RESEAUX DE NEURONES ET ANALYSE DES CORRESPONDANCES

Ludovic Lebart

CNRS, Ecole Nationale Supérieure des Télécommunications,
46 rue Barrault, 75013, Paris

L'analyse des correspondances des tables de contingences peut être présentée de différents points de vue. Dès l'origine, les travaux de précurseurs comme Guttman [Gut41] et Hayashi [Hay56] puis le traité de Benzécri [Ben73] ont effectivement donné lieu à des présentations distinctes. On peut ainsi présenter cette méthode comme un cas particulier de décomposition aux valeurs singulières, et aussi comme une analyse discriminante particulière.

Dans le cadre des réseaux de neurones, l'analyse des correspondances des tables de contingence (ou tableaux croisés) peut être décrite comme un cas particulier de trois types de réseaux différents [Leb96].

Elle peut être considérée comme un perceptron à une couche cachée (les couches d'entrée et de sortie correspondent alors aux lignes et aux colonnes de la table de contingence) dans un contexte *supervisé* (§ 1).

Elle peut être également décrite comme un perceptron multicouche auto-associatif (cas *non-supervisé*) (§ 2). Dans ce cas ce sont les lignes (ou les colonnes) qui constituent à la fois la couche d'entrée et la couche de sortie.

Enfin, l'analyse des correspondances peut aussi être décrite comme un réseau linéaire adaptatif particulier (§ 4).

1. Un Perceptron multicouche particulier

L'équivalence entre analyse linéaire discriminante (selon Fisher [Fis40]) et perceptron à une couche cachée supervisé (lorsque les fonctions de transfert sont les fonctions

identité) a été démontrée par Gallinari *et al.* [Gal88] et généralisée au cas de modèles non-linéaires par Asoh et Otsu [Aso89].

Un cadre général (cf. par exemple Baldi et Hornik, [Bal89]) permet de traiter simultanément les cas supervisés et non-supervisés.

1.1 Le modèle

On considérera un perceptron multicouche à une couche cachée.

Soit **X** la matrice d'ordre (*n, q*) dont les *n* lignes sont les *n* observations d'un q-vecteur d'entrée, et soit **Y** la matrice d'ordre (*n, p*) contenant les *n* observations d'un *p*-vecteur de sortie.

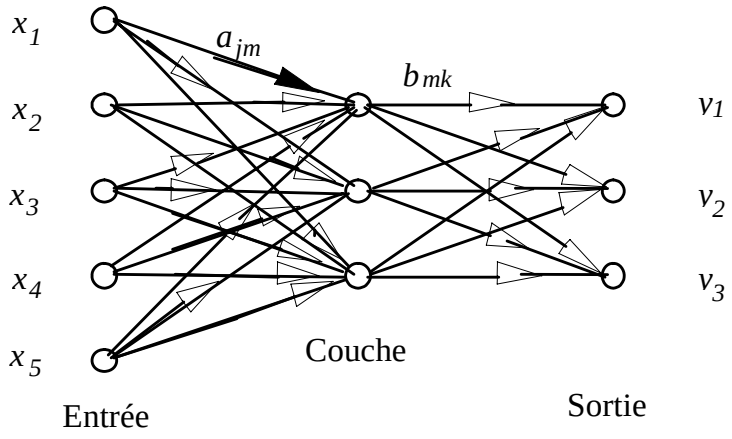

Figure 1 Perceptron à une couche cachée

A désigne la matrice d'ordre (*q, r*) des poids (cf. figure 1) en amont de la couche cachée formée de *r* éléments, et **B** la matrice de poids d'ordre (*r, p*) en aval de celle-ci.

Le modèle général s'écrit :

$$y_{ik} = \Psi\left\{ \sum_{m=1}^{r} b_{mk}\ \Phi\left(\sum_{j=1}^{q} a_{jm} x_{ij} + c_i \right) + d_k \right\} + e_{ik}$$

Nous supposerons dans la suite que les variables sont des variables numériques centrées, et que les termes constants sont nuls.

Nous supposerons également les fonctions de transfert ϕ et ψ égales à la fonction identité.

Le modèle se simplifie alors en:

$$y_{ik} = \left\{ \sum_{m=1}^{r} b_{mk} \left(\sum_{j=1}^{q} a_{jm} x_{ij} \right) \right\} + e_{ik} = \sum_{j=1}^{q} \left(\sum_{m=1}^{r} b_{mk} a_{jm} \right) x_{ij} + e_{ik} \qquad [1]$$

Les *np* équations [1] sont résumées par la notation : **Y** = **XAB**+ **E**.

Un tel système peut être résolu par la méthodes de rétro-propagation de gradient dont on trouvera des exposés dans cette même revue avec les articles de Proriol [Pro91], et de Ciampi et Lechevallier [Cil95]). Le fait de travailler avec des fonctions de transfert identité permet d'obtenir une solution analytique exacte, et des résultats connus des statisticiens.

1.2 Estimation des coefficients

Avec des notations usuelles, on estimera les coefficients inconnus en minimisant, dans un premier temps, la fonction de perte:

$$f = trace\ \mathbf{E}'\mathbf{E} = trace\ (\mathbf{Y} - \mathbf{XAB})'(\mathbf{Y} - \mathbf{XAB}),$$

sous la contrainte

$$\mathbf{BB}' = \mathbf{I}_r \quad (\mathbf{I}_r \text{ est la matrice identité d'ordre } (r,r)\ ).$$

Cette contrainte est nécessaire pour lever l'indétermination du modèle.

