# Classification supervisée par réseaux multicouches (1)

## *Supervised Classification by Multilayer Networks*

**P. COMON**
THOMSON-SINTRA,
Parc de Sophia Antipolis,
BP 138, F-06561 Valbonne Cedex

Pierre Comon a obtenu le Doctorat INPG en 1985, il est affilié au laboratoire CEPHAG de 1983 à 1986, puis en 1987 il visite le laboratoire Information Systems à l'Université de Stanford. En 1988, il quitte le CNRS pour entrer à Thomson-Sintra dans le service d'Études Amont, département de Sophia Antipolis. Activités : théorie du signal, algorithmique numérique adaptative, détection et localisation de signaux sonar, réseaux de neurones. Il peut être joint par courier électronique à l'adresse comon@mirsa.inria.fr.

**RÉSUMÉ**

Le Perceptron MultiCouche (PMC) est un des réseaux de neurones les plus utilisés actuellement, pour la classification supervisée notamment. On fait dans un premier temps une synthèse des résultats acquis en matière de capacités de représentation dont jouit potentiellement l'architecture PMC, indépendamment de tout algorithme d'apprentissage. Puis on montre pourquoi la minimisation d'une erreur quadratique sur la base d'apprentissage semble être un critère mal approprié, bien que certaines propriétés asymptotiques soient aussi exhibées. Dans un second temps, l'approche bayesienne est analysée lorsqu'on ne dispose que d'une base d'apprentissage de taille finie. A l'aide de certains estimateurs de densité, dont les propriétés remarquables sont résumées, il est possible de construire un réseau de neurones stratifié réalisant la classification bayesienne. Cette technique de discrimination directe semble être supérieure au PMC classique à tous points de vue en dépit de la similarité des architectures.

**MOTS CLÉS**

Neurone, perceptron, rétro-propagation, classification, Bayes, discrimination, apprentissage supervisé, représentation, noyau de probabilité.

***ABSTRACT***

*The Multi-Layer Perceptron (PMC in French) is one of the neural networks the most widely used, particularly for supervised classification. First, existing results on general representation capabilities enjoyed by the PMC architecture are surveyed, independently of any learning algorithm. Then it is shown why the minimization of a quadratic error over the learning set seems an awkward optimization criterion, though some asymptotic properties are also proved. In a second stage, the bayesian approach is analyzed when leaming sets of finite size are at disposal. With the help of certain density estimators whose basic properties are emphasized, it is possible to build a feed-forward neural network implementing the bayesian classification. This technique of direct discrimination seems to perform better than the classical MLP in all respects despite of the similarities of the architectures.*

***KEY WORDS***

*Neuron, perceptron, back-propagation, classification, Bayes, discrimination, supervised learning, representation, probability kernel.*

## 1. Introduction

*Objectifs.* Dans un réseau de neurones, deux modes de fonctionnement peuvent être distingués. Dans le premier, les paramètres du réseau sont ajustés grâce à la présentation d'exemples pour lesquels on connaît la réponse désirée : ce mode de fonctionnement est appelé « phase d'apprentissage ». Dans le deuxième mode, il s'agit d'exploiter le réseau en lui présentant des données inconnues : ce mode d'utilisation est souvent appelé « phase de reconnaissance », ou « phase de généralisation ». Cette dernière appellation sera utilisée dans la suite. Notons que le terme de « phase » peut sembler mal adapté, étant donné que les « phases » d'apprentissage et de reconnaissance peuvent être théoriquement entrelacées. Il ne s'agit pas en réalité de phases distinctes mais bien de modes de fonctionnement.

Ce point de terminologie étant précisé, on peut définir la classe des réseaux « non bouclés » (feed-forward networks). Dans les réseaux de cette classe, les données transitent toujours à sens unique au cours des phases de généralisation, de sorte qu'elles entrent en certains points, suivent un trajet bien défini, et sortent en d'autres points bien précis ; autrement dit, les graphes de connexions de ces réseaux ne comportent pas de boucles. De tels réseaux s'apparentent fortement aux réseaux systoliques (flot de données régulier et localisé, graphe de connexions sans

(1) Travail en partie financé par la D.R.E.T. Soumis à la revue *Traitement du Signal* le 7/9/91 ; révisé le 17/3/92.

boucle, cellules identiques), surtout si les connexions entre couches sont partielles (locales).

Plus précisément, parmi les réseaux de neurones non bouclés décrits dans la littérature [45], [57], [69], deux ont fait l'objet d'utilisations beaucoup plus poussées ; il s'agit du Perceptron MultiCouche (PMC) [25], [34], [35], [43], [44], [66], [67] et du réseau Adaline polynomial (ALN) [36], [49], [67]. On trouvera également des exemples d'application dans les ouvrages plus généraux tels que [45], [57], [69]. Dans cet article, notre attention sera accaparée uniquement par le PMC, et ses propriétés lorsqu'il est utilisé pour la classification supervisée. Lorsqu'il sera fait référence au PMC, il ne s'agira que de l'architecture physique du réseau (de nature stratifiée), mais aucun algorithme particulier ne sera sous-entendu. Le but de cet article est de présenter les propriétés du PMC de façon synthétique, vues sous l'éclairage du traitement du signal.

*Définitions.* La définition et les fonctionnalités du PMC sont décrites par exemple dans [57, chapitres 2 et 8], [37], [34]. Rappelons simplement quelques définitions de base. Un neurone formel peut être considéré comme une application particulière de $\mathbb{R}^M$ dans $\mathbb{R}$ définie comme suit :

$$(1)\quad \forall \mathbf{x} \in \mathbb{R}^M, \quad \mathbf{x} \to g_{w,\theta}(\mathbf{x}) = \\ = f\left(\sum_{m=1}^{M} w_m x_m - \theta\right) = f(\mathbf{w}^t \mathbf{x} - \theta),$$

où $f$ est une fonction scalaire réelle fixée, en général non linéaire, appelée « fonction de transition » ; $w_m$ est le vecteur de « poids synaptiques » pondérant les entrées $x_m$ ; et $\theta$ est un scalaire réel, appelé parfois « offset ». On convient de noter les vecteurs en minuscules grasses, et les matrices en majuscules. Comme cela est souligné dans [69] entre autres, une autre écriture est plus simple dans $\mathbb{R}^{M+1}$, en rajoutant une entrée constante de valeur $-1$, ce qui permet de supprimer le terme constant $\theta$ :

$$(2)\quad g_w(\mathbf{x}) = f'(\mathbf{w}'\,\mathbf{x}), \text{ avec } x_{M+1} = -1 \text{ et } w_{M+1} = \theta.$$

Ces deux écritures conduisent aux représentations schématiques des figures 1 *a* et 1 *b*.

Un Perceptron MultiCouche (PMC) est une architecture stratifiée de neurones formels. La figure 2 *a* représente un perceptron à 3 couches. Par convention, la couche d'entrée n'exécute rien, et deux couches seulement sont donc actives. La couche du milieu est la seule couche « cachée », dans le sens où ses sorties ne sont pas directement observables. La figure 2 *b* schématise la simple extension du PMC permettant de se ramener à des fonctions linéaires, et non plus affines, en dimension M + 1. Si P cellules sont présentes dans la couche cachée, les sorties du réseau de la figure 2 *a* vérifient par conséquent une relation du type :

$$(3)\quad y_k = f_3\left(\sum_{p=1}^{P} a_{kp}\, f_2\left(\sum_{m=1}^{M} w_{pm}\, x_m - \theta_p\right) - \mu_k\right).$$

Les fonctions de transition $f_2$ et $f_3$ des deuxième et troisième couches peuvent être différentes. En outre, on suppose souvent dans le PMC que les fonctions de transition sont identiques au sein d'une même couche [25], [67].

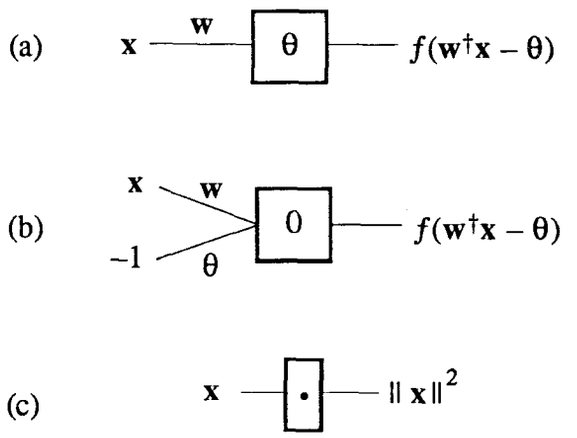


**Figure 1. — Cellules élémentaires, fonctionnalités, et convention de représentation...**

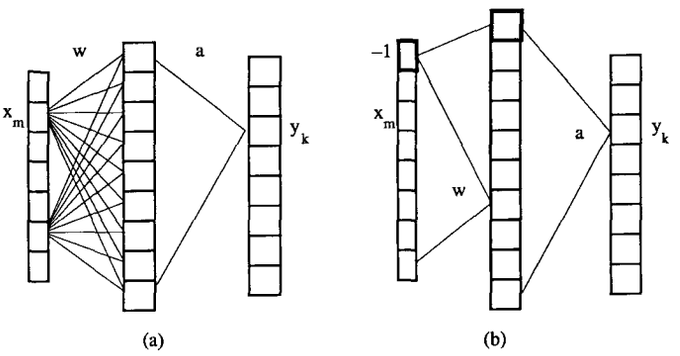


**Figure 2. — Perceptron multicouche ; (*a*) PMC à trois couches standard ; (*b*) PMC à trois couches avec adjonction d'une entrée constante et égale à 1. C'est principalement ce deuxième type de réseau qui est utilisé, car il permet d'éviter de représenter les offsets.**

Dans la suite un autre type de cellule moins standard sera utilisé, délivrant en sortie la norme carrée de son vecteur d'entrée. Cette cellule sera distinguée des autres par un point (*fig.* 1 *c*). L'usage de fonctions radiales a été préconisé initialement dans [42], puis ultérieurement dans [50], [46], mais des cellules polynomiales de forme plus générale ont déjà été expérimentées [17], [64], sans parler du réseau ALN [67], [49], [36].

*Les problèmes pour le Traitement du Signal.* En traitement du signal, les signaux mesurés aussi bien que les signaux délivrés en sortie de traitement sont toujours bornés, en raison de la dynamique finie des appareils, qui est d'ailleurs a priori connue. Si on suppose en outre que les signaux sont à valeurs réelles, alors on est en droit d'admettre que les entrées, $\mathbf{x}$, et les sorties $\mathbf{y}$ sont à valeurs dans des ensembles $\mathbb{E}$ et $\mathbb{F}$, tous deux compacts de $\mathbb{R}^M$ et $\mathbb{R}^K$ respectivement. Un classifieur bayesien par exemple, peut être considéré comme une application $\varphi$ de

$\mathbb{E}$ dans $\mathbb{F} = [0, 1]^K$ qui, à tout vecteur d'observation $\mathbf{x}$, associe le K-uplet des probabilités d'appartenance de $\mathbf{x}$ à chacune des classes $\omega_k$, notées $p(\mathbf{x}, \omega_k)$.

Le problème de l'apprentissage supervisé consiste à identifier cette application à partir d'un ensemble d'exemples, $A(N) = \{\mathbf{x}^{(n)}, \mathbf{y}^{(n)}, 1 \leq n \leq N\}$, appelé base d'apprentissage. Deux problèmes peuvent être abordés. Le premier est celui de la représentation exacte de l'application

$$(4) \qquad \mathbf{y}^{(n)} = \varphi(\mathbf{x}^{(n)}), \quad \forall n \in \{1, ..., N\} .$$

Ce problème est rarement rencontré en pratique (mémoires associatives par exemple [2]) : il consiste à représenter sans erreur une application à l'aide de fonctions imposées (c'est un codage sans perte d'information). Pour ce faire le réseau sera en général de grande taille (comparable à N). Le deuxième problème est celui de l'approximation d'une application avec un niveau de tolérance donné et le minimum de cellules :

$$(5) \qquad \mathbf{y}^{(n)} \approx \varphi(\mathbf{x}^{(n)}), \quad \forall n \in \{1, ..., N\} .$$

Il existe aussi le problème dual du précédent, à savoir celui de la représentation la plus fidèle possible avec un nombre de cellules donné. Ces problèmes seront abordés de différentes façons dans cet article.

Dans le chapitre qui va suivre, on fera une synthèse des résultats connus concernant les capacités de représentation de l'architecture PMC à trois ou quatre couches, en ne tenant compte que des contraintes imposées par les définitions précédentes. Dans un troisième chapitre, on s'intéressera au choix du critère d'optimisation et aux conséquences qu'il entraîne en classification. Dans le chapitre quatre, les performances de l'algorithme d'apprentissage le plus répandu pour le PMC seront analysées. Une alternative sera proposée dans le chapitre cinq, avec une architecture légèrement différente, et un algorithme d'apprentissage approprié. Enfin, un problème de classification binaire construit à l'aide de simulations sera traité par les deux méthodes à des fins d'illustration dans une dernière partie.

## 2. Capacités ultimes de représentation de l'architecture PMC

On passe en revue dans cette partie les capacités qu'a le PMC à représenter des applications quelconques lorsque son nombre de cellules peut être arbitrairement grand. C'est en cela qu'elles sont ultimes. Une architecture incapable de représenter une application $\varphi$ avec une infinité de cellules le sera a fortiori avec un nombre fini. Étudier les capacités ultimes a donc un sens.

### 2.1. REPRÉSENTATION EXACTE

Depuis quatre ou cinq ans, on a pu voir dans la littérature proliférer un grand nombre d'articles présentant des règles plus ou moins intuitives permettant d'anticiper les capacités du réseau PMC à représenter tel ou tel type d'application, notamment booléenne. Un bon exemple de ce genre de présentation est l'article de Lippmann [37] où les arguments avancés sont de nature uniquement expérimentale, tandis que [41] et [39] en sont de moins convaincants. Sur le plan théorique, le premier résultat susceptible de nous intéresser remonte à 1957 avec le théorème de Kolmogorov.

(6) Théorème de Kolmogorov [32].

Soit $\varphi(\mathbf{x})$ une application réelle continue quelconque de $[0, 1]^M$ dans $[0, 1]$, et $\{f_{pq}, 0 \leq q \leq 2M, 1 \leq p \leq M\}$ des applications continues bijectives sur $[0, 1]$, dépendant de M (elles sont donc fixées lorsque M est fixé). Alors il existe des fonctions continues $\chi_q$, dépendant de $\varphi$, telles que :

$$\varphi(\mathbf{x}) = \sum_{q=0}^{2M} \chi_q \left( \sum_{p=1}^{M} f_{pq}(\mathbf{x}_p) \right) .$$

La difficulté d'utilisation de ce théorème vient du fait que les fonctions $\chi_q$ dépendent de $\varphi$, ce qui est contraire à notre définition du neurone formel où les fonctions de transition sont fixées à l'avance. Ce théorème a été sensiblement amélioré dans les années 60 avec le résultat suivant.

(7) Théorème de Sprecher [59].

Soit $\varphi(\mathbf{x})$ une application réelle continue quelconque de $[0, 1]^M$ dans $[0, 1]$, $\lambda$ une constante et $f$ une application lipschitienne sur $[0, 1]$ de rapport inférieur à 1, fixées à M fixé. Alors il existe une fonction continue $\chi$, dépendant de $\varphi$, et un nombre rationnel $\varepsilon$ tels que :

$$\varphi(\mathbf{x}) = \sum_{q=0}^{2M} \chi \left( \sum_{p=1}^{M} \lambda^p f(\mathbf{x}_p + \varepsilon q) + q \right) .$$

Ici, seule la fonction $\chi$ dépend de la fonction $\varphi$, ce qui est un mieux, mais rend le théorème malgré tout inapplicable. De plus, ces deux théorèmes ne donnent aucune idée de procédé constructif permettant de calculer les paramètres intervenant dans la décomposition [24]. Ceci s'améliore un peu avec le théorème suivant, dont la démonstration utilise les propriétés de la transformée de Fourier.

(8) Théorème de Irie et Miyake [27].

Soient $\varphi(\mathbf{x})$ une application réelle continue quelconque de $\mathbb{R}^M$ dans $[0, 1]$, et $f$ une application fixée de $\mathbb{R}$ dans $[0, 1]$, toutes deux absolument intégrables. Alors il existe une fonction poids $B(\mathbf{w}, v)$ définie sur $\mathbb{R}^m \times \mathbb{R}$ telle que :

$$\varphi(\mathbf{x}) = \int_{\mathbb{R}^M} \int_{-\infty}^{\infty} B(\mathbf{w}, v) f(\mathbf{w}^t \mathbf{x} + v) \, d\mathbf{w} \, dv .$$

Ce théorème est le premier qui, historiquement, a été réellement applicable aux réseaux de neurones. La figure 3 représente le réseau PMC à trois couches, sous-jacent au théorème, dont la couche cachée est un continuum de cellules neuro-mimétiques.

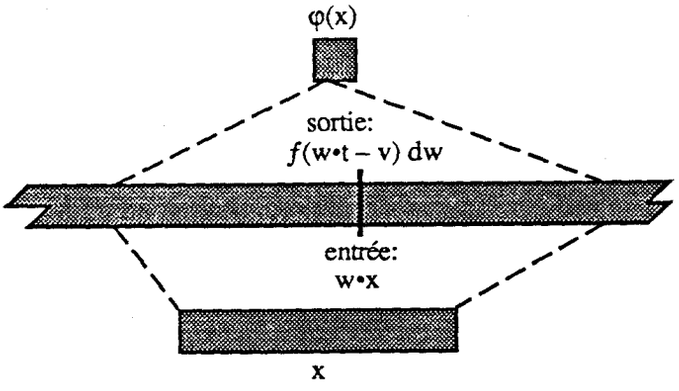


**Figure 3. — Réseau infini de Irie et Miyakie ; la couche cachée est continue.**

## 2.2. REPRÉSENTATION APPROCHÉE

Les résultats analysés dans cette section sont plus exploitables dans la mesure où ils concernent les capacités du PMC à approximer une application (continue) avec une précision arbitraire, mais finie.

