Draft of the invited paper at the 1996 IFCS Conference (Kobe, Japan), published (same title) in : . In: *Data Science, Classification and Related Methods*. Hayashi C., Ohsumi N., Yajima K., Tanaka Y., Bock H.-H., Baba Y. (eds), Springer, Berlin, p 423-430, (1997).

Correspondence Analysis, Discrimination, and Neural Networks [1]

Ludovic Lebart

Centre National de la Recherche Scientifique
Ecole Nationale Supérieure des Télécommunications
46 rue Barrault, 75013, Paris, France.

Summary: Correspondence Analysis of contingency tables (CA) is closely related to a particular Supervised Multilayer Perceptron (MLP) or can be described as an Unsupervised MLP as well. The unsupervised MLP model is also linked to various types of stochastic approximation algorithms that mimic the cognition process involved in reading and comprehending a data table.

1. CA: a tool at the junction of many different methods

Correspondence Analysis of contingency tables (CA), independently discovered by various authors, can be presented from nearly as many points of views. It can be viewed, for example, as a particular case of both Linear Discriminant Analysis (LDA) (performed on dummy variables) and Singular Value Decomposition (SVD) (performed after a proper scaling of the original data). After the seminal papers of Guttman (1941), Hayashi (1956) and Benzécri (1969a), various presentations of CA can be found in the available literature (see, for instance, Lebart et al. (1984), Greenacre (1984), Gifi (1990), Benzécri (1992), Gower and Hand (1996)).

In the context of neural networks - cf. the recent reviews of this fast-growing field by Cheng and Titterington (1994), Murtagh (1994), Ripley (1994) - Correspondence Analysis is also at the meeting point of many different techniques.

It can be described as a particular *Supervised Multilayer Perceptron* (MLP, section 2) (in that case, the input and the output layers are respectively the rows and the columns of the contingency table) or as an *Unsupervised Multilayer Perceptron* (UMLP, section 3) (in such a case the input layer, and the output layer as well, could be the rows, whereas the observations - also named examples, or elements of the training set - could be the columns of the table). In both situations, the networks make use of the identity function as a transfer function. More general transfer functions might lead to interesting non-linear extensions of the method. CA can also be obtained from *Linear Adaptive Network*s (section 4), a series of methods closely related to stochastic approximation algorithms.

2. A particular supervised Multilayer Perceptron

2.1 Reminder about the Multilayer Perceptron

Equivalence between Linear Discriminant Analysis and supervised Multilayer Perceptron (when transfer functions are identity functions) has been proved by Gallinari

[1] Published in "*Data Science, Classification, and Related Method*", C. Hayashi et al, (eds), Springer, 1998, p 423-430.

et al. (1988) and generalized to the case of more general models (such as non-linear discriminant analysis) by Asoh and Otsu (1989).

A general framework (see, e.g., Baldi and Hornik (1989)) can deal simultaneously with the supervised and the unsupervised cases.

Let **X** be the (n, q) matrix whose n rows contain the n observations of an *input* q-vector, and let **Y** be the (n, p) matrix containing (as rows) the n observations of an *output* p-vector.

A designates the (q, r) matrix of weights (a_{jm}) (see fig.1) before the hidden layer, and **B** the (r, p) matrix of weights (b_{mk}) following it (r ² p and r ² q).

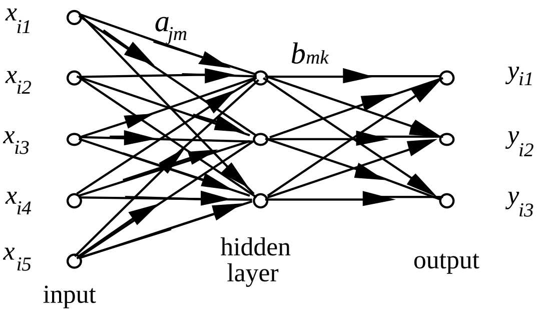

Fig. 1: Perceptron with one hidden layer (i-th observation)

