la séquence considérée, alors que la dernière est la moins significative car impliquant une grande dispersion de la sous-séquence ACG dans AGGCATGA.

Afin de pondérer les occurrences des sous-séquences, on introduit un facteur d'amoindrissement $\lambda \in [0,1]$. Ainsi, si la sous-séquence u de longueur $|u|$ est trouvée dans la séquence s avec une longueur $l(u)$, alors on pondère la sous-séquence u avec le poids $\lambda^{l(u)-|u|}$.

— Exemple **Pondération des sous-séquences (2)** —

En reprenant l'exemple précédent, et avec $\lambda = 0.9$, on aurait le poids de la sous-séquence $u =$ ACG valant 1 dans ACGT, $0.9^{5-3} = 0.81$ dans ACTTGA et $0.9^{7-3} = 0.9^4 \approx 0.66$.

5.3.4 Fonctions noyau à convolution

Étant donné un ensemble d'objets composites, c'est-à-dire constitués de sous-parties, est-il possible de définir une relation de similarité, c'est-à-dire ici une fonction noyau, entre ces objets ?

Par exemple, supposons que l'objet[10] $\mathbf{x} \in \mathcal{X}$ soit composé de sous-parties $x_p \in \mathcal{X}_p$, où $p = 1, \ldots, P$ (les ensembles $\mathcal{X}_p$ n'ayant pas à être égaux ou de tailles égales). Ces sous-parties elles-mêmes pouvant à leur tour être des objets composites.

— Exemple **Chaînes** —

Soit la chaîne $x =$ ATG et $P = 2$. La chaîne peut être considérée comme composée de $x_1 =$ AT et $x_2 =$ G, ou bien de $x_1 =$ A et de $x_2 =$ TG.

On peut considérer l'ensemble des décompositions licites comme une relation $R(x_1, x_2, \ldots, x_P, \mathbf{x})$ signifiant que « $x_1, x_2, \ldots, x_P$ constituent l'objet composite $\mathbf{x}$ ».

Haussler [Hau99] a été l'un des premiers à examiner comment définir une fonction noyau entre objets composites à partir de mesures de similarité entre leurs sous-parties, c'est-à-dire de fonctions noyau plus locales κ_p définies sur $\mathcal{X}_p \times \mathcal{X}_p$. L'idée est de mesurer la similarité entre les sous-objets grâce à des fonctions noyau locales, puis en remontant récursivement, niveau par niveau, de faire la somme des différentes contributions à la ressemblance globale. On parle de

[10] On continue ici à noter $\mathbf{x}$ les objets de $\mathcal{X}$ même si ce ne sont plus des vecteurs.

noyau de convolution car le calcul réalise une sorte de moyenne sur les différents choix possibles de décompositions en sous-objets.

Définition 14.7 (Fonction noyau de convolution)

Le noyau de convolution *de $\kappa_1, \kappa_2, \ldots, \kappa_P$ selon la relation de décomposition R est défini par :*

$$(\kappa_1 \star \ldots \star \kappa_P)(\mathbf{x}, \mathbf{x}') = \sum_R \prod_{p=1}^{P} \kappa_p(x_p, x'_p) \tag{14.62}$$

où la somme se fait selon toutes les décompositions permises par R pour décomposer l'objet $\mathbf{x}$ en $x_1, \ldots, x_P$ et l'objet $\mathbf{x}'$ en $x'_1, \ldots, x'_P$.

Par convention, on posera qu'une somme « vide » est égale à 0, c'est-à-dire que si $\mathbf{x}$ ou $\mathbf{x}'$ ne peut pas être décomposé, alors $(\kappa_1 \star \ldots \star \kappa_P)(\mathbf{x}, \mathbf{x}') = 0$.

Si il n'y a qu'un nombre fini de décompositions possibles on dit que R est finie et on peut montrer que la fonction de convolution ainsi définie est une fonction noyau.

Les exemples les plus simples de fonctions noyau à convolution sont les noyaux *tout sous-ensembles* (« all-subsets kernels »), les noyaux polynomiaux et les noyaux ANOVA.

5.3.5 Calcul des fonctions noyau à convolution

Notons que la définition récursive d'une fonction noyau a une traduction algorithmique permettant d'organiser de manière efficace les calculs. La programmation dynamique est en général utilisée. Une description des algorithmes correspondants peut être trouvée dans la littérature, à commencer par [STC04].

Nous présentons l'algorithme de calcul du noyau « toutes sous-séquences » pour donner un exemple de type d'algorithme. Le noyau « toutes sous-séquences » associe à chaque chaîne un vecteur dont chaque composante est le nombre d'occurrences $\Phi_u^U(x)$ d'une sous-séquence continue ou non dans la chaîne x.

Par exemple, toutes les sous-séquences des chaînes `bac`, `baa` et `cab` sont données dans la table suivante (e.g. la chaîne `baa` contient les sous-chaînes `ba` et `b_a` d'où le 2 dans la colonne `ba`) :

	λ	a	b	c	aa	ab	ac	ba	bc	ca	cb	bac	baa	cab
bac	1	1	1	1	0	0	1	1	1	0	0	1	0	0
baa	1	2	1	0	1	0	0	2	0	0	0	0	1	0
cab	1	1	1	1	0	1	0	0	0	1	1	0	0	1

Cela permet de compter le nombre d'alignements entre chaînes grâce à la fonction noyau : $\kappa(x, y) = \sum_{u \in \Sigma^\star} \Phi_u^U(x)\, \Phi_u^U(y)$. Par exemple :

$$\kappa(\texttt{baa, bac}) = 1 + 3 + 2 = 6$$

L'expression $\kappa(x_1, x_2, \ldots, x_n, y_1, y_2, \ldots, y_m)$ compte le nombre d'alignements entre les chaînes `x` et `y` que l'on a décomposées en caractères. On peut noter que ce calcul peut être décomposé en les alignements dans lesquels x_n n'est pas utilisé, soit $\kappa(x_1, x_2, \ldots, x_{n-1}, y_1, y_2, \ldots, y_m)$ et les alignements dans lesquels x_n est aligné avec l'une des $y_1 \ldots y_m$. Ce dernier compte est stocké dans une variable auxiliaire notée `Aux`.

Algorithme 22 : **Calcul du noyau toutes sous-séquences**

Entrées : $x_1, \ldots, x_n, y_1, \ldots, y_m$
Résultat : $\mathrm{K}[n][m] = \kappa(x_1, \ldots, x_n, y_1, \ldots, y_m)$
début
  pour $j \in \{1, .., m\}$ `/* L'alignement vide */`
  faire
    $\mathrm{K}[0][j] \leftarrow 1$
  fin Pour
  pour $i \in \{1, .., n\}$ **faire**
    $dernier \leftarrow 0$
    $\mathrm{Aux}[0] \leftarrow 0$
    pour $j \in \{1, .., m\}$ **faire**
      $\mathrm{Aux}[j] \leftarrow \mathtt{Aux}[dernier]$
    fin Pour
    if $x_i = y_i$ **then**
      $\mathrm{Aux}[j] \leftarrow \mathtt{Aux}[dernier] + \mathtt{K}[i-1][j-1]$
    end
    $dernier \leftarrow j$
    pour $j \in \{1, .., m\}$ **faire**
      $\mathrm{K}[i][j] \leftarrow \mathtt{K}[i-1][j] + \mathtt{Aux}[j]$
    fin Pour
  fin Pour
  retourner $K[n][m]$
fin

5.3.6 Fonctions noyau sur les arbres

Les arbres et les graphes s'inscrivent naturellement parmi les objets composites et structurés. La comparaison de tels objets a été abordée depuis longtemps, en particulier dans ce qui s'appelait la « reconnaissance des formes structurelle ». Cependant, l'avènement des méthodes à noyaux a conduit à un renouvellement des recherches dans la mesure où ces méthodes sont génériques et peuvent s'appliquer à une grande variété de domaines dès lors que l'on est capable de concevoir une fonction noyau, c'est-à-dire, *in-fine*, une mesure de similarité appropriée.

Les techniques de calcul récursif découvertes pour certains types de données ont amorcé une pléthore de travaux pour définir, selon une même approche, des noyaux sur des objets structurés susceptibles d'être décomposés en sous-objets, et ainsi de suite.