(9) Définition.

On dira qu'une fonction $f$ est saturante si elle vérifie :

$$\lim_{u \to -\infty} f(u) = 0 \quad \text{et} \quad \lim_{u \to +\infty} f(u) = 1 .$$

Notons que la fonction $f(u)$ ne décrit pas nécessairement l'intervalle [0, 1], et n'est pas forcément continue.

(10) Théorème de Hornik [26].

Soit $f$ une fonction saturante croissante fixée. Alors toute fonction continue $\varphi(\mathbf{x})$ définie sur un compact $\mathbb{K}$ de $\mathbb{R}^M$, est limite uniforme (sur $\mathbb{K}$) de fonctions de la forme :

$$h_n(\mathbf{x}) = \sum_{j=1}^{J(n)} \alpha_j(n)\, f(\mathbf{w}_j(n)^t\, \mathbf{x} + \theta_j(n)) ,$$

où $\alpha_j(n)$ et $\theta_j(n)$ sont des scalaires réels, et $\mathbf{w}_j(n)$ des vecteurs de $\mathbb{R}^M$.

Ce résultat est valable pour des fonctions de transition croissantes. Le résultat ci-dessous le complète dans le sens où la monotonie est remplacée par la continuité. Rien n'empêche bien sûr d'utiliser des fonctions de transition à la fois continues et croissantes.

(11) Théorème de Cybenko [12].

Soient $f$ une fonction continue discriminante fixée, et $\mathbb{K}$ un fermé borné de $\mathbb{R}^M$. Alors l'espace $\mathbb{G}$ des fonctions de $\mathbb{K}$ dans $\mathbb{R}$ de la forme

$$h(\mathbf{x}) = \sum_{j=1}^{J} \alpha_j\, f(\mathbf{w}_j^t\, \mathbf{x} + \theta_j) ,$$

est dense dans l'espace $C(\mathbb{K})$. La définition exacte d'une fonction discriminante nous mènerait trop loin, et peut être renvoyée à [12]. La propriété importante qu'il est suffisant de connaître pour mesurer la portée du théorème ci-dessus est que [12] :

(12) Toute fonction bornée saturante est discriminante.

Il existe un certain nombre de références moins percutantes à notre avis, citons [19] et [14], entre autres. Le fond du raisonnement est de toutes façons toujours le même, et il est basé sur le théorème de Stone-Weierstrass [56]. Si cela ne se voit pas immédiatement dans l'énoncé des théorèmes, cela apparaît dans la démonstration. En effet, dans [26] il est fait clairement usage des polynômes trigonométriques et de leurs propriétés. Dans [27] ou [19], c'est la transformée de Fourier qui est utilisée. En revanche dans [12], Cybenko utilise successivement les théorèmes de Hahn-Banach et de Riesz, de sorte que l'on n'y voit pas apparaître le théorème Stone-Weierstrass. En réalité, les polynômes trigonométriques interviennent dans [12] lorsqu'il s'agit de montrer que toute fonction bornée saturante est discriminante.

Les théorèmes (10) et (11) montrent de façon claire qu'un PMC à trois couches peut approcher n'importe quelle application, pourvu qu'il ait suffisamment de cellules et que la fonction de transition de ses neurones formels satisfasse certaines propriétés (assez faibles). Toutefois, le nombre total de cellules requises peut fort bien être plus faible avec un PMC à plus de trois couches pour approcher la même application avec la même tolérance. Ce problème reste ouvert sur le plan théorique, et peut sans doute être résolu pour certaines classes d'applications, mais certainement pas pour des applications continues quelconques. En effet, il n'existe pas d'architecture qui soit optimale pour tous les problèmes [12], [42], mais plutôt des architectures qui soient statistiquement optimales pour des classes de problèmes [4].

Cette constatation nous conduirait à l'étude de la complexité de l'apprentissage dans sa définition générale si nous voulions aller plus loin. Par exemple selon Valiant [62], on dira qu'un algorithme est optimal pour l'apprentissage d'une application quelconque $\varphi$ d'une famille $\Phi$ si, pour toute densité $p(\mathbf{x})$, tout exemple $\{\mathbf{x}, \varphi(\mathbf{x})\}$ tiré selon $p(\mathbf{x})$, et pour tout couple de réels positifs $(\varepsilon, \delta)$, il est capable de fournir avec une probabilité au moins égale à $1 - \delta$ une solution $\hat{\varphi}$ telle que la probabilité $pr(\hat{\varphi}(\mathbf{x}) \neq \varphi(\mathbf{x})) < \varepsilon$. De plus, l'algorithme ne doit nécessiter qu'un nombre d'opérations qui soit fonction polynomiale de $\delta$ et $\varepsilon$. Toutefois, cette définition est beaucoup trop exigeante en pratique si aucune restriction n'est faite sur la densité $p(\mathbf{x})$. D'autres approches existent [3], [4c], [13], [40] ; citons à titre indicatif seulement l'approche de Baum [4] où la densité $p(\mathbf{x})$ est soumise à la contrainte d'être issue d'une famille donnée, et d'être choisie indépendamment de l'application $\varphi$. Cette définition de la complexité est probablement la plus réaliste à l'heure actuelle.

## 2.3. PROCÉDURES CONSTRUCTIVES DANS LE CAS BOOLÉEN

Les théorèmes précédents prouvaient l'existence de représentations sans pouvoir inspirer de procédures viables pour les construire. La procédure décrite maintenant est très simple et constructive mais n'est malheureusement valable que pour des fonctions booléennes. Elle peut être vue comme une variante de la procédure succinctement décrite dans [37, p. 16], utilisant des polyèdres au lieu

d'hypercubes ; elle permet de construire n'importe quelle fonction booléenne donnée définie sur $\mathbb{R}^M$ avec un nombre réduit de cellules. La fonction de transition $f$ désigne dans cette section la fonction échelon unité (donc croissante discontinue).

Les résultats énoncés ci-dessous sont immédiats. Avec une cellule, on peut réaliser :

(13) la synthèse d'une fonction indicatrice du demi espace $\{\mathbf{w}^t \mathbf{x} + \theta \geq 0\}$ avec

$$g : \quad \mathbf{x} \in \mathbb{R}^M \rightarrow g(\mathbf{x}) = f(\mathbf{w}^t \mathbf{x} + \theta)$$

(14) la synthèse de la complémentation grâce à la relation

$g(\mathbf{x})$ indicatrice d'un ensemble $\mathbb{D} \Leftrightarrow$

$\Leftrightarrow 1 - g(\mathbf{x})$ indicatrice de son complémentaire $\bar{\mathbb{D}}$ .

(15) la synthèse de l'opération de réunion, notée $\vee$, en remarquant que

si $y_n = g_n(\mathbf{x})$ sont les fonctions indicatrices de N ensembles $\mathbb{D}_n$, $y_n \in \{0, 1\}$, alors :

$$\bigvee_{n=1}^{N} y_n = f\left(\sum_{n=1}^{N} y_n - 0.5\right)$$

est la fonction indicatrice de $\bigcup_{n=1}^{N} \mathbb{D}_n$.

(16) la synthèse de l'opération d'intersection, notée $\wedge$, en remarquant de façon similaire que l'application

$$\bigwedge_{n=1}^{N} y_n = f\left(\sum_{n=1}^{N} y_n - N + 0{,}5\right)$$

est la fonction indicatrice de $\bigcap_{n=1}^{N} \mathbb{D}_n$.

Par conséquent, la fonction indicatrice de tout polyèdre convexe peut être représentée exactement par un PMC à trois couches, la couche cachée réalisant la synthèse des hyperplans-frontière, et la troisième couche réalisant leur intersection. Si le polyèdre a P côtés, alors P + 1 cellules actives sont nécessaires dans la couche cachée.

Supposons maintenant que $y_n$, $1 \leq n \leq N$, désignent N variables booléennes. Alors toute fonction booléenne des $y_n$ peut se décomposer sous Forme Normale Conjonctive (FNC) :

$$(17) \qquad g(\mathbf{y}) = \bigwedge_{p=1}^{P} \left\{ \bigvee_{i \in I(p)} y_i \vee \bigvee_{j \in J(p)} \bar{y}_j \right\} .$$

On aboutit donc tout naturellement au théorème suivant :

(18) La fonction indicatrice, $\varphi$, de tout domaine compact $\mathbb{K}$ de $\mathbb{R}^M$ peut être approximée par un réseau PMC à quatre couches (deux couches cachées) de façon constructive.

*Démonstration*

En effet, un domaine compact $\mathbb{K}$ quelconque de $\mathbb{R}^M$ peut être approché par une réunion finie $\mathbb{P}$ de polyèdres convexes (et même de pavés plus simplement) avec une précision arbitraire $\varepsilon$. Il suffit pour cela d'extraire un sous-recouvrement fini $\mathbb{P}$ d'un recouvrement de $\mathbb{K}$ par des polyèdres convexes de diamètre $\varepsilon$. Or un moyen permettant de construire un polyèdre quelconque avec un PMC à trois couches a été décrit plus haut. En adjoignant simplement une couche supplémentaire réalisant l'opération de réunion, on obtient le résultat escompté. Ce résultat s'applique à tout fermé borné de $\mathbb{R}^M$.

*Exemple*

On se propose de montrer dans cet exemple comment représenter la fonction indicatrice d'un tétragone non convexe, représenté sur la figure 4. On construit les quatre hyperplans de la deuxième couche avec quatre cellules :

$$(19) \quad y_1 = f(3x_1 + x_2 - 2) ; \quad y_2 = f(x_1 + 3x_2 - 2) ;$$
$$y_3 = f(x_1 - x_2 + 1{,}5) ; \quad y_4 = f(x_2 - x_1 + 1{,}5) ;$$

puis on réalise l'opération

$$(20) \qquad g(\mathbf{x}) = (y_1 \times y_2) \wedge y_3 \wedge y_4 ,$$

soit sous FNC :

$$(21) \qquad g(\mathbf{x}) = (y_1 \vee y_2) \wedge (\bar{y}_1 \vee \bar{y}_2) \wedge y_3 \wedge y_4 .$$

Ceci conduit, d'après les relations (14) à (16) au réseau de la figure 5.

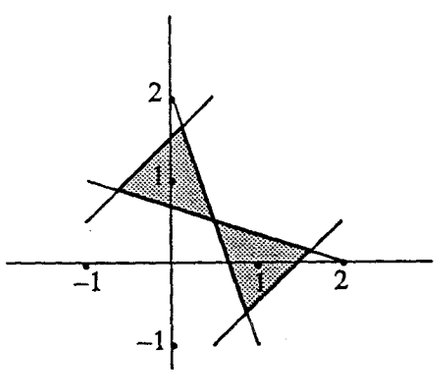


**Figure 4. — Un exemple de domaine dont on peut construire la fonction indicatrice avec un PMC à quatre couches.**

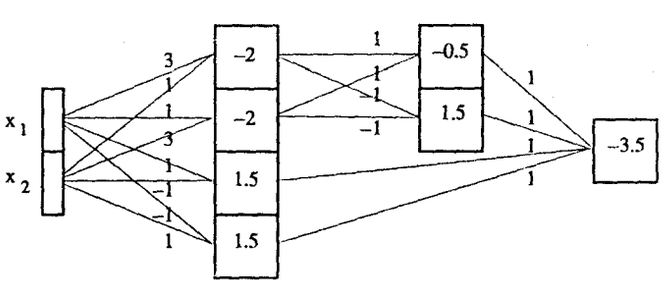


**Figure 5. — Réseau PMC à quatre couches construisant la fonction indicatrice du domaine de la figure 4. Si nécessaire, on peut rajouter deux cellules pass-tout dans la troisième couche pour se conformer aux contraintes de l'architecture standard.**

On cite parfois le « ou exclusif » (XOR) comme un exemple pathologique. En réalité, cette opération booléenne ne présente aucune difficulté particulière, et n'est qu'une application banale de la procédure décrite plus haut. Comme notre tétragone contient les points (0, 1) et (1, 0) du plan $\mathbb{R}^2$, à toutes entrées $a$ et $b$ de $\{0, 1\}$ la sortie délivrée sera $a \times b$. L'exemple ci-dessus est donc une façon de réaliser le XOR lorsque les entrées sont booléennes. Mais il y a bien entendu une infinité de façons de construire le ou exclusif avec cinq cellules actives (PMC à quatre couches) en construisant la fonction indicatrice d'un cône contenant les points (1, 0) et (0, 1) et excluant les points (0, 0) et (1, 1).

### 2.4. REMARQUES

Les résultats théoriques précédents nécessitent la connaissance de $\varphi$, qui n'est malheureusement pas donnée en pratique mais seulement connue en un nombre fini de points, constitués par les éléments de la base d'apprentissage. Or, il y a une infinité de fonctions continues à support borné passant par N points. Diverses questions se posent alors. Faut-il que $\varphi$ passe exactement par les N points pour maximiser le taux de reconnaissance des exemples futurs ? Le problème n'est-il pas mal posé puisqu'apparemment une infinité de solutions existent ? Faut-il chercher les fonctions les plus lisses possible, selon un certain critère [50] ? Quelles sont les performances de la solution la plus utilisée à l'heure actuelle pour l'apprentissage du PMC ? Les sections qui vont suivre vont aborder ces questions.

## 3. Apprentissage et critères d'optimisation

Considérons un PMC fixé. L'application qu'il va représenter est fonction d'un ensemble $\mathbb{W}$ de paramètres qu'il faut déterminer (poids et terme constant). Notons $\varphi(\mathbb{W}, \mathbf{x})$ l'application ainsi construite :

$$\mathbf{x} \in \mathbb{E} \subset \mathbb{R}^M \rightarrow \mathbf{y} = \varphi(\mathbb{W}, \mathbf{x}) \in \mathbb{F} \subset \mathbb{R}^K , \tag{22}$$

et $\Phi$ l'ensemble des fonctions $\varphi(\mathbb{W}, \bullet)$, obtenu lorsque $\mathbb{W}$ décrit toutes les valeurs possibles des paramètres.

### 3.1. CODAGE DES SORTIES

Lorsqu'il s'agit de traiter un problème de classification, les sorties délivrées doivent permettre d'attribuer le vecteur d'entrée à telle ou telle classe. Il y a donc une infinité de façons de coder les sorties. Les sorties peuvent notamment appartenir à un ensemble fini de valeurs, $\mathbb{F}$, dont le cardinal coïncide avec le nombre K de classes, par exemple

$$\mathbb{F} = \{\mathbf{e}_1, \mathbf{e}_2, ..., \mathbf{e}_K\} , \tag{24}$$

où $\mathbf{e}_k$ désigne la $k$-ième colonne de la matrice identité de dimension K. Un codage également très utilisé lorsque le PMC est destiné à des fins de classification est celui obtenu à partir de (24) en remplaçant les $\{0\}$ par des $\{-1\}$ [57], [25], [35], [36]. Pour le codage (24), on définit la correspondance

$$\mathbf{x}^{(n)} \in \omega_j \Leftrightarrow \mathbf{y}_k^{(n)} = \delta_{jk} . \tag{25}$$

Lorsque les sorties sont les probabilités $p(\mathbf{x}, \omega_k)$, L'ensemble $\mathbb{F}$ est de cardinal infini et

$$\mathbb{F} = [0, 1]^K . \tag{26}$$

Contrairement à ce que l'on pourrait croire au premier abord, le choix de ce codage est absolument crucial pour certains critères d'optimisation, et en particulier pour le critère MEQ qui va être analysé ci-après. Les théorèmes de la section 3.5 l'illustrent bien.

### 3.2. MINIMISATION DE L'ERREUR QUADRATIQUE (MEQ)

L'objectif est de chercher l'ensemble des poids, $\mathbb{W}$, de façon à minimiser l'écart entre les sorties désirées, $\mathbf{y}^{(n)}$, et les sorties obtenues, $\varphi(\mathbb{W}, \mathbf{x}^{(n)})$, lorsque $\mathbf{x}^{(n)}$ décrit la base d'apprentissage :

$$\mathbb{W} = \text{Arg} \min_{\mathbb{W}} \sum_{n=1}^{N} \left\| \mathbf{y}^{(n)} - \varphi(\mathbb{W}, \mathbf{x}^{(n)}) \right\|^2 . \tag{27}$$

Ce critère présente un inconvénient majeur. En effet, on voudrait en réalité minimiser $\left\| \mathbf{y}^{(n)} - \varphi(\mathbb{W}, \mathbf{x}^{(n)}) \right\|$ pour des observations $\mathbf{y}^{(n)}$ qui n'appartiennent pas à la base d'apprentissage, mais à une « base de généralisation » **inconnue,** ce qui apparaît plus clairement dans une approche statistique que dans (27).

De façon générale, l'apprentissage nécessite l'existence d'un lien entre la base de généralisation et la base d'apprentissage, par exemple de nature statistique [11]. A l'instar de [25], on supposera que les exemples des deux bases sont des réalisations tirées aléatoirement conformément à une même densité de probabilité, $p(\mathbf{x}/\omega_k)$, sous chacune des classes, et que les classes ont la même probabilité d'occurrence, $P_k$, dans chacune des bases.

