QUATORZIEME COLLOQUE GRETSI - JUAN-LES-PINS - DU 13 AU 16 SEPTEMBRE 1993

APPLICATION DES RESEAUX DE NEURONES A L'ESTIMATION PARAMETRIQUE POUR UN MULTICAPTEUR

Pascale Hirschauer, Pascal Larzabal, Henri Clergeot

LESiR-ENS. Cachan URA CNRS D1375 Tel: 47 40 27 09 Fax: 47 40 21 99
61, av. du Président Wilson 94235 Cachan Cedex France

RÉSUMÉ

Les réseaux statiques de neurones à couches, du type MLP, sont capables d'effectuer des fonctions logiques, de partitionner l'espace d'entrée en vue d'un problème de classification, et d'approximer des fonctions non linéaires. L'objet de cet article est d'étudier leurs comportements et leurs performances lorsqu'ils sont utilisés pour effectuer une estimation paramétrique sans modèle explicite (estimation supervisée). L'accent sera mis plus particulièrement sur l'amélioration introduite par l'utilisation conjointe d'une rétropropagation par gradients conjugués et d'une initialisation performante des poids. Ces résultats sont ensuite appliqués à l'extraction de paramètres à partir de signaux réels issus d'un multicapteur.

ABSTRACT

The static network as multilayer perceptron (MLP) are able to implement boolean logic functions, to partition the pattern space for classification problems and to approximate nonlinear functions. The aim of this paper is to study their capabilities when they are used to effect a parametric estimation without explicit model (supervised estimation). This paper focuses on the combinated use of conjugate gradient retropropagation and performant initial values of the adaptive weights. These results are then applied to the extraction of parameters from real multisensor signals.

1. INTRODUCTION

L'interprétation des informations délivrées par un multicapteur est actuellement réalisée par des méthodes classiques de traitement du signal [1]. Nous nous sommes attachés à résoudre ce problème à l'aide d'un réseau de neurones en vue d'une estimation paramétrique sans modèle explicite. Le réseau réalisera ainsi une fonction non linéaire Φ qui relie les paramètres recherchés à l'observation.

Il n'est plus besoin de démontrer les possibilités des réseaux de neurones à couches. Toutefois, certains aspects d'ordre pratique subsistent. Il s'agit de la taille du réseau (nombre de couches cachées, et nombre de neurones par couches), du prétraitement des données, de la durée et de la technique retenue pour l'apprentissage, de l'initialisation des poids, de l'algorithme de rétropropagation, du critère d'arrêt de l'apprentissage, des capacités d'interpolation et de généralisation. La conception "optimale" d'un estimateur paramétrique neuronal nécessite une méthodologie globale qui tient compte de tous ces problèmes forts délicats. Un tel travail dépasse, bien entendu, largement le cadre de cet article.

Nous choisissons d'utiliser un réseau à une seule couche cachée, ses capacités "d'universalité" ayant été largement démontrées et admises.

Nous décrivons l'application traitée dans le second paragraphe. Elle servira de support à l'étude que nous avons menée sur la conception d'un estimateur paramétrique neuronal. Les algorithmes de rétropropagation du gradient et du gradient conjugué sont comparés dans le paragraphe 3. L'influence du préconditionnement des données sur la vitesse d'apprentissage est examinée dans le paragraphe 4. Afin d'accélérer la convergence et d'éviter les minima locaux, il faut tenir compte de la spécificité de l'application pour l'initialisation des poids. Dans cette optique, une initialisation performante est proposée dans le paragraphe 5, puis comparée aux techniques classiques (aléatoires et Nguyen-Widrow). Les performances de cet estimateur sont présentées dans le paragraphe 6.

2. POSITION DU PROBLEME

A partir de signaux réels issus d'un multicapteur à courant de Foucault, donnant une image très imparfaite du profil d'une pièce en V, le réseau doit retrouver ses paramètres caractéristiques. Le multicapteur possède 16 capteurs et la pièce testée est définie par les 4 paramètres suivants : d (distance entre la pièce et le capteur), l (largeur du V), x (décentrement entre la pièce et le capteur) et a (inclinaison). Soit $\mathbf{p}$ le vecteur de paramètres $\mathbf{p} = [d, l, x, a]^t$.