En effet, pour toute matrice non-singulière **H** d'ordre (*r*,*r*), **AH** et $\mathbf{H}^{-1}\mathbf{B}$ sont solutions, dès lors que **A** et **B** sont des solutions du problème de minimisation. D'autres fonctions de perte et contraintes sont possibles, comme celles présentées au paragraphe 1.3.

La dérivation de *f* conduit aux équations [2] et [3] :

$$\mathbf{BY}'\mathbf{X} = \mathbf{A}\mathbf{X}'\mathbf{X}, \qquad [2]$$

$$\mathbf{Y}'\mathbf{XA} = \mathbf{B}'\mathbf{L} \qquad [3]$$

(**L** étant une matrice d'ordre (*r*, *r*) de multiplicateurs de Lagrange).

Les équations [2] et [3], avec la contrainte précédente, conduisent aux relations :

$$\mathbf{MB}' = \mathbf{B}'\mathbf{L},$$

La matrice **M** étant définie comme:

$$\mathbf{M} = \mathbf{Y}'\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{Y},$$

On peut alors écrire le critère *f* de la façon suivante:

$$f = trace\ \mathbf{Y}'\mathbf{Y} - trace\ \mathbf{L}.$$

Il est donc équivalent de minimiser *f* ou de maximiser *trace* **L**.

On déduit aisément de ces relations que **L** est une matrice diagonale contenant les *r* plus grandes valeurs propres de **M** sur sa diagonale, les *r* lignes de **B** étant les vecteurs propres unitaires correspondants.

On peut alors calculer **A**:

$$\mathbf{A} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{Y}\mathbf{B}'.$$

Cette formule généralise celle obtenue en régression multiple simultanée (comportant plusieurs variables endogènes) lorsque la matrice des coefficients de régression est soumise à une contrainte de rang.

Notons que **P** telle que :

$$\mathbf{P} = \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'$$

est la matrice idempotente qui projette sur le sous-espace engendré par les colonnes de **X**.

On peut donc écrire : $\mathbf{M} = (\mathbf{PY})'(\mathbf{PY})$.

Ainsi, le perceptron multi-couche réalise une *analyse projetée* de **Y** (une *analyse en composantes principales projetée* si **Y** est centrée en colonnes) sur le sous espace engendré par les colonnes de **X** (cf. [Rao64], [Leb73], [Sab87], [Sab89]).

1.3 Le cas des tableaux disjonctifs

Quand **Y** et **X** sont des tableaux disjonctifs complets (tableaux binaires décrivant les deux partitions des *n* observations en *p* et *q* classes, illustrés par la figure 2), la matrice **C** telle que :

$$\mathbf{C} = \mathbf{Y}'\mathbf{X}$$

n'est autre que la table de contingence d'ordre (*p*, *q*) croisant les deux partitions, la matrice $\mathbf{D}_q$ (resp. $\mathbf{D}_p$) telle que :

$$\mathbf{D}_q = \mathbf{X}'\mathbf{X} \text{ (resp. } \mathbf{D}_p = \mathbf{Y}'\mathbf{Y})$$

est la matrice diagonale dont les *q* (resp. *p*) éléments diagonaux sont les effectifs des *q* classes (resp. *p* classes).

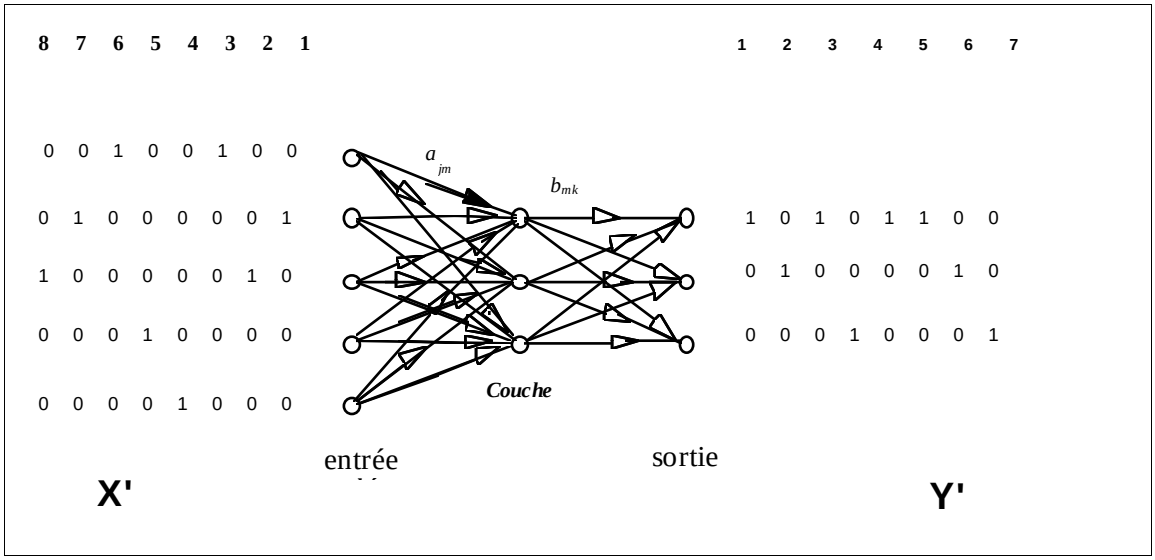

Figure 2.
Perceptron alimenté par des données disjonctives

Le perceptron multi-couche, en diagonalisant la matrice **M** telle que:

$$\mathbf{M} = \mathbf{C}\mathbf{D}_q^{-1}\mathbf{C}',$$

réalise une *analyse non-symétrique des correspondances* (Lauro et D'Ambra, [Lau84]) de la table de contingence **C**.