A perceptron with a unique hidden layer is a model of the form:
$y = \sum ax + b$

$$y_{ik} = \Psi\left\{ \sum_{m=1}^{c} b_{mk} \; \Phi\left(\sum_{j=1}^{p} a_{jm} x_{ij} + c_m \right) + d_k \right\} + e_{ik} \qquad (1)$$

In the case of identity transfer functions (Φ and Ψ) and null constant terms, the model collapses to the simpler form:

$$y_{ik} = \left\{ \sum_{m=1}^{c} b_{mk} \left(\sum_{j=1}^{p} a_{jm} x_{ij} \right) \right\} + e_{ik} = \sum_{j=1}^{p} \left(\sum_{m=1}^{c} b_{mk} a_{jm} \right) x_{ij} + e_{ik} \quad (2)$$

2.2 Estimating the parameters

The *np* equations (2) are summarized by:

$$\mathbf{Y} = \mathbf{XAB} + \mathbf{E}. \qquad (3)$$

Denoting by $\mathbf{M}^T$ the transpose of matrix **M**, the loss function to be minimized can be written:

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

under the constraint:

$$\mathbf{B}\mathbf{B}^T = \mathbf{I}_r \quad (\mathbf{I}_r \text{ is the identity (r, r) matrix).}$$

This last constraint is introduced to remedy the indeterminacy of the model, since for any non-singular (r, r) matrix **H**, **AH** and $\mathbf{H}^{-1}\mathbf{B}$ are solutions of the minimization problem as well as **A** and **B**.

A and **B** could be estimated through a back-propagation algorithm, complemented with an orthonormalization of the rows of **B** at each step.

Since we are dealing here with the simpler case of identity transfer functions, we will focus on a direct analytical solution.

The minimization of *f* leads to equations (4) and (5):

$$\mathbf{B}\mathbf{Y}^T\mathbf{X} = \mathbf{A}^T\mathbf{X}^T\mathbf{X}, \qquad (4)$$

$$\mathbf{Y}^T\mathbf{X}\mathbf{A} = \mathbf{B}^T\mathbf{L} \qquad (5)$$

(**L** is an (r, r) matrix of Lagrange multipliers).

Equations (4) and (5), together with the previous constraint, lead to the following equation:

$$\mathbf{M}\mathbf{B}^T = \mathbf{B}^T\mathbf{L},$$

the matrix **M** being defined as:

$$\mathbf{M} = \mathbf{Y}^T\mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y} \qquad (6)$$

We get a new expression for the criterion *f*:

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

Minimizing *f* is then equivalent to maximizing *trace* **L**.

We can easily deduce from the preceding relationships and from this new criterion that **L** is a diagonal matrix containing the r largest eigenvalues of **M** as diagonal elements, the r rows of **B** being the corresponding unit eigenvectors.

We can then derive the value of **A**:

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

This formula provides a generalization of that obtained in the simultaneous multiple regression, since (3) can be written:

$$\mathbf{Y} = \mathbf{X}\mathbf{W} + \mathbf{E}, \qquad (\text{with } \mathbf{W} = \mathbf{A}\mathbf{B}).$$

This generalization concerns the new situation where the matrix of coefficients **W** undergoes a constraint of rank.

Note that:

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

is the (idempotent) projector onto the subspace spanned by the columns of **X**.

Hence :

$$\mathbf{M} = (\mathbf{P}\mathbf{Y})^T(\mathbf{P}\mathbf{Y}).$$

Thus, the Multilayer Perceptron performs a *projected principal axes analysis* of **Y**, the projection being performed onto the space spanned by the columns of **X**. This analysis is also a *projected Principal Component Analysis*, if **Y** is centered columnwise.

2.3 The case of binary disjunctive data

When **Y** and **X** are binary disjunctive tables (dummy variables describing two partitions of the n observations into p and q classes), the matrix **C** defined as:

$$\mathbf{C} = \mathbf{Y}^T\mathbf{X}$$

is the (p, q) contingency table crossing the two partitions.

The matrix $\mathbf{D}_q$ (resp. $\mathbf{D}_p$) such that:

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

is the diagonal matrix whose q (resp. p) diagonal elements are the counts of the q classes (resp. p classes).