Le risque de surapprentissage (souvent dénommé « overfitting » en franglais) est présent dans tous les problèmes d'apprentissage supervisé, et notamment dans (27) de façon particulièrement cachée. En effet, si la base d'apprentissage A(N) est de faible taille comparée au nombre de paramètres libres dans $\mathbb{W}$, l'erreur peut atteindre des valeurs très faibles, voire nulles. Pourtant, absolument rien n'assure que les probabilités d'erreurs soient minimisées ; au contraire, le surapprentissage a bien souvent pour conséquence l'augmentation des erreurs de classification : expérimentalement, on vérifie que la probabilité d'erreur admet un minimum pour une certaine valeur de l'erreur quadratique. Ceci a été analysé dans le cas linéaire (Perceptron à deux couches) [52], et expérimentalement dans le cas non-linéaire par de nombreux auteurs (PMC à plus de 2 couches, et réseau ALN). Le surapprentissage est d'autant plus sensible que la taille du réseau est grande (beaucoup de paramètres en comparaison à la taille de la base), si le nombre de recyclages est élevé.

### 3.3. RISQUE BAYESIEN

Dans le contexte bayesien, on minimise la fonction risque :

$$(28) \qquad R = \sum_{i,j=1}^{K} \kappa(i,j)\, P_j \int_{u \in \omega_i} p(\mathbf{u}/\omega_j)\, du ,$$

où $p(\mathbf{u}/\omega_j)$ = probabilité qu'une observation issue de la classe $\omega_j$ prenne la valeur $\mathbf{u}$,
$P_j$ = probabilité a priori qu'une observation soit issue de la classe $\omega_j$,
$\kappa(i,j)$ = coût de la classification dans la classe $\omega_i$ d'un élément de la classe $\omega_j$.

Il vient avec ces notations la relation donnant la densité de probabilité marginale de l'observation :

$$(29) \qquad p(\mathbf{u}) = \sum_{k=1}^{K} P_k\, p(\mathbf{u}/\omega_k) .$$

Rappelons ce que donne ce critère dans quelques cas particuliers. Si les réponses correctes ne sont pas pénalisées, i.e. si $\kappa(i,i) = 0$, la solution bayesienne est obtenue en affectant à $\mathbf{x}$ la classe $\omega_i$ pour laquelle la quantité suivante est la plus faible :

$$(30) \qquad B_i(\mathbf{x}) = \sum_{i \neq j} \kappa(i,j)\, P_j\, p(\mathbf{x}/\omega_j) .$$

Ceci découle directement de (29). Un cas particulier d'importance est celui des coûts uniformes : $\kappa(i,j) = 1 - \delta_{ij}$. Dans ce cas la relation (29) permet d'écrire (30) sous une autre forme :

$$B_i(\mathbf{x}) = p(\mathbf{x}) - P_i\, p(\mathbf{x}/\omega_i) .$$

Autrement dit, puisque $p(\mathbf{x})$ ne dépend pas de $\omega_i$, on peut de manière équivalente affecter $\mathbf{x}$ à la classe pour laquelle la quantité ci-dessous est la plus grande :

$$(31) \qquad b_i(\mathbf{x}) = P_i\, p(\mathbf{x}/\omega_i) .$$

C'est sous cette dernière forme que le critère bayesien est le plus souvent utilisé. Notons au passage que $b_i(\mathbf{x})$ n'est autre que la probabilité conjointe $p(\mathbf{x}, \omega_i)$, probabilité d'observer un vecteur d'entrée $\mathbf{x}$ issu de la classe $\omega_i$.

Pour définir un classifieur bayesien, il est donc nécessaire d'estimer les densités conditionnelles $p(\mathbf{x}, \omega_i)$, qui sont en général inconnues. L'objet de la section suivante est de proposer des estimateurs dont les performances sont remarquables pour des échantillons de taille limitée. Cette section a également pour fonction d'introduire les notations nécessaires à l'étude asymptotique des performances du PMC.

### 3.4. ESTIMATEURS À NOYAU DE PROBABILITÉ

Les performances des estimateurs nucléaires sur des échantillons courts sont remarquables lorsque la densité cherchée est $C^\infty$. Les propriétés fondamentales de ces estimateurs sont démontrées dans [7], tandis que [23] en fait un tour d'horizon synthétique.

Soit $\{\mathbf{x}(n),\ 1 \leq n \leq N,\ \mathbf{x}(n) \in \mathbb{R}^M\}$ une base d'apprentissage de taille N, que l'on suppose être la réalisation d'une variable aléatoire unique, de densité $p_x(\mathbf{u})$. Alors on définit l'estimateur à noyau radial, $\hat{p}_x(\mathbf{u})$ comme suit.

$$(32) \qquad \hat{p}_x(N, \mathbf{u}) = \frac{1}{N} \sum_{n=1}^{N} \frac{1}{h(N)^M} K\left( \frac{\mathbf{u} - \mathbf{x}(n)}{h(N)} \right) ,$$

où K est une fonction de $\mathbb{R}^M$ dans $\mathbb{R}$ ne dépendant que de la norme de son argument (fonction radiale), et $h(N)$ une suite réelle positive tendant vers zéro quand $N \to \infty$. Cacoullos [7] énonce un certain nombre de conditions suffisantes sur $h(N)$ pour que l'estimateur soit performant. Nous avons en particulier le théorème de convergence suivant :

Si $K(\mathbf{u})$ et $p_x(\mathbf{u})$ vérifient les conditions

$$(33) \qquad \int_{\mathbb{R}^M} K(\mathbf{u})\, d\mathbf{u} = 1 ,$$

$$(34) \qquad \int_{\mathbb{R}^M} K(\mathbf{u})\, u_i\, u_j\, d\mathbf{u} = v_{ij} \text{ borné} , \quad \forall i, j,\ 1 \leq i, j \leq M ,$$

(35) $p_x(\mathbf{x})$ est deux fois continûment différentiable ,

et si $h(N)$ vérifie

$$(36) \qquad \lim_{N \to \infty} h(N) = 0 \quad \text{et} \quad \lim_{N \to \infty} N h(N)^M = \infty ,$$

alors l'estimateur (32) converge en moyenne quadratique ; autrement dit, l'erreur

$$(37) \qquad e(N) = E\left\{ \int_{\mathbb{R}^M} [\hat{p}_x(N, \mathbf{u}) - p_x(\mathbf{u})]^2\, d\mathbf{u} \right\}$$

tend vers zéro quand $N \to \infty$. On supposera dorénavant que les conditions (33) à (36) sont satisfaites.

### 3.5. LIMITATIONS INTRINSÈQUES DU CRITÈRE MEQ

Le but de cette section est de vérifier mathématiquement l'assertion :

(39) *« Les sorties du PMC tendent vers les probabilités bayesiennes lorsque la taille de la base d'apprentissage tend vers l'infini. »*

Ici encore, on analyse les limites « ultimes » du critère MEQ, c'est-à-dire dans le cas le plus favorable où l'algorithme de minimisation MEQ atteint le minimum absolu de l'erreur. Les quatre théorèmes suivants (dont trois sont originaux) ont été démontrés en annexe.

**Théorème I**

Soit $A_0(N) = \{\mathbf{x}^{(n)}, \mathbf{y}^{(n)},\ 1 \leq n \leq N\}$ une base d'apprentissage, contenant $N_k$ exemples dans chacune des classes $\omega_k$, $1 \leq k \leq K$. Soit l'application $\varphi(\mathbb{W}, .)$ obtenue en

minimisant l'erreur

$$(40) \qquad \varepsilon(N) = \frac{1}{N} \sum_{n=1}^{N} \left\| \mathbf{y}^{(n)} - \varphi(\mathbb{W}, \mathbf{x}^{(n)}) \right\|^2$$

par rapport à $\mathbb{W}$. Alors cette application tend lorsque $\{N_k \to \infty,\ 1 \leq k \leq K\}$ vers la meilleure approximation quadratique dans la classe $\Phi$ du récepteur optimal bayesien équipénalisé, à condition de choisir le codage (25), i.e. la fonction indicatrice de classe.

**Théorème II**

Soit $A_0(N) = \{\mathbf{x}^{(n)}, \mathbf{y}^{(n)}, 1 \leq n \leq N\}$ une base d'apprentissage de taille finie N, contenant $N_k$ exemples dans chacune des classes $\omega_k$, $1 \leq k \leq K$. Alors la solution $\varphi(\mathbb{W}, .)$ obtenue en minimisant l'erreur

$$(41) \qquad \varepsilon(N) = \frac{1}{N} \sum_{n=1}^{N} \left\| \mathbf{y}^{(n)} - \varphi(\mathbb{W}, \mathbf{x}^{(n)}) \right\|^2$$

tend vers la meilleure approximation dans $\Phi$ de la solution bayesienne générale (30) lorsque $\{N_k \to \infty,\ 1 \leq k \leq K\}$, à condition de choisir pour codage

$$(42) \qquad y_i^{(n)} = \kappa(i, j) \quad \text{pour} \quad \mathbf{x}^{(n)} \in \omega_j .$$

Lorsque N est fini, une méthode couramment pratiquée pour étendre artificiellement la taille de la base $A_0(N) = \{\mathbf{x}^{(n)}, \mathbf{y}^{(n)}, 1 \leq n \leq N\}$ consiste à la dupliquer en la bruitant à l'aide de NR réalisations indépendantes d'un bruit centré de densité $p_z(\mathbf{u})$. On obtient ainsi R bases d'apprentissage de taille N :

$$(43) \quad A_r(N) = \{\mathbf{x}^{(n)} + \mathbf{z}^{(n,r)}, \mathbf{y}^{(n)}, 1 \leq n \leq N\}\ ,\ 1 \leq r \leq R .$$

Posons alors

$$(44) \qquad A(N, R) = \bigcup_{r=1}^{R} A_r(N) .$$

**Théorème III**

Soit $A_0(N) = \{\mathbf{x}^{(n)}, \mathbf{y}^{(n)}, 1 \leq n \leq N\}$ une base d'apprentissage de taille finie contenant $N_k$ exemples ($N_k > 0$) dans chacune des classes, et $A(N, R)$ sa version étendue conformément à (44). On suppose en outre que le codage adopté est celui de (25). Alors lorsque R tend vers l'infini, la solution $\varphi(\mathbb{W}, .)$ obtenue en minimisant l'erreur euclidienne

$$(45) \quad \varepsilon(N, R) = \frac{1}{N}\frac{1}{Q} \sum_{n=1}^{N} \sum_{r=1}^{R} \left\| \mathbf{y}^{(n)} - \varphi(\mathbb{W}, \mathbf{x}^{(n)} + \mathbf{z}^{(n,r)}) \right\|^2$$

tend vers la meilleure approximation dans $\Phi$ de la solution bayesienne équipénalisée (31) construite à partir de l'estimateur nucléaire (32) de noyau $p_z(\mathbf{u})$.

**Théorème IV**

Dans les mêmes conditions que le théorème ci-dessus, et lorsque les sorties $\mathbf{y}^{(n)}$ sont définies par le codage (42), la solution obtenue en minimisant l'erreur $\varepsilon(N, R)$ tend vers la solution bayesienne générale (30) où les densités sont remplacées par leurs estimations nucléaires de noyau $p_z(\mathbf{u})$.

Les démonstrations, données en annexe, exhibent des conditions portant sur le bruit **z** permettant à la solution d'être consistante. En effet, on y constate que les densités conditionnelles $p(\mathbf{v}/\omega_k)$ tendent vers leurs estimations nucléaires (32) de noyau

$$(46) \qquad p_z(\mathbf{u}) = K(\mathbf{u}/h)/h^M .$$

Par conséquent, grâce au théorème de Cacoullos, on constante qu'il convient de choisir un bruit additif **z** tel que sa densité soit définie par (46) où $h$ vérifie (38) :

$$h(N) \sim N^{-\frac{1}{M+4}} .$$

On conclut de ces quatre théorèmes que, lorsqu'on créera les bases $A_r(N)$, on pourra notamment choisir un bruit gaussien isotrope d'écart-type $\sigma = \alpha N^{-1/M+4}$, où $\alpha$ est une constante fixée. On conclut également que l'assertion (39) n'est vraie que si une infinité d'exemples différents est traitée, et que si le minimum absolu de l'erreur quadratique est atteint. Ces deux conditions ne sont évidemment pas satisfaites en pratique, de sorte que l'on peut s'attendre à ce que les sorties du PMC ne soient pas égales aux densités bayesiennes estimées. Notons que notre théorème I a été indépendamment obtenu par d'autres auteurs dans [55] ou [4d], tandis que nos théorèmes II à IV le généralisent.

# 4. L'algorithme de Rétro-Propagation du Gradient (RPG)

## 4.1. DÉFINITION DE L'ALGORITHME

L'algorithme RPG a été découvert indépendamment par plusieurs auteurs, le premier remontant semble-t-il à 1973. Par ailleurs, il existe [5] de très fortes similarités entre l'algorithme de RPG et l'algorithme de Viterbi, utilisé pour l'identification des modèles de Markov cachés [51] ; de plus, cet algorithme a été découvert quelques années plus tôt (vers 1966). La présentation faite ci-après de l'algorithme de RPG est similaire à celle de [34], et est à notre avis la plus rationnelle ; la présentation de l'algorithme RPG de Rumelhart [57] est également à recommander.

Considérons un réseau PMC à Q couches, $Q \geq 2$, et une base d'apprentissage $A_0(N) = \{\mathbf{x}^{(n)}, \mathbf{y}^{(n)}, 1 \leq n \leq N\}$. Notons $m(q)$ le nombre de cellules dans la couche $q$, et $\mathbf{s}^{(n)}(q)$ le vecteur de dimension $m(q)$ délivré à la sortie de la couche $q$. Avec ces notations, $m(Q) = K$, $m(1) = M$, et $\mathbf{x}^{(n)} = \mathbf{s}^{(n)}(1)$. Conformément à (2), la structure du réseau est telle que

$$(50) \qquad \mathbf{s}^{(n)}(q+1) = f(\mathbf{W}(q)\, \mathbf{s}^{(n)}(q)) ,$$

où on a supposé que toutes les couches avaient la même

fonction de transition pour simplifier (ce n'est nullement nécessaire sur le plan théorique), ici supposée dérivable. Dans (50), la matrice $W(q)$ est de dimension $m(q+1) \times m(q)$. L'apprentissage du PMC peut être considéré comme un problème d'optimisation sous contraintes. Supposons que l'on veuille minimiser la distance euclidienne entre les sorties désirées et les sorties réellement obtenues :

$$\sum_{n=1}^{N} \|\mathbf{s}^{(n)}(Q) - \mathbf{y}^{(n)}\|^2 .$$

Certaines contraintes sont imposées par l'architecture du réseau et définies par (50) de sorte que la fonction objectif peut être écrite sous la forme variationnelle suivante pour $1 \leqslant q < Q$, $1 \leqslant n < N$, $1 \leqslant j \leqslant m\ (q+1)$ :

(51)

$$L\left\{W(q), \mathbf{s}^{(n)}(q+1), \lambda_j(n, q)\right\} = \sum_{n=1}^{N} \|\mathbf{s}^{(n)}(Q) - \mathbf{y}^{(n)}\|^2 +$$

$$+ \sum_{n=1}^{N} \sum_{q=1}^{Q-1} \sum_{j=1}^{m(q+1)} \lambda_j(n, q) \times$$

$$\times \left[ s_j^{(n)}(q+1) - f\left( \sum_{i=1}^{m(q)} w_{ji}(q)\, s_i^{(n)}(q) \right) \right]$$

où les vecteurs $\lambda(n, q)$ sont de dimension $m(q+1)$ et désignent les multiplicateurs de Lagrange associés aux contraintes (50). Une condition nécessaire pour qu'un point $\mathbb{W}$ soit minimum local de L est qu'il soit stationnaire selon toutes ses composantes (toutes celles figurant entre accolades dans (51)), ce qui donne en plus de (50) deux familles d'équations supplémentaires :

(52)

$$\forall q, r, j\ ,\ 1 \leqslant q < Q\ ,\ 1 \leqslant r \leqslant m(q)\ ,\ 1 \leqslant j \leqslant m(q+1)$$

$$\partial L / \partial w_{jr}(q) =$$

$$- \sum_{n=1}^{N} \lambda_j(n, q)\, s_r^{(n)}(q)\, f'\left( \sum_{i=1}^{m(q)} w_{ji}(q)\, s_i^{(n)}(q) \right) .$$

Soit, en posant

$$(53) \quad z_j^{(n)}(q) = \lambda_j(n, q)\, f'\left( \sum_{i=1}^{m(q)} w_{ji}(q)\, s_i^{(n)}(q) \right) ,$$

on obtient le gradient

$$(54) \quad \partial L / \partial w_{jr}(q) = - \sum_{n=1}^{N} z_j^{(n)}(q)\, s_r^{(n)}(q) ,$$

$$\forall j, q, r \,/\, 1 \leqslant q < Q\ ,\ 1 \leqslant j \leqslant m(q+1)\ ,\ 1 \leqslant r \leqslant m(q) .$$

Par ailleurs l'annulation des dérivées partielles de L par rapport aux $s_r^{(n)}(q)$, $\forall n, j$, $1 \leqslant n < N$, $1 \leqslant r \leqslant m(q)$, fournit pour $q = Q$ :

$$(55) \quad \lambda(n, Q-1) = -2(\mathbf{s}^{(n)}(Q) - \mathbf{y}^{(n)}) ,$$

et pour $1 < q < Q$ :

$$\lambda_r(n, q-1) =$$

$$= \sum_{j=1}^{m(q+1)} \lambda_j(n, q)\, w_{jr}(q)\, f'\left( \sum_{i=1}^{m(q)} w_{ji}(q)\, s_i^{(n)}(q) \right) ,$$

qui s'écrit d'après (53)

$$(56) \quad \lambda(n, q-1) = W(q)^t\, \mathbf{z}^{(n)}(q)\ ,\ 1 \leqslant r \leqslant m(q) .$$

La première opération de l'algorithme RPG est la « propagation », et consiste à passer les exemples dans le sens direct (entrée vers sortie) afin d'obtenir tous les $\mathbf{s}^{(n)}(q)$. Ensuite, il s'agit de modifier les poids $W(q)$ de façon à diminuer la fonction objectif (51) ; la difficulté vient du fait que l'erreur n'est connue que dans la couche de sortie Q et non dans les couches cachées, $1 < q < Q$. On utilise à cette fin un algorithme du gradient :

$$W(q) \leftarrow W(q) - \mu\, \partial L / \partial W(q) ,$$

qui s'écrit d'après (54) :

$$(57) \quad W(q) \leftarrow W(q) + \mu \sum_{n=1}^{N} \mathbf{z}^{(n)}(q)\, \mathbf{s}^{(n)}(q)^t ,$$

à la fin de chaque cycle. L'équation (53) exprime $\mathbf{z}(q)$ en fonction de $\lambda(q)$, tandis que (56) exprime $\lambda(q-1)$ à partir de $\mathbf{z}(q)$. Ces équations permettent donc obtenir par récurrence descendante les quantités $\lambda(q)$ et $\mathbf{z}(q)$ pour $1 \leqslant q < Q$, la première valeur, $\lambda(Q-1)$, étant donnée par (55). La « rétro-propagation » de ces grandeurs auxiliaires permet de modifier les poids conformément à (57) puisque toutes les sorties $\mathbf{s}^{(n)}(q)$ ont été calculées pendant la propagation directe.