Ce perceptron coïncidera avec l'*analyse des correspondances* usuelle si $\mathbf{D}_p$ est une matrice scalaire (les *p* classes ont les mêmes effectifs) ou si la matrice de sortie **Y** a subit un changement d'échelle (en: $\hat{\mathbf{Y}} = \mathbf{Y}\mathbf{D}_p^{-1/2}$) au cours d'une étape préliminaire.

1.4 Perceptron et analyse canonique

En fait, le changement d'échelle précédent revient à prendre comme fonction de perte:

$$f = trace\ \mathbf{E}\ (\mathbf{Y}'\mathbf{Y})^{-1}\mathbf{E}' = trace\ (\mathbf{Y} - \mathbf{XAB})\,(\mathbf{Y}'\mathbf{Y})^{-1}\ (\mathbf{Y} - \mathbf{XAB})',$$

sous la nouvelle contrainte, pour les coefficients b_{mk} :

$$\mathbf{B}\,(\mathbf{Y}'\mathbf{Y})^{-1}\mathbf{B}' = \mathbf{I}_r \quad (\mathbf{I}_r \text{ est la matrice identité d'ordre } (r,r)).$$

Avec cette fonction de perte et ces contraintes, le perceptron (toujours dans le cas d'une fonction de transfert identité) réalise dans le cas général pour **X** et **Y** une analyse canonique.

Cette analyse, on le sait, coïncide avec une analyse discriminante de Fisher si les variables à prédire sont les indicatrices d'une partition (le tableau **Y** est alors un tableau disjonctif complet), et avec une analyse des correspondances de la table de contingence **X'Y** lorsque les prédicteurs (tableau **X**) sont aussi les indicatrices d'une partition.

Ce critère et cette contrainte supposent cependant que l'ensemble des lignes de **Y** (individus ou "exemples") sont connues a priori. Le critère initial peut être en fait utilisé "ligne à ligne" et donner lieu à une mise à jour permanente de coefficients.

2. Un perceptron multi-couche non-supervisé

Alors que les modèles supervisés (pour lesquels on dispose d'un échantillon d'apprentissage permettant d'estimer les paramètres) correspondent tout à fait à la démarche de la régression et de l'analyse discriminante, les modèles non-supervisés ou auto-organisés sont le pendant des méthodes purement exploratoires comme les méthodes factorielles descriptives. L'analyse des correspondances des tables de contingence a le privilège d'appartenir aux deux familles.

Dans les réseaux de neurones auto-associatifs, la sortie **Y** coïncide avec l'entrée **X**, une situation qui peut paraître triviale *a priori.*

En fait, ces réseaux sont d'un grand intérêt si la couche cachée est beaucoup plus étroite que les autres, réalisant une compression du signal d'entrée (figure 3).

Bourlard et Kamp [Bou88], Baldi et Hornik [Bal89] ont montré le lien entre la décomposition aux valeurs singulières (SVD) - et par conséquent l'analyse en composantes principales - et ces réseaux particuliers.

La démonstration en est immédiate si l'on remplace **Y** et **X** par leur valeur commune, notée **T**, dans les formules de la section précédente.

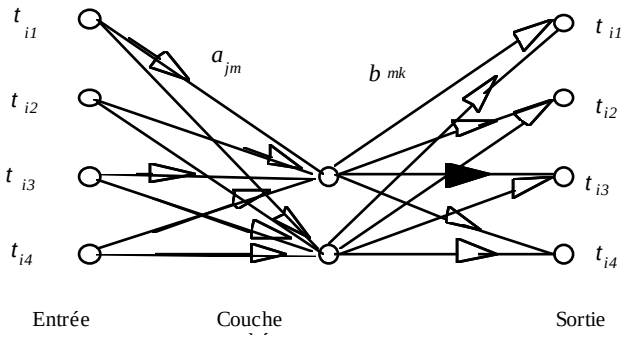

Figure 3.
Réseau auto-associatif "étranglé".

La matrice **M** précédente n'est alors autre que la matrice **T'T** des moments d'ordre deux.

$$\mathbf{M} = \mathbf{Y'X(X'X)^{-1}X'Y = T'T}$$

Dans ce cadre, l'équivalence avec l'analyse des correspondances suppose que la matrice **T** est obtenue partir d'une table de contingence **K** en opérant la transformation (avec des notations usuelles):

$$t_{ij} = \frac{k_{ij} - k_{i.}k_{.j}}{\sqrt{k_{i.}k_{.j}}} \qquad [4]$$

On notera que la nature et la taille des données d'entrée impliquées dans les deux approches des sections 1 et 2 sont radicalement différentes.

Les tableaux de données correspondent respectivement aux premiers et derniers tableaux explicités par le tableau 1 et la figure 3.

Le réseau de la section 1 est alimenté par *n* observations individuelles : un vecteur disjonctif en entrée (une ligne de **X**), un autre vecteur disjonctif en sortie (la ligne correspondante de **Y**).

Le réseau de la section 1 ***apprend à prédire,*** pour l'observation i, sa *catégorie de sortie* à partir de la connaissance de sa *catégorie d'entrée*.

