Traffic matrix estimation using the Levenberg-Marquardt neural network of a large IP system

S. Mekaoui[1], C. Benhamed[1] & K. Ghoumid[2]
[1]*Université des Sciences et de la Technologie H. Boumediene, Alger, Algérie*
[2]*ENSAO, Complexe Universitaire, Oujda, Maroc*

Abstract

This paper deals with a method using a specific class of neural networks whose learning phase is based on the Levenberg-Marquardt algorithm and which had been applied to the estimation of the traffic matrix (TM) of a large scale IP network. The neural network had been implemented with the help of the specific neural toolbox of the source software Matlab. Such neural networks are within the class of feed forward and recurrent types. The simulation tests have been processed on the available data base of the very reputed American observatory data base on the Internet of a very large scale IP network, the so-called Abilene network on both categories of neural networks. The simulated results using this method have been found to be very accurate as compared to one another. The static model converges rapidly but was less accurate in the estimation of the Traffic Matrix of such a kind of large IP System (the Abilene System) than the dynamic model which in this way earned the challenge of yielding a perfect estimation.
Keywords: IP networks, Traffic Matrix (TM), neural networks (NN), Levenberg-Marquardt learning algorithm, Traffic Matrix estimation, linear regression.

1 Introduction

For quantitative reasons, telecommunication operators are usually interested in a good knowledge of the values of the volumes of information that pass through their network versus time. These volumes are generally expressed in flow rates of information going through the different points and links that constitute the topology of their network. These links volumes are also known as charges of


WIT Transactions on Information and Communication Technologies, Vol 45, © 2013 WIT Press
www.witpress.com, ISSN 1743-3517 (on-line)
doi:10.2495/DATA130081

information and are measured today with specific network protocols such as NetFlow from Cisco and Simple Network Manager Protocol (SNMP) in each link or point of presence (PoP) of the network and constitute themselves what we globally define as the traffic through the considered network. Essential information that they need is what we call the traffic matrix that indicates the volumes of the traffic in terms of flow rates between the nodes origin-destination since a given period of time. The direct measurement of this matrix is very hard usually because not all the points of the network are fed with information at a time, not all important nodes called routers are equipped with the necessary software and usually because the direct measurement is very expensive and can affect the quality of the service of the network if the operator is obliged to stop the network since the measurement is progressing. So, the researchers oriented this measurement to estimation by developing many methods and techniques to estimate this traffic matrix. These techniques were categorized into three major classes such as; deterministic approach which consider this estimation as an optimization problem. In that way, Goldschmidt [1] and Eum *et al.* [2] used Linear Programming (LP) to maximize the global sum of the weights of the flow rates through the links of the network and considered this function as an objective one. Others, [3, 4], of the same class, use the minimization of the distance of a given traffic matrix integrated to a real one**;** statistical approaches which consider the measurements of the different link charges as random variables measured since continuous intervals of time and the flows between the origin-destination pairs as parameters of a statistical model. Hence, Tebaldi and West [5] utilize a Bayesian approach when Medina *et al.* [6] perform the maximum likelihood function used in Expectation Maximization (EM) approach. Other methods in this second class use a Gaussian statistical model or alternatively a Poisson statistical one; dynamic approach whose principle is to reroute the traffic flows by changing the weights of links. This concept will be responsible in the generation of new links charges. By applying repeatedly new weights to redirect and reroute the traffic flows, it will be then possible to solve the problem of this estimation. This last approach requires finding out a set of links that will be able to reroute the flows under the constraint that the rerouted flows will not cause any saturation in the network. To determine the link weights that can satisfy this condition, some authors, like in [7–9], have used heuristic approaches. Finally, another approach which can be classified as more close to the third class is the neural network approach. In this category, Jiang *et al.* [10, 11], Vardi [12, 14] and Eum *et al.* [13] developed a method called back propagation for traffic matrix estimation (BPTME) based on a specific class of artificial neural network which consisted in modifying the weighing coefficients (w_i) to estimate the traffic matrix. We, in our work, have developed a multi-layer neural network technique for the estimation of the traffic matrix (TM) of large scale IP Network (Abilene) based on the Levenberg-Marquardt [15] algorithm and carried out a comparison to validate this estimation with various previous methods.

