

UNIVERSITÉ DU QUÉBEC

MÉMOIRE PRÉSENTÉ À
L'UNIVERSITÉ DU QUÉBEC À TROIS-RIVIÈRES

COMME EXIGENCE PARTIELLE
DE LA MAÎTRISE EN PHYSIQUE

PAR
JAMAL LAMRANI

ÉTUDE ET TENTATIVE D'OPTIMISATION DES PARAMÈTRES
D'UN RÉSEAU DE NEURONES DE TYPES AUTO-ASSOCIATIF.

JANVIER 1994

Université du Québec à Trois-Rivières

Service de la bibliothèque

Avertissement

L'auteur de ce mémoire ou de cette thèse a autorisé l'Université du Québec à Trois-Rivières à diffuser, à des fins non lucratives, une copie de son mémoire ou de sa thèse.

Cette diffusion n'entraîne pas une renonciation de la part de l'auteur à ses droits de propriété intellectuelle, incluant le droit d'auteur, sur ce mémoire ou cette thèse. Notamment, la reproduction ou la publication de la totalité ou d'une partie importante de ce mémoire ou de cette thèse requiert son autorisation.

TABLE DES MATIÈRES

LISTE DES FIGURES	ii
LISTE DES TABLEAUX	vi
LISTE DES SYMBOLES	vii
RÉSUMÉ	x
INTRODUCTION	1
CHAPITRE I	
L'APPROCHE CONNEXIONNISTE ET LE TRAITEMENT PAR RÉSEAUX NEURONAUX	3
• Analogie entre neurone biologique et neurone formel	4
• Le modèle connexionniste	6
CHAPITRE II	
LE MODÈLE EIDOS	9
• L'architecture du modèle	9
• La règle de transmission	10
• La règle d'apprentissage	11
CHAPITRE III	
OPTIMISATION DU MODÈLE EIDOS	14
• Phase d'apprentissage	15
• Conséquence du choix des valeurs des paramètres sur la performance du réseau	17
LA MÉMOIRE ET SES DIFFÉRENTS ASPECTS	23
CARACTÉRISTIQUES DES FONCTIONS DE TRANSMISSION	25
CHAPITRE IV	
ÉTUDE DU MODÈLE BSB/EIDOS OPTIMISÉ : RÉSULTATS DE SIMULATIONS	28

• Méthodologie générale des simulations	28
• Simulation 1 : Apprentissage des prototypes	31
• Simulation 2 : Exemple d'un essai de rappel	32
• Simulation 3 : Apprentissage à partir d'exemples	34
• Simulation 4 : Modification de la valeur du paramètre de maintien	35
• Comparaison des résultats obtenus avec et sans paramètre de maintien	39
UTILISATION D'UNE NOUVELLE FONCTION DE TRANSMISSION	41
• Phase d'apprentissage	41
• Phase de rappel	41
CHAPITRE V	
DISCUSSION	50
• Développement de l'espace stimuli	50
• Performance en simulation	53
• Développements futurs	54
CONCLUSION	57
RÉFÉRENCES	59
ANNEXES	
LISTE DES PROGRAMMES	61
EXEMPLE ILLUSTRANT L'ÉVOLUTION DE LA MATRICE A	63

Remerciements

Je tiens à remercier mon directeur de recherche, Monsieur Mounir Boukadoum, pour son aide et ses critiques constructives, ainsi que pour ses conseils qui m'ont été une source constante d'encouragement et de connaissances. J'aimerais adresser un remerciement tout particulier à Monsieur Elie Boridy, qui a contribué au développement du programme de la maîtrise en physique en collaboration avec l'Université du Québec à Trois-Rivières.

LISTE DES FIGURES

Figure	Page
CHAPITRE I	
1. Les correspondances fonctionnelles entre le neurone biologique-type et son modèle, le neurone formel	5
CHAPITRE II	
2. Architecture neuronale du modèle Eidos	9
3. Illustration géométrique de la règle de transmission	11
CHAPITRE III	
4. Évolution de la valeur propre de A en fonction du nombre du nombre d'essais d'apprentissage d'un seul prototype avec les valeurs de paramètres de Proulx et Bégin	14
5. Évolution de $f(\lambda)$ pour les valeurs de paramètres utilisées par Proulx et Bégin	17
6. Valeurs de λ_c et de $f(\lambda_{\max})$ en fonction du choix de C	19
7. Nombre d'essais d'apprentissage requis avant que la matrice synaptique se stabilise en fonction de la valeur de C	20
8. Courbe de $f(\lambda)$ pour $C = 25$	21
9. Évolution de λ_c la valeur propre de A en fonction du nombre d'essais d'apprentissage d'un seul prototype ($C = 25$)	22
10. Aspect typique du signal de sortie d'un circuit réverbérant	24