Tableau 1

Equivalences classiques entre trois analyses des correspondances

Tableau analysé	Dimension	Facteur	Valeur propre
Z = [Y, X] tableau disjonctif complet	*(p,n)* où $p = p_1 + p_2$.	$\Phi = \begin{vmatrix} \psi \\ \varphi \end{vmatrix}$	$\lambda = \frac{1+\sqrt{\mu}}{2}$
B = Z'Z Tableau de Burt	*(p,p)*	$\Phi_B = \Phi\sqrt{\lambda}$	λ^2
K=Y'X tableau de contingence	*(p1 , p2)*	ψ dans R^{p1} φ dans R^{p2}	μ

Le réseau de la section 2 est alimenté simultanément par *p* (ou *q*) catégories. Ce sont les lignes ou colonnes de **T**, issues de la transformation [4] opérée sur la table de contingence **K**=**Y'X** (cf. tableau 1 et figure 4). Il ***apprend à résumer*** l'information d'entrée.

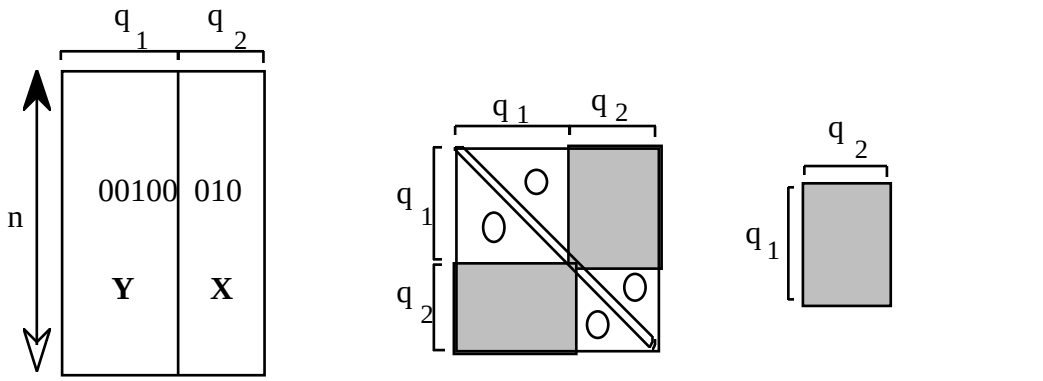

tableau disjonctif complet **Z** (réseau section 1)

tableau de Burt **Z'Z**

tableau de contingence **K= Y'X** (réseau section 2)

Figure 4

Les trois tableaux conduisant aux mêmes premières directions propres.

Notons que la section 2 (comme, on le verra, la section 4 ci-après) traite de propriétés communes à l'analyse des correspondances et à l'analyse en composantes principales, ce qui n'était pas le cas de la section 1 dévolue au cas supervisé.

3. Généralisation et mise en oeuvre

Les exposés précédents s'appuient sur un fonction de transfert "identité". Le fait d'introduire une fonction de transfert sigmoïde conduit immédiatement à des généralisations de l'analyse canonique et de l'analyse des correspondances (et aussi de l'analyse en composantes principales dans le cas auto-associatif évoqué en section 2).

Le modèle (1) prend alors la forme, dans le cas ou les deux fonctions de transfert sont égales à une même fonction ϕ:

$$y_{ik} = \phi\left\{ \sum_{m=1}^{r} b_{mk}\ \phi\left(\sum_{j=1}^{q} a_{jm} x_{ij} \right) \right\} + e_{ik} \qquad \text{[1-bis]}$$

Une forme de fonction de transfert fréquemment retenue par la simplicité des calculs qu'elle engendre est la sigmoïde (fonction d'un paramètre t, ("température") souvent pris égal à 1 ou 0.5, mais pouvant également constituer un paramètre supplémentaire à estimer)

$$\phi(x) = \frac{1 - e^{-x/t}}{1 + e^{-x/t}}$$

dont la dérivée peut s'écrire (ce qui sera économique du point de vue du calcul):

$$\phi'(x) = \frac{(1-\phi(x))(1+\phi(x))}{2t}$$

Les calculs analytiques précédents ne sont plus valables, mais ils peuvent avantageusement fournir une estimation de départ pour les algorithmes itératifs. L'algorithme de rétro-propagation de gradient [Web90], [Cil95], [Pro91] permet alors d'estimer les coefficients de ces analyses généralisées.

Une version pédagogique de cet algorithme (Fortran), reprenant les notations utilisées ici, est présentée en appendice ci-après.

Dans le cas des méthodes prédictives (régressions multiples simultanées [reduced rank regression, dans ce cas] , analyse discriminante, ces généralisations donne des résultats souvent meilleurs que les analyses plus classiques. La validation croisée est alors le critère de plus sûr d'évaluation des résultats.

Dans le cas des méthodes descriptives (composantes principales, correspondances simples et multiples), l'intérêt est peut-être moins évident, car la plupart des propriétés, et donc des règles d'interprétation, sont perdues. On doit également renoncer à la simplicité des représentations géométriques, la transparence du fonctionnement, enfin l'extrême rapidité et précision des calculs.

4 Un réseau linéaire adaptatif

Les calculs nécessités par une analyse des correspondances peuvent être réalisés par des algorithmes très variés : la méthode des moyennes réciproques (Reciprocal averaging), peu performante mais pédagogique, l'algorithme de la puissance itérée de Hotelling [Hot33] (qui revient, dans le cas de l'analyse des correspondances, à regrouper par deux les itérations de l'algorithme des moyennes réciproques), la méthode de Jacobi, de Householder, de Lanczos (cf., par exemple, pour une revue de ces derniers algorithmes : Chatelin, [Cha88]).