En pratique, les exemples $[\mathbf{x}(n), \mathbf{y}(n)]$ sont présentés au réseau séquentiellement, de sorte qu'il est aussi possible de remettre à jour les matrices $W(q)$ à chaque fois que $n$ est incrémenté, sans attendre la fin du cycle. Dans ce cas, la remise à jour (57) prend la forme (que l'on peut qualifier de gradient stochastique) suivante :

$$W(q) \leftarrow W(q) + \mu \mathbf{z}^{(n)}(q)\, \mathbf{s}^{(n)}(q)^t ,$$

pour chaque exemple de chaque cycle. On augmentera considérablement la vitesse de convergence de l'algorithme en ajustant la valeur du pas $\mu$ (appelé parfois taux d'apprentissage) à sa valeur optimale. Pour ce faire, comme la fonctionnelle à minimiser n'est pas fonction explicite de $\mu$, on peut utiliser des méthodes de réduction d'intervalles ou d'interpolation polynomiales [20]. Une autre façon encore plus performante mais plus coûteuse est d'utiliser des algorithmes de type quasi-Newton [20]. Un excellent article en présente les avantages (rapidité) et inconvénients (coûts) dans le cadre du PMC [65]. Ces méthodes sont parfois appelées méthodes du second ordre [47]. Pour information, l'application de ce type d'algorithme d'apprentissage a été récemment rapportée dans [63], [28].

### 4.2. VITESSE DE CONVERGENCE DES POIDS POUR UN RÉSEAU LINÉAIRE

L'étude de la convergence de l'algorithme de RPG présente des difficultés insurmontables s'il est exécuté sur un PMC de forme générale. En effet, on ne connaît pas la limite vers laquelle tend l'erreur quadratique, entre autres raisons parce que ce critère admet a priori plusieurs minima locaux. L'étude de la convergence d'une suite sans en connaître la limite ne peut être qu'incomplète. Par exemple, dans [60] l'étude ne concerne que la variation du critère avec le nombre d'itérations, lorsqu'on se trouve dans un voisinage de la limite (inconnue). Or sur le plan pratique, c'est plutôt le temps nécessaire pour parvenir dans ce voisinage qui est intéressant.

C'est pourquoi notre analyse a été effectuée dans le cas linéaire (2 couches avec $f(x) = x$) [30a], qui présente l'avantage d'être analytiquement tractable et d'avoir un critère MEQ toujours convexe, mais qui est aussi sans aucun doute plus favorable que le cas multicouche. En d'autres termes, le temps d'apprentissage d'une fonction linéaire avec un PMC sans couche cachée est vraisemblablement plus court que l'apprentissage de fonctions complexes sur un PMC de forme générale. On étudie donc bien une borne inférieure au temps d'apprentissage, assez éloignée de la borne atteinte par l'algorithme, donc optimiste. Mais insistons bien sur le fait que dans cette section le gradient n'est pas rétropropagé.

Une pratique usuelle de l'algorithme de RPG sur des bases d'apprentissage finies consiste à recycler la base plusieurs fois intégralement. Dans des cas plus raffinés, la base est bruitée lors de chaque recyclage (cf. section 3.5). Au cours du $(r+1)$-ième recyclage, le gradient s'écrit, si le pas $\mu$ est maintenu constant au sein de chaque recyclage :

$$(58)\quad W^{(rN+n+1)} = W^{(rN+n)} - \mu(r)(W^{(rN+n)}\mathbf{x}^{(n)} - \mathbf{y}^{(n)})\,\mathbf{x}^{(n)t},$$

pour $r \geq 0$ et $1 \leq n \leq N$. Une analyse très récente [38] utilise implicitement l'hypothèse (très forte) que les vecteurs $\mathbf{x}^{(n)}$ sont décorrélés, ce qui confère au résultat un caractère excessivement optimiste à notre avis. On ne fera pas cette hypothèse ici, de sorte que nos conclusions seront moins flatteuses.

Pour alléger, utilisons temporairement la notation $\mu$ au lieu de $\mu(r)$, tant que $r$ est fixé. Après présentation des deux premiers vecteurs de la base $A_0(N)$ dans le premier cycle, la matrice des poids prend la forme

$$(59)\quad W^{(2)} = W^{(0)}[I - \mu(\mathbf{x}^{(1)}\mathbf{x}^{(1)t} + \mathbf{x}^{(2)}\mathbf{x}^{(2)t})] + \mu[\mathbf{y}^{(1)}\mathbf{x}^{(1)t} + \mathbf{y}^{(2)}\mathbf{x}^{(2)t}] + \mu^2 g^{(2)},$$

où $g$ s'écrit :

$$(60)\quad g^{(2)} = W^{(0)}\mathbf{x}^{(1)}\mathbf{x}^{(1)t}\mathbf{x}^{(2)}\mathbf{x}^{(2)t} - \mathbf{y}^{(1)}\mathbf{x}^{(1)t}\mathbf{x}^{(2)}\mathbf{x}^{(2)t}.$$

Si nous poursuivons le calcul jusqu'à la fin du cycle, nous obtenons

$$(61)\quad W^{(N)} = W^{(0)}\left[I - \mu\sum_{n=1}^{N}\mathbf{x}^{(n)}\mathbf{x}^{(n)t}\right] + \mu\sum_{n=1}^{N}\mathbf{y}^{(n)}\mathbf{x}^{(n)t} + \mu^2 g^{(N,0)}(\mu).$$

où le terme $g^{(N)}(\mu)$ est un polynôme de degré $N-2$ en $\mu$, dont on peut exprimer le premier terme :

$$(62)\quad g^{(N,0)}(\mu) = W^{(0)}\sum_{1\leq i<j\leq N}[\mathbf{x}^{(i)}\mathbf{x}^{(i)t}\mathbf{x}^{(j)}\mathbf{x}^{(j)t} - \mathbf{y}^{(i)}\mathbf{y}^{(i)t}\mathbf{x}^{(j)}\mathbf{x}^{(j)t}] + O(\mu).$$

A N fixé, le polynôme $g^{(N,r)}(\mu)$ peut être simplement considéré comme une grandeur bornée dans (61) puisque $0 < \mu < 1$. Si $\mu$ est suffisamment petit, la relation (61) décrit le cycle avec une bonne précision. Posons maintenant

$$X = [\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(N)}],$$
$$Y = [\mathbf{y}^{(1)}, \mathbf{y}^{(2)}, \ldots, \mathbf{y}^{(N)}],$$

et la SVD de X : $X = V\Sigma U^t$.

Les matrices V, $\Sigma$ et U sont respectivement de dimension $M \times M$, $M \times N$, et $N \times N$, tandis que X et Y sont respectivement de dimensions $M \times N$ et $K \times N$. Étudions la suite

$$(63)\quad Z(r) = (W^{(Nr)} - YX^-)\,V,$$

où $X^-$ désigne la pseudo-inverse de X, satisfaisant $X^- = U\Sigma^- V^t$, où $\Sigma$ est diagonale et où U et V sont orthogonales. Cette suite est régie par la récurrence

$$(64)\quad Z(r+1) = Z(r)[I - \mu(r)\,\Sigma\Sigma^t] + \mu(r)^2 g^{(N,r)}(\mu(r))\,V.$$

(65) **Lemme.**

Si $\sum_{r=1}^{\infty}\mu(r) = \infty$, la suite $\tilde{Z}(r)$ définie ci-dessous converge vers zéro :

$$(66)\quad \tilde{Z}(r+1) = \tilde{Z}(r)[I - \mu(r)\,\Sigma\Sigma^t].$$

(67) **Lemme.**

Si $\sum_{r=1}^{\infty}\mu(r) = \infty$, et $\sum_{r=1}^{\infty}\mu(r)^2 < \infty$, alors la suite définie par (64) converge vers zéro.

(68) **Théorème V.**

Soient $\lambda$ la plus petite valeur propre non nulle de la matrice $XX^t$, et $\mu(r)$ une suite satisfaisant les hypothèses du lemme (67), e.g., $\mu(r) = 1/r$. Alors l'exécution de $O(\varepsilon^{-1/\sqrt{\lambda}})$ cycles garantit la convergence de la suite des poids à $\varepsilon$ près.

Les démonstrations sont renvoyées en annexe. Le théorème V montre que si $\lambda$ est petite, la convergence de l'algorithme RPG peut être arbitrairement lente. L'apprentissage du PMC par RPG est donc de complexité non polynomiale, bien que les conditions dans lesquelles elle ait été abordée aient été favorables (optimistes). Enfin lorsque la base $A_0(N)$ est recyclée avec un bruitage additif, un résultat analogue au théorème V pourrait être obtenu, pourvu que N soit suffisamment grand pour que

$XX^t$ puisse être remplacée par une matrice de covariance empirique du type $(XX^t + \sigma^2 I)$, ce qui aurait aussi le mérite d'assurer une valeur minimale à $\lambda$. Si par contre N est faible, l'étude devient sans doute plus compliquée.

L'étude de la convergence de ce type d'algorithme a déjà été abordée depuis longtemps dans le cadre du filtrage adaptatif, et des résultats plus généraux ont été obtenus [4b]. Toutefois, les développements mathématiques correspondants sont beaucoup plus complexes (voir par exemple [4b] et les références qui y sont citées).

# 5. Discrimination bayesienne directe

Quelques propriétés des réseaux de neurones en couches ont été passées en revue dans les sections précédentes. On a notamment vu pourquoi il est raisonnable (mais optimiste) de croire que ces réseaux minimiseront la probabilité totale d'erreurs de classification. Montrons à présent comment la solution bayesienne, censée minimiser cette probabilité d'erreur par définition, peut elle aussi être implantée (mais de façon approchée il est vrai) sur un réseau de neurones en couches.

## 5.1. CHOIX OPTIMAL DE LA SUITE $h$(N)

Dans cette section, on propose une façon de choisir la suite $h(N)$, basée sur la minimisation de l'erreur quadratique (37). Afin d'écrire explicitement cette erreur, il faut reprendre en détail les calculs de Cacoullos, et en particulier calculer le biais et la variance de l'estimateur (32). Sous les hypothèses énoncées dans la section 3.4, et lorsque N tend vers l'infini (et donc lorsque $h$ tend vers zéro), les résultats asymptotiques suivants peuvent être obtenus

$$\varepsilon(N, \mathbf{x}) = \frac{h^2}{2} \sum_{ij=1}^{M} v_{ij} \frac{\partial^2 p_x(\mathbf{x})}{\partial x_i \, \partial x_j} + O(h^3) \tag{70}$$

pour le biais, et

$$\text{var}\,(N, \mathbf{x}) = \frac{1}{N} \frac{1}{h^M} p_x(\mathbf{x}) \int_{\mathbb{R}^M} K^2(\mathbf{z})\, d\mathbf{z} + O\left( \frac{1}{N} \frac{1}{h^{M-1}} \right) \tag{71}$$

pour la variance. Les démonstrations sont données en annexe. Or l'erreur quadratique, puisqu'elle vérifie

$$e(N) = \int_{\mathbb{R}^M} [\text{var}\,(N, \mathbf{u}) + \varepsilon(N, \mathbf{u})^2]\, d\mathbf{u} .$$

s'écrit par conséquent

$$e(N) = \frac{h(N)^4}{4} \int_{\mathbb{R}^M} J(\mathbf{x})^2\, d\mathbf{x} + \frac{1}{N} \frac{1}{h^M} \int_{\mathbb{R}^M} K^2(\mathbf{z})\, d\mathbf{z} + O(h^5) + O\left( \frac{1}{N} \frac{1}{h^{M-1}} \right) , \tag{72}$$

en posant

$$J(\mathbf{x}) = \sum_{ij=1}^{M} v_{ij} \frac{\partial^2 p_x(\mathbf{x})}{\partial x_i \, \partial x_j} , \tag{73}$$

et où $v_{ij}$ est définie par (34). On montre alors en annexe que l'erreur minimale est atteinte pour

$$h(N)^{M+4} = \frac{M}{N} \frac{\int_{\mathbb{R}^M} K^2(\mathbf{z})\, d\mathbf{z}}{\int_{\mathbb{R}^M} J^2(\mathbf{x})\, d\mathbf{x}} , \tag{74}$$

et vaut

$$e(N)_{\text{mini}} = \left( \frac{M}{4} + 1 \right) \frac{1}{N} \frac{1}{h^M} \int_{\mathbb{R}^M} K^2(\mathbf{z})\, d\mathbf{z} . \tag{75}$$

La relation (74) donne la valeur de $h(N)$ à choisir, à condition de connaître $J(\mathbf{x})$ et $K(\mathbf{x})$. La seconde constatation que l'on peut faire est que l'ordre de grandeur de $e(N)$ n'est pas affecté par la forme exacte de $K(\mathbf{x})$ pourvu que les conditions de la section 3.4 soient respectées. Choisissons (arbitrairement) le noyau gaussien standardisé :

$$K(\mathbf{v}) = (2\pi)^{-M/2} \exp(-\|\mathbf{v}\|^2/2) . \tag{76}$$

L'estimateur (32) est alors un mélange gaussien, qui permet de bien représenter la plupart des densités observées dans le monde réel [22]. La relation (74) devient dans ce cas

$$h(N)_{\text{opt}}^{M+4} = \frac{M}{N} \frac{(2\sqrt{\pi})^{-M}}{\int_{\mathbb{R}^M} \Delta p(\mathbf{x})^2\, d\mathbf{x}} , \tag{77}$$

où $\Delta p$ désigne le laplacien de $p$.

## 5.2. APPRENTISSAGE PRATIQUE D'UNE DENSITÉ DE PROBABILITÉ

L'estimateur (32) est peu pratique si la base d'apprentissage est de grande taille. Si l'on veut que la procédure proposée soit compétitive face au PMC, il est nécessaire d'imposer une limite aux exigences mémoire de notre algorithme d'apprentissage. Or avec (32), il est nécessaire de mémoriser toute la base d'apprentissage, comme c'est le cas dans [58] par exemple. Afin de rendre une telle solution réalisable, on peut effectuer une compression de données, par agglomération automatique de l'ensemble $\{\mathbf{x}(n),\ 1 \leq n \leq N\}$ en P groupes, $E(p)$, de centroïde $\mathbf{w}(p)$ et contenant chacun $a(p)$ éléments. De plus, si la vraie densité cherchée, $p_x(\mathbf{u})$ est effectivement un mélange gaussien, elle sera bien approximée par un estimateur du type de (32) avec le noyau gaussien (76) pourvu que le nombre de modes de la densité $p_x(\mathbf{u})$ soit inférieur ou égal au nombre maximal de modes de l'estimateur. Supposons que les contraintes matérielles nous imposent un nombre P

de gaussiennes. L'estimateur (32) peut s'écrire

$$\hat{p}_x(\mathrm{N}, \mathbf{u}) = \\ = \frac{1}{\Sigma a(p)} \sum_{p=1}^{\mathrm{P}} \sum_{x(n) \in \mathrm{E}(p)} \frac{1}{h(a(p))^{\mathrm{M}}} \mathrm{K}\left( \frac{\mathbf{u} - \mathbf{x}(n)}{h(a(p))} \right),$$

qu'il est possible d'approximer par l'estimateur à P modes :

$$(78) \quad \hat{p}(\mathbf{u}) = \frac{1}{\Sigma a(p)} \sum_{p=1}^{\mathrm{P}} \frac{a(p)}{h(a(p))^{\mathrm{M}}} \mathrm{K}\left( \frac{\mathbf{u} - \mathbf{w}(p)}{h(a(p))} \right).$$

en assimilant les échantillons $\mathbf{x}(n)$ de chacun des groupes $\mathrm{E}(p)$ à leur centroïde $\mathbf{w}(p)$. Au sein de chaque groupe $\mathrm{E}(p)$, on peut calculer la valeur de $h(a(p))$ en appliquant la formule (77). En effet, si $\mathbf{x}(n) \in \mathrm{E}(p)$, alors $p_x(\mathbf{u})$ est gaussienne de moyenne $\mathbf{w}(p)$ et de variance $\sigma(p)^2 \mathrm{I}$. Le calcul donné en annexe conduit à

$$(79) \quad h(a(p)) = \sigma(p) \left( \frac{4}{3} \right)^{\frac{1}{\mathrm{M}+4}} a(p)^{-\frac{1}{\mathrm{M}+4}},$$

où l'écart-type $\sigma(p)$ peut être remplacé par son estimation du maximum de vraisemblance (sous hypothèse gaussienne isotrope) :

$$(80) \quad \hat{\sigma}(p)^2 = \frac{1}{\mathrm{M} a(p)} \sum_{x \in \mathrm{E}(p)} \| \mathbf{x} - \mathbf{w}(p) \|^2 .$$

En résumé, l'estimateur utilisé est donc de la forme

$$(81) \quad \hat{p}(\mathbf{u}) = (2\pi)^{-\mathrm{M}/2} \frac{1}{\Sigma a(p)} \sum_{p=1}^{\mathrm{P}} \frac{1}{h(a(p))^{\mathrm{M}}} \times \\ \times a(p) \exp(- \| \mathbf{u} - \mathbf{w}(p) \|^2 / 2 h(a(p))^2) .$$

Pour terminer cette section, signalons l'existence de techniques plus élaborées (mais plus coûteuses) permettant d'identifier des mélanges gaussiens, notamment [68] et [9]. En revanche, nous n'avons pas encore étudié les possibilités d'implantation de ces techniques sur des réseaux stratifiés tels que celui de la figure 6.