Figure	Page
11. La fonction linéaire saturante	26
12. La fonction sigmoïde	27
CHAPITRE IV	
13. Représentation graphique des neuf prototypes utilisés pour les simulations	29
14. Évolution du spectre des valeurs propres de A obtenues après apprentissage par présentation de neuf prototypes	31
15. Exemple de rappel avec bruit de 100% pour la lettre A	32
16. Erreurs de catégorisation pour la règle de transmission $x[t+1] = (A + I) x[t]$	33
17. Temps de réponse pour la règle de transmission $x[t+1] = (A + I) x[t]$	34
18. Évolution du spectre des valeurs propres de A pour une proportion de bruit de 25%	35
19. Évolution du spectre des valeurs propres de A lors d'une modification de la règle de transmission	36
20. Erreurs de catégorisation pour la règle de transmission $x[t+1] = (A+0.95 I)x[t]$	37
21. Temps de réponse pour la règle de transmission $x[t+1] = (A+0.95 I)x[t]$	37
22. Évolution du spectre des valeurs propres de A pour une proportion de bruit de 25%	38

Figure	Page
23. Erreurs de catégorisation pour la règle de transmission $x[t+1] = (A+0.95 I)x[t]$	39
24. Temps de réponse pour la règle de transmission $x[t+1] = (A+0.95 I)x[t]$	39
25. Erreurs de catégorisation pour $x[t+1] = \tanh[2 (A + I)] x[t]$	43
26. Temps de réponse pour $x[t+1] = \tanh[2 (A + I)] x[t]$	43
27. Erreurs de catégorisation pour $x[t+1] = \tanh[10 (A + I)] x[t]$	44
28. Temps de réponse pour $x[t+1] = \tanh[10 (A + I)] x[t]$	44
29. Erreurs de catégorisation pour $x[t+1] = \tanh[25 (A + I)] x[t]$	45
30. Temps de réponse pour $x[t+1] = \tanh[25 (A + I)] x[t]$	45
31. Erreurs de catégorisation pour $x[t+1] = \tanh[2 (A +0.95 I)] x[t]$	46
32. Temps de réponse pour $x[t+1] = \tanh[2 (A +0.95 I)] x[t]$	46
33. Erreurs de catégorisation pour $x[t+1] = \tanh[10 (A +0.95 I)] x[t]$	47
34. Temps de réponse pour $x[t+1] = \tanh[10 (A +0.95 I)] x[t]$	47
35. Erreurs de catégorisation pour $x[t+1] = \tanh[25 (A +0.95 I)] x[t]$	48
36. Temps de réponse pour $x[t+1] = \tanh[25 (A +0.95 I)] x[t]$	48
 CHAPITRE V	
37. Évolution du spectre des valeurs propres de A pour les 26 lettres de l'alphabet	51

Figure	Page
38. Erreurs de catégorisation pour les 26 lettres de l'alphabet	52
39. Temps de réponse pour les 26 lettres de l'alphabet	52

LISTE DES TABLEAUX

Tableau	Page
CHAPITRE III	
1. Nombre d'essais d'apprentissage requis pour neuf prototypes en fonction des valeurs de C	20
CHAPITRE IV	
2. Indice de proximité entre les lettres utilisées lors des simulations	30
3. Effets de varier la valeur du paramètre de maintien	36
4. Résumé des résultats obtenus lors des simulations	40
5. Temps de réponse et erreurs de catégorisation pour une proportion de bruit de 100%	49
CHAPITRE V	
6. Comparaison des résultats pour $\psi = 1$	54
7. Comparaison des résultats pour $\psi = 0.95$	54
8. Comparaison des performances pour différentes fonctions de transmission	55

LISTE DES SYMBOLES

Symbole	Signification
α	Facteur d'amplification de la modification Hebbienne dans le modèle Eidos
β	Facteur d'amplification de la modification anti-Hebbienne dans le modèle Eidos
γ	Valeur de saturation des unités
η	Facteur d'amplification de la modification Hebbienne dans le modèle BSB
κ	Facteur déterminant le pas pour la fonction tangente hyperbolique
λ	Valeur propre de la matrice A
$\Delta\lambda$	Variation de la valeur de λ suite à un essai d'apprentissage
π	La proportion de bruit
σ	L'écart type de la distribution normale
ψ	Facteur de maintien de l'activité des unités
ζ	Facteur d'affaiblissement des connexions entre unités
λ_c	Valeur propre critique de la matrice A