L'utilisation de la méthode de rétro-propagation de gradient et d'autres techniques associés aux perceptrons multi-couches permet de nouvelles approches numériques qui peuvent présenter un intérêt pour la compréhension du fonctionnement des méthodes.

Notons que l'algorithme de la puissance itérée est très voisin des algorithmes de gradient dans le cas de la recherche des extrema d'une forme quadratique, les algorithmes d'approximation stochastique dont nous allons parler en constituant la version *gradient stochastique*.

4.1 Les algorithmes d'approximation stochastique

Certains modèles non supervisés sont aussi très proches des méthodes d'approximation stochastique qui simulent le processus cognitif de lecture d'un tableau [Ben85]. Ces algorithmes peuvent traiter de grands tableaux clairsemés comme ceux que l'on rencontre en recherche documentaire ou *Automatic Information Retrieval* [Leb82].

Benzécri [Ben69], Krasulina [Kra70] ont proposés indépendamment des algorithmes d'approximation stochastique pour déterminer les plus grandes valeurs propres de l'espérance mathématique d'une matrice aléatoire. Lebart [Leb74] a donné une preuve de convergence de l'algorithme de Benzécri dans un cadre purement numérique.

Oja et Karhunen [Oja81] ont proposé des algorithmes similaires avec des démonstrations différentes, utilisant notamment les résultats de Kushner et Clark [Kus78].

Les travaux de Monnez [Mon94], Bouamaine [Bou86] concernent la convergence du modèle statistique correspondant. Une revue de nombreuses approches se trouve dans [Bou96].

La première traduction de ces algorithmes en termes de réseaux de neurones est faite par Oja [Oja82], qui a proposé depuis une large variété d'algorithmes [Oja92].

L'idée de base est la suivante : $\mathbf{X}$ étant la matrice d'ordre (n, p) des données éventuellement centrées et réduites ou transformées selon la formule [4] ci-dessus, la matrice des moments empiriques $\mathbf{X}'\mathbf{X}$ peut être écrite comme une somme de n termes $\mathbf{A}_i$.

$$X'X = \sum_i A_i \text{ , avec } A_i = x_i x_i^{'} \text{ , } \quad (x_i = i^{th} \text{ colonne de } X').$$

L'algorithme de la puissance itérée peut alors être mis en oeuvre à partir de cette matrice décomposée [Wol66] ce qui permet de tirer éventuellement avantage du caractère clairsemé du tableau $\mathbf{X}$ (notamment, dans le cas des tableaux disjonctifs complets ou des tableaux d'incidence utilisés en recherche documentaire).

Partant d'un vecteur quelconque $\mathbf{u}_0$, l'étape k de cet algorithme, après avoir posé : $\mathbf{u}_k = \mathbf{0}$, consiste à effectuer n fois l'affectation :

$$\textit{pour } i = 1 \text{ à } n \text{ , } \textit{faire : } \mathbf{u}_k \leftarrow \mathbf{u}_k + \mathbf{A}_i \mathbf{u}_{k-1} \qquad [5]$$

Le vecteur $\mathbf{u}_k$ est inchangé durant l'étape.

On peut espérer améliorer la performance de l'algorithme en modifiant $\mathbf{u}_k$ à chaque affectation, ce qui conduit au processus :

$$\textit{pour } j = 1 \text{ à } \infty \text{ , } \textit{faire : } \mathbf{u}_j \leftarrow \mathbf{u}_{j-1} + \gamma(j)\mathbf{A}_{i(j)} \mathbf{u}_{j-1} \qquad [6]$$

Dans cette formule, $\gamma(j)$ est une fonction de gain.

Pour assurer la convergence de $\mathbf{u}_j$ vers le vecteur propre de $\mathbf{X}'\mathbf{X}$ correspondant à la plus grande valeur propre, la série $\gamma(j)$ doit diverger, alors que la série $\gamma^2(j)$ doit converger.

A l'étape k, l'indice $i(j)$ de la matrice **A** prend des valeurs de 1 à n .

(à l'étape k, j varie de $(k-1)n + 1$ à kn, et $i(j) = j - (k-1)n$).

En fait, l'algorithme d'approximation stochastique [6] remplace, pour l'étape k, l'opérateur

$$\mathbf{P}_5 = \sum_i \mathbf{A}_i \quad \text{(utilisé dans l'algorithme [5]),}$$

par l'opérateur

$$\mathbf{P}_6 = \prod_j \left(I + \gamma(j)\mathbf{A}_{i(j)} \right). \quad \text{(algorithme [6])}$$

Ces opérateurs ont approximativement les mêmes vecteurs propres, mais, si $\gamma(j)$ est bien choisie, les valeurs propres du second sont plus écartées, ce qui devrait assurer une convergence plus rapide.

En ce sens, l'algorithme [6] peut être considéré comme une simple technique d'accélération de l'algorithme [5] [Leb82].

Une série usuelle $\gamma(j)$ est la série harmonique : $\gamma(j) = \frac{c}{j^d}$ avec: $d = 1$.