Par exemple, les résultats obtenus pour un profil placé à 4 distances différentes sont reportés à la figure 1.

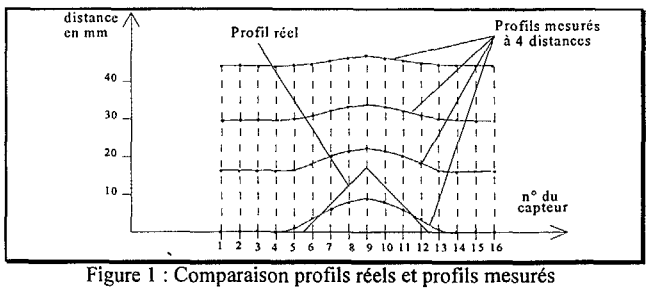


Figure 1 : Comparaison profils réels et profils mesurés par le multicapteur pour x=0 et a=0.

Les profils mesurés sont "adoucis" par rapport à la réalité. Ceci provient, entre autre, de l'épanouissement des lignes de champ

et du couplage important entre capteurs élémentaires. Le traitement efficace d'un tel problème nécessite souvent l'exploitation laborieuse d'un algorithme itératif de reconstruction d'image [1]. L'utilisation d'un réseau de neurones en vue d'une telle reconstruction devenait, dès lors, très séduisante.
Le réseau utilisé comporte donc 16 entrées et 4 sorties. Le synoptique de l'apprentissage est explicité figure 2.

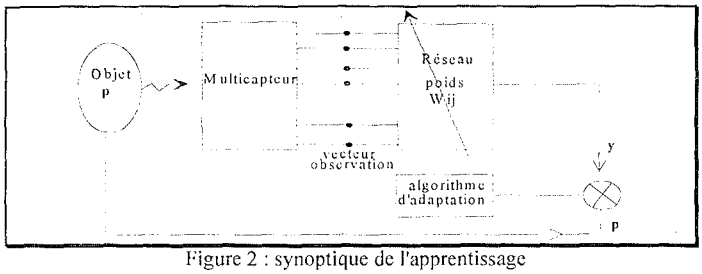


Figure 2 : synoptique de l'apprentissage

Les différentes bases d'apprentissage obtenues à partir de cette application serviront de support dans les paragraphes suivants.

3. ALGORITHMES DE RETROPROPAGATION

Le critère d'apprentissage utilisé est la minimisation de l'erreur quadratique moyenne entre les N vecteurs de la base d'apprentissage et les sorties du réseau à l'itération k :

$$\xi_k = \frac{1}{N}.\sum_{i=1}^{N}\left(\mathbf{p}^i - \mathbf{y}_k^i\right)^2 = \frac{1}{N}.\|\mathbf{P} - \mathbf{Y}_k\|^2 \quad (1)$$

3.1 Algorithmes du premier ordre

Les techniques d'apprentissage par rétropropagation, décrites par Widrow [2] et Rumelhart, utilisent un développement de l'erreur à l'ordre 1 :

$$\xi(\mathbf{w}_{k+1}) = \xi(\mathbf{w}_k + \Delta\mathbf{w}) = \xi(\mathbf{w}_k) + (\mathbf{g}_k)^t.\Delta\mathbf{w} + o\|\Delta\mathbf{w}\| \quad (2)$$

où g_k est le gradient de l'erreur quadratique moyenne.
On obtient ainsi *l'algorithme du gradient* ou "algorithme de descente suivant la ligne de plus grande pente":

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \lambda.\mathbf{g}_k \quad (3)$$

Si le pas λ est fixé à une valeur constante, la qualité de la convergence est mauvaise. Il faut envisager un pas variable dont la valeur peut être obtenue par diverses techniques de relaxation. Le pas est calculé à chaque itération soit en utilisant une procédure de recherche unidimensionnelle (on peut supposer que l'erreur est une fonction quadratique du pas), soit en l'adaptant en fonction des variations de l'erreur (augmentation ou diminution).
Les directions de recherche étant perpendiculaires aux isocritères, la convergence de l'algorithme sera d'autant plus rapide que les isocritères seront sphériques (figure 3).