La structure d'arbre, par exemple, est omniprésente en informatique, et est spécialement utile en biologie évolutive, en traitement du langage naturel, dans le traitement des programmes informatiques ou des documents XML, ou encore pour certains traitements d'image. Il est donc important d'être capable de détecter des régularités dans ces types de données et cela requiert de pouvoir calculer des ressemblances entre arbres.

Étant donné un arbre T, l'arbre S est un *sous-arbre* de T si et seulement si il est sous-graphe connecté de T et les étiquettes des sommets et des arcs de S s'apparient avec les étiquettes des sommets et arcs correspondants dans T. Nous noterons $\mathcal{S}(T)$ l'ensemble des sous-arbres de T.

La plupart des fonctions noyau partagent la même structure : les fonctions noyau évaluées sur les arbres T_1 et T_2, $\kappa(T_1, T_2)$, sont exprimées comme une somme de fonctions noyau locales $\kappa_{\mathcal{S}}(S_1, S_2)$ définies sur toutes les paires de sous-arbres possibles S_1 et S_2 des arbres T_1 et T_2.

À titre d'illustration, nous considérons la fonction noyau *tous sous-arbres*.

Définition 14.8 (Fonction noyau *tous sous-arbres*)

La fonction noyau tous sous-arbres *est définie comme :*

$$\kappa(T_1, T_2) = \sum_{S_1 \in \mathcal{S}(T_1),\, S_2 \in \mathcal{S}(T_2)} \kappa_{\mathcal{S}}(S_1, S_2)$$

où $\kappa_{\mathcal{S}}(S_1, S_2)$ est elle-même une fonction noyau entre arbres.

Il en découle que la fonction κ est un noyau, si les fonctions $\kappa_{\mathcal{S}}$ le sont. Plusieurs fonctions locales $\kappa_{\mathcal{S}}$ sont possibles.

1. Si cette fonction locale retourne 1 lorsque les sous-arbres comparées sont égaux et 0 sinon (fonction δ de Kronecker) :
$$\kappa_{\mathcal{S}}(S_1, S_2) = \delta(S_1 = S_2)$$
alors $\kappa(T_1, T_2)$ compte simplement le *nombre de sous-arbres communs* entre T_1 et T_2.
2. Il est aussi possible de compter le *nombre de sous-arbres communs d'une taille N donnée* (mesurée en nombre de nœuds).
$$\kappa_{\mathcal{S}}(S_1, S_2) = \delta(S_1, S_2)\, \delta(n(S_1) = N)\, \delta(n(S_2) = N)$$
où $n(S_i)$ est le nombre de nœuds de S_i, et N une profondeur prédéfinie.
3. On peut également prendre en compte des *sous-arbres communs qui ne sont pas à la même profondeur* dans T_1 et dans T_2 dans une limite fixée à D.
$$\kappa_{\mathcal{S}}(S_1, S_2) = \delta(S_1, S_2) \max(D + 1 - |d(S_1) - d(S_2)|, 0)$$
où $d(S_i)$ est la profondeur de la racine de S_i dans l'arbre T_i, et D est la différence maximale tolérée entre les profondeurs des sous-arbres.

La figure 14.14 montre une base possible de sous-arbres définissant l'espace de redescription $\mathcal{R}$.

Ces fonctions noyau peuvent être calculées en utilisant la programmation dynamique avec une complexité proportionnelle au carré du nombre de sommets [STV04].

5.3.7 Fonctions noyau sur et entre graphes

La description sous forme de graphes est plus générale encore que celle utilisant des arbres. Les réseaux sociaux, les molécules complexes rencontrées en biologie, les codes HTML ou XML, les ontologies, appellent naturellement une représentation par graphes. La possibilité de classification automatique est, par exemple, spécialement importante pour l'aide à la découverte de nouvelles molécules ou pour la prédiction de la toxicité ou de la bio-activité de nouvelles structures chimiques.

Plusieurs **tâches de comparaison** peuvent intervenir dans les graphes :

- Comparaison *entre deux nœuds* d'un graphe. Les fonctions noyau à diffusion sont dédiées à ce problème (« diffusion kernels »).
- Comparaison *entre deux chemins*, faisant appel à des fonctions noyau entre chemins (« path kernels »).
- Comparaison *entre deux graphes*, faisant appel à des fonctions noyau pour graphes (« graph kernels »).

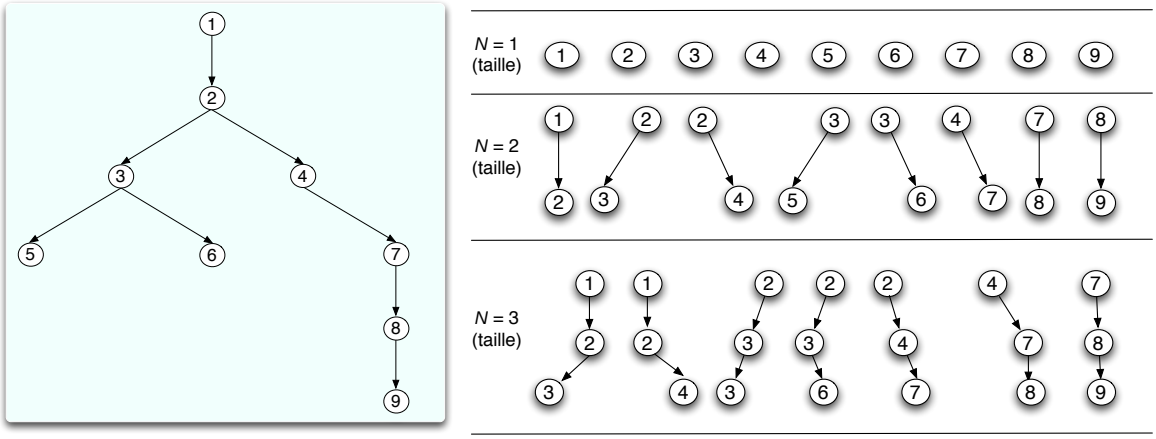


FIG. 14.14: À gauche : *un arbre.* À droite, *une décomposition en sous-arbres de tailles croissantes.*

De plus, une **tâche d'apprentissage** sur graphe consiste à *induire la valeur (réelle) associée à certains nœuds à partir de la connaissance de la valeur en certains autres nœuds.* Il s'agit en fait de la traduction du problème classique d'induction d'une fonction réelle à partir d'exemples, sauf que la notion de voisinage est maintenant induite par la notion d'adjacence sur un graphe et que l'on a une connaissance *a priori* plus forte des points sur lesquels il faut faire la prédiction (tâche d'apprentissage transductif).

Les fonctions noyau classiques sont définies entre vecteurs, comme des produits scalaires dans un espace de redescription. Pour définir des fonctions noyau entre graphes, l'idée générale, suivant l'idée originale de Haussler, est d'utiliser une décomposition en sous-structures (e.g. sous-séquences, sous-arbres ou sous-graphes) puis de calculer un décompte général à partir de la similarité entre ces sous-structures pour deux objets donnés en entrée.

Fonctions noyaux entre chemins

Les fonctions noyau de comparaison entre arbres s'appuyaient sur une décomposition en sous-arbres et sur des fonctions noyau locales de comparaison de sous-arbres. Il est tentant de recourir à la même approche dans le cas des graphes, en décomposant ceux-ci en *chemins* et en comparant ces derniers entre eux.

La majorité des fonctions noyau proposées pour la comparaison de graphes font appel à des sous-graphes sous la forme de chemins ou de séquences d'étiquettes obtenues par traversées des graphes considérés (voir par exemple la figure 14.15).

L'idée générale est de définir un espace de redescription $\mathcal{R}$ en engendrant des chemins par marches aléatoires dans l'espace des graphes selon une distribution de probabilités à définir. À partir de ces chemins $\mathbf{c} = (x_1, x_2, \ldots, x_l)$, on obtient des séquences $\mathbf{s} = (v_{x_1}, e_{x_1,x_2}, v_{x_2}, \ldots, v_{x_l})$ où les v_{x_i} sont des sommets du graphe, et les e_{x_i,x_j} sont les arcs entre sommets.