Symbole	Signification
λ_{\max}	Valeur propre maximale de la matrice A
A	Matrice synaptique carrée symétrique représentant les poids des connexions des unités dans le réseau
x	Vecteur d'état colonne de dimension n représentant un des prototypes du répertoire stimuli
\mathbf{x}^T	La transposée du vecteur x
ΔA	Changement de la valeur de A lors de l'application de la règle d'apprentissage
$A_{[0]}$	Valeur initiale de la matrice A
b	Vecteur bruit de même dimension que le prototype x
n	Nombre d'itérations du vecteur d'état dans la modification anti-Hebbienne
p	Nombre d'itérations du vecteur d'état dans la modification Hebbienne
C	Facteur d'amplification définissant le rapport entre α et $-\beta$
D	Distance de Hamming séparant l'exemple du prototype
I	Matrice identité
$L[x_i]$	Fonction non linéaire saturante

Symbole	Signification
$N(0,\sigma)$	La distribution normale de moyenne 0 et d'écart type σ
0	La matrice nulle

RÉSUMÉ

Parmi les modèles de reconnaissance de forme connus, le modèle BSB « Brain States in a Box » de Anderson et al. semble des plus aptes à expliquer comment un réseau de neurones biologique développe ses capacités de catégorisation et de généralisation. Ce réseau neuronique est basé sur un hypercube dont certains sommets, grâce à un apprentissage non supervisé, deviennent des pôles d'attraction vers lesquels convergent les stimuli à catégoriser. Le modèle original a récemment été amélioré par Proulx et Bégin, qui ont résolu certains des problèmes qui lui étaient associés en ajoutant un terme anti-hebbien à sa règle d'apprentissage qui était strictement hebbienne. Le nouveau modèle, « Eidos », utilise les règles de transmission et d'apprentissage suivantes :

$$\begin{aligned}\bar{x}[t+1] &= L((A + \psi I)\bar{x}[t]) \quad t = 0, 1, 2, \dots, T \\ \Delta A &= \alpha \bar{x}[p]\bar{x}[p]^t - \beta \bar{x}[n]\bar{x}[n]^t\end{aligned}$$

Dans ces équations, n et p représentent la $n^{\text{ième}}$ et la $p^{\text{ième}}$ itération d'un vecteur test $\bar{x}[0]$, α et β représentent l'importance des termes hebbien et anti-hebbien, ψ est le paramètre de maintien de l'état précédent (limité par $0 < \psi \leq 1$), A est la matrice synaptique du réseau, I est la matrice identité et L représente une fonction linéaire à saturation définie dans un intervalle $[-\gamma, \gamma]$. Proulx et Bégin avaient utilisé les valeurs de paramètres $p=5$, $n=10$, $\alpha=0.005$ et $\beta=0.0025$, sans donner de justification pour ce choix.

Le travail présent tente de trouver des valeurs optimales pour un de ces paramètres. Une étude théorique du modèle combinée avec des simulations a révélé que pour le choix des valeurs de p , n et α utilisé par Proulx et Bégin, et si la performance du réseau est évaluée en terme de la vitesse d'apprentissage, du temps de réponse et du taux d'erreur en rappel, une valeur optimale pour β est donnée par $\beta=\alpha/25$. Nous avons utilisé les mêmes données que ceux de Proulx et Bégin à des fins de comparaison, et nos résultats montrent que le nombre moyen de présentations requis pour apprendre un nouveau stimulus est coupé de 50%, de même que le temps de réponse moyen qui diminue de 76% et de l'erreur de catégorisation qui baisse de 19%.

On a aussi tenté de trouver une fonction de transmission optimale pour améliorer encore plus la performance du réseau. Nos résultats montrent que si on garde la fonction de transmission linéaire à saturation pendant la phase d'apprentissage, mais qu'on la remplace par

$$\bar{x}[t+1] = \tanh(10 (A + I)) \bar{x}[t]$$

durant la phase de rappel, le temps de réponse du réseau s'améliore encore de 68% et l'erreur moyenne de catégorisation de 34%.