Durant l'itération k : $j = i + (k-1)n$ [$i=1, n$]

4.2 Principe des preuves de convergence numérique

On considérera à titre d'illustration une série constante durant l'itération k:

$$\gamma(j) = \frac{1}{m_k}$$

avec, par exemple : $m_k = kn$

Les majorations permettant d'établir la convergence de la procédure utilisent la norme suivante:

$$|\mathbf{A}| = \sup \|\mathbf{Ax}\| \quad \text{pour} \quad \|\mathbf{x}\| = 1$$

Si l'on suppose, a étant un réel positif:

$$: \quad |\mathbf{A_i}| \le a \ , \quad \|\mathbf{A}\| \le 1, \quad \text{et} \quad m_k \le \frac{1}{na}$$

Durant l'étape k, l'opérateur (2) s'écrit alors:

$$\prod_j \left(I + \gamma(j)\mathbf{A}_{i(j)}\right) = \prod_{i=1}^{n}\left(I + \frac{1}{m_k}\mathbf{A_i}\right)$$

et l'on obtient la majoration suivante:

$$\left\| \prod_{\mathbf{i=1}}^{\mathbf{n}}\left(\mathrm{I} + \frac{1}{m_k}\mathbf{A}_i\right) - e^{(n/m_k)\mathbf{A}} \right\| \leq (a^2+1)\left(\frac{n}{m_k}\right)^2$$

Des majorations bien meilleures pourraient être obtenues si l'algèbre à laquelle appartiennent les $\mathbf{A_i}$ était commutative...

On déduit de cette majoration l'inégalité suivante, qui prouve la convergence de l'opérateur associé à l'approximation stochastique vers un opérateur projection sur la première direction propre de A.

$$\left\| \prod_k \prod_{\mathbf{i=1}}^{\mathbf{n}}\left(\mathrm{I} + \frac{1}{m_k}\mathbf{A}_i\right) - e^{\Sigma(1/m_k)n\mathbf{A}} \right\| \leq 2n^2(a^2+1)\, e^{\Sigma(1/m_k)n} \sum_k \left(\frac{1}{m_k}\right)^2$$

Contrairement à l'algorithme [5], l'algorithme [6] dépend de l'ordre dans lequel sont présentées les matrices $\mathbf{A_i}$.

Ce dernier algorithme est d'ailleurs accéléré si l'ordre des matrices $\mathbf{A_i}$. est inversé lors de deux itérations successives [Leb74]. On obtient analytiquement de meilleures majorations (on remédie partiellement à la non-commutativité des $\mathbf{A_i}$) et observe empiriquement des vitesses de convergence notablement accrues.

Les réseaux linéaires adaptatifs [5] et [6] peuvent donner simultanément plusieurs vecteurs propres, si des orthonormalisations sont effectuées avec une fréquence qui dépend de la précision exigée.

Contrairement à ce que proposent beaucoup d'algorithmes dans la littérature neuromimétiticienne, il n'est nullement nécessaire (et inutilement coûteux en calcul) d'orthonormaliser à chaque pas.

Enfin, il convient de signaler que ces algorithmes convergent très lentement (comme la série harmonique diverge).

Plutôt que de les utiliser pour isoler des vecteurs propres, il vaut mieux les utiliser pour trouver un sous-espace à *s* dimension qui contienne les *r* premiers vecteurs propres ($r < s$).

On isolera alors ceux-ci en diagonalisant (par une procédure classique, susceptible de séparer des directions propres voisines) une matrice d'ordre (*s*, *s*) obtenue après projection.

Un listage complet de la procédure d'approximation stochastique (Fortran) appliquée à des variables nominales (en vue d'une analyse des correspondances multiples) se trouve dans [Leb77]. L'approximation stochastique n'y est utilisée que pour accélérer les premières itérations de la puissance itérée.

Appendice

Procédure élémentaire d'estimation des coefficients du modèle [1-bis] par rétro-propagation.

Les procédures (Fortran) présentées sont valables soit dans le cas d'une fonction de transfert égale à la fonction identique (paramètre it = 1, modèle [1] du paragraphe 1.1), soit dans le cas d'une sigmoïde (paramètre it = 0, modèle [1-bis] du paragraphe 3).

La même procédure permet donc d'apprécier les apports de l'intervention de la sigmoïde, et donc de comparer les modèles classiques aux modèles généralisés, qu'il s'agisse de décrire (analyse des correspondances ou en composantes principales) ou de prédire (régression multiple ou analyse linéaire discriminante).

Le modèle [1-bis] s'écrit:

$$y_{ik} = \phi\left\{ \sum_{m=1}^{layer(2)} b_{mk} \;\; \phi\left(\sum_{j=1}^{layer(1)} a_{jm} x_{ij} \right) \right\} + e_{ik} \qquad (i = 1,\ n;\quad k = 1,\ layer(3))$$

Pour un individu i donné, décrit par x(j), la procédure *score* calcule l'approximation (premier terme du second membre de l'équation ci-dessus) de y_{ik}, puis la procédure *retrop* corrige les estimations précédentes de a et b, par une descente de gradient stochastique (ou "on line" : la correction intervient au niveau de chaque individu).