On suppose l'existence de fonctions noyau pour la comparaison de nœuds ($\kappa_n(n, n')$) et pour la comparaison d'arcs ($\kappa_a(a, a')$).

Un exemple de fonction noyau sur les nœuds pourrait être : $\kappa_n(n, n') = \delta(n, n')$, où δ est la fonction dirac. Si les étiquettes des nœuds sont à valeur réelle, on peut alors prendre la fonction noyau gaussienne : $\kappa_n(n, n') = \exp(-||f(n) - f(n')||^2/2\sigma^2)$.

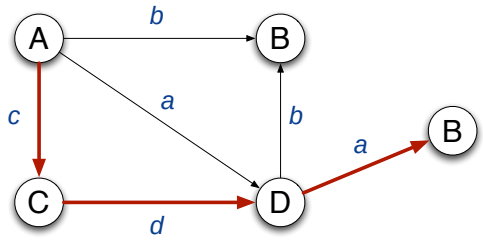


FIG. 14.15: *Exemple de graphe étiqueté. Les sommets sont étiquetés en majuscules et les arcs en minuscules. Le chemin en gras produit la séquence* (A,c,C,d,D,a,B).

Une *fonction noyau entre chemins* peut alors prendre la forme ([KTI03]) :

$$\kappa_c(\mathbf{c},\mathbf{c}') = \begin{cases} 0 & \text{si l} \neq \text{l'} \\ \kappa_n(n_{c_1}, n'_{c'_1}) \prod_{i=2}^{l} \kappa_a(a_{c_{i-1},c_i}, a'_{c'_{i-1},c'_i}) \times \kappa(n_{h_i}, n'_{h'_i}) & \text{si l} = \text{l'} \end{cases} \tag{14.63}$$

où c et c' sont deux chemins de longueur l et l' respectivement.

Fonctions noyaux entre graphes

Les fonctions noyau entre graphes (estimant une ressemblance entre graphes) diffèrent essentiellement par la manière d'engendrer des chemins et par les pondérations éventuellement utilisées.

En effet, lorsque les graphes considérés sont acycliques, les calculs impliqués dans la définition des noyaux à convolution peuvent s'effectuer par programmation dynamique. En revanche, il faut prendre des précautions pour les graphes cycliques, sous peine de considérer des séquences infinies. Dans ce cas, on utilise en particulier des poids inférieurs à 1 pour parvenir à des équations de point fixe.

Par exemple, une fonction noyau a été proposée (voir [STV04]) de la forme :

$$\kappa_G(G_1,G_2) = \sum_{\mathbf{c}_1} \sum_{\mathbf{c}_2} \kappa_l(\mathbf{c}_1,\mathbf{c}_2)\, \mathbf{P}(\mathbf{c}_1|G_1)\, \mathbf{P}(\mathbf{c}_2|G_2) \tag{14.64}$$

où $\kappa_l(\mathbf{c}_1,\mathbf{c}_2)$ est une fonction noyau locale définie sur des chemins de même longueur et inférieure à l. Les probabilités conditionnelles $\mathbf{P}(\mathbf{c}_i|G_i)$ sont définies à partir des probabilités de départ, de transition et de terminaison sur les sommets de G_i. Cette fonction noyau consiste à mesurer la similarité des distributions de probabilité de chemins aléatoires sur les deux graphes G_1 et G_2.

Nous avons utilisé la notation κ_G pour souligner que l'espace de redescription $\mathcal{R}$ associé est défini globalement, à partir de la notion de graphe et indépendamment des arguments de la fonction noyau. De même, la description en « sacs de mots » est indépendante des textes comparés. On parle de **redescription globale**. Nous allons voir dans la suite que la définition des espaces de redescription peut dépendre des données comparées.

Fonctions noyaux entre nœuds d'un graphe

La similarité entre nœuds d'un graphe peut être évaluée de plusieurs manières.

1. La similarité entre les nœuds n_1 et n_2 peut correspondre à la *« facilité » avec laquelle on passe de l'un à l'autre dans le graphe*. Une mesure consiste à considérer un chemin

optimal entre les deux nœuds. La sensibilité de cette mesure à l'insertion ou à la délétion de nœuds dans le graphe, nœuds correspondant généralement aux données d'apprentissage, fait préférer une mesure plus globale, correspondant à une sorte de « résistance » (par analogie à une résistance électrique) dans le graphe entre les nœuds. Une fonction ***noyau de diffusion*** a été proposée dans cette optique (voir [KL02]). L'idée est de mesurer la facilité de passage entre deux nœuds du graphe. Pour cela, une possibilité est de faire appel au *Laplacien de graphe* (voir plus bas pour une définition plus précise) défini par :

$$L_{ij} \;=\; \begin{cases} d_i & \text{si } i=j \text{ et le degré du nœud } n_i \text{ est } d_i \\ -1 & \text{si les nœuds } n_i \text{ et } n_j \text{ sont directement connectés} \\ 0 & \text{sinon} \end{cases}$$

On définit alors la matrice de Gram $\mathbf{G}_{i,j} = \kappa(n_i, n_j)$ associée au noyau par :

$$\boxed{\mathbf{G} \;=\; \exp\left(-\beta L\right)} \tag{14.65}$$

Parce que la fonction exponentielle est associée au développement limité : $1 + \frac{x}{1!} + \frac{x^2}{2!} + \ldots + \frac{x^n}{n!} + \ldots$, la matrice de Gram associée au noyau de diffusion prend successivement les corrélations de plus en plus lointaines entre nœuds dans le graphe en les affectant de poids de plus en plus réduits.

2. Une autre mesure de similarité prend en compte la *similarité entre les « relations » des deux nœuds*. Deux nœuds sont jugés d'autant plus semblables qu'ils sont liés (étroitement) à des nœuds communs dans le graphe.

Ces deux notions de similarité font appel à la notion de marches aléatoires dans le graphe, comme pour la comparaison de graphes. Cependant, cette fois-ci, l'espace de redescription associé à la fonction noyau $\kappa_n(n_1, n_2)$ dépend des deux nœuds comparés puisqu'elle s'appuie sur la comparaison des nœuds voisins de n_1 et de n_2. Il s'agit donc de **redescriptions locales**.

Noyaux pour l'apprentissage de fonctions définies sur les nœuds d'un graphe

De même que la transformation de Fourier est essentielle dans l'analyse fonctionnelle, la notion de ***Laplacien de graphe*** est au cœur de l'étude des graphes et des variations sur les graphes. La matrice d'incidence et la matrice « Laplacien de graphe » peuvent être considérées comme les analogues discrets des opérateurs gradient et Laplacien dans les espaces euclidiens.

Soit G un graphe arbitraire doté d'un ensemble N de m sommets[11] et d'un ensemble d'arcs entre ces sommets A. Pour chaque arc $a = \{n_1, n_2\}$, on choisit arbitrairement l'un des sommets comme « positif » et l'autre comme « négatif ». La matrice ∇ définie par :

$$\nabla_{ij} \;=\; \begin{cases} +1 & \text{si le sommet } n_i \text{ est l'extrémité positive de l'arc } a_j \\ -1 & \text{si le sommet } n_i \text{ est l'extrémité négative de l'arc } a_j \\ 0 & \text{sinon} \end{cases}$$

est appelée *matrice d'incidence* du graphe G.

Soit maintenant une fonction f définie sur les sommets du graphe G, c'est-à-dire $f : N \to \mathbb{R}$, alors $\nabla f : A \to \mathbb{R}$ est définie par :

$$(\nabla f)(a) \;=\; f(n_1) - f(n_2) \qquad \text{où } a = \{n_1, n_2\}$$

[11] On ne prend pas m au hasard ici. Souvent en effet, les sommets du graphe correspondront aux exemples d'apprentissage, voir par exemple le chapitre 19 sur l'apprentissage transductif et l'apprentissage semi-supervisé.

où n_1 est l'extrémité positive de l'arc a. Il s'agit de l'analogue le plus proche possible de l'opérateur différentiel sur un graphe.

La *matrice d'adjacence* d'un graphe G est telle que chaque composant $\mathbf{A}_{ij}$ est le poids de l'arc $a = \{n_i, n_j\}$. Ces poids doivent être non négatifs et symétriques.