INTRODUCTION

Un système de reconnaissance de forme devrait être capable d'expliquer les mécanismes par lesquels un organisme intelligent parvient à s'adapter et à fonctionner dans un milieu sans cesse variable, en développant de façon autonome des représentations conceptuelles de chacun des objets ou événements pertinents de son environnement. Il devrait disposer d'une capacité de généralisation suffisante pour lui permettre d'identifier correctement des objets en présence de bruit ou d'erreurs, et il devrait posséder le pouvoir de se développer et de s'adapter à de nouvelles situations sans l'intervention d'un agent extérieur pour redéfinir sa base des connaissances. Finalement, la possibilité d'exécuter ces opérations de catégorisation en temps réel serait aussi un atout, puisque dans la majorité des cas pratiques, le processus de reconnaissance de forme constitue l'élément de base d'un système de prise de décision automatique plus général, qui doit opérer en cours de processus.

Parmi les modèles de reconnaissance de forme connus, le modèle BSB « Brain States in a Box » (Anderson et al.; 1977) semble des plus aptes à reproduire ce processus de catégorisation. Malheureusement, le modèle initial proposé par Anderson et al. utilise une règle d'apprentissage hebbienne (Hebb; 1949) qui devient inefficace lorsque le réseau est soumis à des stimuli corrélés ou bruyants (Proulx; 1986). On obtient alors une performance médiocre lors de tentatives de reconnaissances d'un prototype appris. Récemment, grâce aux travaux de J. Bégin, un début de solution a vu le jour (1990) avec le développement du modèle « Eidos ». Dans ce modèle, la modification de la règle d'apprentissage par l'ajout d'un terme anti-hebbien a permis de résoudre ce problème de convergence que Anderson et al. avaient rencontré.

Dans cette étude, on veut analyser la performance du modèle « Eidos » qu'on soumet à une tâche de reconnaissance de caractères alphabétiques, en fonction des valeurs de paramètres choisies. Le répertoire utilisé contient neuf lettres et le réseau est composé de 35 unités de traitement. Dans un premier temps, l'apprentissage du réseau se fait à partir de prototypes

purs (sans bruit ajouté). Ensuite, la performance du système est examinée lors d'une phase de rappel pour différentes conditions d'opération. Finalement, l'effet d'appliquer plusieurs types de fonctions de transmission sur la performance du système est étudié. Dans tous les cas, le temps de réponse du réseau et le taux d'erreur de catégorisation sont les variables mesurées.

CHAPITRE I

L'approche connexionniste et le traitement par réseaux neuronaux.

Imaginons un ordinateur qui apprend tout seul : nourri par des exemples de comportement qu'on lui fournit, il saura développer ses propres capacités de décision, et il fonctionnera juste à l'aide de ces composantes. Cet ordinateur évoluera au rythme des connaissances acquises, faisant une myriade d'erreurs en cours de chemin mais apprenant de ces dernières, jusqu'à ce qu'il développe sa propre intelligence et puisse mener ses tâches avec succès. Ce comportement est presque humain, et pour concevoir une telle machine on s'est inspiré de nos connaissances sur la structure du cerveau, avec ses cellules nerveuses, leurs interconnexions et leurs interactions.

L'informatique construite sur la base des réseaux de neurones artificiels (*RNA*) présente une alternative prometteuse aux approches séquentielles présentes de traitement de l'information. Elle peut résoudre des problèmes complexes de reconnaissance de forme, de la voix, et d'identification des caractères manuscrits pour lesquels une solution séquentielle est soit impossible, soit demande un temps de traitement excessif.

L'intérêt récent pour les RNA est dû au désir de construire de nouvelles générations d'ordinateurs capables de résoudre des problèmes que l'ordinateur conventionnel ne peut aborder, et au souci d'utiliser des modèles cognitifs qui pourraient servir de fondations à l'intelligence artificielle. Il est bien connu que les machines informatiques actuelles offrent des vitesses de traitement qui surpassent de loin celle du cerveau humain, sans pour autant égaler sa performance pour solutionner des problèmes divers. L'idée derrière l'usage des RNA est que, en intégrant dans ces machines certains aspects des fonctions connues (ou supposées) du cerveau, on leur permettrait d'accomplir des tâches comme l'association, la catégorisation, la

généralisation, la classification, l'extraction de caractéristiques et l'optimisation pour lesquelles les moyens informatiques présents offrent des performances médiocres.