2 Network background

2.1 General background

As cited above, the traffic matrix represents the traffic between the different Origin-Destination pairs of the network simply denoted **OD.** The traffic flows go through physical links that connect the network nodes between them and the concatenation of the links of an OD flow path constitutes the route. All of the routes being crossed by the OD flows are represented by a routing matrix that contains only ones ("1") and zeros ("0"). Let A= ($a_{i,j}$) be this matrix, if the link between the origin I and the destination j does exist then $a_{i,j}$=1 and if the link does not $a_{i,j}$= 0. Let n be the number of nodes and l the number of physical links that connect the nodes one another, then the number of OD pairs is N= n^2. Let X(t) be the traffic matrix of N x M dimension so that M represents the number of time samples. We, then, define the traffic matrix of the charges at the nodes by Y (t) which has an l x M dimension and the routing matrix by A which has an l x N dimension. The relation that links the traffic matrix and this of the charges is simply given by the following expression:

$$Y(t) = A.X(t) \tag{1}$$

The problem is to estimate the traffic matrix $X(t)$ from the links charges $Y(t)$ and the routing matrix A. This problem is an ill addressed problem because the number of the unknown variables is greater than the number of the known data ($n^2 > l$).

2.2 Topology of the IP system

The studied large IP system is the well known American large IP network called Abilene network whose topology is given in Figure 1.

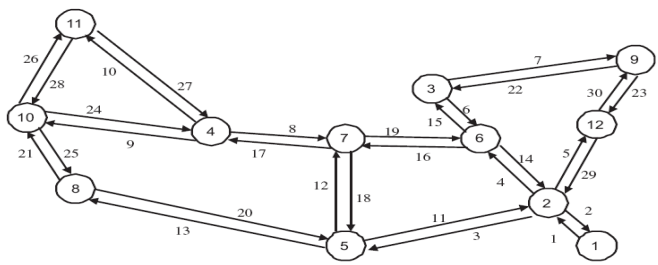

Figure 1: Abilene network topology (US large scale IP/network).

This network consists of twelve (12) principal nodes which distribute information through 144 OD (Origin-Destination) pairs and through 54 links that connect the nodes one another. Data are collected through measurements that have been processed from 01-03-2004 to 07-03-2004 (that is to say for one week) every five minutes a day. So, we got 288 samples a day and 2016 samples a week.

3 Neural network background

3.1 Dynamic neural network

Because of certain properties, we deliberately have chosen this model of neural networks. This kind of neural network is known as a model having unity feedbacks in its hidden layers [15–17]. Figure 2 depicts such a recurrent model.

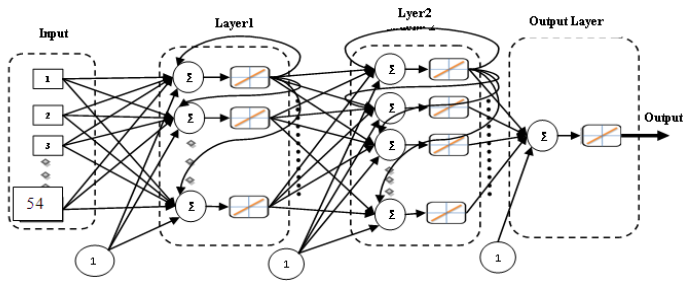

Figure 2: Dynamic Neural Network structure with 3 layers and one output.

Let $w_{i,i}$ be the weights of the different recurrent connections, then, the following equations can be derived from the depicted above neural network dynamic model:

$$n_1(t) = (w_1 \times p(t)) + (w_{1,1} \times p((t-1)) + 1; \quad (2)$$

$$n_2(t) = (w_{2,1} \times n_1(t)) + (w_{2,2} \times n_1(t-1)) + 1; \quad (3)$$

$$n_3(t) = (w_{3,2} \times n_2(t)) + (w_{2,2} \times n_2(t-1)) + 1; \quad (4)$$

$n_3(t)$ from equation (4) is the final output of the dynamic network as shown in figure 2 [16, 17]. Moreover, the model that we conceived comprises two hidden layers and one output layer. It also contains a prior input layer. The neurons activation functions are linear and the first hidden layer comprises 16 neurons whereas the second hidden layer consists in 13 neurons. Each input in the input layer is connected to all the input neurons of the first hidden layer which themselves are simultaneously connected between them recursively (same layer) and to all of the input neurons of the second layer. The outputs of the neurons of the second layer are all connected by a summation process to one input neuron of the final output layer $(n_3(t))$.