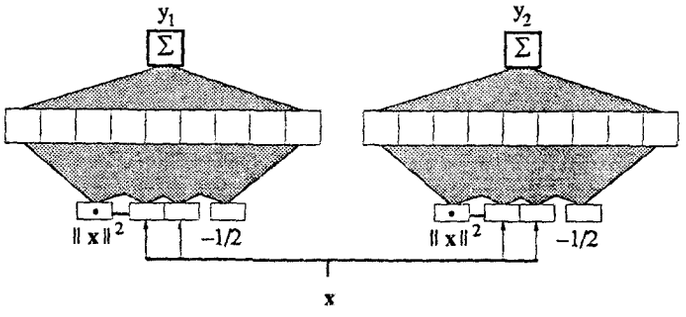


**Figure 6. — Réseau à trois couches permettant d'implanter le classifieur bayesien estimé décrit dans la section 5. A chaque entrée x présentée, ce réseau délivre en sortie les probabilités conditionnelles $\hat{p}(\mathbf{x}/\omega_i)$.**

### 5.3. IMPLANTATION PARALLÈLE

Les estimateurs à noyaux radiaux peuvent être implantés sur une architecture parallèle s'apparentant au PMC à trois couches, auquel on aurait adjoint une cellule de quadration (*fig.* 1 *c*). En effet, pour calculer la valeur de la fonction radiale, on a besoin de

$$(82) \quad \frac{1}{2} \| \mathbf{u} - \mathbf{w} \|^2 = \frac{1}{2} \| \mathbf{u} \|^2 + \frac{1}{2} \| \mathbf{w} \|^2 - \mathbf{u}^\dagger \mathbf{w} .$$

Si on pose

$$(83) \quad \bar{\mathbf{w}} = \begin{pmatrix} -1/2 \\ \mathbf{w} \\ \| \mathbf{w} \|^2 \end{pmatrix} \quad \text{et} \quad \bar{\mathbf{u}} = \begin{pmatrix} \| \mathbf{u} \|^2 \\ \mathbf{u} \\ -1/2 \end{pmatrix},$$

alors on peut écrire (82) sous la forme d'un produit scalaire

$$(84) \quad \frac{1}{2} \| \mathbf{u} - \mathbf{w} \|^2 = \bar{\mathbf{u}}^\dagger \bar{\mathbf{w}} .$$

Par ailleurs, définissons la fonction de transition saturante

(85)

$f(z) = e^{-z}/2$, si $z \geqslant 0$ ; et $f(z) = 1 - f(-z)$ si $z \leqslant 0$.

Alors il est évident que pour le noyau gaussien isotrope (76) on a :

$$(86) \quad \mathrm{K}(\mathbf{u} - \mathbf{w}) = 2(2\pi)^{-\mathrm{M}/2} f(\bar{\mathbf{u}}^\dagger \bar{\mathbf{w}}) .$$

Similairement, si on avait défini un noyau radial à partir de la fonction sigmoïde

$$(87) \quad f_s(v) = \frac{1}{1 + e^v},$$

alors on aurait

$$(88) \quad \mathrm{K}_s(\mathbf{u} - \mathbf{w}) = \mathrm{C}_s f_s(\bar{\mathbf{u}}^\dagger \bar{\mathbf{w}}), \quad \text{avec}$$

avec $\quad \mathrm{C}_s^{-1} = (2^{\mathrm{M}/2} - 2)\, \pi^{\mathrm{M}/2}\, \Gamma(\mathrm{M}/2)\, \zeta(\mathrm{M}/2) ,$

où $\zeta(\mathbf{k})$ est la fonction zêta de Riemann [1, p. 807]. Le noyau $\mathrm{K}_s$ approche le noyau gaussien pour de grandes valeurs de l'argument, et peut être utilisé si les moyens matériels sont plus adaptés à la synthèse de (85) plutôt que (87).

L'apprentissage consiste alors simplement à stocker les vecteurs centroïdes, $\mathbf{w}(p)$, ainsi que les effectifs, $\mathbf{a}(p)$, dans le réseau. Ceci est immédiat si le nombre d'exemples de la base, N, est inférieur ou égal à la taille du réseau, P. Mais en général, on a $\mathrm{N} \gg \mathrm{P}$, et l'apprentissage se ramène donc à l'exécution d'un algorithme d'agglomération [10].

### 5.4. ALGORITHME D'AGGLOMÉRATION

L'algorithme proposé est une version récursive des nuées dynamiques [16]. Rappelons en quelques lignes les instructions de l'algorithme des nuées. Deux opérations sont exécutées dans chaque itération.

(89) *Algorithme des Nuées Dynamiques.*

- Une opération d'affectation :

chaque individu $\mathbf{x}(n)$ est affecté au groupe $E(p^{\{x\}})$ pour lequel le centroïde $\mathbf{w}(p^{\{x\}})$ est le plus proche.

- Une opération de représentation :

les centroïdes $\mathbf{w}(p)$ sont remplacés par les centres de gravité des vecteurs qui leur sont affectés.

Pour des bases de taille N finie, on peut montrer que l'algorithme se termine au bout d'un nombre fini d'itérations [16]. En outre, l'algorithme minimise le critère :

$$\varepsilon = \sum_{p=1}^{P} \sum_{x \in E(p)} \|\mathbf{w}(p) - \mathbf{x}\|^2 ,$$

où $E(p)$ est le groupe engendré par les $\mathbf{w}(p)$ à chaque itération. Rien n'assure que le minimum atteint soit absolu, mais cela n'a pas forcément une grosse importance. Cela veut simplement dire que l'on peut approximer plus finement la base de données avec le même nombre de groupes. La version récursive proposée peut se décrire comme suit [10].

(90) *Algorithme des Nuées Récursives*

- Les centroïdes $\mathbf{w}(p)$ sont initialement fixés à des valeurs arbitraires, par exemple équiparties sur une grille.
- Les effectifs $a(p)$ valent initialement 1, bien que les groupes soient encore vides.
- On présente un nouveau vecteur $\mathbf{x}$.
- On sélectionne le vecteur $\mathbf{w}(p^{\{x\}})$ le plus proche de $\mathbf{x}$.
- On remet à jour l'ensemble des centroïdes par la règle d'évolution

$$(91) \quad \mathbf{w}(p) \leftarrow (a(p)\,\mathbf{w}(p) + \delta(p - p^{\{x\}})\,\mathbf{x}) \times \frac{1}{a(p) + \delta(p - p^{\{x\}})}, \quad \forall p ;$$

autrement dit seul le centroïde $\mathbf{w}(p^{\{x\}})$ est modifié.
- On remet à jour les effectifs par la règle :

$$(92) \qquad a(p) \leftarrow \eta(a(p) + \delta(p - p^{\{x\}})), \quad \forall p ;$$

- Les données sont recyclées une ou deux fois, après quoi l'algorithme est interrompu.

Le coefficient $\eta$ est un facteur d'oubli que l'on choisira dans l'intervalle (0,9, 1). Cet algorithme s'apparente à celui des cartes auto-organisatrices de Kohonen [31], de sorte que sa convergence pourrait être étudiée à l'aide des outils d'analyse des processus linéaires markoviens, comme cela a été fait dans [54] pour les cartes auto-organisatrices.

## 6. Un exemple de Classification binaire

Dans cet exemple, des individus caractérisés par les densités suivantes ont été générés. Pour les individus de la classe 1 :

$$(93) \qquad p_x(\mathbf{x}) = \frac{1}{4\pi}\left[e^{-x^t x/2} + \frac{1}{10} e^{-x^t x/20}\right],$$

et pour ceux de la classe 2

$$(94) \qquad p_x(\mathbf{x}) = \{p_1(\mathbf{x}) + p_2(\mathbf{x}) + 2\,p_3(\mathbf{x})\}/4 ,$$

avec :

$$p_j(\mathbf{x}) = \frac{1}{2\pi}\frac{1}{\sigma_{j,x}\,\sigma_{j,y}} \exp\left\{-\frac{(x - m_{j,x})^2}{2\,\sigma_{j,x}^2} - \frac{(y - m_{j,y})^2}{2\,\sigma_{j,y}^2}\right\} .$$

$\sigma_{1,x} = 0{,}2$ , $\sigma_{1,y} = 0{,}2$ , $m_{1,x} = 0$ , $m_{1,y} = 0$ ;
$\sigma_{2,x} = 0{,}2$ , $\sigma_{2,y} = 0{,}2$ , $m_{1,x} = 0$ , $m_{1,y} = 2$ ;
$\sigma_{3,x} = 0{,}2$ , $\sigma_{3,y} = 1$ , $m_{1,x} = 0{,}2$ , $m_{1,y} = 1$ .

L'échantillon sur lequel on travaille est de taille 50 pour chacune des classes, et est représenté en figure 7. Afin de

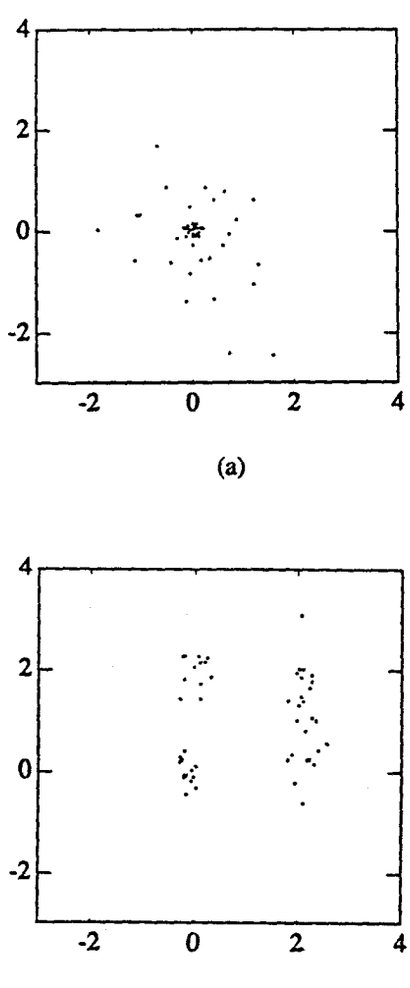


**Figure 7. — Échantillons de taille 50 générés par simulation et utilisés dans la partie 6 ; (*a*) classe 1, (*b*) classe 2.**

mesurer les performances de la classification réalisée on calcule l'ensemble des probabilités d'erreurs, que l'on convient de ranger dans une matrice dite de confusion. En accord avec (28), cette dernière est définie par :

$$C_{ij} = \text{prob (choisir } \omega_i/\mathbf{x} \in \omega_j) .$$

Cette matrice est calculable exactement, puisque nous connaissons les vraies densités. Elle a pour expression, pour un classifieur décidant $\omega_i$ sur un domaine $D_i$ :

$$C(i, j) = \int_{x \in D_i} p(\mathbf{x}/\omega_j)\, d\mathbf{x} . \tag{95}$$

Si on convient de poser $d_i(\mathbf{u})$ la fonction indicatrice du domaine $D_i$, l'expression (95) s'écrit :

$$C(i, j) = \int_x d_i(\mathbf{x})\, p(\mathbf{x}/\omega_j)\, d\mathbf{x} . \tag{96}$$

Pour un classifieur binaire bayesien, on peut notamment écrire :

$$d_i(\mathbf{x}) = \frac{1}{2} [\text{sign } \{\hat{p}_i(\mathbf{x}) - \hat{p}_j(\mathbf{x})\} + 1] , \tag{97}$$

où sign $(z) \in \{+ 1, - 1\}$, et où les $\hat{p}_i(\mathbf{x})$ approximent les $p(\mathbf{x}/\omega_i)$. En utilisant notre algorithme d'apprentissage décrit dans la section 5, on obtient la matrice de confusion ci-dessous après 2 recyclages, $\eta = 0{,}99$, et avec $P = 6$ groupes dans chaque classe :

$$C_{est} = \begin{pmatrix} 0{,}916 & 0{,}160 \\ 0{,}084 & 0{,}840 \end{pmatrix} . \tag{98}$$

Sur les 6 groupes, 4 seulement ont été retenus dans la classe 1, les 2 groupes restants ayant des effectifs nuls au terme de l'algorithme d'agglomération. La confusion (98) doit être comparée à la confusion ultime que l'on obtient avec le classifieur bayesien idéal, i.e., celle que l'on obtient en remplaçant $\hat{p}_i(\mathbf{x})$ par $p(\mathbf{x}/\omega_i)$ dans (97) :

$$C_{ideal} = \begin{pmatrix} 0{,}864 & 0{,}024 \\ 0{,}136 & 0{,}976 \end{pmatrix} . \tag{99}$$

La figure 6 donne une vue générale de l'ensemble du réseau bayesien qui serait utilisé dans cet exemple. Les figures 8 *a* et 8 *b* représentent les domaines de décision bayesiens idéaux, et bayesiens estimés à travers (81) et (97), respectivement.

Ensuite, l'algorithme RPG a été appliqué d'une part à un PMC (3, 9, 1), ayant 3 cellules d'entrée, 9 cellules cachées et 1 cellule de sortie, et d'autre part à un PMC (4, 5, 2, 1) pour lequel l'entrée $\|\mathbf{x}\|^2$ a été rajoutée. L'apprentissage a été exécuté dans les conditions suivantes :

(100)

- La valeur initiale des éléments des matrices $W(q)$ a été choisie aléatoirement dans l'intervalle $[- 1/5, 1/5]$.
- La version stochastique du RPG a été utilisée.
- Le facteur d'apprentissage $\mu$ a été pris égal à 0,1 initialement, puis
  - — a été diminué exponentiellement si l'erreur augmentait : $\mu \leftarrow \mu/2$ ; dans ce cas l'itération suivante n'a pas pris en compte la valeur de $\mathbb{W}$ obtenue,
  - — a été augmenté si l'erreur décroissait : $\mu \leftarrow \mu * 1{,}1$,
  - — l'apprentissage s'arrêtait dès que le pas était inférieur à $10^{-7}$.
- La fonction de transition utilisée est celle définie par (85), avec un facteur de dilatation de 5, c'est-à-dire : $f(x) = [1 + s - s \exp(- s\alpha x)]/2$, avec $s = \text{sign } (x)$ et $\alpha = 5$.
- Dix exécutions complètes de l'algorithme ont été faites à partir de dix valeurs initiales tirées aléatoirement, et seule celle conduisant à la confusion la plus faible a été conservée.

Pour le PMC (3, 9, 1), l'erreur est devenue stationnaire à partir du 19-ième recyclage, et ce jusqu'à ce que le pas atteigne $\mu \approx 10^{-7}$ au recyclage 40. La matrice de confusion apparente obtenue sur l'ensemble d'apprentissage (en effectuant un comptage simple non pondéré par les densités) avait alors atteint la valeur

$$A_{pmc\ (3, 9, 1)} = \begin{pmatrix} 0{,}97 & 0{,}13 \\ 0{,}03 & 0{,}87 \end{pmatrix} . \tag{101}$$

Les domaines de décision correspondants sont représentés figure 8 *c*. Pour le PMC (4, 5, 2, 1), l'apprentissage a été stoppé à l'itération 77. La matrice de confusion apparente sur l'ensemble d'apprentissage avait alors atteint la valeur

$$A_{pmc\ (4, 5, 2, 1)} = \begin{pmatrix} 0{,}95 & 0{,}13 \\ 0{,}05 & 0{,}87 \end{pmatrix} . \tag{101}$$

Les domaines de décision correspondants sont représentés figure 8 *d*. Les matrices de confusion réelles pour ces deux classifieurs ont été estimées à (en utilisant les vraies densités, qui sont connues dans le cas de simulations)

$$C_{pmc\ (3, 9, 1)} = \begin{pmatrix} 0{,}775 & 0{,}266 \\ 0{,}225 & 0{,}734 \end{pmatrix} , \tag{102}$$

$$C_{pmc\ (4, 5, 2, 1)} = \begin{pmatrix} 0{,}600 & 0{,}270 \\ 0{,}400 & 0{,}730 \end{pmatrix} .$$

On a supposé que les exemples étaient bien classés en dehors du pavé $[- 2{,}5, 4] \times [- 2{,}5, 4]$, ce qui est évidemment un peu optimiste, surtout pour le PMC ; les probabilités d'occurrence des exemples sont par contre faibles à l'extérieur de ce pavé, de sorte que le biais de la confusion reste raisonnable.