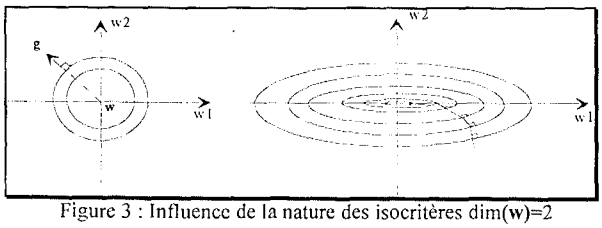


Figure 3 : Influence de la nature des isocritères dim(w)=2

La topologie des surfaces d'erreur étant complexe [2], le gradient converge lentement. Seule une initialisation adéquate évitera de converger vers un minimum local.

Le gradient converge très mal en phase finale. L'exploitation de la courbure de la fonction ξ améliore énormément la qualité de cette convergence en terme de précision et de vitesse. Il devient dès lors intéressant de développer des algorithmes du second ordre dont la particularité est justement d'exploiter cette courbure.

3.2 Algorithmes du second ordre

On utilise un développement à l'ordre 2 du critère :

$$\xi(\mathbf{w}_{k+1}) = \xi(\mathbf{w}_k) + (\mathbf{g}_k)^t.\Delta\mathbf{w} + \frac{1}{2}\Delta\mathbf{w}^t.\mathbf{H}_k.\Delta\mathbf{w} + o\|\Delta\mathbf{w}\|^2 \quad (4)$$

où $\mathbf{H}_k$ est le Hessien défini par $\frac{\partial^2\xi}{\partial\mathbf{w}\partial\mathbf{w}^t}$.
Si le Hessien est inversible, *l'algorithme de Newton* s'écrit :

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \mathbf{H}_k^{-1}.\mathbf{g}_k \quad (5)$$

L'inconvénient majeur de cette méthode est qu'elle nécessite le calcul du Hessien et son inversion. Malgré ses performances très séduisantes, cette méthode est rarement utilisée à cause de sa complexité numérique.
Plusieurs méthodes, telles que quasi-Newton ou les gradients conjugués, permettent de combiner la simplicité du gradient et les performances de Newton : la courbure de la fonction est prise en compte sans calculer le Hessien [3]. Là encore, on suppose que l'erreur est une fonction quadratique des poids.

L'algorithme *des gradients conjugués*, que nous avons choisi de développer pour sa plus grande simplicité, est décrit ci-après. A l'itération k, un certain nombre de gradients $\mathbf{g}_1,\mathbf{g}_2,...,\mathbf{g}_k$ et un certain nombre de directions de recherche $\mathbf{d}_1,\mathbf{d}_2,..,\mathbf{d}_{k-1}$ ont été calculés. Appelons V_k le sous-espace engendré par ces vecteurs. La méthode consiste alors à chercher $\mathbf{w}_{k+1}$ minimisant le critère ξ dans le sous-espace passant par $\mathbf{w}_k$ et parallèle à V_k. Ceci est bien meilleur que de minimiser le critère dans une variété de dimension 1 (cas du gradient). On a donc :

$$\mathbf{w}_{k+1}(\lambda_k) = \mathbf{w}_k + \lambda_k.\mathbf{d}_k \quad \text{où} \quad \mathbf{d}_k \in V_k \quad (6)$$

Les gradients sont ainsi orthogonaux les uns par rapport aux autres comme le montre la figure 4 :