Procédure score:

Calcul, pour i fixé, des valeurs au niveau de la couche cachée [v(2,m)] et au niveau de la couche de sortie [v(3,k)]:

$$v(2,m) = \phi\left(\sum_{j=1}^{layer(1)} a_{jm} x_{ij} \right), \qquad v(3,k) = \phi\left\{ \sum_{m=1}^{layer(2)} b_{mk} \;\; \phi\left(\sum_{j=1}^{layer(1)} a_{jm} x_{ij} \right) \right\}$$

nx = *nombre maximal de neurones dans une couche*
layer(c) = *nombre de neurones de la couche c*
x(j) = *le vecteur d'entrée pour l'individu i*
v(1,j)=x(j); v(2,m)= *valeurs couche cachée*; v(3,k)= *valeurs sortie.*

a(j,m) et b(m,k) = *estimations des coefficients au niveau (i-1)*

t = p*aramètre de la sigmoïde (fonction* sigm, *calculant phi à partir de xx)*

it =1 : *transfert=identité*; it = 0 : *transfert = sigmoïde*.

```
      subroutine score(nx,layer,x,v,a,b, t, it)
c---------------------------------------
      dimension layer(3), x(nvar), v(3,nx), a(nx,nx), b(nx,nx)
c---------------------------------------
      do j=1,layer(1)
          v(1,j)=x(j)
      enddo
      do m=1,layer(2)
           xx=0.0
               do j=1,layer(1)
                    xx=xx + v(1,j)*a(j,m)
               enddo
          phi = xx
               if (it.ne.1) call sigm(t, xx, phi)
          v(2,m)=phi
      enddo
      do k=1, layer(3)
          xx=0.0
               do m=1,layer(2)
                    xx=xx + v(2,m)*b(m,k)
               enddo
          phi = xx
               if (it.ne.1) call sigm(t, xx, phi)
          v(3,k)=phi
      enddo
      return
      end
```

Procédure retrop:

Mise à jour des coefficients a(j, m) et b (m, k) durant la lecture de l'individu i. Si *f* désigne la fonction de perte relative à l'individu "i", on a, par exemple, dans le cas de a(j, m):

$$[a(j,m)]_i = [a(j,m)]_{i-1} \quad - \quad \varepsilon \left| \frac{\partial f}{\partial a(j,m)} \right|_{i-1}$$

y(k) *est* y_{ik} *pour l'individu i.*

cora, corb, tb *sont des tableaux de travail*

eps (ε) *est le pas du gradient (e.g.* eps= *0.01)*

a *et* b *en entrée proviennent de l'étape* (i-1)

```
      subroutine retrop (nx,layer,y,v,a,b,cora,corb,tb,t,eps,it)
c---------------------------------------
      dimension y(nx),v(3,nx), tb(3,nx), a(nx,nx), b(nx,nx)
      dimension layer(3),dimension cora(nx,nx), corb(nx,nx)
```

```
c
c----------------- première phase: corrections sur  b
      do k=1,layer(3)
            phi=v(3,k)
            deriv = 1
                  if (it.ne.1) deriv=(1. + phi)*(1. - phi)/(2*t)
            tb(3,k)=eps*deriv*(y(k)-v(3,k))

                  do m=1,layer(2)
                        corb(k,m)=tb(3,k)*v(2,m)
                  enddo
      enddo
c ------------ deuxieme phase: corrections sur a
      do m=1,layer(2)
            som=0.0
                  do k=1,layer(3)
                        som=som+b(m,k)*tb(3,k)
                  enddo
            phi=v(2,m)
            deriv = 1
                  if (it.ne.1) deriv=(1. + phi)*(1. - phi)/(2*t)
            tb(2,m)=deriv*som

                  do j=1,layer(1)
                        cora(m,j)=tb(2,m)*v(1,j)
                  enddo
      enddo
c--------- troisième phase: report des corrections de a(j,m) et
de b(m,k)

      do m=1,layer(2)
            do j=1,layer(1)
                  a(j,m) = a(j,m)+ cora(m,j)
            enddo
      enddo

      do k=1,layer(3)
            do m=1,layer(2)
                  b(m,k) = b(m,k)+corb(k,m)
            enddo
      enddo
      return
      end
```

Il y a donc un appel du couple (*score, retrop*) pour chaque individu, et donc n appels par lecture complète du tableau. Quelques centaines de lectures complètes suffisent souvent pour obtenir des résultats satisfaisant.

Dans le cas d'une fonction de transfert identité, si l'on désire interpréter les coefficients, il est nécessaire de faire intervenir la contrainte sur b, sous forme d'orthonormalisation (modèle du paragraphe 1.1).

Comme en approximation stochastique, l'orthonormalisation de la matrice des coefficients b n'est pas nécessaire après chaque lecture d'individu.

Références

[Aso89] **Asoh, H. et Otsu, N**., "Nonlinear Data Analysis et Multilayer Perceptrons." *IEEE, IJCNN*-89, 2, p 411-415, 1989.

[Bal89] **Baldi P., Hornik K**., "Neural networks and principal component analysis : learning from examples without local minima". *Neural Networks*, 2, p 52-58, 1989.

[Ben69] **Benzécri**, **J.-P.**, "Approximation stochastique dans une algèbre normée non commutative." *Bull. Soc. Math. France*, 97, p 225-241, 1969.

[Ben73] **Benzécri J.-P**. *L'Analyse des Données*. Tome 1: *La Taxinomie*. Tome 2: *L'Analyse des Correspondances* (2de. éd. 1976). Dunod, Paris, 1973.

[Ben85] **Benzécri J.P**. "Intégration de l'information dans la mémoire et approximation stochastique". *Les Cahiers de l'Analyse des Données*, 10, p 495-498, 1985.

[Bou86] **Bouamaine A.** *Analyse factorielle sequentielle par approximation stochastique*. These, Université de Nancy, 1986.