## 7. Conclusions

On retiendra que le PMC à 3 couches peut approcher n'importe quelle application, mais avec un nombre grand de cellules, et pourvu que cette application soit connue.

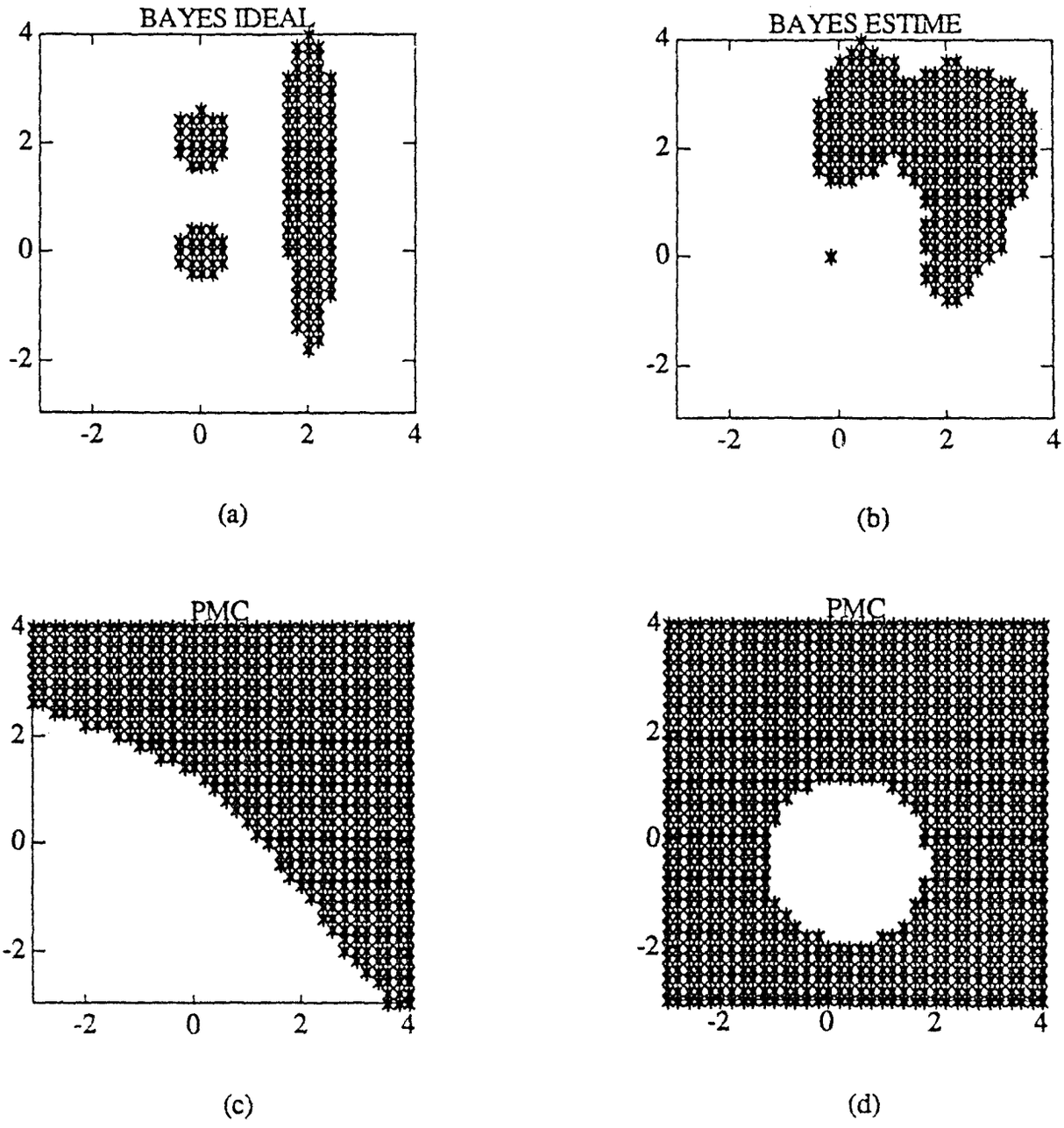


**Figure 8. — Domaines de décision obtenus sur une grille de 29 × 29 points ; la décision en faveur de la classe 2 est représentée par la zone noircie. (*a*) Solution bayesienne idéale, obtenue avec les densités exactes. (*b*) Solution bayesienne, obtenue avec les estimateurs de densité à noyau radial variable. (*c*) Solution obtenue avec un PMC (3, 9, 1). (*d*) Solution obtenue avec un PMC (4, 5, 2, 1), en adjoignant aux 3 entrées précédentes l'entrée $\| x \|^2$.**

Lorsque l'application n'est connue qu'à travers N exemples, l'apprentissage du PMC par RPG nécessite du doigté. Ses performances en classification sont sensibles à la valeur initiale que l'on donne à l'ensemble des paramètres $\mathbb{W}$, comme nous l'avons montré dans la section 6. De plus, dans le cas favorable où la RPG converge vers le minimum absolu du critère d'erreur, le PMC ne délivre qu'asymptotiquement (pour de grandes bases ou pour un grand nombre de recyclages bruités) la solution Bayesienne (théorèmes I à IV, section 3). Pourtant, même la convergence de l'algorithme RPG vers un minimum local peut prendre un temps arbitrairement long (théorème V, section 4). Ces limitations de l'algorithme RPG ont motivé la recherche d'une autre procédure. La seconde procédure d'apprentissage que nous avons proposée est basée sur l'estimation des densités conditionnelles, et soulève moins

de difficultés pratiques (section 5). En outre, la complexité de sa mise en œuvre est beaucoup plus faible pour des performances plutôt meilleures.

En conclusion, l'approche bayesienne, bien que plus standard, semble à nos yeux plus maniable et pour l'instant moins coûteuse en nombre d'opérations que la RPG. De plus, cette approche peut être elle aussi implantée sur un réseau stratifié de cellules neuro-mimétiques. Cette solution pourrait inspirer un certain nombre d'améliorations à apporter aux algorithmes d'apprentissage pour les réseaux en couches dédiés à la classification.

Par ailleurs, un des problèmes encore ouvert est celui de l'évaluation des performances de généralisation lorsque seulement un nombre limité d'exemples est disponible. Certains auteurs abordent différentes facettes de ce problème dans [21], [18], [33]. Ce problème ne s'est pas imposé à nous dans ce travail puisqu'il s'agissait de simulations (il était possible d'utiliser les vraies densités pour calculer les matrices de confusion). L'expression (96) suppose en effet que les densités $p(\mathbf{x}/\omega_i)$ sont connues. Dans certains utilitaires informatiques actuellement disponibles, notamment Soma [15b] ou *Neuroclass* [8], les performances sont estimées par tirage aléatoire de partitions dans l'ensemble des données : dans chaque tirage une partie sert pour l'apprentissage, l'autre pour l'estimation des performances. Ce procédé n'est pas le meilleur, mais a l'avantage d'être simple à mettre en œuvre. D'autres utilitaires incluent des procédés similaires, mais aucun ne semble à l'heure actuelle totalement satisfaisant. Des recherches sont en cours autour de ce problème, qui ne date pas d'aujourd'hui d'ailleurs, puisque l'obtention de résultats fiables ne peut se passer d'une prédiction de performances.

## 8. Remarques

Pendant la rédaction du présent article, nous avons remarqué la parution de deux communications dont le contenu rejoint certains de nos résultats. Dans [6], un algorithme d'agglomération (quantification vectorielle) est proposé en vue de rendre utilisable le réseau de classification probabiliste [58] ; cet algorithme est une alternative à notre algorithme [10] exposé en section 5.5. Dans [29], la convergence de l'erreur quadratique vers les probabilités empiriques a été abordée, à première vue d'une façon sensiblement différente de la nôtre.

Par ailleurs, les rapporteurs sur l'article ont suggéré tardivement un certain nombre de références complémentaires que l'auteur n'a pas voulu passer sous silence, bien qu'elles lui soient peu familières pour la plupart.

Le livre [69] semble exceptionnellement clair et rigoureux, et serait une bonne sélection parmi la myriade de livres disponibles sur le sujet.

Un certain nombre d'applications des réseaux de neurones peut être trouvé dans la série Neural Information Processing Systems, de 1989 à 1991, publiée par Morgan Kaufmann, ainsi que dans les actes des International Joint Conferences on Neural Networks, de 1988 à 1991, publiées par IEEE.

Pour les problèmes de minimisation de risque empirique, un expert recommande le livre de Vapnik [70]. En ce qui concerne les relations entre les réseaux de neurones et l'approche Bayesienne, les rapporteurs indiquent aussi [71] et [72].

Enfin, l'approximation de fonctions booléennes est abordée dès 1970 dans [73], et les approximations polyédrales dans [74].

*Manuscrit reçu le 7 septembre 1991.*

### BIBLIOGRAPHIE

[1] M. Abramowitz and I. A. Stegun, *Handbook of Mathematical Functions*, Dover Publ. Inc., New York, 1965, 1972.

[2] S. Amari, « Mathematical Foundations of Neuro-Computing », *Proceedings of the IEEE*, special issue on neural networks, September 1990, 1443-1463.

[3] E. B. Baum, « A Proposal for More Powerful Learning Algorithms », *Neural Computation*, 1, 1989, 201-207.

[4] E. B. Baum, « The Perceptron Algorithm is Fast for Non Malicious Distributions », *Neural Computation*, 2, 1990, 248-260.

[4b] A. Benveniste, M. Métivier et P. Priouret, *Algorithmes adaptatifs et approximations stochastiques*, Masson, 1987.

[4c] A. Blumer *et al.*, « Learnability and the Vapnik-Chervonenkis Dimension », *Journal of the ACM*, vol. 36, n° 4, October 1989, 929-965.

[4d] J. Bridle, « Probabilistic Interpretation of Feedforward Classification Network Outputs with relationships to Statistical Pattern Recognition », *Neurocomputing*, Fogelman and Herault editors, NATO ASI series, vol. F68, 1990, Springer Verlag, 227-236.

[5] H. Boulard and C. J. Wellekens, « Links between Markov Models and Multilayer Perceptrons », *Advances in Neural Information Processing Systems 1*, Morgan Kauffmann, 1989, 502-510.

[6] P. Burrascano, « Learning Vector Quantization for the Probabilistic Neural Network », *IEEE Trans. Neural Networks*, vol. 2, n° 4, July 1991, 458-461.

[7] T. Cacoullos, « Estimation of a Multivariate Density », *Annals of Inst. of Stat. Math.*, 18, 1966, 178-189.

[8] J. G. Cailton, F. Vallet *et al.*, « Neuroclass, Manuel d'utilisation », version 1.1 pour stations SUN, *Rapport Thomson LCR*, 9 août 1990.

[9] G. Celeux and D. Diebolt, « Identification of a densities mixture and classification », *INRIA Report*, 1984.

[10] P. Comon, « Distributed bayesian classification », *Revue Technique Thomson CSF*, numéro spécial Reconnaissance de Formes et Réseaux Neuronaux, vol. 22, n° 4, December 1990, 543-562.

[11] P. Comon, « Supervised Detection and Estimation », *Esprit BRA Workshop on Neural Networks and Artificial Vision*, January 1991.

[12] G. Cybenko, « Approximation by Superpositions of Sigmoidal Functions », *Mathematics of Control, Signals and Systems*, 2, n° 4, 1989.

[13] M. Cosnard, « Neural Algorithms », *Les entretiens de Lyon*, mars 1990.

[14] N. E. Cotter, « The Stone-Weierstrass Theorem and its Application to Neural Networks », *IEEE Trans. Neural Networks*, vol. 1, December 1990, 290-295.

[15] P. Delsarte and Y. Kamp, « Low Rank Matrices with a given Sign Pattern », *SIAM Jour. Disc. Math.*, vol. 2, n° 1, February 1989, 51-63.

[15b] P. Demartines et L. Tettoni, « SOMA » : Simulateur logiciel de modèles neuronaux adaptatifs », rapport interne LAMI (EPFL, Ecublens, CH1015 Lausanne), 1991.

[16] E. Diday *et al.*, *Éléments d'Analyse de Données*, Dunod, 1982.

[17] R. Durbin and D. E. Rumelhart, « Product Units : A Computationally Powerful and Biologically Plausible Rxtension to Backpropagation Networks », *Neural Computation*, 1, 1989, 133-142.

[18] K. Fukunga and D. L. Kessel, « Estimation of Classification Error », *IEEE Trans. Computers*, vol. 20, n° 12, December 1971, 1521-1527.

[19] K. Funahasashi, « On the Approximate Realization of Continuous Mappings by Neural Networks », *Neural Networks*, vol. 2, n° 3, 1989, 183-192.

[20] P. E. Gill, W. Murray and M. H. Wright, *Practical Optimization*, Academic Press, 1981.

[21] W. Greblecki, « Asymptotically Optimal Pattern Recognition... », *IEEE Trans. Inf. Theory*, 1978, 24, 250-251.

[22] F. R. Hampel, « Robust Estimation : A Condensed Partial Survey », *Wahrscheinlischskeits Theorie Verw. Geb.*, 27, 1973, 87-104.

[23] D. J. Hand, *Kernel Discriminant Analysis*, 1982, RSP press.

[24] R. Hecht-Nielsen, « Kolmogorov's Mapping Neural Network Existence Theorem », *First IEEE ICNN Conference*, 3, 1987, 11-14.

[25] R. Hecht-Nielsen, « Theory of Backpropagation Neural Networks », *IEEE IJCNN*, 1, 1989, 593-605.

[26] K. Hornik, « Approximation Capabilities of Multilayer Feedforward Networks », *Neural Networks*, vol. 4, n° 2, 1991, 251-258.

[27] B. Irie and S. Miyake, « Capabilities of Three-Layered Perceptrons », *Second IEEE ICNN Conference*, 1, 1988, 641-648.

[28] R. A. Jacobs, « Increased Rates of Convergence through Learning Rate Adaptation », *Neural Networks*, vol. 1, n° 4, 1988, 295-307.

[29] F. Kanaya and S. Miyake, « Bayes Statistical Behavior and Valid Generalization... », *IEEE Trans. NN*, vol. 2, July 1991, 471-475.

[30] D. Kazakos, « Bayes Error Probability with Inaccurate Decision Threshold », *Electronic Letters*, vol. 25, n° 14, July 1989, 894-895.

[30a] S. Knerr, L. Personnaz, G. Dreyfus, « Single-layer Learning Revisited : a Stepwise Procedure for Building and Training a Neural Network », *Neurocomputing*, Fogelman and Herault editors, NATO ASI series, vol. F68, 1990, Springer Verlag, 41-50.

[31] T. Kohonen, « The Self-Organizing Maps », *Proceedings of the IEEE*, special issue on neural networks, September 1990, 1464-1480.

[32] A. N. Kolmogorov, « On the Representation of Continuous Functions of Many Variables by Superposition of Continuous Functions of One Variable and Addition », *American Math. Soc. Series Trans.*, series 2, vol. 88, 1963, 55-59 (orig. : Dolk. Akad. Nauk. SSSR, 114, 1957, 953-956).

[33] A. Krzyzak, « On Exponential Bounds on the Bayes Risk of Kernel Classification Rule », *IEEE Trans. Information Theory*, vol. 37, n° 3, May 1991, 490-499.

[34] Y. Lecun, « A Theoretical Framework for Back-Propagation », *Proceedings of the 1988 Connectionist Summer School*, Carnegie-Mellon University, Touretzky, Hinton and Sejnowski editors, Morgan Kaufmann 1989.

[35] T. Lefebvre *et al.*, « Arachyde, Analyse et Reconnaissance Acoustique par Systeme Hybride », *Revue Thomson CSF*, numéro spécial Reconnaissances des Formes, tome 2, vol. 23, n° 1, mars 1991, 185-198.

[36] D. Legitimus and L. Schwab, « Natural Underwater Sounds Identification by Neural Networks and Classical techniques », *Revue Thomson CSF*, numéro spécial Reconnaissance des Formes, tome 2, vol. 23, n° 1, mars 1991, 161-184.

[37] R. P. Lippmann, « An Introduction to Computing with Neural Nets », *IEEE ASSP Magazine*, April 1987, 4-19.

[38] Z. Q. Luo, « On the Convergence of the LMS Algorithm with Adaptive Learning Rate », *Neural Computation*, 3, 1991, 226-245.

[39] J. Makhoul *et al.*, « Classification Capabilities of two layer Neural Nets », *ICASSP 89*, 635-638.

[40] G. Mauri and B. Apolloni, « Computational Aspects of Learning in Neural Networks », *Les entretiens de Lyon*, mars 1990.

[41] G. Mirchandani and W. Cao, « On Hidden Nodes for Neural Nets », *IEEE Trans. CAS*, special issue, May 1989, vol. 26, n° 5, 661-664.

[42] J. Moody and C. J. Darken, « Fast Learning in Networks of Locally-Tune Processing Units », *Neural Computation*, 1, 1989, 281-294.