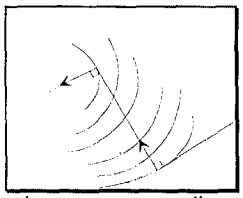


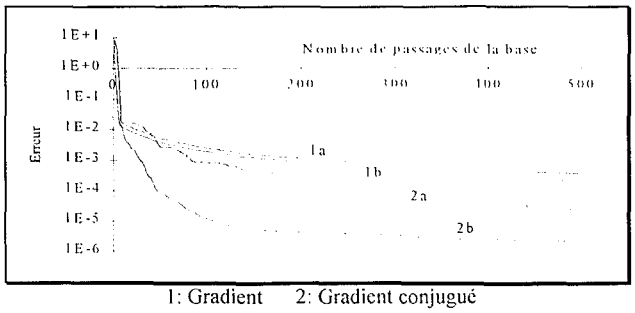
Figure 4 : convergence en dimension 2

$\mathbf{d}_k$ est donnée par la méthode de Newton. Les directions de recherche successives sont alors conjuguées par rapport aux Hessien. La suite $\mathbf{w}_k$ peut être obtenue sans connaître le Hessien du critère. L'algorithme est alors connu sous le nom de *Polak-Ribière* :
a) On choisit $\mathbf{w}_0$ puis on calcule $\mathbf{d}_0 = -\mathbf{g}_0$.
b) On calcule $\mathbf{w}_{k+1} = \mathbf{w}_k + \lambda_k.\mathbf{d}_k$ où λ_k est choisi de façon à minimiser $\xi(\mathbf{w}_k + \lambda_k.\mathbf{d}_k)$
c) La direction suivante est donnée par

$$\mathbf{d}_{k+1} = -\mathbf{g}_{k+1} + \beta_k.\mathbf{d}_k \text{ avec } \beta_k = \frac{\mathbf{g}_{k+1}^t.(\mathbf{g}_{k+1} - \mathbf{g}_k)}{\mathbf{g}_k^t.\mathbf{g}_k} \quad (7)$$

La méthode des gradients conjugués repose sur le fait que les λ_k sont déterminés avec exactitude. Or, cette recherche de minimum est effectuée avec les techniques de relaxation préalablement exposées. Le pas peut être adapté suivant les

performances effectives ou être calculé par recherche unidimensionnelle par approximation quadratique. Les valeurs λ_k ainsi obtenues ne sont donc que des estimations de λ_{kopt} .
Ces différents algorithmes ont été appliqués à l'estimation paramétrique du multicapteur. Les résultats après 500 passages de la base d'apprentissage (N=100) sont reportés à la figure 5. Notre objectif, dans ce paragraphe, n'est pas de trouver le nombre optimal de neurones cachés, mais de comparer les algorithmes. Nous avons vérifié que la hiérarchie des performances ne dépendait pas de ce nombre. L'initialisation des poids est identique pour les 4 algorithmes.



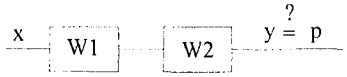
1: Gradient 2: Gradient conjugué
a: pas adapté suivant les performances
b: recherche unidimensionnelle du pas par approximation quadratique
Figure 5 : comparaison des méthodes d'apprentissage

Deux résultats importants apparaissent. La relaxation par approximation quadratique est la plus efficace. Cette différence est très accentuée dans la méthode du gradient conjugué qui nécessite un pas λ précis. La méthode des gradients conjugués est plus performante que les méthodes de rétropropagation classiques. Au vu de ces résultats, nous utiliserons pour les simulations suivantes l'algorithme de rétropropagation par gradients conjugués, le pas étant calculé par approximation quadratique de l'erreur.

4. PRECONDITIONNEMENT DES DONNEES

4.1 Normalisation

Afin d'éviter un problème de saturation du réseau, il est nécessaire que l'ordre de grandeur des poids des deux couches soit semblable. En effet, les poids de la couche d'entrée évoluent proportionnellement à ceux de la couche de sortie et réciproquement. Illustrons ceci par un exemple :



l'erreur vaut $\xi = (p-y)^2$, soit: $\frac{\partial \xi}{\partial w1} = -2.(p-y).w2.x$

de même $\frac{\partial \xi}{\partial w2} = -2.(p-y).w1.x$

Donc, en cas de disproportionnalité entre w1 et w2, les neurones saturent rapidement. Pour éviter ce problème, les vecteurs d'entrées et les vecteurs désirés seront *normalisés.*

4.2 Centrage

Afin d'augmenter la rapidité de la convergence, on *centre les signaux*. A l'initialisation, les neurones fonctionnent dans la zone linéaire de la sigmoïde. Les poids s'adaptent ensuite pour obtenir la fonction non linéaire Φ souhaitée par l'application. Les différences de convergence sont reportées à la figure 6.