3.2 Learning principle

The learning principle of a neural network is to minimize in the sense of the last mean squares the cost function. Since the neural networks are not linear systems, then the cost function obtained by solving the gradient to zero is not linear either. This fact makes the parameters estimation more complicated. Another problem is the existence of local minimums in the cost function. As this function is not a quadratic one, it has consequently several minimums as shown in Figure 3.

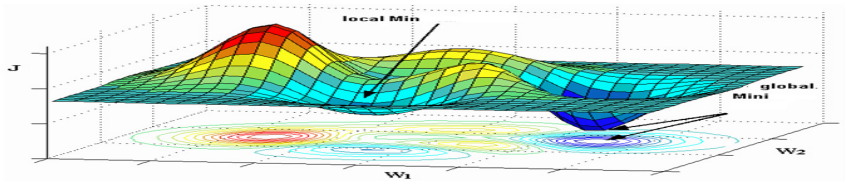

Figure 3: Cost function representation with local and global minimums of a neuron with two weighted inputs.

The cost function minimization of a non linear model involves the use of iterative methods that modify the model [15–17]. Each operation of the iterative process is denoted as an epoch of the learning process and requires two major steps namely:

- Gradient cost function evaluation in order to approach a minimum of the function.
- Function parameters modification versus the gradient in order to approach such a fixed minimum.

3.2.1 Data pre-treatment

Once the data are available, we should proceed to a pre-treatment [16] that allows the modeling to be as efficient as possible. In anyway, the minimum pre-treatment consists in normalizing and centralizing the data as in such a way that we can avoid any oversized variables and make the learning algorithm most efficient one. Therefore, a simple pre-treatment consists in changing the variables to central variables. So, we get the following:

$$u' = \frac{u - \langle u \rangle}{S_u}; \tag{5}$$

where $\langle u \rangle$ is denoting the mean of the considered quantity u, which can be estimated by:

$$\langle u \rangle = \frac{1}{N} \sum_{k=1}^{N} u_k, \tag{6}$$

and S_u the standard deviation of the quantity u is then defined by:

$$S_u = \sqrt{\frac{1}{N-1} \sum_{k=1}^{N} (u - \langle u \rangle)^2} \tag{7}$$

3.2.2 Levenberg-Marquardt learning algorithm

The Levenberg-Marquardt algorithm (MLA) helps to find out a solution to the minimization of a cost function when this function depends on several variables. The LMA is a powerful algorithm compared to the Gauss-Newton or the gradient algorithms in terms of convergence and stability [15]. Furthermore, the LMA algorithm modifies the weights of the neural network by moderating them with the value Δw given in equation (8):

$$\Delta w = (J^T J + \mu I)^{-1} J^T e \tag{8}$$

where: w; is the weights vector of the neural network,
I; is the identity matrix,
J; is the Jacobi matrix of (PxM) xN size,
μ; The combination coefficient and (^{T}); the transpose operand,
e; The error vector.

Thus, P is the number of the considered samples; M the number of the outputs of the neural network and N is the number of the weights. The elements of the vector e are calculated using the following equation:

$$e_{dm} = d_{pm} - o_{pm} \tag{9}$$

where d_{pm} and o_{pm} are respectively the desired output and the current output of the neural network, at respectively the output m for the input sample p. The matrices forms of J and e are given by the following equalities:

$$J = \begin{bmatrix} \frac{\partial e_{11}}{\partial w_1} & \frac{\partial e_{11}}{\partial w_2} & \cdots & \frac{\partial e_{11}}{\partial w_N} \\ \frac{\partial e_{12}}{\partial w_1} & \frac{\partial e_{12}}{\partial w_2} & \cdots & \frac{\partial e_{12}}{\partial w_N} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial e_{1M}}{\partial w_1} & \frac{\partial e_{1M}}{\partial w_2} & \cdots & \frac{\partial e_{1M}}{\partial w_N} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial e_{P1}}{\partial w_1} & \frac{\partial e_{P1}}{\partial w_2} & \cdots & \frac{\partial e_{P1}}{\partial w_N} \\ \frac{\partial e_{P1}}{\partial w_1} & \frac{\partial e_{P1}}{\partial w_2} & \cdots & \frac{\partial e_{P1}}{\partial w_N} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial e_{PM}}{\partial w_1} & \frac{\partial e_{PM}}{\partial w_2} & \cdots & \frac{\partial e_{PM}}{\partial w_N} \end{bmatrix}, \tag{10a}$$

$$e = \begin{bmatrix} e_{11} \\ e_{12} \\ \cdots \\ e_{1M} \\ \cdots \\ e_{P1} \\ e_{P2} \\ \cdots \\ e_{PM} \end{bmatrix}, \tag{10b}$$