This particular Multilayer Perceptron, whose training entails the diagonalization of the matrix **M**:

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

performs a *Non Symmetrical Correspondence Analysis* (Lauro and D'Ambra (1984)) of the contingency table **C**.

A classical Correspondence Analysis would imply a diagonalization of the matrix $\mathbf{M}^*$ such that :

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

Note that **M*** involves symmetrically the two sets (p columns of **X** on the one hand, q columns of **Y** on the other).

The Multilayer Perceptron will coincide with *Correspondence Analysis* if $\mathbf{D}_p$ is a scalar matrix (all the p classes have the same number of elements) or if the output matrix **Y** has been properly re-scaled during a preliminary step into $\hat{\mathbf{Y}}$ according to the following formula:

$$\hat{\mathbf{Y}} = \mathbf{Y}\mathbf{D}_p^{-1/2}$$

The new matrix to be diagonalized :

$$\mathbf{M}_s = \mathbf{D}_p^{-1/2}\mathbf{C}\mathbf{D}_q^{-1}\mathbf{C}'\,\mathbf{D}_p^{-1/2}$$

has the same eigenvalues as $\mathbf{M}^*$, and has eigenvectors that can be easily derived from those of $\mathbf{M}^*$.

3. An unsupervised Multilayer Perceptron

In auto-associative neural networks, the output **Y** coincides with the input **X**. The common value of **X** and **Y** is denoted by **Z**.

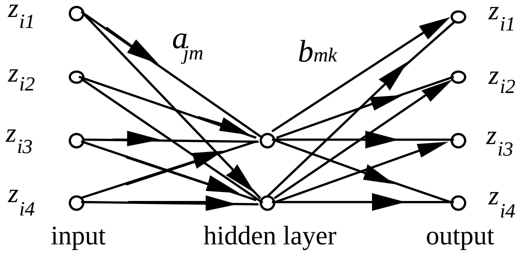

Fig. 2: Auto association strangulated network

It is an apparently trivial situation. In fact, these networks are of great interest if the hidden layer is narrower than the others, thus realizing a compression of the input signal (fig. 2).

Bourlard and Kamp (1988), Baldi and Hornik (1989) have stressed the link between SVD - and consequently Principal Component Analysis (PCA) - and these particular networks. The proof is straightforward if we replace both **Y** and **X** by **Z** in the formulas obtained in the previous section.

In this context, the matrix **M** given by the equation (6) is nothing but the product-moment matrix $\mathbf{Z}^T\mathbf{Z}$.

In this setting, the equivalence with Correspondence Analysis is obtained if **Z** is derived from a contingency table **K** according to the transformation (with usual notations):

$$z_{ij} = \frac{k_{ij} - k_{i.}k_{.j}}{\sqrt{k_{i.}k_{.j}}} \qquad (7)$$

Note that the nature and the size of the input data involved in the two approaches of section 2 and 3 are radically different.

The network of section 2 is "fed" by n individual observations. It learns how to predict the output category corresponding to observation i, from the knowledge of its input category.

The network of section 3 is fed simultaneously by q observations of p categories (rows of **Z**) or equivalently by p observations of q categories (columns of **Z**). It learns how to summarize the input information.

Note that section 3 deals with properties common to Principal Component Analysis and Correspondence Analysis.

4. A Linear Adaptive Network

4.1 Brief review of some computational techniques involved in CA

Several distinct computational algorithms could be involved in Correspondence Analysis: Reciprocal averaging, iterated power, QR and QL algorithms, Jacobi method and its generalizations, Lanczos method, as well as other classical numerical procedure for SVD, (see, for example, Parlett (1980)).

The use of Back-Propagation method and other techniques usually associated with Multilayer Perceptron provides new numerical approaches and a better insight into the method. The unsupervised MLP model is also closely related to various types of stochastic approximation algorithms that could roughly outline the cognition process involved in perusing a data table. These algorithms are able to tackle huge data sets like those encountered in Automatic Information Retrieval.