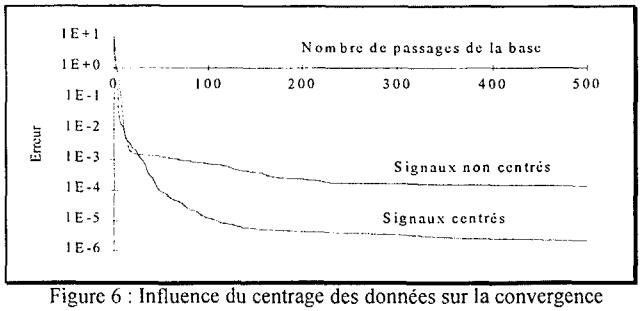


Figure 6 : Influence du centrage des données sur la convergence

5. INITIALISATION DES POIDS

L'initialisation des poids est cruciale : si le réseau est mal initialisé, il converge vers un minimum local. L'initialisation aléatoire classique paraît être une très mauvaise solution. Une amélioration sensible peut être obtenue si l'on tient compte du problème traité. Dans cette optique, deux initialisations vont être décrites.
Afin d'accélérer la convergence, Nguyen et Widrow [4] ont proposé une initialisation originale. Un réseau à couches étant capable d'approximer toute fonction non linéaire, chaque neurone de la couche cachée est responsable de l'approximation d'une partie de la réponse souhaitée (le nombre de neurones cachés dépend ainsi de la complexité du problème et du degré d'approximation souhaité). Leur initialisation est fondée sur les remarques précédentes.
Une autre initialisation des poids peut être obtenue en cherchant par décomposition en valeurs singulières la solution explicite du problème d'estimation. La méthode que nous proposons s'appuie sur les travaux de H. Boulard et Y. Kamp [5], qu'ils ont développés dans le cas d'un problème d'auto-association.
La structure du réseau ainsi que les notations utilisées sont données dans la figure 7 :

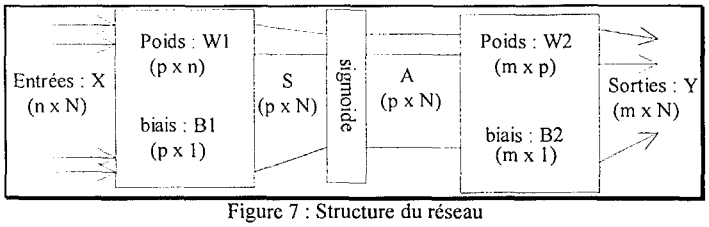


Figure 7 : Structure du réseau

Le réseau est donc défini par les équations suivantes où **u** est un vecteur identité (Nx1) et f la fonction d'activation :

$$\begin{cases} \mathbf{S} = \mathbf{W1}.\mathbf{X} + \mathbf{b1}.\mathbf{u}^t \\ \mathbf{A} = f(\mathbf{S}) \\ \mathbf{Y} = \mathbf{W2}.\mathbf{A} + \mathbf{b2}.\mathbf{u}^t \end{cases} \quad (8)$$

Lors de l'apprentissage, les poids et les biais vont évoluer de façon à minimiser l'erreur quadratique moyenne :

$$\xi = \frac{1}{N}.\|\mathbf{P}-\mathbf{Y}\|^2 = \frac{1}{N}.\|\mathbf{P}-\mathbf{W2}.\mathbf{A}+\mathbf{b2}.\mathbf{u}^t\|^2 \quad (9)$$

La démarche consiste ensuite à chercher la solution optimale à ce problème.

- Le vecteur biais $\hat{\mathbf{b}}2$ est obtenu en dérivant l'erreur par rapport à **b**2, on obtient :

$$\hat{\mathbf{b}}2 = \frac{1}{N}(\mathbf{P}-\mathbf{W2}.\mathbf{A}).\mathbf{u} \quad (10)$$

Le critère devient alors :

$$\xi = \frac{1}{N}.\|\mathbf{P}'-\mathbf{W2}.\mathbf{A}'\|^2$$

où $\mathbf{P}' = \mathbf{P}.(\mathbf{I}-\mathbf{u}.\mathbf{u}^t/N)$ et $\mathbf{A}' = \mathbf{A}.(\mathbf{I}-\mathbf{u}.\mathbf{u}^t/N)$ (11)