The different steps of the algorithm are first concerned by the introduction of the available data samples for the learning and the training of the neural network. After what, the Jacobi matrix and the error vector e are calculated following (10a) and (10b). Then, the weights are modified using equation (8) that yields the new refreshed values of the weights that will better control and better converge and stabilize the neural network. Then the LMA proceeds to a better controlled learning. To learn more, one should refer to [15–17].

4 Traffic matrix estimation

Once the neural network has learnt the adequate model at the corresponding OD (Origin-Destination) of the IP network pair that's to say to the optimal weight adjusting, it is enough to introduce at any time of the considered week only the states of the links charges that had been measured by the SNMP (Simple Network Management Protocol) to obtain the links charges at any OD pair of the IP network. Since the traffic variations between the OD pairs present different variations functions, it is necessary to assign to each OD pair the most appropriate model in order to obtain the optimal minimum error.

5 Results

In this section, we present some results obtained from our simulations operated on specific OD pairs of the Abilene IP network for the estimation of the traffic matrices.

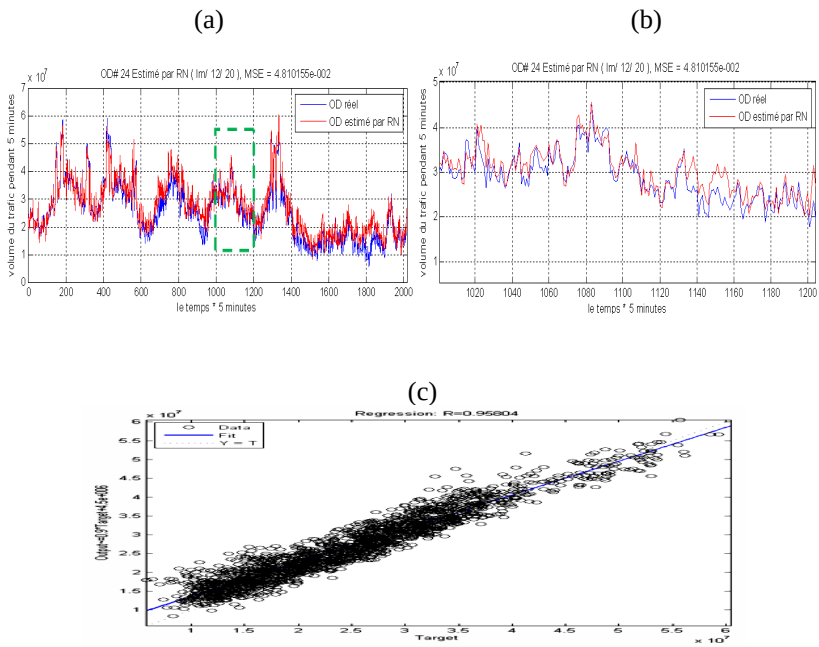

Figure 4: (a) Traffic flow estimation OD#24, Red= real, Blue=estimated, (b) A zoom on the green area, (c) Linear regression of TM_{real} on the $TM_{estimated}$ for OD#24.