Définition 14.9 (Laplacien de graphe)

Soit $\boldsymbol{D}$ la matrice diagonale des degrés de chaque sommets, c'est-à-dire que $\boldsymbol{D}_{nn}$ est le nombre d'arcs incidents à n, et soit $\boldsymbol{A}$ la matrice d'adjacence de G. Alors la matrice

$$-\Delta \;=\; \boldsymbol{D} - \boldsymbol{A}$$

est appelée le Laplacien *du graphe G.*

Ce Laplacien est l'analogue du Laplacien défini sur $\mathbb{R}^d$:

$$f \mapsto \nabla f = \sum_{i=1}^{d} \frac{\partial^2 f}{\partial x_i^2}$$

Il s'agit d'un opérateur symétrique, semi-défini positif et singulier. Le vecteur propre $(1, \ldots, 1)$ est associé à la valeur propre $\lambda_1 = 0$, dont la multiplicité égale le nombre de composantes connexes du graphe G.

Soient les valeurs propres $0 = \lambda_1 \leq \ldots \leq \lambda_m$ du Laplacien Δ et $\{\phi_i\}_{i=1,\ldots,m}$ l'ensemble des vecteurs propres orthogonaux associés. Alors, de même que les fonctions de base de Fourier dont des fonctions propres du Laplacien continu sur $\mathbb{R}^d$, les vecteurs propres de Δ peuvent être considérés comme les équivalents discrets d'une base de Fourier sur le graphe G, dont les fréquences augmentent avec la valeur des valeurs propres associées. En d'autres termes, toute fonction définie sur les sommets d'un graphe G peut être vue comme somme de fonctions de base définies sur ce graphe, et ces fonctions de base « oscillent » d'autant plus fortement sur le graphe que la valeur propre associée est élevée (pour $\lambda_1 = 0$, les fonctions propres associées sont constantes sur chaque composante connexe du graphe). (Voir [Sta96] pour plus de détails, et les figures 19.8 et 19.9 du chapitre 19 pour une illustration).

Une fonction f peut donc être décomposée comme somme de fonctions de base sur G :

$$f \;=\; \sum_{i=1}^{m} \hat{f}_i \, \phi_i$$

où les ϕ_i sont les fonctions de base et les $\hat{f}_i$ sont les coefficients associés (voir aussi la section 5.2 du chapitre 19).

La matrice du Laplacien étant semidéfinie positive, elle peut être employée comme matrice de Gram correspondant à une fonction noyau. On peut en effet démontrer que, pour toute fonction $f : n \in N \to \mathbb{R}$, on a :

$$f^\top \nabla f \;=\; \frac{1}{2} \sum_{i,j} \mathbf{A}_{ij} \left(f(n_i) - f(n_j) \right)^2 \;\geq 0 \tag{14.66}$$

où l'inégalité est due au fait que $\mathbf{A}$ n'a que des composantes non négatives.

L'équation 14.66 mesure aussi la régularité de la fonction f sur le graphe. En gros, f est régulière si les variations $f(n_i) - f(n_j)$ sont faibles quand $\mathbf{A}_{ij}$ est grand.

On peut voir comme un cas particulier le cas de chaque fonction de base, pour lesquelles on a :

$$\phi_i^\top \mathbf{A} \phi_i \ = \ \lambda_i \tag{14.67}$$

Ainsi, les fonctions de base associées à des petites valeurs propres sont plus régulières. En conséquence, on peut aussi ré-exprimer la régularité de la fonction f comme :

$$f^\top \nabla f \ = \ \sum_{i=1}^{m} \hat{f}_i^2 \, \lambda_i \tag{14.68}$$

Pour réaliser l'induction de f à partir de la connaissance de valeurs sur un sous-ensemble de noeuds (on parle aussi d'apprentissage par transduction dans ce cas), on fait le plus souvent l'hypothèse que la fonction f est aussi régulière que possible sur le graphe.

En supposant qu'il y ait n sommets dans le graphe G, on considérera des fonctions noyau associées à des matrices de Gram de la forme :

$$\boxed{K \ = \ \sum_{i=1}^{m} \mu_i \phi_i \phi_i^\top} \tag{14.69}$$

où les ϕ_i sont les vecteurs propres du Laplacien du graphe, et les $\mu_i \geq 0$ sont les valeurs propres de la matrice de Gram associée à κ.

La matrice de Gram K définit un espace de Hilbert à noyau auto-reproduisant de norme associée :

$$||f||_K^2 \ = \ \langle f, f \rangle_K \ = \ \sum_{i=1}^{m} \frac{\hat{f}_i^2}{\mu_i}$$

pour une fonction prenant la forme : $f \ = \ \sum_{i=1}^{m} \hat{f}_i \phi_i$ (on prendra $\frac{\hat{f}_i^2}{\mu_i}$ si $\mu_i = 0$).

5.3.8 Les noyaux génératifs

Il arrive que l'on dispose d'information sur le processus générant les données. Il peut alors être intéressant d'en tirer parti pour définir des fonctions noyau mieux adaptées à la tâche considérée. Plusieurs approches ont été proposées. Nous décrivons brièvement ici les fonctions noyau dérivées de la matrice d'information de Fisher[12].

L'idée sous-jacente dans cette approche est de mesurer l'effort d'adaptation nécessaire du modèle existant pour accommoder la donnée considérée. En particulier, il est nécessaire de disposer d'un modèle de génération des données qui soit paramétré continûment de telle manière que les dérivées par rapport à ces paramètres puissent être calculées.

Soit un modèle paramétrique $\mathbf{p}_{\theta^0}(x) = \mathbf{p}(\mathbf{x}|\theta^0)$ correspondant à la probabilité que l'entrée $\mathbf{x}$ soit produite par le processus génératif dont le vecteur de paramètre θ est mis) θ^0.

On peut mesurer ce qui est nécessaire pour que le modèle accommode une nouvelle donnée en calculant les dérivées partielles de $\mathbf{p}_{\theta^0}(\mathbf{x})$ par rapport à chacun des paramètres et en mesurant dans quelle direction aller pour augmenter la vraisemblance de cette donnée. On peut alors comparer deux données $\mathbf{x}_i$ et $\mathbf{x}_j$ en comparant les adaptations nécessaires pour augmenter la vraisemblance de chacune. Si les adaptations sont similaires, les données seront considérées

[12] Sir Ronald Aylmer Fisher (1890-1962) est l'un des pionniers qui a posé les fondations des statistiques actuelles. Il a en particulier posé les principes de l'analyse d'expériences et élaboré les outils d'analyse de la variance. Il est également à l'origine du principe de vraisemblance maximale.

comme proches. Plus précisément, on comparera les gradients associés à l'accommodation de chacune des données.

Formellement, soit $\log \mathcal{L}_{\theta^0}$ la log-vraisemblance du modèle associé à $\theta^0(\mathbf{x})$. Le *score de Fisher* est le gradient de la log-vraisemblance.

$$\mathbf{g}(\theta, \mathbf{x}) \;=\; \left(\frac{\partial \log \mathcal{L}_\theta(\mathbf{x})}{\partial \theta_i}\right)_{i=1}^{N}$$

en supposant qu'il y ait N paramètres pour définir le modèle.

La *matrice d'information de Fisher* pour le modèle est :

$$\mathbf{I} \;=\; \mathbb{E}\big[\mathbf{g}(\theta^0, \mathbf{x})\, \mathbf{g}(\theta^0, \mathbf{x})^\top\big]$$

où l'espérance est calculée par rapport aux points $\mathbf{x}$ engendrés par la distribution générative des données.

Le score de Fisher correspond à une application sur $\mathbb{R}^N$, ce qui suggère une fonction noyau.

Définition 14.10 (Noyau de Fisher)

Le noyau de Fisher *par rapport au modèle génératif de vecteur de paramètre θ^0 est :*

$$\kappa(\mathbf{x}, \mathbf{x}') \;=\; \mathbf{g}(\theta^0, \mathbf{x})^\top\, \mathbf{I}^{-1}\, \mathbf{g}(\theta^0, \mathbf{x}')$$

Le noyau de Fisher a été utilisé pour plusieurs types de données. Nous renvoyons le lecteur à [STC04] pour plus d'information.