[43] J. M. Nicolas, A. Lemer et D. Legitimus, « Identification Automatique de Bruits Impulsifs en Acoustique Sous-Marine par Réseaux Multicouches », *Neuro-Nimes*, 13-16 novembre 1989.

[44] J. M. Nicolas, A. Lemer and J. C. Delvigne, « Automatic Identification of Transient Biological Noises using Arborescent Wavelets and Neural Networks », Colloque *Ondelettes et Premières Applications*, Marseille, 29 mai-3 juin 1989.

[45] OFTA ed., *Les réseaux de Neurones*, Masson, 1991.

[46] J. Park and I. W. Sandberg, « Universal Approximation Using Radial-Basis Function Networks », *Neural computation*, 3, 1991, 246-257.

[47] D. B. Parker, « Optimal Algorithms for Adaptive Networks : second Order Back-Propagation, Second Order Direct Propagation, and Second Order Hebbian Learning », *First IJCNN*, San Diego, June 1987, vol. II, 593-600.

[48] L. Personnaz *et al.*, « Les Machines Neuronales », *La Recherche*, vol. 19, novembre 1988, 1362-1373.

[49] T. Poggio, « On Optimal Non Linear Associative Recall », *Biol. Cybernetics*, 19, 1975, 201-209.

[50] T. Poggio and F. Girosi, « Networks for Approximation and Learnings », *Proceedings of the IEEE*, special issue on neural networks, September 1990, 1481-1497.

[51] L. R. Rabiner, « A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition », *Proc. of the IEEE*, vol. 77, n° 2, February 1989, 257-286.

[52] P. Refregier and J. M. Vignolle, « An Improved Version of the Pseudo-Inverse Solution for Classification and Neural Networks », June 1989, *Europhysics Letters*, October 1989, 387-392.

[53] P. Refregier, A. Jaffre and F. Vallet, « A Probabilistic Approach for Multiclass Discrimination with Neural Networks », *Revue Technique Thomson CSF*, numéro spécial Reconnaissance de Formes et Réseaux Neuronaux, vol. 22, n° 4, December 1990, 563-572.

[54] H. Ritter and K. Schulten, « Convergence Properties of Kohonen's Topology Conserving Maps : Fluctuations, Stability and Dimension Selection », *Biological Cybernetics*, vol. 60, 1988, 59-71.

[55] D. W. Ruck, S. K. Rogers *et al.*, « The Multilayer Perceptron as an Approximation to Bayes Optimal Discriminant Func-

tion », *IEEE Trans. Neural Networks*, vol. 1, December 1990, 296-298.

[56] W. Rudin, *Analyse Réelle et Complexe*, Masson, 1978.

[57] D. E. Rumelhart and J. L. McClelland, *Parallel Distributed Processing*, vol. 1, 1986, MIT Press.

[58] D. F. Specht, « Probabilistic Neural Networks », *Neural Networks*, vol. 3, n° 1, 1990, 109-118.

[59] D. A. Sprecher, « On the Structure of Continuous Functions of Several Variables », *Transactions of the American Mathematical Society*, vol. 115, n° 3, March 1965, 340-355.

[60] G. Tesauro, Y. He and S. Ahmad, « Asymptotic Convergence of Backpropagation », *Neural Computation*, vol. 1, 1989, 382-391.

[61] F. Vallet, « Approche globale du problème de discrimination : aspects probabilistes », *Revue Technique Thomson CSF*, numéro spécial Reconnaissance de Formes et Réseaux Neuronaux, vol. 22, n° 4, décembre 1990, 519-542.

[62] L. G. Vaillant, « A Theory of the Learnable », *Communications of the ACM*, November 1984, vol. 27, 1134-1142.

[63] T. P. Vogl *et al.*, « Accelerating the Convergence of the Back-Propagation Method », *Biol. Cybernetics*, 59, 1988, 257-263.

[64] D. J. Volper and S. E. Hampson, « Quadratic function Nodes ; use, structure and training », *Neural Networks*, vol. 3, 1990, 93-107.

[65] R. L. Watrous, « Learning Algorithms for Connectionist Networks : Applied Gradient Methods of Nonlinear Optimization », *First IJCNN*, San Diego, California, June 21-24, 1987, vol. II, 619-627.

[66] B. Widrow and R. Winter, « Neural Nets for Adaptive Filtering and Adaptive Pattern Recognition », *Computer*, vol. 21, n° 3, March 1988, 25-39.

[67] B. Widrow and M. A. Lehr, « 30 Years of Adaptive Neural Networks », *Proceedings of the IEEE*, special issue on neural networks, September 1990, 1415-1442.

[68] S. Yakowitz and J. D. Spragins, « On the Identifiability of Finite Mixtures », *Annals Math. Stat.*, vol. 39, n° 1, 1968, 209-214.

[69] J. Hertz, A. Krogh and R. G. Palmer, Introduction to the Theory of Neural Computation, Addison-Wesley, 1991.

[70] Vapnik, Estimation of Dependences Based on Empirical Data, Springer, 1982.

[71] S. Geman, E. Bienenstock and R. Doursat, « Neural Networks and the Bias-Variance Dilemna », *Neural Computation*, vol. 4, n° 1, 1992, 1-58.

[72] R. J. Williams and J. Peng, « An Efficient Gradient-Based Algorithm for On-Line Training of Recurrent Network Trajectories », *Neural Computation*, vol. 2, n° 4, 1990, 490-501.

[73] M. L. Minsky and S. A. Papert, Perceptrons, MIT Press, 1969.

[74] T. M. Cover, « Geometrical and Statistical Properties of Systems of linear inequalities with Applications in Pattern Recognition », *IEEE Trans. Electronic Computers*, 14, 1965, 326-334.

## Annexes

• Démonstration des théorèmes I et II.

Les démonstrations des théorèmes I et II débutent de la même façon. La définition de l'erreur donnée en (40) peut s'écrire

$$\varepsilon(\mathrm{N}) = \sum_{k=1}^{\mathrm{K}} \frac{\mathrm{N}_k}{\mathrm{N}} \frac{1}{\mathrm{N}_k} \sum_{x^{(n)} \in \omega_k} \| \mathbf{y}^{(n)} - \varphi(\mathbb{W}, \mathbf{x}^{(n)}) \|^2 .$$

Faisons tendre les $\mathrm{N}_k$ vers l'infini, nous obtenons :

$$\varepsilon(\infty) = \sum_{k=1}^{\mathrm{K}} \mathrm{P}_k \int p(\mathbf{u}/\omega_k) \| \mathbf{y}^{(n)} - \varphi(\mathbb{W}, \mathbf{u}) \|^2 \, d\mathbf{u} .$$

Puis en utilisant la relation

$$p(\mathbf{u}) = \sum_{k=1}^{\mathrm{K}} \mathrm{P}_k \, p(\mathbf{u}/\omega_k) ,$$

nous obtenons finalement :

$$\text{(A-1)} \quad \varepsilon(\infty) = \int p(\mathbf{u}) \| \varphi(\mathbb{W}, \mathbf{u}) \|^2 \, d\mathbf{u} - 2 \sum_{k=1}^{\mathrm{K}} \mathrm{P}_k \int p(\mathbf{u}/\omega_k) \sum_{j=1}^{\mathrm{K}} y_j^{(n)} \varphi_j(\mathbb{W}, \mathbf{u}) \, d\mathbf{u} + \varepsilon_0 ,$$

où $\varepsilon_0$ est indépendant des $\varphi_j$. ◇

**Démonstration du théorème I**

Ici, $y_j^{(n)} = \delta_{jk}$ si $\mathbf{x}^{(n)} \in \omega_k$. Dans (A-1), la somme sur $j$ se réduit donc à un seul terme, $\varphi_k(\mathbb{W}, \mathbf{u})$. Posons $g_k(\mathbf{u}) = b_k(\mathbf{u})/p(\mathbf{u})$, où $b_k(\mathbf{u})$ a été défini en (31) ; remarquons que $g_k(\mathbf{u})$ n'est autre que la probabilité conditionnelle de la classe $k$, $p(\omega_k/\mathbf{u})$. L'expression (A-1) devient dans ce cas

$$\text{(A-2)} \quad \varepsilon(\infty) = \int p(\mathbf{u}) \| \varphi(\mathbb{W}, \mathbf{u}) \|^2 \, d\mathbf{u} - 2 \int p(\mathbf{u}) \sum_{k=1}^{\mathrm{K}} g_k(\mathbf{u}) \varphi_k(\mathbb{W}, \mathbf{u}) \, d\mathbf{u} + \varepsilon_1 ,$$

où $\varepsilon_1$ ne dépend pas des $\varphi_k$. Minimiser $\varepsilon(\infty)$ revient donc à minimiser

$$\text{(A-3)} \quad \varepsilon(\infty) - \varepsilon_2 = \int p(\mathbf{u}) \| \mathbf{g}(\mathbf{u}) - \varphi(\mathbb{W}, \mathbf{u}) \|^2 \, d\mathbf{u} .$$

L'application $\varphi$ construite approxime donc $\mathbf{g}(\mathbf{u})$. Autrement dit, si $\Phi$ est un ensemble suffisamment large, trouver la sortie $\varphi_k$ la plus élevée revient à trouver la sortie $g_k$ (ou $b_k$) la plus élevée. ◇

**Démonstration du théorème II**

Ici, $y_i^{(n)} = \kappa(i, k)$ si $\mathbf{x}^{(n)} \in \omega_k$. D'après (A-1) nous avons donc

$$\text{(A-4)} \quad \varepsilon(\infty) = \int p(\mathbf{u}) \| \varphi(\mathbb{W}, \mathbf{u}) \|^2 \, d\mathbf{u} - 2 \sum_{k=1}^{\mathrm{K}} \sum_{i=1}^{\mathrm{K}} \int p(\mathbf{u}/\omega_k) \, \mathrm{P}_k \, \kappa(i, k) \, \varphi_i(\mathbb{W}, \mathbf{u}) \, d\mathbf{u} + \varepsilon_3 ,$$

où $\varepsilon_3$ ne dépend pas des $\varphi_i$. On pose cette fois $\mathrm{G}_i(\mathbf{u}) = \mathrm{B}_i(\mathbf{u})/p(\mathbf{u})$, où $\mathrm{B}_i(\mathbf{u})$ est définie en (30). La relation (A-4) devient alors

$$\varepsilon(\infty) = \int p(\mathbf{u}) \| \varphi(\mathbb{W}, \mathbf{u}) \|^2 \, d\mathbf{u} - 2 \sum_{i=1}^{\mathrm{K}} \int p(\mathbf{u}) \, \mathrm{G}_i(\mathbf{u}) \, \varphi_i(\mathbb{W}, \mathbf{u}) \, d\mathbf{u} + \varepsilon_4 .$$

Minimiser $\varepsilon(\infty)$ revient donc finalement à minimiser

$$(A\text{-}5)\quad \varepsilon(\infty) - \varepsilon_5 = \int p(\mathbf{u}) \| G(\mathbf{u}) - \varphi(\mathbb{W}, \mathbf{u}) \|^2 \, d\mathbf{u} ,$$

ce qui montre que $\varphi(\mathbb{W}, \mathbf{u})$ approxime $G(\mathbf{u})$. Ici encore, si $\Phi$ est suffisamment large, pour la plupart des $\mathbf{u}$ fixés les applications $G(\mathbf{u})$ et $\varphi(W, \mathbf{u})$ seront minimales pour la même composante $k$. ◇

• Démonstration des théorèmes III et IV.

Les théorèmes III et IV résultent d'un lemme commun. Écrivons l'erreur quadratique (45) sous la forme

$$\varepsilon(N, R) = \sum_{k=1}^{K} \frac{N_k}{N} \frac{1}{N_k} \sum_{x^{(n)} \in \omega_k} \times \frac{1}{R} \sum_{r=1}^{R} \| \mathbf{y}^{(n)} - \varphi(\mathbb{W}, \mathbf{x}^{(n)} + \mathbf{z}^{(n, r)}) \|^2 .$$

Posons à présent

$$(A\text{-}6)\qquad \hat{P}_k = \frac{N_k}{N} ,$$

$$(A\text{-}7)\quad \xi_k(\mathbf{u}) = \| \mathbf{y}^{(n)} - \varphi(\mathbb{W}, \mathbf{u}) \|^2 , \text{ pour } \mathbf{x}^{(n)} \in \omega_k ,$$

ce qui est possible car la sortie $\mathbf{y}^{(n)}$ ne dépend que de $k$ lorsque $\mathbf{x}^{(n)} \in \omega_k$. Quand R tend vers l'infini, nous avons :

$$\varepsilon(N, \infty) = \sum_{k=1}^{K} \hat{P}_k \int p_z(\mathbf{u}) \frac{1}{N_k} \sum_{x^{(n)} \in \omega_k} \xi_k(\mathbf{x}^{(n)} + \mathbf{u}) \, d\mathbf{u} .$$

Soit après le changement de variable $\mathbf{v} = \mathbf{x}^{(n)} + \mathbf{u}$, et en posant

$$(A\text{-}8)\qquad \hat{p}(\mathbf{v}/\omega_k) = \frac{1}{N_k} \sum_{x^{(n)} \in \omega_k} p_z(\mathbf{v} - \mathbf{x}^{(n)}) ,$$

nous obtenons

$$(A\text{-}9)\qquad \varepsilon(N, \infty) = \sum_{k=1}^{K} \hat{P}_k \int \hat{p}(\mathbf{v}/\omega_k) \, \xi_k(\mathbf{v}) \, d\mathbf{v} . \qquad ◇$$

**Démonstration du théorème III**

Ici, $y_j^{(n)} = \delta_{jk}$ si $\mathbf{x}^{(n)} \in \omega_k$. On définit les quantités suivantes :

$$(A\text{-}10)\qquad \hat{p}(\mathbf{v}) = \sum_{k=1}^{K} \hat{P}_k \, \hat{p}(\mathbf{v}/\omega_k)$$

et

$$(A\text{-}11)\qquad \hat{g}_k(\mathbf{v}) = \hat{P}_k \, \hat{p}(\mathbf{v}/\omega_k)/\hat{p}(\mathbf{v}) .$$

Alors l'expression (A-9) de l'erreur devient

$$\varepsilon(N, \infty) = \int \hat{p}(\mathbf{v}) \| \varphi(\mathbf{v}) \|^2 \, d\mathbf{v} - 2 \int \sum_{k=1}^{K} \hat{g}_k(\mathbf{v}) \, \varphi_k(\mathbb{W}, \mathbf{v}) \, d\mathbf{v} + \varepsilon_1 .$$

Et finalement

$$(A\text{-}12)\quad \varepsilon(N, \infty) = \int \hat{p}(\mathbf{v}) \| \varphi(\mathbb{W}, \mathbf{v}) - \hat{g}(\mathbf{v}) \|^2 \, d\mathbf{v} + \varepsilon_2$$

montre que la fonction $\varphi(\mathbb{W}, \mathbf{v})$ approxime la fonction $\hat{g}(\mathbf{v})$, qui est elle-même une estimation de la fonction bayesienne $\mathbf{g}(\mathbf{v}) = \mathbf{b}(\mathbf{v})/p(\mathbf{v})$ définie en (31). La conclusion est similaire à celle du théorème I. ◇

**Démonstration du théorème IV**

Le codage est défini maintenant par (42) :

$$y_i^{(n)} = \kappa(i, k) \quad \text{pour} \quad \mathbf{x}^{(n)} \in \omega_k .$$

On garde la notation (A-10) et on pose $\hat{G}_i(\mathbf{u}) = \hat{B}_i(\mathbf{u})/\hat{p}(\mathbf{u})$, avec

$$(A\text{-}13)\qquad \hat{B}_i(\mathbf{x}) = \sum_{j=1}^{K} \kappa(i, j) \, \hat{P}_j \, \hat{p}(\mathbf{x}/\omega_j) .$$

Il vient alors d'après (A-9) que

$$\varepsilon(N, \infty) = \int \hat{p}(\mathbf{v}) \| \varphi(\mathbf{v}) \|^2 \, d\mathbf{v} - 2 \int \sum_{i=1}^{K} \hat{G}_i(\mathbf{v}) \, \hat{p}(\mathbf{v}) \, \varphi_i(\mathbb{W}, \mathbf{v}) \, d\mathbf{v} + \varepsilon_3 .$$

D'où finalement

$$(A\text{-}14)\quad \varepsilon(N, \infty) = \int \hat{p}(\mathbf{v}) \| \varphi(\mathbb{W}, \mathbf{v}) - \hat{G}(\mathbf{v}) \|^2 \, d\mathbf{v} + \varepsilon_4 . \, ◇$$

• Démonstration du lemme (65).

On peut trouver des résultats beaucoup plus généraux dans [4b], mais dont la complexité de la démonstration est sans comparaison à celle de ce petit lemme. Pour cette raison, nous allons le démontrer. Les éléments diagonaux de $\tilde{Z}(r)$ vérifient une relation du type

$$(A\text{-}15)\qquad z(r) \, \mu(r) \, \sigma = z(r) - z(r+1) .$$

Quant aux éléments extra-diagonaux, ils sont constants, et donc nuls si $\tilde{Z}(0) = 0$. Par conséquent, les sommes partielles

$$(A\text{-}16)\qquad S(r) = \sigma \sum_{p=1}^{r} z(p) \, \mu(p)$$

sont bornées, puisqu'elles peuvent s'écrire $S(r) = z(1) - z(r+1)$ d'après (A-15). Or, puisque $\mu(r)$ tend vers zéro, la suite $z(r)$ est de signe constant à partir d'un certain rang. Supposons par exemple $z(r) \geq 0$. Comme $\mu(r)$ est aussi positive, la suite $S(r)$ est par ailleurs croissante. Elle converge donc vers une limite finie. En conséquence, comme la série des $\mu(r)$ diverge, il est nécessaire que 0 soit valeur d'adhérence de $z(r)$ pour que $S(r)$ converge. Enfin, $z(r) = z(1) - S(r-1)$ montre que $z(r)$ est convergente. Donc $z(r)$ converge vers 0. ◇

• Démonstration du lemme (67).