- Si $p \leq m$, le produit $\hat{\mathbf{W}}2.\hat{\mathbf{A}}'$ minimisant ξ est la meilleure approximation de rang p de $\mathbf{P}'$. La solution est obtenue par décomposition de $\mathbf{P}'$ en valeurs singulières :

$$\hat{\mathbf{W}}2.\hat{\mathbf{A}}' = \mathbf{U}_p.\Sigma_p.\mathbf{V}_p^t \qquad (12)$$

Une décomposition possible peut être :

$$\hat{\mathbf{W}}2 = \mathbf{U}_p.\mathbf{T}^{-1} \qquad \hat{\mathbf{A}}' = \mathbf{T}.\Sigma_p.\mathbf{V}_p^t \qquad (13)$$

où $\mathbf{T}$ est une matrice (pxp) arbitraire non singulière jouant le rôle d'une matrice d'échelle, afin que les ordres de grandeur des poids $\mathbf{W}1$ et $\mathbf{W}2$ soient identiques.

- L'obtention de $\hat{\mathbf{W}}1$ et $\hat{\mathbf{b}}1$ dépend de la fonction d'activation. On suppose pour l'initialisation des poids, soit que f = Identité, soit qu'on travaille dans la partie linéaire de la caractéristique ($f(x) \approx \alpha + \beta.x$ avec pour notre exemple $\alpha = 0$ et $\beta = 1$).

En combinant les équations (8), (11) et (13), on obtient :

$$\hat{\mathbf{A}}' = \mathbf{T}.\Sigma_p.\mathbf{V}_p^t = \mathbf{W}1.\mathbf{X}' + \mathbf{b}1.\mathbf{u}^t.(\mathbf{I} - \mathbf{u}.\mathbf{u}^t / N) \qquad (14)$$

Du fait que $\mathbf{u}^t.\mathbf{u} = N$, $\mathbf{b}1$ peut être choisi de manière arbitraire, on a $\mathbf{T}.\Sigma_p.\mathbf{V}_p^t = \mathbf{W}1.\mathbf{X}'$. En notant $\mathbf{X}'^{\#}$ la pseudo-inverse de $\mathbf{X}'$, on obtient :

$$\hat{\mathbf{W}}1 = \mathbf{T}.\Sigma_p.\mathbf{V}_p^t.\mathbf{X}'^{\#} \qquad (15)$$

Dans le cas d'une base d'apprentissage trop importante, pour des raisons de complexité numérique, seule une partie caractéristique de la base peut être traitée lors de l'initialisation.

Les figures suivantes représentent la convergence du réseau pour les trois initialisations décrites précédemment. Les courbes de la figure 8a sont relatives à une base d'apprentissage où seuls deux paramètres caractéristiques de la pièce testée varient. Celles de la figure 8b correspondent à une base d'apprentissage où les 4 paramètres décrivent, de façon aléatoire et uniforme, le domaine utile de fonctionnement.

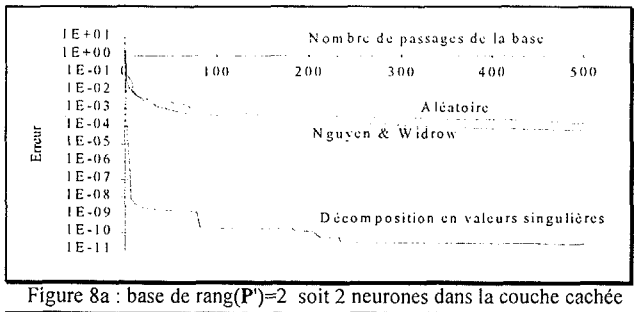


Figure 8a : base de rang($\mathbf{P}'$)=2 soit 2 neurones dans la couche cachée

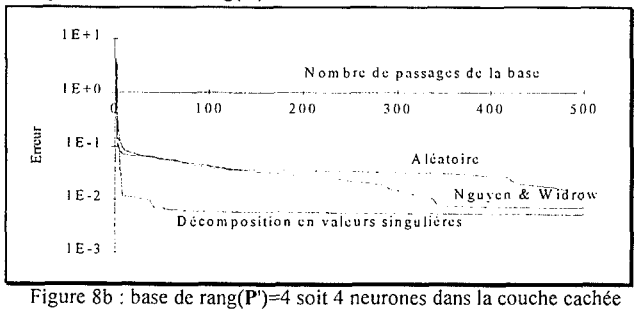


Figure 8b : base de rang($\mathbf{P}'$)=4 soit 4 neurones dans la couche cachée

L'amélioration introduite par l'initialisation par décomposition en valeurs singulières est spectaculaire.