Benzécri (1969b), Krasulina (1970) have proposed independently stochastic approximation algorithms for determining the largest eigenvalues of the expectation of a random matrix. Lebart (1974) has given a numerical proof of the convergence of Benzécri algorithm, and shown its interest in the case of sparse data matrices, such as those involved in Multiple Correspondence Analysis. Oja and Karhunen (1981) have proposed similar algorithms, adding new proofs and developments, reinforced by the results of Kushner and Clark (1978). The first mention of neural networks can be found in Oja (1982), who has proposed since then a wide variety of algorithms (see: Oja (1992)).

4.2 Basics of stochastic approximation algorithms

From our point of view, the basic idea is as follows:

X being the (n,p) matrix of properly re-scaled data, the product moment matrix $\mathbf{X}^T\mathbf{X}$ can be written as a sum of n terms $\mathbf{A}_i$.

$$\mathbf{X}^T\mathbf{X} = \sum_{i=1}^{i=n} \mathbf{A}_i$$

with:

$$\mathbf{A}_i = \mathbf{x}_i\mathbf{x}_i^T \text{ ,} \quad (\mathbf{x}_i \text{ being the i}^{\text{th}} \text{ column of } \mathbf{X}^T)$$

The classical *iterated power algorithm* can then be performed using this decomposition, (cf. Wold (1966)) taking advantage of the possible sparsity of the data matrix **X**.

Starting from a random vector $\mathbf{u}_0$, the step k of this algorithm, after setting $\mathbf{u}_k = \mathbf{0}$, consists of n assignments such as:

$$\textit{for} \quad i = 1 \quad \textit{to} \quad i = n, \quad \textit{do :} \qquad \mathbf{u}_k \leftarrow \mathbf{u}_k + \mathbf{A}_i\,\mathbf{u}_{k-1} \tag{8}$$

The vector $\mathbf{u}_k$ remains unchanged during the whole step k.

We can try to improve the algorithm by modifying the estimate of $\mathbf{u}_k$ during each assignment, according to the process:

$$\textit{for} \quad j = 1 \quad \textit{to} \quad j = °, \quad \textit{do :} \qquad \mathbf{u}_j \leftarrow \mathbf{u}_{j-1} + \gamma(j)\,\mathbf{A}_{i(j)}\,\mathbf{u}_{j-1} \tag{9}$$

where $\gamma(j)$ is a gain parameter.

During each step k, the index $i(j)$ of the matrix **A** takes values *1* to *n* .

At step *k* : $i(j) = j - (k-1)n$.

To ensure the convergence of $\mathbf{u}_j$ towards the largest eigenvector of $\mathbf{X}^T\mathbf{X}$, the series $\gamma(j)$ must diverge whereas the series $\gamma^2(j)$ must converge. The series $\gamma(j)$ could be chosen among series closely related to the *harmonic series* such as: $\gamma(j) = a/(b+j)$.

In fact, during step k, the iterated power algorithm (algorithm (8)) involves the operator:

$$\sum_i \mathbf{A}_i \tag{10}$$

whereas the stochastic approximation algorithm (algorithm (9)) replaces the operator (10) with the operator:

$$\prod_j \left(I + \gamma(j)\mathbf{A}_{i(j)}\right) \tag{11}$$

4.3 Stochastic approximation *versus* iterated power

Actually, if the terms of the series*(j)* are small enough, the two operators defined by (10) and (11) have similar unit eigenvectors. However, if the terms of the series $\gamma(j)$ are not too small, the operator (11) may have more separated eigenvalues, inducing a faster convergence of algorithm (9). Therefore, there is a trade-off between two options: fast convergence towards approximate eigenvectors, or slower convergence towards the exact values.

After several steps, because of the decrease in the values of $\gamma(j)$, operator (10) is definitely superior to operator (11).

In this sense, algorithm (9) can be considered as a mere technique of acceleration of the algorithm (8) (Lebart (1982)).

Unlike the algorithm (8), (9) depends on the order of the $\mathbf{A}_i$ within the sequence ($\mathbf{A}_1$, $\mathbf{A}_2$, ..., $\mathbf{A}_i$, ..., $\mathbf{A}_n$). It can be shown that the speed of convergence can be improved if two consecutive sequences are read in reverse order (Lebart (1974)).