Le lemme (65) aurait montré que la suite (64) converge vers zéro si les termes en $\mu(r)^2$ ne s'y trouvaient pas. Considérons de nouveau la suite des éléments diagonaux de $Z(r)$. Ils sont régis par une récurrence du type

$$(A\text{-}17) \quad z(r+1) = z(r) - \sigma\mu(r)\, z(r) + \nu g(r)\, \mu(r)\,,$$

où $\sigma$ et $\nu$ sont constantes, et où $g(r)$ est bornée. Pour la même raison que précédemment, on peut supposer $z(r) \geqslant 0$, et le même raisonnement permet de conclure. Plus précisément la somme

$$(A\text{-}18) \qquad \sum_{r=1}^{\infty} \mu(r)^2\, g(r)$$

étant finie puisque $g(r)$ est bornée, la suite $S(r)$ est croissante majorée, et donc converge. La série des $\mu(r)$ étant divergente, on en déduit que 0 est valeur d'adhérence de $z(r)$. Comme $z(r)$ converge par ailleurs, 0 est sa limite. ◇

• Démonstration du théorème V.

Il est clair d'après (63), que la convergence de la suite (64) vers zéro entraîne la convergence de la suite (61) vers $YX^-$. Il s'agit ici d'approximer la vitesse de cette convergence. les éléments diagonaux de la suite $Z(r)$ vérifient (A-17), et si $\mu(r) = 1/r$, nous avons

$$(A\text{-}19) \quad z(r+1) = z(r)[1 - \sigma/r] + \nu g(r)\, \mu(r)\,.$$

Montrons que cette suite est équivalente à $r^{-\sigma}$. On peut toujours trouver une fonction $C^1$, notée $\Omega(x)$, positive et décroissante sur $\mathbb{R}^+$, et coïncidant avec $z(r)$ sur $\mathbb{N}^*$ : $z(r) = \Omega(r)$, $r \in \mathbb{N}^*$. Cherchons à approximer $\Omega(x)$. Nous adoptons la condition

$$(A\text{-}20) \quad \Omega(x+1) - \Omega(x) = -\,\Omega(x)\,\sigma/x\,, \quad \forall x \in \mathbb{R}^+\,,$$

conforme à la définition de la suite $z(r)$. En appliquant le théorème des accroissements finis, il vient que

$$\Omega'(x) \leqslant \Omega(x+1) - \Omega(x) \leqslant \Omega'(x+1)$$

puis en tenant compte de la décroissance de $\Omega(x)$, nous avons :

$$\begin{aligned} \Omega'(x)/\Omega(x) &\leqslant -\,\sigma/x \leqslant \Omega'(x+1)/\Omega(x) \\ &\leqslant \Omega'(x+1)/\Omega(x+1)\,. \end{aligned}$$

Si nous intégrons maintenant entre 2 et $n$, nous obtenons :

$$\Omega(n)/\Omega(2) \leqslant 2^{\sigma}/n^{\sigma} \leqslant \Omega(n+1)/\Omega(3)\,, \quad \forall n \in \mathbb{N}^*\,.$$

D'où nous déduisons

$$(A\text{-}21) \quad \Omega(3)\, 2^{\sigma} \frac{1}{(n-1)^{\sigma}} \leqslant \Omega(n) \leqslant \Omega(2)\, 2^{\sigma} \frac{1}{n^{\sigma}}\,.$$

La suite $z(n)$ est donc bien équivalente à $n^{-\sigma}$. La chute vient alors d'elle-même. Soit $z(r)$ une composante diagonale quelconque, et $\lambda = \sigma^2$ la valeur propre correspondante dans la matrice $\Sigma\Sigma'$. Supposons que $r$ soit tel que $z(r)$ ait atteint l'erreur résiduelle $\varepsilon$. Alors $\varepsilon \sim c/r^{\sigma}$, ce qui équivaut à $r \sim c^{1/\sigma}\, \varepsilon^{-1/\sigma}$. Enfin, il est clair que la valeur propre imposant la convergence est la plus petite valeur propre non nulle. ◇

• Obtention du biais (70).

Puisque tous les vecteurs $\mathbf{x}(n)$ suivent la même loi, nous avons d'après (32) :

$$(A\text{-}22) \quad \mathrm{E}\left\{\hat{p}_x(\mathrm{N}, \mathbf{u})\right\} = \frac{1}{h(\mathrm{N})^{\mathrm{M}}} \int_{\mathbb{R}^{\mathrm{M}}} \mathrm{K}\left(\frac{\mathbf{u}-\mathbf{v}}{h(\mathrm{N})}\right) p_x(\mathbf{v})\, d\mathbf{v}\,.$$

Notons $\varepsilon(\mathrm{N}, \mathbf{u}) = \mathrm{E}\left\{\hat{p}_x(\mathrm{N}, \mathbf{u})\right\} - p_x(\mathbf{u})$ le biais. Après changement de variable, nous avons les écritures :

$$\varepsilon(\mathrm{N}, \mathbf{u}) = \int_{\mathbb{R}^{\mathrm{M}}} \mathrm{K}(\mathbf{z})\{p_x(\mathbf{u} - h(\mathrm{N})\,\mathbf{z}) - p_x(\mathbf{u})\}\, d\mathbf{z}$$

ou bien

$$(A\text{-}23) \quad \varepsilon(\mathrm{N}, \mathbf{u}) = \frac{1}{h(\mathrm{N})^{\mathrm{M}}} \int_{\mathbb{R}^{\mathrm{M}}} \mathrm{K}\left(\frac{\mathbf{y}}{h(\mathrm{N})}\right) p_x(\mathbf{u} - \mathbf{y})\, d\mathbf{y} - p_x(\mathbf{u})\,.$$

Comme $p_x(\mathbf{x})$ est deux fois dérivable, elle admet le développement de Taylor-Young à l'ordre deux suivant :

$$p_x(\mathbf{x} - h\mathbf{z}) = p_x(\mathbf{x}) - h \sum_{i=1}^{\mathrm{M}} z_i \frac{\partial p_x(\mathbf{x})}{\partial x_i} + \frac{h^2}{2} \sum_{i,j=1}^{\mathrm{M}} z_i\, z_j \frac{\partial^2 p_x(\mathbf{x})}{\partial x_i\, \partial x_j} + (\mathbf{x} - h\mathbf{z})^2\, \rho(h)\,.$$

En reportant ce développement dans l'espression (A-23), nous obtenons :

$$(A\text{-}24) \quad \varepsilon(\mathrm{N}, \mathbf{u}) = \frac{h^2}{2} \sum_{i,j=1}^{\mathrm{M}} \int_{\mathbb{R}^{\mathrm{M}}} \mathrm{K}(\mathbf{z})\, z_i\, z_j \frac{\partial^2 p_x(\mathbf{x})}{\partial x_i\, \partial x_j}\, d\mathbf{z} + \mathrm{O}(h^3)\,.$$

En effet, le terme d'ordre 1 a disparu car, la fonction $\mathrm{K}(\mathbf{u})$ étant radiale, tous ses moments d'ordre 1 sont nuls. L'expression du biais est finalement

$$(A\text{-}25) \quad \varepsilon(\mathrm{N}, \mathbf{x}) \overset{\mathrm{N}\to\infty}{} \frac{h^2}{2} \sum_{i,j=1}^{\mathrm{M}} v_{ij} \frac{\partial^2 p_x(\mathbf{x})}{\partial x_i\, \partial x_j} + \mathrm{O}(h^3)\,. \qquad \diamond$$

• Obtention de la variance (71).

Par ailleurs, repartant de la définition (32), nous avons pour tout N fixé

$$(A\text{-}26) \quad \mathrm{E}\left\{\hat{p}_x(\mathrm{N}, \mathbf{u})^2\right\} = \frac{1}{\mathrm{N}^2} \frac{1}{h^{2\mathrm{M}}} \sum_{i,j=1}^{\mathrm{M}} \int_{\mathbb{R}^{\mathrm{M}}} \int_{\mathbb{R}^{\mathrm{M}}} \mathrm{K}\left(\frac{\mathbf{u}-\mathbf{x}_i}{h}\right) \mathrm{K}\left(\frac{\mathbf{u}-\mathbf{x}_j}{h}\right) p_x(\mathbf{x}_i, \mathbf{x}_j)\, d\mathbf{x}_i\, d\mathbf{x}_j\,.$$

Il est nécessaire de distinguer les cas $i \neq j$ et $i = j$ afin de prendre en compte l'indépendance statistique des $x(n)$.

Après calcul, il vient que :

$$(A\text{-}27)\quad \mathrm{E}\left\{\widehat{p}_x(\mathrm{N},\mathbf{u})^2\right\} = \mathrm{E}\left\{\widehat{p}_x(\mathrm{N},\mathbf{u})\right\}^2 - \frac{1}{\mathrm{N}}\frac{1}{h^{2\mathrm{M}}}\left\{\int_{\mathbb{R}^{\mathrm{M}}} \mathrm{K}\left(\frac{\mathbf{u}-\mathbf{v}}{h}\right) p_x(\mathbf{v})\,d\mathbf{v}\right\}^2 + \frac{1}{\mathrm{N}}\frac{1}{h^{2\mathrm{M}}}\int_{\mathbb{R}^{\mathrm{M}}} \mathrm{K}^2\left(\frac{\mathbf{u}-\mathbf{v}}{h}\right) p_x(\mathbf{v})\,d\mathbf{v}\,.$$

Après le changement de variable approprié, la variance s'écrit simplement :

$$(A\text{-}28)\quad \mathrm{var}\,(\mathrm{N},\mathbf{x}) = -\frac{1}{\mathrm{N}}\left\{\int_{\mathbb{R}^{\mathrm{M}}} \mathrm{K}(\mathbf{z})\,p_x(\mathbf{x}-h\mathbf{z})\,d\mathbf{z}\right\}^2 + \frac{1}{\mathrm{N}}\frac{1}{h^{\mathrm{M}}}\int_{\mathbb{R}^{\mathrm{M}}} \mathrm{K}^2(\mathbf{z})\,p_x(\mathbf{x}-h\mathbf{z})\,d\mathbf{z}\,.$$

Lorsque $h(\mathrm{N})$tend vers zéro, le premier terme est dominé par le second, puisque les deux intégrales sont bornées. En remplaçant $p_x(\mathbf{x}-h\mathbf{z})$ par son développement de Taylor-Young à l'ordre 1 dans le second terme nous obtenons finalement :

$$(A\text{-}29)\quad \mathrm{var}\,(\mathrm{N},\mathbf{x}) = \frac{1}{\mathrm{N}}\frac{1}{h^{\mathrm{M}}}p_x(\mathbf{x})\int_{\mathbb{R}^{\mathrm{M}}} \mathrm{K}^2(\mathbf{z})\,d\mathbf{z} + \mathrm{O}\left(\frac{1}{\mathrm{N}}\frac{1}{h^{\mathrm{M}-1}}\right)\,. \quad \diamond$$

• Démonstration de (74)

La relation (72) montre qu'il est nécessaire que $h(\mathrm{N})$ tende vers zéro et que $\mathrm{N}h(\mathrm{N})$ tende vers l'infini lorsque N tend vers l'infini, pour que l'erreur $e(\mathrm{N})$ tende vers zéro. Nous nous plaçons donc dans ces conditions, et procédons de la même manière que dans [7]. Cherchons les valeurs de $e(\mathrm{N})$ stationnaires par rapport à $h$ en dérivant (72). Nous avons

$$(A\text{-}30)\quad h(\mathrm{N})^3\int_{\mathbb{R}^{\mathrm{M}}} \mathrm{J}(\mathbf{x})^2\,d\mathbf{x} - \frac{1}{\mathrm{N}}\frac{\mathrm{M}}{h^{\mathrm{M}+1}}\int_{\mathbb{R}^{\mathrm{M}}} \mathrm{K}^2(\mathbf{z})\,d\mathbf{z} = 0\,.$$

Il est facile de vérifier que le seul réel positif satisfaisant cette équation est celui défini par (74). Or, d'après (72), $e(\mathrm{N})$ est infinie en $h = 0$ et en $h = \infty$. Cet unique point stationnaire admissible est donc bien un minimum. $\diamond$

• Démonstration de (79).

Dans notre cas particulier où $p(\mathbf{x}/\mathbf{x} \in \mathrm{E}(p))$ est une gaussienne centrée en $\mathbf{w}(p)$ et de variance $\sigma(p)^2\,\mathrm{I}$, nous avons en utilisant (73) et (74) :

$$(A\text{-}31)\quad h(a(p))_{\mathrm{opt}}^{\mathrm{M}+4} = \frac{\mathrm{M}}{a(p)}\,\frac{\displaystyle\int_{\mathbb{R}^{\mathrm{M}}} \mathrm{K}^2(\mathbf{z})\,d\mathbf{z}}{\displaystyle\int_{\mathbb{R}^{\mathrm{M}}} \Delta p\,(\mathbf{x}/\mathbf{x} \in \mathrm{E}\,(p))^2\,d\mathbf{x}}\,,$$

où $\Delta$ désigne l'opérateur laplacien. Pour conclure, il est plus clair de montrer les trois lemmes suivants.

**Lemme**

Soit K le noyau défini par $\mathrm{K}(\mathbf{u}) = (2\,\pi)^{-\mathrm{M}/2}\exp(-\,\|\mathbf{u}\|^2/2)$. Alors

$$(A\text{-}32)\quad \int \ldots \int [\Delta \mathrm{K}(\mathbf{u})]^2\,d\mathbf{u} = \frac{3\,\mathrm{M}}{4}\,(2\,\sqrt{\pi})^{-\mathrm{M}}\,,$$

où $\Delta\mathrm{K}$ désigne le laplacien de K.

En effet, par définition de $\mathrm{K}(\mathbf{u})$, il vient que

$$\partial \mathrm{K}/\partial u_i(\mathbf{u}) = -\,u_i\,\mathrm{K}(\mathbf{u})$$
$$\partial^2 \mathrm{K}/\partial u_i^2(\mathbf{u}) = (u_i^2 - 1)\,\mathrm{K}(\mathbf{u})\,.$$

L'intégrale (A-32) s'écrit donc

$$\Omega = (2\,\pi)^{-\mathrm{M}}\int \ldots \int (\Sigma u_i^2 - \mathrm{M})^2 \exp(-\,\|\mathbf{u}\|^2)\,d\mathbf{u}\,,$$

En posant le changement de variable $\mathbf{v} = \sqrt{2}\,\mathbf{u}$, on obtient

$$\Omega = 2^{-\mathrm{M}}\,\pi^{-\mathrm{M}/2}\int \ldots \int (\Sigma v_i^2/2 - \mathrm{M})^2\,\mathrm{K}(\mathbf{v})\,d\mathbf{v}\,.$$

En remarquant enfin que les $v_i$ sont indépendantes d'une part, et que les moments de K d'ordre 4 et 2 valent respectivement 3 et 1 d'autre part, on en déduit le résultat escompté.

**Lemme**

$$(A\text{-}33)\quad \int \ldots \int [\Delta p\,(\mathbf{x})]^2\,d\mathbf{x} = \sigma^{-\mathrm{M}-4}\,\frac{3\,\mathrm{M}}{4}\,(2\,\sqrt{\pi})^{-\mathrm{M}}\,.$$

Démontrons ce deuxième lemme. Nous avons d'après les définitions de $p(\mathbf{u})$ et de $\mathrm{K}(\mathbf{u})$

$$p(\mathbf{x}) = \sigma^{-\mathrm{M}}\,\mathrm{K}(\mathbf{x}/\sigma)\,,$$

donc aussi

$$\partial p/\partial x_i(\mathbf{x}) = -\,\sigma^{-\mathrm{M}-1}\,x_i\,\partial \mathrm{K}/\partial u_i(\mathbf{x}/\sigma)$$
$$\partial^2 p/\partial x_i^2(\mathbf{x}) = \sigma^{-\mathrm{M}-2}(x_i^2 - 1)\,\partial^2 \mathrm{K}/\partial u_i^2(\mathbf{x}/\sigma)\,.$$

Il vient par conséquent que

$$(A\text{-}34)\qquad \Delta p\,(\mathbf{x}) = \sigma^{-\mathrm{M}-2}\,\Delta \mathrm{K}(\mathbf{x}/\sigma)\,.$$

Il suffit alors de faire le changement de variable $\mathbf{x} = \sigma\mathbf{y}$ pour pouvoir conclure.

Par le même raisonnement, on montrerait aussi que :

**Lemme**

$$(A\text{-}35)\qquad \int \ldots \int \mathrm{K}(\mathbf{u})^2\,\mathrm{d}\mathbf{u} = (2\,\sqrt{\pi})^{-\mathrm{M}}\,.$$

Il suffit à présent de remplacer dans (90) les intégrales par leur valeur pour obtenir :

$$(A\text{-}36)\qquad h(a(p))_{\mathrm{opt}}^{\mathrm{M}+4} = \frac{\mathrm{M}}{a(p)}\,\frac{4}{3\,\mathrm{M}\sigma(p)^{-\mathrm{M}-4}}$$

d'où (79) découle. $\diamond$