5.4 Aspects calculatoires

Il est important que le calcul des quantités $\kappa(\mathbf{x}, \mathbf{x}')$ soit efficace puisqu'il doit être répété de très nombreuses fois. Une complexité de l'ordre de $\mathcal{O}(|\mathbf{x}| + |\mathbf{x}'|)$ ou de l'ordre de $\mathcal{O}(|\mathbf{x}| \cdot |\mathbf{x}'|)$ est généralement considérée comme correcte.

Tout un ensemble de techniques ont été développées pour améliorer l'efficacité des approches à noyau, comme nous allons voir.

6. Les méthodes à noyaux en pratique

6.1 Aperçu des approches calculatoires

Par contraste avec les méthodes d'optimisation adaptées par exemple aux réseaux connexionnistes multi-couche qui explorent un espace à multiples optima locaux, les méthodes requises pour les méthodes à noyaux explorent un problème *a priori* plus simple puisqu'il s'agit d'une optimisation d'un problème quadratique qui ne présente qu'un optimum global, qui plus est dans un paysage quadratique.

Plus précisément, le problème à résoudre a la forme générale de la minimisation d'un risque régularisé :

$$h^\star_{\mathcal{S}} \;=\; \underset{h \in \mathcal{H}}{\mathrm{ArgMin}}\big[R_{\mathrm{Emp}}(h) \;+\; \lambda ||h||^2\big] \tag{14.70}$$

pour lequel il existe une solution de la forme :

$$h^\star_{\mathcal{S}}(\mathbf{x}) \;=\; \langle \mathbf{w}(\alpha), \Phi(\mathbf{x}) \rangle \;=\; \big\langle \sum_{i=1}^{m} \alpha_i \Phi(\mathbf{x}_i), \Phi(\mathbf{x}) \big\rangle \;=\; \big\langle \sum_{i=1}^{m} \alpha_i \, \kappa(\mathbf{x}_i, \mathbf{x}) \big\rangle \tag{14.71}$$

Notons que l'on a :

$$\|\mathbf{w}(\alpha)\|^2 \;=\; \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j \, \kappa(\mathbf{x}_i, \mathbf{x}_j) \;=\; \alpha^\top \mathrm{K} \, \alpha \tag{14.72}$$

où $\mathbf{K}$ est la matrice noyau symétrique (ou encore matrice de Gram) de coefficients $\mathrm{K}_{i,j} := \kappa(\mathbf{x}_i, \mathbf{x}_j)$ et $\alpha^\top$ est la transposée du vecteur α.

En introduisant les équations 14.71 et 14.72 dans l'équation 14.70, on obtient :

$$\begin{aligned} h^\star_{\mathcal{S}} \;&=\; \underset{h \in \mathcal{H}}{\mathrm{ArgMin}} \Big[\frac{1}{m} \sum_{i=1}^{m} \ell(h(\mathbf{x}_i), u_i) \;+\; \lambda ||h||^2 \Big] \\ &=\; \underset{\mathbf{w}}{\mathrm{ArgMin}} \Big[\frac{1}{m} \sum_{i=1}^{m} \ell(\langle \mathbf{w}, \Phi(\mathbf{x}_i) \rangle, u_i) \;+\; \lambda ||h||^2 \Big] \\ &=\; \underset{\alpha}{\mathrm{ArgMin}} \Big[\frac{1}{m} \sum_{i=1}^{m} \ell\big(\sum_{i=1}^{m} \alpha_i \, \kappa(\mathbf{x}_i, \mathbf{x}), u_i \big) \;+\; \lambda \alpha^\top \mathbf{K} \, \alpha \Big] \end{aligned} \tag{14.73}$$

Une somme finie de fonctions convexes, toutes définies sur le même ensemble convexe, est aussi une fonction convexe. De plus, la fonction noyau κ étant, par définition, définie positive, la forme quadratique $\alpha^\top \mathbf{K} \, \alpha$ est convexe par rapport à α. L'hypothèse $h^\star_{\mathcal{S}}$ est donc la solution d'un problème convexe de dimension finie.

De nombreuses **méthodes** existent pour résoudre ces problèmes. Ce qui rend spécifique le problème lié aux SVM est le grand nombre d'exemples qui peut être traité et la dimension éventuellement élevée de $\mathcal{X}$. En revanche, on peut essayer de tirer profit du fait que, souvent, le nombre d'exemples critiques, ceux qui déterminent la solution, est réduit. Parmi les approches existantes, on citera, sans prétention à l'exhaustivité :

- Les *méthodes de gradient* et de *gradient conjugué* sont des méthodes de référence pour ces problèmes. Elles requièrent cependant, pour être efficaces, de stocker en mémoire la matrice noyau $\mathbf{K}$ afin de faire des inversions de matrices. Cette matrice n'est pas, en général, parcimonieuse, et il faut donc être prêt à stocker de l'ordre de $m(m+1)/2$ coefficients, chacun requérant de l'ordre de 8 octets (en double précision). Cela limite l'usage de ces méthodes à des échantillons de taille inférieure au million d'éléments pour le moment.

- Les *méthodes de point intérieur* sont appropriées pour des tailles d'échantillon petites à modérées et elles sont à la fois fiables et précises. L'idée générale de ces méthodes est de résoudre simultanément la forme primale et la forme duale du problème, en imposant les conditions de Karush-Kuhn-Tucker de façon itérative. Cependant, comme pour les méthodes de gradient, ces méthodes précalculent la matrice $\mathbf{K}$, ce qui est maladroit lorsque de nombreux exemples ont une influence faible ou nulle sur la solution.

- Lorsque l'échantillon de données est de grande taille (plusieurs centaines de milliers ou plusieurs millions), les *méthodes de décomposition* sont souvent une solution envisageable. Elles consistent à décomposer le problèmes en sous-problèmes qui sont résolus de manière itérative. Ici, c'est le vecteur α qui est considéré et optimisé par sous-parties. C'est en particulier

le cas de la méthode SMO (*Sequential Minimal Optimization*) proposée par Platt [Pla99] qui pousse l'approche à son extrême en ne considérant à chaque pas que deux composantes du vecteur α.

- Une autre approche adaptée aux très grands échantillons d'apprentissage est une *optimisation en-ligne* dans laquelle les exemples sont considérés séquentiellement, sans ré-examen ultérieur en général. Cela permet à la fois de réduire la complexité calculatoire, mais aussi d'avoir une méthode qui peut bénéficier naturellement de la connaissance préalable d'une solution approchée. Cela ouvre aussi la voie à un apprentissage adaptatif, lorsque les données ne sont plus stationnaires. La méthode LASVM [LCB06] en est un exemple.

6.2 Aperçu de logiciels spécifiques

Il existe de nombreuses boîtes à outils et packages numériques dédiés à l'optimisation convexe (quadratique) sous contraintes et qui peuvent donc être utilisés dans les méthodes à noyau et en particulier pour les SVM. Cependant, des logiciels ont été spécifiquement conçus pour les méthodes à noyau. Ils sont généralement libres d'utilisation et de droit. Une liste assez complète peut être trouvée sur des sites tels que : `www.kernel-machines.org` et `www.support-vector-machines.org`. Nous en mentionnons certains des plus connus ici, sans prétendre qu'ils sont meilleurs que les autres. Chaque package a en général son domaine d'application privilégié, et il est intéressant de déterminer à quel domaine se rapporte son propre problème. À ce sujet, le tableau 10.1 figurant en p.280 de [SS02] constitue un arbre de décision utile pour savoir à quel type de problème on a affaire et quelle famille de méthode est *a priori* la plus adaptée.

`LIBSVM`

Ce logiciel permet de traiter des problèmes de classification, de régression et d'estimation de densité. Il est codé en `C++` et en `Java`. Il est régulièrement mis à jour et des interfaces existent avec d'autres langages tels que `R`, `MATLAB`, `Python`, `Perl` ou logiciels tels que `Weka`. Par ailleurs, ce logiciel est accompagné d'outils appelés `LIBSVM Tools` ainsi que d'une interface graphique très appréciable pour faire des démonstrations pédagogiques en classification ou en régression.