Both linear adaptive networks corresponding to algorithms (8) and (9) can produce simultaneously several eigenvectors, provided that orthonormalizations are carried out with a frequency that depends on the available precision. It is by no mean necessary to orthonormalize the estimates of eigenvectors at each reading (i.e. for each value of the index j when using the algorithm (9)).

It must be stressed that stochastic approximation algorithms such as algorithm (9) converge very slowly, their convergence being based on the divergence of the harmonic series. Iterated power algorithms (8) (whose firts steps could be speeded up by using stochastic approximation (9)) perform well if they confine themselve to finding a s-dimensional space V_S containing the t first eigenvectors (with: t << s). Then, the t dominant eigenvectors (and their corresponding eigenvalues) can be efficiently computed through a classical diagonalization algorithm applied to the (s, s) product-moment matrix obtained after projection onto the subspace V_S.

5. References

Asoh, H. and Otsu, N. (1989): Nonlinear Data Analysis and Multilayer Perceptrons. *IEEE, IJCNN*-89, **2**, 411-415.

Baldi, P. and Hornik, K. (1989): Neural networks and principal component analysis : learning from examples without local minima. *Neural Networks,* **2**, 52-58.

Benzécri J.-P. (1969a): Statistical analysis as a tool to make patterns emerge from clouds. In : *Methodology of Pattern Recognition,* S.Watanabe, (ed.) Academic Press, 35-74.

Benzécri, J.-P. (1969b): Approximation stochastique dans une alg bre normée non commutative. *Bull. Soc. Math. France,* **97**, 225-241.

Benzécri J.-P. (1992): *Correspondence Analysis Handbook.* Marcel Dekker, New York.

Bourlard, H. and Kamp, Y. (1988): Auto-association by Multilayers perceptrons and singular value decomposition. *Biological Cybernetics,* **59**, 291-294.

Cheng, B. and Titterington, D.M. (1994): Neural networks: a review from a statistical perspective. *Statistical Science,* **9**, 2-54.

Gallinari, P., Thiria, S. and Fogelman-Soulie, F. (1988): Multilayers perceptrons and data analysis, *International Conference on neural Networks,* IEEE,, **1**, 391-399.

Gifi A. (1990): *Non Linear Multivariate Analysis,* J. Wiley, Chichester.

Greenacre M. (1984): *Theory and Applications of Correspondence Analysis.* Academic Press, London.

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

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

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

Krasulina, T. P. (1970): Method of stochastic approximation in the determination of the largest eigenvalue of the mathematical expectation of random matrices. *Automation and Remote Control*, Feb., 215-221.

Kushner, H. and Clark, D. (1978): *Stochastic approximation methods for constrained and unconstrained systems*, Springer, New York.

Lauro, N. C and D'Ambra, L. (1984): L'Analyse non-symètrique des Correspondances. In : *Data Analysis and Informatics*, III, Diday et al. (eds.), North-Holland, 433-446.

Lebart, L. (1974): On the Benzécri's method for finding eigenvectors by stochastic approximation. *COMPSTAT, Proceedings in Computational. Statist.*, Physica verlag, Vienna, 202-211.

Lebart, L. (1982): Exploratory analysis of large sparse matrices with application to textual data. *COMPSTAT, Proceedings in Computational. Statist.*, Physica Verlag, Vienna, 67- 76.

Lebart L., Morineau A., Warwick K. (1984): *Multivariate Descriptive Statistical Analysis.* J. Wiley, New York.

Murtagh, F. (1994): Neural network and related massively parallel methods of statistics: a short overview. *International Statistical Review*, **62**, 275-288.

Oja, E. (1982): A simplified neuron model as a principal components analyzer. *J. of Math. Biology*, **15**, 267-273.

Oja, E. (1992): Principal components, minor components, and linear neural networks. *Neural Networks*, **5**, 927-935.

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

Parlett B. N. (1980): *The Symmetric Eigenvalue Problem.* Prentice Hall, Englewood Cliffs, N.J.

Ripley, B. D. (1994): Neural nerworks and related methods of classification. *J. R. Statist. Soc. B*, **56**, 3, 409-456.

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