6. PERFORMANCES DE L'ESTIMATEUR

La fonction non linéaire Φ, reliant les paramètres recherchés aux observations, est uniquement connue en un nombre fini de points constitués par les éléments de la base d'apprentissage. Le réseau doit être capable de généraliser et d'interpoler les points appris. La stratégie d'apprentissage est liée à la base d'apprentissage disponible :

-Si peu de vecteurs d'apprentissage sont disponibles, un dilemne apparait entre l'apprentissage de cette base proprement dite et les facultés de généralisation et d'interpolation du réseau. La figure 9 illustre, dans un cas extrême, la perte de généralisation résultant d'un trop long apprentissage.

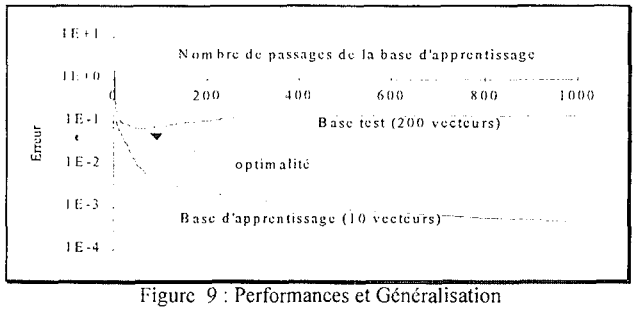


Figure 9 : Performances et Généralisation

-Si par contre, la base d'apprentissage couvre de manière exhaustive le domaine utile de variation des paramètres, la minimisation de l'erreur sur la base d'apprentissage est le critère à retenir. Le nombre de passage de cette base devra être aussi élevé que possible.

Dans notre application, l'apprentissage a pu être effectué à partir d'une base de données pour laquelle le ou les paramètre(s) recherché(s) couvre(nt), uniformément de manière aléatoire, leur domaine utile de variation. Les performances de cet estimateur, testées sur une base similaire, sont comparables à l'erreur obtenue sur la base d'apprentissage.

Les paramètres recherchés ont ainsi pu être estimés avec une précision très satisfaisante.

7. CONCLUSIONS ET PERSPECTIVES

L'application des réseaux de neurones à l'estimation paramétrique s'avère très intéressante. Nous avons montré que les performances ont été grandement améliorées par l'utilisation des gradients conjugués et sont très sensibles à l'initialisation des poids. Pour cela, une initialisation performante a été proposée, mais doit encore être améliorée. Les premiers résultats obtenus sont très encourageants et semblent très prometteurs pour la résolution de problèmes similaires.

Références

[1] P. Aknin, "Algorithmes itératifs pour l'inversion d'un modèle non linéaire de multicapteur à courants de Foucault". Thèse de Docteur ès Sciences. Orsay 1990.

[2] B. Widrow and M.A. Lehr, "30 Years of Adaptative Neural Network : Perceptron, Madaline, and Backpropagation". Proceedings of the IEEE, vol. 78, N°9 Sept. 1990.

[3] E. Walter et L. Pronzato. "Outils et Méthodes pour la construction de Modèles Paramétriques à partir de données expérimentales". LSS Polycopié Supelec n°6080. 1990.

[4] D. Nguyen and B. Widrow, " Improving the Learning Speed of 2-Layer Neural Networks by choising Initial Values of the Adaptative Weigths". Proc. Intl. Joint Conf. on Neural Networks, San Diego, CA, June 1990.

[5] H. Bourlard and Y. Kamp, "Auto-Association by Multilayer Perceptrons and Singular Value Decomposition". Biological Cybernetics 59, 291-294 (1988).