`SVM`^{light}

`SVM`^{light}, écrit en `C` a été l'une des premières implantations permettant de faire de la classification ou de la régression sur des bases de données de taille importante. Ce logiciel fait maintenant partie d'un logiciel plus général, `SVM`^{struct}, dédié à la prédiction de données multivariées ou structurées, tels que des arbres, des séquences ou des ensembles. Cela permet en particulier d'aborder des problèmes comme l'analyse de texte ou l'alignement de séquences biologiques.

`R`

`R` est un logiciel statistique qui peut être utilisé pour le calcul de SVMs pour la classification ou la régression en ayant recours, par exemple, à la fonction `svm` développée par D. Meyer à partir du package `e1071`. L'emploi conjoint des ressources graphiques de `R` permet d'aborder des échantillons de taille moyenne et de réaliser des interfaces faciles avec d'autres traitements statistiques. De plus, le package `svmpath` développé par T. Hastie permet de calculer le chemin de régularisation complet pour des petits échantillons et avec recours à la fonction de perte coude (« hinge loss ») (voir section 6.3).

`SimpleSVM`

Écrite en `Matlab`, cette boîte à outils utilise la méthode d'optimisation itérative et de maintien des contraintes actives pour pouvoir traiter des échantillons d'apprentissage de (très) grande taille. `SimpleSVM` permet de réaliser de la classification à une classe (« One Class SVM »), de

la classification et de la régression. Réalisant un apprentissage en-ligne, `SimpleSVM` permet à la fois de faciliter l'apprentissage si une solution approchée est connue et fournie en entrée, et de réaliser des apprentissages lorsque l'environnement est non stationnaire.

6.3 La détermination des hyper-paramètres

Les paramètres à régler pour utiliser les SVM sont :

1. la *fonction noyau* employée (avec éventuellement ses propres paramètres)
2. le *paramètre* C contrôlant le compromis entre fidélité aux données et tolérance aux écarts.

Leur détermination revient à sélectionner le modèle (la classe d'hypothèses) le plus approprié (voir chapitre 3).

Le choix de la fonction noyau est prépondérant. Les figures 14.16 et 14.17 réalisées grâce à un logiciel de démonstration en Java[13] pour des données décrites sur un plan montrent les frontières de décision obtenues avec des fonctions noyau différentes.

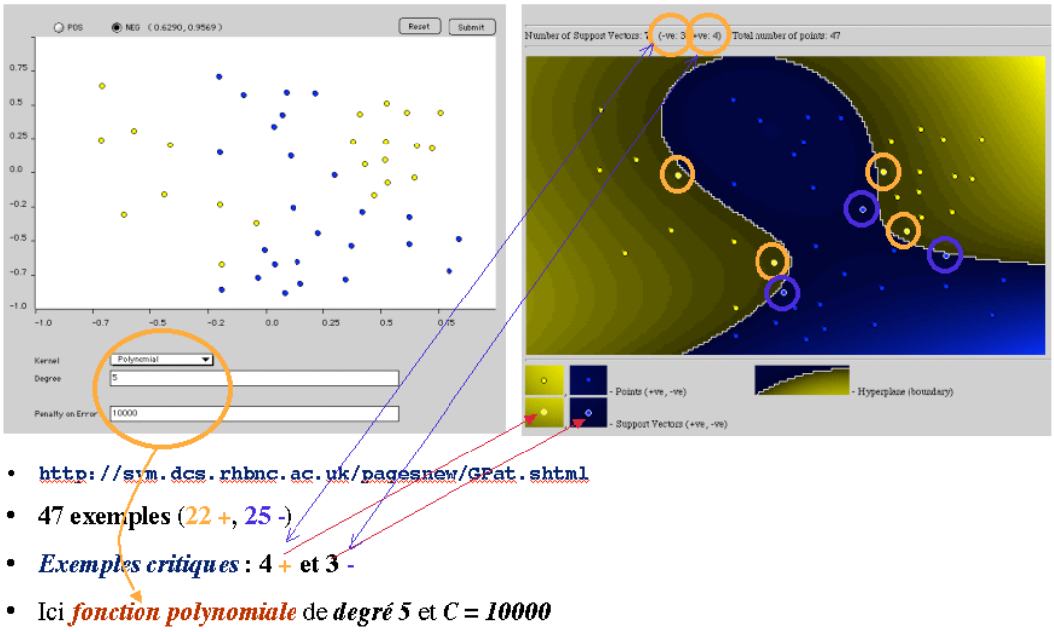


FIG. 14.16: *Dans la fenêtre de gauche, quarante-sept exemples d'apprentissage (vingt-deux de la classe '+' et vingt-cinq de la classe '-') ont été disposés par l'utilisateur. La fenêtre de droite montre la frontière de décision obtenue avec un SVM de noyau polynomial de degré 5 avec une constante* $C = 10000$ *(c'est-à-dire avec une faible tolérance aux exemples mal classés). La frontière s'appuie sur sept exemples critiques dont quatre dans la classe '+' et trois dans la classe '-'.*

La démarche habituelle pour choisir la fonction noyau et le paramètre C consiste à utiliser une procédure de validation croisée pour estimer la performance correspondant à chaque choix. Cependant, il s'agit là d'une approche par essais et erreurs seulement guidée par l'art de l'expert. Des recherches récentes ont porté sur des méthodes plus systématiques, et si possible automatiques, d'exploration.

[13] Disponible à http://svm.dcs.rhbnc.ac.uk/pagesnew/GPat.shtml

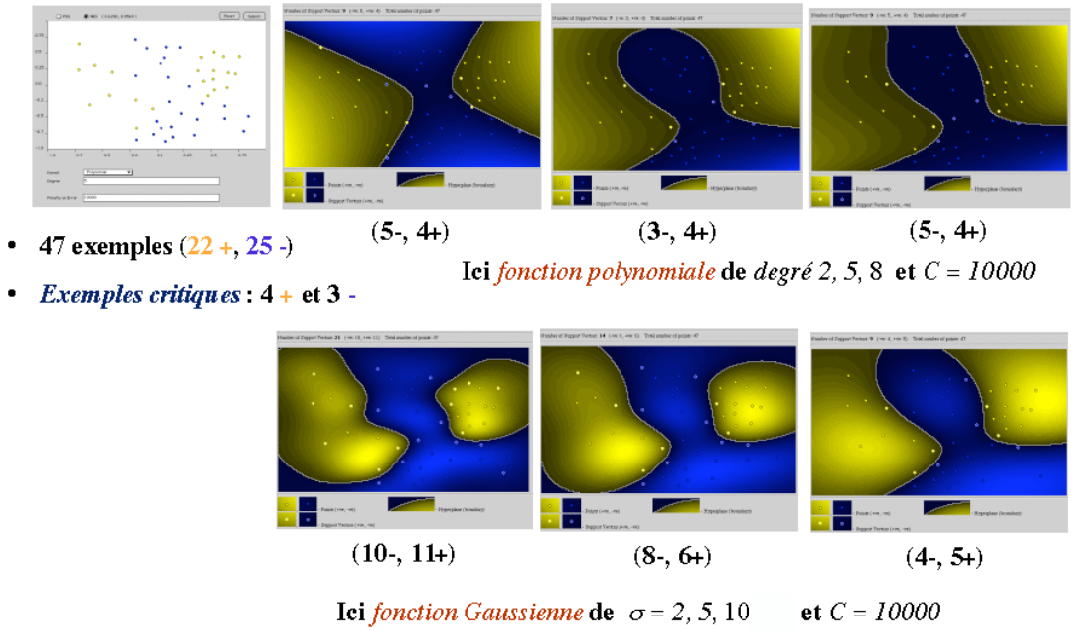


FIG. 14.17: *Pour les mêmes exemples d'apprentissage que ceux de la figure 14.16, on voit l'effet de différents choix de fonctions noyau (polynomiale de degré 2, 5 et 8 en haut et Gaussienne d'écart-type 2, 5 et 10 en bas). Le nombre d'exemples critiques de chaque classe est indiqué entre parenthèses en-dessous des imagettes.*

Ainsi, on voudrait pouvoir sélectionner automatiquement la meilleure fonction noyau[14], ou bien même apprendre une combinaison de fonctions noyaux pour faire face à des données hétérogènes par exemple (voir [SRSS06]).