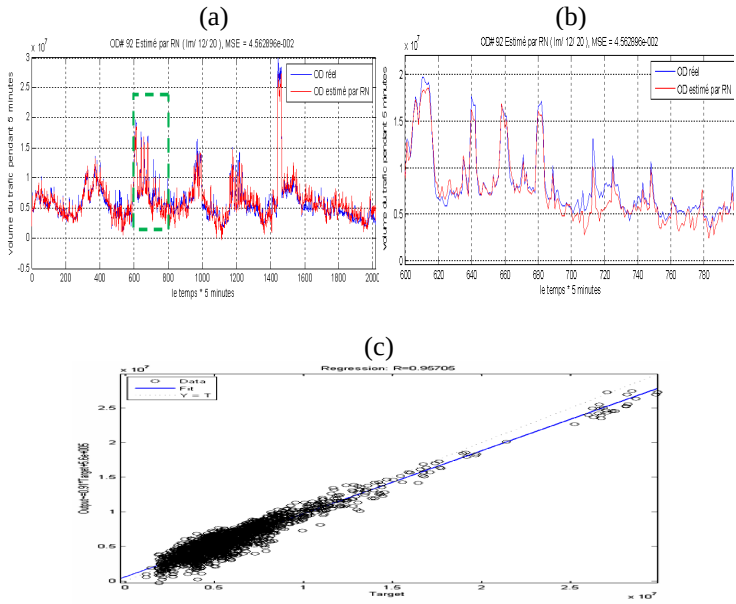

Figure 5: (a) Traffic flow estimation OD#92, Red= real, Blue=estimated, (b) A zoom on the green area, (c) Linear regression of TM_{real} on the $TM_{estimated}$ for OD#92.

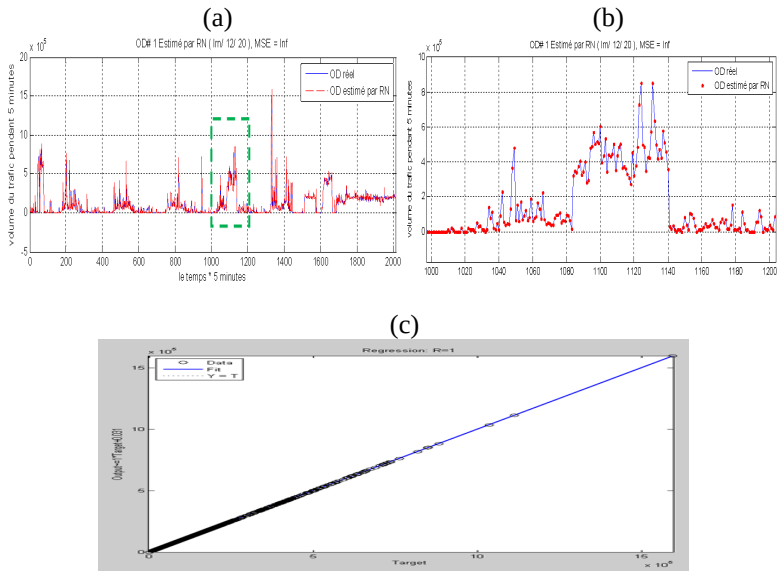

Figure 6: (a) Traffic flow estimation OD#1, Red= real, Blue=estimated, (b) A zoom on the green area, (c) Linear regression of TM_{real} on the $TM_{estimated}$ for OD#1.

6 Discussion and conclusion

Our work had been focused on the implementation and testing of two types of neural networks based on LMA learning algorithm namely the static feed forward neural network and the dynamic one. Unfortunately, because of template limitations it was not possible to detail the first category and thanks to its exceptional precision in the estimation of the traffic matrix of the IP system, we deliberately have chosen to detail the second category (dynamic). The static neural network also comprises 54 inputs and three layers, except that in this case, the first hidden layer consists in 9 neurons and the second in 17 neurons and the model is not recurrent. The output layer is the same with the dynamic model and the neural activation functions are linear either. 400 samples were used in our simulations and the criterion adopted to stop the learning process was that of the 'Early-Stopping' [15], which is based on the validation error. The learning is stopped when this error starts to rise up from a global minimum.

In Figure 4(a), an estimation of the traffic flow in OD#24 is performed and compared at a time in the same figure with the real flow with an MSE error of about 4.81 10^{-2}. Figure 4(b) represents a zoom of the time interval [600-800] that shows a good correlation between the real flow and the estimated one with the mean of the LMA (NN). A linear regression of the estimated values on the real values had been performed and displayed in Figure 4(c) to validate the estimation process by this mean. Indeed this figure shows a good correlation between both parameters with a regression coefficient of about 0.95804. Same procedure and displays are applied to OD#92 (Figures 5(a)–5(c)) and OD#1 (Figures 6(a)–6(c)). In the case of OD#92, we have found approximately similar results compared to those of OD#24. Whereas, in the case of OD#1 and with a dynamic neural network model, the coefficient of regression is equal to one and we can observe that the precision of the estimation is perfect and the estimated values of the traffic matrix coincide exactly with the real values without any kind of error.