[Bou96] **Bouamaine A.** *Méthodes d'approximation stochastique en analyse des données.* Thèse d'Etat, Université Mohammed 5, Rabat, 1996.

[Bok88] **Bourlard H., Kamp Y**., "Auto-association by multi-layers perceptrons and singular value decomposition". *Biological Cybernetics*, 59, p 291-294, 1988.

[Cha88] **Chatelin F.**, "*Valeurs propres de matrices*," Masson, Paris, 1988.

[CiL95] **Ciampi A., Lechevallier Y.** "Réseaux de neurones et modèles statistiques" *La revue de Modulad*, 15, p 27-46, 1995.

[Eck36] **Eckart C., Young G.** "The approximation of one matrix by another of lower rank". *Psychometrika*, l, p 211-218, 1936.

[Fis40] **Fisher R.A.** "The precision of discriminant functions". *Ann. Eugen*., 10, p 422-429, 1940.

[Gal88] **Gallinari P., Thiria S., Fogelman-Soulié F.** "Multilayer perceptrons and data analysis", *International Conference on neural Networks, IEEE,*, 1, p 391-399, 1988.

[Gut41] **Guttman L.** "The quantification of a class of attributes: a theory and method of a scale construction". In : *The prediction of personal adjustment* (Horst P., ed.) p 251 -264, SSCR New York, 1941 .

[Hay56] **Hayashi C.** " Theory and examples of quantification. (II)" *Proc. of the Institute of Statist. Math.* 4 (2), p 19-30, 1956 .

[Hor94] **Hornik, K.**, "Neural networks : more than "statistics for amateurs"." In : *COMPSTAT*, Dutter R., Grossmann W. (eds), Physica Verlag, Heidelberg, p 223-235, 1994.

[Hot33] **Hotelling H.** "Analysis of a complex of statistical variables into principal components". *J. Educ. Psy.* 24, p 417-441, p 498-520, 1933.

[Kra70] **Krasulina, T. P.** Method of stochastic approximation in the determination of the largest eigenvalue of the mathematical expectation of random matrices. *Automation et Remote Control,* Feb., p 215-221, 1970

[Kus78] **Kushner H., Clark, D.** *Stochastic approximation methods for constrained et unconstrained systems*, Springer, New York, 1978.

[Lau84] **Lauro, N. C., D'Ambra, L.** "L'Analyse non-symétrique des Cor-respondances." In : *Data Analysis et Informatics*, III, Diday et al. Ed., North-Holland, p 433-446, 1984.

[Leb73] **Lebart L., Fénelon J.P..** *Statistique et Informatique Appliquées. Dunod, Paris,* 1973.

[Leb74] **Lebart L.** "On the Benzécri's method for finding eigenvectors by stochastic approximation." *COMPSTAT, Proceedings in Computational. Statist.*, Physica verlag, Vienna, p 202-211, 1974.

[Leb77] **Lebart L., Morineau A., Tabard N.** *Techniques de la description statistique.* Dunod, Paris, 1977.

[Leb82] **Lebart L.** "Exploratory analysis of large sparse matrices with application to textual data. " *COMPSTAT, Proceedings in Computational. Statist.*, Physica Verlag, Vienna, p 67- 76, 1982.

[Leb96] **Lebart L.** Correspondence analysis, discrimination, and neural networks. in: *Data Science, Classification and Related Methods*, IFCS96, (Fifth International Conference of the International Federation of Classification Societies, Kobe, Japan), p 3-6, 1996.

[Lel92] **Lelu A.** *Modèles neuronaux pour l'analyse de données documentaires et textuelles*. Thèse Université Paris 6 (Pierre et Marie Curie), 1992.

[Mon94] **Monnez J.M.** Convergence d'un processus d'approximation stochastique en analyse factorielle. Publ. de l'ISUP, p 37-56, 1994.

[Oja82] **Oja E.** "A simplified neuron model as a principal components analyzer." *J. of Math. Biology*, 15, p 267-273, 1982.

[Oja92] **Oja E.** (1992): Principal components, minor components, et linear neural networks. *Neural Networks*, 5, p 927-935.

[Oja81] **Oja E. et Karhunen, J.**, "*On stochastic approximation of the eigenvectors et eigenvalues of the expectation of a random matrix.*" Report of the Helsinki University of Technology (Dept of Technical Physics). Otaniemi, Finland, 1981.

[Pro91] **Proriol J.** "NLP: Programme de réseau neuronal multicouche" *La Revue de Modulad,* 8, p 23-29, 1991.

[Rao64] **Rao C.R.** "The use and interpretation of principal component analysis in applied research". *Sankhya* serie A, 26, p 329-357, 1964 .

[Sab87] **Sabatier R.** "Analyse factorielle de données structurées et métriques". *Statist. et Anal. des Données*, 12, (3), p 75-96, 1987.

[Sab89] **Sabatier R., Lebreton J.-D., Chessel D.** "Principal component analysis with instrumental variables as a tool for modeling composition data". In : *Mutiway Data Analysis*, Coppi R., Bolasco S. (eds), Elsevier, Amsterdam, 1989.

[Sap90] **Saporta G.** *Probabilités, analyse des données et statistiques.* Technip, Paris, 1990.

[Wol66] **Wold H.**, "Estimation of principal components et related models by iterative least squares", in : *Multivariate Analysis*, Krishnaiah et al. (eds), Academic Press, New York, p 391-420, 1966.