De même, des techniques ont été testées pour le réglage automatique du paramètre C. Celui-ci peut en effet être considéré comme un paramètre de régularisation : plus la valeur de C est grande, plus on favorise les frontières de décision respectant les exemples, c'est-à-dire les plaçant du bon côté des marges. Inversement, des valeurs de C petites correspondent à des solutions plus régulières. Il est donc tentant de chercher à importer les méthodes d'optimisation du paramètre de régularisation vues pour les méthodes linéaires (voir section 5.2 dans le chapitre 9) dans le cadre des méthodes à noyaux. Là aussi on fera varier continûment la valeur de C sur $[0, \infty[$, et on cherchera la valeur pour laquelle l'erreur estimée par validation croisée est minimale. Au lieu de faire varier les coefficients attachés à chaque variable, comme dans la régression linéaire par exemple, ce sont les multiplicateurs de Lagrange α_i attachés aux exemples d'apprentissage qui seront ici contrôlés par les variations de C. La formulation duale du problème d'optimisation à résoudre (voir l'équation 14.22) montre que l'on est proche d'une pénalisation L_1, ce qui est conforme à la parcimonie observée des solutions en terme d'exemples support. Par ailleurs, l'exploration du chemin de régularisation peut se faire par morceaux (car les multiplicateurs de Lagrange $\alpha_i(C)$ sont des fonctions linéaires par morceaux en fonction de C), donc avec un coût raisonnable (voir [HRTZ04]). Lorsque C s'accroît (ce qui correspond à un λ décroissant), la marge décroît car le système tente de mettre tous les exemples du bon côté de la marge. Les valeurs α_i associées aux points initialement mal placés par rapport à la marge passent de la valeur $\alpha_i = 1$

[14] Voir le Workshop « Kernel Learning : Automatic Selection of Optimal Kernels » à l'url : http://www.cs.nyu.edu/learning_kernels/ comme point d'entrée vers d'autres références sur ce sujet.

à la valeur $\alpha_i = 0$ quand ils passent en dehors de la marge. Leur valeur passe de 1 à 0 tandis qu'ils sont exactement sur la marge.

La figure 14.18 (tirée de la thèse de G. Loosli [Loo06]) illustre l'effet de la variation de C dans le système ν-SVM.

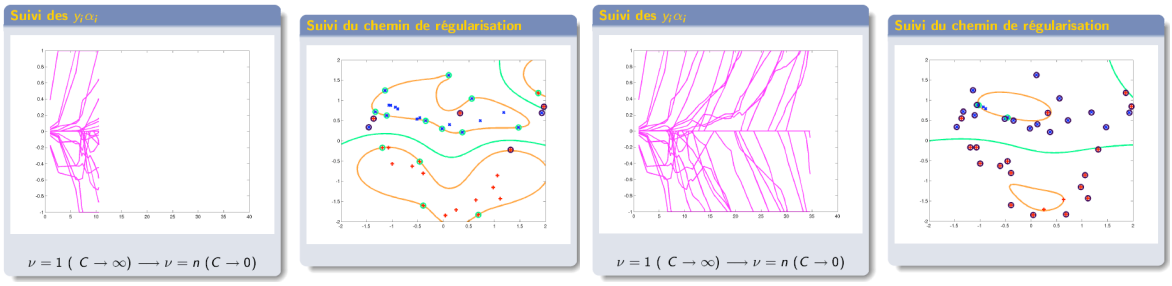


FIG. 14.18: *Pour deux valeurs de C. À gauche, la valeur de C en abscisse variant de ∞ à 0, et les valeurs des coefficients des exemples d'apprentissage en ordonnées ($y_i\alpha_i$). À droite, la frontière de décision correspondante. Les exemples mal positionnés sont en encerclés.*

6.4 Limites des méthodes à noyaux

Les méthodes à noyau présentent certains inconvénients et difficultés de mise en œuvre qui peuvent limiter leur mise en pratique. Le premier problème est lié au manque d'interprétabilité des hypothèses produites. Il est en général impossible de comprendre d'où vient la puissance (ou le manque de puissance) de la fonction de décision obtenue. Par exemple, il n'est pas possible de distinguer les variables pertinentes de celles qui ne le sont pas. De même, sauf par la définition des fonctions noyaux utilisées, il est impossible de tirer parti de connaissances *a priori* pour influencer et aider l'apprentissage. En ceci, encore davantage que les réseaux de neurones, les méthodes à noyaux sont des méthodes de type « boîte noire ».

Dans cet ordre idée, il est essentiel de réaliser que la force des méthodes à noyau, c'est-à-dire leur généralité, s'appuyant sur la propriété que de nombreux algorithmes peuvent se mettre sous une forme dans laquelle les données n'interviennent que par des produits scalaires $\langle \mathbf{x}, \mathbf{x}' \rangle$, est aussi une faiblesse, dans la mesure où seules les informations de distances entre points d'apprentissage sont prises en compte. Si d'autres informations, telles que l'orientation dans l'espace $\mathcal{X}$, sont importantes, elles seront, par nécessité, ignorées.

Par ailleurs, les méthodes à noyaux n'échappent pas au « fléau de la dimensionalité » dénoncé par Bellman en 1961 ([Bel61]) selon lequel, dès que le nombre de dimensions de l'espace des entrées est important (supérieur à une dizaine), la performance en apprentissage, pour un échantillon d'apprentissage de taille m limitée, dépend de manière cruciale de la fonction de distance choisie pour comparer les entrées. Cela se traduit par la nécessité de définir la ou les fonctions noyau utilisées de manière appropriée au problème, ce qui est souvent loin d'être trivial. Malheureusement, les résultats sont généralement très sensibles à la fonction noyau retenue.

Grâce à la propriété de parcimonie qui peut être associée à l'utilisation de SVM, à savoir qu'une faible proportion des exemples sont critiques lorsque les classes ont peu de recouvrement (faible erreur de Bayes), la fonction de prédiction est facile à calculer. En revanche, le problème d'optimisation quadratique de l'apprentissage d'un hyperplan optimal implique une complexité calculatoire en m^2 multipliée par le nombre d'exemples critiques. De nombreux travaux visent

à étendre le domaine d'application des SVM à de grandes tailles de bases de données. Il reste cependant difficile, pour le moment, de traiter des bases de taille $m \gtrsim 10^5$.

7. Bilan et perspectives

Le développement des méthodes à noyaux et particulièrement des séparateurs à Vastes Marges marque le point de convergence de plusieurs concepts essentiels : le *passage au non linéaire*, grâce à l'astuce des noyaux, d'une grande famille d'algorithmes linéaires ne s'appuyant que sur des produits scalaires dans l'espace des entrées $\mathcal{X}$, la *régularisation automatique* grâce au théorème de représentation, la *marge comme mesure de capacité* de l'espace des hypothèses, la notion de *sélection de modèle a posteriori*.

Finalement, les méthodes à noyaux encapsulent les données par rapport à l'algorithme et permettent ainsi l'utilisation d'un algorithme générique sur différents types de données et pour plusieurs tâches.

On comprend dès lors l'excitation des théoriciens et des praticiens pour cette nouvelle boîte à outils. De plus, ces méthodes peuvent être étendues à des espaces non vectoriels par la définition de distances et de similarité via des fonctions noyau adéquates. Finalement, il existe un lien fort entre ces méthodes et la recherche d'expressions parcimonieuses des régularités du monde, grâce à la régularisation en utilisant la norme ℓ_1,, ce qui est au cœur de la théorie de l'information et qui est très séduisant comme démarche générale pour l'apprentissage.

Il reste cependant beaucoup à faire pour mieux maîtriser ces méthodes et les étendre à d'autres applications.

Par exemple, on sait encore très peu de choses sur la convergence du risque réel vers le risque empirique dans le cas de données et de sorties structurées (voir chapitre 20). Globalement, les solutions trouvées sont surtout de nature heuristique sans que les outils formels n'aient, encore, suivi.

Un domaine de recherche lié à celui-ci concerne l'*apprentissage des fonctions noyau* à partir des données.