This fact was also observed in other cases of OD pairs and compelled us to conclude that the dynamic model of the neural network based on the LMA learning is the best which earned the challenge. In most of the time, this neural network structure was able to track the real matrix dynamics and that the linear regression between the estimated values and the real values served as testimony of the perfect estimation. Other validation parameters can be implemented to validate the proposed method. We personally have carried out three of them such as: time convergence; the algorithm complexity; and the estimation efficiency which will certainly be reported in another paper. Work is in progress to attain this goal.

References

[1] O. Goldschmidt, "ISP Backbone Traffic Inference Methods to Support Traffic Engineering," *In Internet Statistics and Metrics Analysis (ISMA) Workshop*, San Diego, CA, USA, pp. 1063–1075, December 2000.

[2] S. Eum, J. Murphy, and R. Harris, "A Fast Accurate LP Approach for Traffic Matrix Estimation," in *International Telegraphic Congress, (ITC19)*, Beijing, China, September 2005.
[3] Y. Zhang, M. Roughan, N. Duffield, and A. Greenberg, "Fast accurate computation of large scale IP traffic matrices from link loads," in *Proceedings of the 2003 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*. San Diego, CA, USA: ACM Press, 2003, pp. 206–217.
[4] Y. Zhang *et al.* "Information Theoretic Approach to Traffic Matrix Estimation," in *Special Interest Group on Data Communication, (SIGCOMM'03)*, Karlsruhe, Germany, August 2003.
[5] C. Tebaldi and M. West "Bayesian inference on network traffic using link count data (with discussion)," *Journal of Amer. Stat. Assoc.*, pp. 557–576, June 1998.
[6] A. Medina, K. Salamatian, N. Taft, I. Matta, and C. Diot, "A Two-step Statistical Approach for Inferring Network Traffic Demands," Technical Report BUCS-2004-001, March 2004.
[7] A. Nucci, R. Cruz, N. Taft, and C. Diot, "Design of IGP Link Weight Changes for Estimation of Traffic Matrices," in *The 23rd Conference of the IEEE Communications Society (INFOCOM'04)*, Hong Kong, 2004.
[8] Dingde Jiang, Xingwei Wanga, Lei Guo, Haizhuan Ni, Zhenhua Chen "An Accurate Estimation of large-scale IP traffic matrix" Int. J. Electron. Commun. (AEU¨) (2011) pp 75–86.
[9] A. Soule, A. Lakhina, N. Taft, K. Papagiannaki, K. Salamatian, A. Nucci, M. Crovella, and C. Diot. "Traffic matrices: Balancing measurements, inference and modeling", ACM Sigmetrics, Banff, June 2005.
[10] D. Jiang, X. Wang, L. Guo, 'An optimization method of large-scale IP traffic matrix estimation', AEU-International Journal of Electronics and Communications 64 (7) (2010) pp. 685–689.
[11] D. Jiang, Z. Xu, Z. Chen, Y. Han, H. Xu, 'Joint time–frequency sparse estimation of large-scale network traffic 'Computer Networks 55 (2011) 3533–3547.
[12] Y. Vardi, "Network tomography: Estimating source-destination traffic intensities from link data," J. Amer. Stat. Assoc., vol. 91, no. 433, pp. 365-377, 1996.
[13] S. Eum *et al.* 'Generalized Kruithof Approach for Traffic Matrix Estimation' 14*th IEEE Intern. Conf. on Networks.* ICON'06, Singapore, Sep. 2006.
[14] Y. Vardi, "Network Tomography: Estimating Source-Destination Traffic Intensities from Link Data," *J. Am. Stat. Assoc.*, pp. 365–377, 1996
[15] B.M. Wilamowski, *et al*. Improved Computation for Levenberg–Marquardt Training, IEEE transactions on neural networks, vol. 21, no. 6, June 2010.

[16] G.Dreyfus, J.-M.Martinez, M.Samelides, M.B.Gordon, F.Badran, S.Thiria. '*Apprentissage statistique Réseau de neurones, Cartes topologique, Machine à vecteurs supports'*. editor : Eyrolles, 2008, Paris.
[17] Léon Personnaz, Isabelle Rivals,'Réseaux de neurones pour la modélisation, la commande et la classification. CNRS, 2003, Paris.