De nombreux travaux portent sur le développement d'*algorithmes à noyaux en-ligne* afin de pouvoir traiter de grandes bases de données (puisque le coût computationnel des méthodes à noyaux est à peu près une fonction quadratique du nombre d'exemples) et les flux de données. L'adaptation à des concepts cibles dérivants reste encore à explorer (voir chapitre 20).

Finalement, il faut aussi réaliser **les limites** potentielles de ces approches. Ainsi la plupart des travaux privilégient l'utilisation de fonctions noyau gaussiennes, ou en tous les cas à support borné. Si les praticiens se sentent plus à l'aise avec ce genre de fonction noyau en raison de leur plus grande interprétabilité comme mesure de similarité, cela se traduit aussi, implicitement, par un biais vers la recherche de régularité « lisses », c'est-à-dire dont les dérivées sont sévèrement bornées. Or, il existe de nombreux domaines dans lesquels ce biais n'est pas approprié. En général, les régularités complexes impliquent des interdépendances non locales dans les données qui ne peuvent pas être capturées simplement si on utilise des noyaux « locaux » (voir la section 6 du chapitre 10, et l'excellent article de Bengio et Le Cun [BL07]). Plus généralement, il est douteux que les méthodes à noyaux puissent représenter les concepts relationnels complexes sous-jacents à certains problèmes. Cependant, il s'agit là de questions ouvertes.

Notes historiques et sources bibliographiques

Le recherche de régularités dans les données en apprentissage artificiel a été dominé par deux courants distincts, l'un se concentrant sur l'apprentissage dans des espaces d'hypothèses à bases d'expressions symboliques et structurés par des opérations logiques, l'autre de nature plus géométrique et statistique. Dans ce dernier courant, trois grandes étapes peuvent être discernées. La *première*, née dans les années trente, en particulier avec Fisher, puis ensuite dans les années soixante avec le Perceptron et autres techniques connexionnistes, s'est intéressée à des méthodes efficaces de recherche de régularités linéaires dans l'espace des entrées. Une *deuxième étape*, démarrée dans les années quatre-vingt a permis de dépasser les limitations des premières techniques en autorisant la recherche de régularités non linéaires. Ce fut la révolution des réseaux de neurones et des arbres de décision. Cependant ces méthodes avaient un caractère très heuristiques et étaient l'objet d'analyses théoriques incomplètes car très difficiles. La *troisième étape* a eu lieu vers le milieu des années quatre-vingt-dix avec l'émergence des approches à noyau permettant la recherche de régularités non-linéaires avec l'efficacité et des fondements théoriques réservés jusque là aux méthodes linéaires. Cela a provoqué une vague de travaux sur des développements de nouveaux algorithmes et de nouvelles analyses dans plusieurs communautés : apprentissage artificiel, statistique et même un renouveau d'intérêt en mathématique pour les espaces fonctionnels définis à partir de fonctions noyaux.

Plus précisément, tout en restant très schématique, l'histoire des méthodes à noyaux en apprentissage est marquée par les avancées suivantes.

- L'idée d'utiliser des fonctions noyau comme équivalent au calcul de produits scalaires dans un espace de redescription date de 1964 avec l'article de Aizermann, Bravermann et Rozoener [ABR64] sur la méthode des fonctions potentielles, cité dans la première édition du livre si influent de Duda et Hart [DH73].
- Cette idée a été reprise par Boser, Guyon et Vapnik en 1992, [BGV92], et combinée avec l'idée de recherche d'hyperplans à vaste marge, pour définir la méthode des Séparateurs à Vaste Marge (SVM).
- Entre temps, Vapnik avait introduit le principe de minimisation du risque structurel (SRM) pour résoudre le problème du contrôle de l'induction (et de la sélection de modèle) [Vap82]. Il en a fait une justification des SVM, remise en cause plus tard comme insuffisante.
- Une analyse plus prometteuse a été proposée en 1998 par Shawe-Taylor et al. [STBWA98] introduisant l'idée de fonction de baraka (« luckiness framework ») caractérisant une probabilité d'adéquation entre l'algorithme d'apprentissage et les données présentes.
- Pour faire face à la tendance des SVM à marge dure de faire du sur-apprentissage, la technique des marges douces et des variables ressort a été publiée en 1995 [Vap95]. Des fonctions de perte appropriées, telle que la fonction coude (« hinge loss »), ont été ensuite proposées. D'autres analyses, fondées sur une idée de robustesse de l'apprentissage à des variations de l'échantillon d'apprentissage ont été également proposées un peu plus tard [BE02].
- La pleine réalisation du potentiel de l'utilisation des fonctions noyau pour étendre leur portée au delà des SVM s'est faite en particulier grâce à Schölkhopf [Sch97]. Haussler, de son côté, a fait œuvre de pionnier en montrant en 1999, dans un rapport technique [Hau99], comment construire des fonctions noyaux calculables pour des structures de données complexes non vectorielles.
- Le lien entre fonctions noyaux et espace fonctionnel de Hilbert, avec en particulier l'équivalence avec un opérateur de régularisation a été explicité en apprentissage par Smola et Schölkhopf [SS98b] puis par d'autres. La preuve originale du théorème de représentation

peut être trouvée dans [SHSW01]. Une version plus simple de ce théorème avait été prouvé préalablement par Kimeldorf et Wahba [KW70].
- Les années récentes ont vu une analyse extensive du coût computationnel des méthodes à noyaux (voir par exemple [Joa99]). Il s'en est suivi la mise au point de tout un ensemble de logiciels, cités en partie dans la section 6.2.

La très large popularité des méthodes à noyau est associée à la publication de nombreux ouvrages entièrement dédiés à ces méthodes. Nous citerons en particulier les ouvrages suivants :
- L'ouvrage de Shawe-Taylor et Cristianini [STC04] qui prend le parti de mettre l'accent sur les fonctions noyaux et toutes leurs déclinaisons à des structures de données variées. Leur livre précédent était tourné vers la description des SVM [CST00].
- L'ouvrage de Herbrich [Her02b] assez théorique mais très complet sur les problèmes de classification.
- L'ouvrage de Schölkhopf et Smola [SS02] plus heuristique que les précédents et présentant en particulier des méthodes à noyaux pour des tâches diverses : classification, régression, analyse en composantes, etc. Un ouvrage très complet.
- L'ouvrage de Steinwart et Christmann [SC08] qui est le plus théorique. Plus difficile à lire que les autres, il offre aussi des points de vue différents et souvent novateurs.

Résumé

- Les méthodes à noyaux permettent de convertir la plupart des méthodes d'apprentissage linéaires en méthodes adaptées à la recherche de régularités non linéaires en conservant en grande partie les avantages des méthodes linéaires : facilité de mise en œuvre et bons fondements théoriques.
- Les fonctions noyaux permettent une redescription implicite de l'espace des entrées dans un espace dont les dimensions sont des corrélations entre dimensions de description des entrées. L'espace de redescription peut également être vu comme un espace fonctionnel dans lequel on cherche une approximation de fonction.
- Les fonctions noyau calculent implicitement le produit scalaire des projections de deux points de l'espace d'entrée.
- Si on peut attacher une sémantique à la projection ϕ vers l'espace de redescription, il est possible d'interpréter le résultat de l'apprentissage. Sinon, l'apprentissage opère comme une « boîte noire ».
- Les fonctions noyaux ont la propriété que toute matrice de Gram associée de dimension finie est positive semi-définie. Le théorème de Mercer est une formulation équivalente de cette propriété pour les espaces vectoriels.
- Il est possible de construire des fonctions noyaux complexes qui peuvent être adaptées à des données non vectorielles en utilisant des opérations simples de combinaison de fonctions noyau plus simples.
- Les SVM se caractérisent par l'utilisation de fonctions noyau, l'absence de minima locaux et la capacité de contrôle de l'induction fournie par l'optimisation des marges. Les SVM permettent la recherche de régularités dans des espaces de grandes dimensions en contrôlant le risque de sur-apprentissage.
- Les méthodes à Vastes Marges forment une classe de méthodes rigoureusement fondée, parmi les plus utilisées et qui offrent un cadre unificateur pour de nombreuses techniques d'apprentissage.