Par le passé, plusieurs investigateurs ont essayé de créer des modèles pour étudier les RNA (Vemuri; 1988). Ces modèles reposent sur une combinaison d'hypothèses tirées de nos connaissances en neurophysiologie et d'analogies tirées de modèles physiques. Les différents investigateurs qui oeuvrent dans le domaine se sont donnés des objectifs de recherche qui peuvent être résumés comme suit :

- 1) Imiter par des moyens artificiels la manière dont le cerveau humain accomplit des tâches cognitives telles que la perception, le rappel par association, le raisonnement et l'apprentissage; pour atteindre ce but, il est nécessaire de déchiffrer comment ces fonctions sont organisées et exécutées dans le cerveau.
- 2) Créer des paradigmes de traitement de l'information inspirés des résultats précédents, qui mettent l'accent sur la puissance du traitement plutôt que sur la fidélité biologique. Ceci rend admissible l'incorporation de certaines caractéristiques aux RNA, même si elles sont neuro-biologiquement peu plausibles.

Ainsi, il y a eu plusieurs approches pour la réalisation de modèles qui simulent des systèmes intelligents de reconnaissance. Parmi elles se trouve l'approche connexionniste, qui tient compte de contraintes biologiques.

- *Analogie entre neurone biologique et neurone formel.*

La nature de la fonction à assurer par le réseau est principalement définie par sa topologie, c'est à dire par la manière dont un ensemble de neurones artificiels (NA) sont interconnectés. Les connexions permettent à chaque neurone j de transmettre à un neurone i une stimulation positive (excitatrice) ou négative (inhibitrice) déterminée par l'état d'activité x_j de j et modulée par un poids synaptique w_{ij} . La figure 1 illustre la correspondance entre ce mécanisme et celui d'un neurone biologique. Les bulles représentent l'évolution temporelle, en différents points du

neurone i du potentiel électrique membranaire après l'arrivée d'un influx x_j délivré par le neurone j .

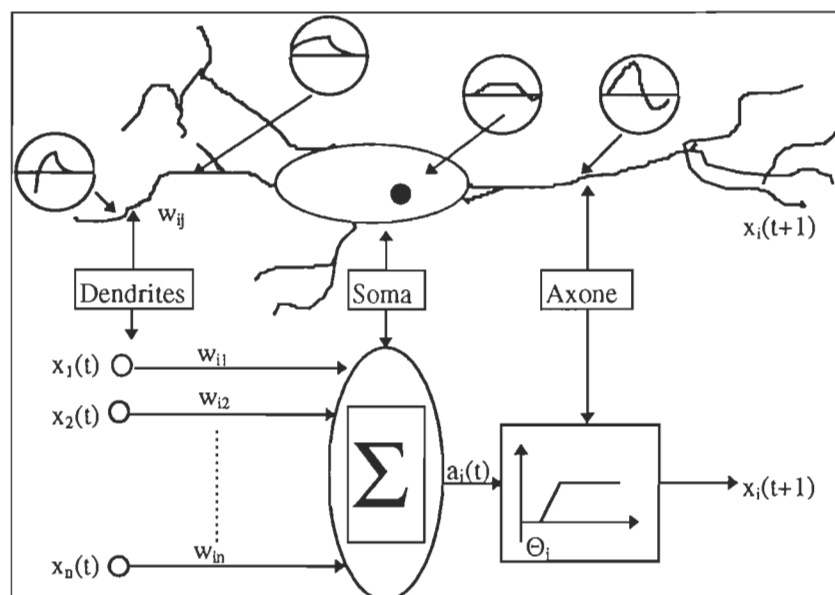


Figure 1. Les correspondances fonctionnelles entre le neurone biologique-type et son modèle, le neurone formel.

Chaque NA est caractérisé, à chaque étape de son évolution (qui est discrète) par un vecteur d'activation $\mathbf{a}(t) = (a_1(t), \dots, a_n(t))$. Ce dernier est défini par la somme pondérée des stimulations afférentes x_j multiplié par le poids synaptique w_{ij} . Le NA est limité par un circuit de sortie qui compare le vecteur d'activation à une valeur de seuil Θ_i , il devient actif lorsque ce seuil est dépassé, sinon il s'éteint (Vemuri; 1988). La transition d'une étape à la suivante résulte d'une mise à jour, soit par tous les NA simultanément, soit par un seul, des valeurs d'activité. Le processus commence par l'imposition d'un certain vecteur d'état \mathbf{x}_0 qui peut être considéré comme la donnée ou *l'intrant*. Il se poursuit par itération d'une règle de transition, et se termine lorsque le système atteint un équilibre caractérisé par un vecteur d'état \mathbf{x}_n , résultat ou *sortie* de ce qu'on peut appeler le calcul effectué par le réseau.

Notons que dans ce modèle, l'effet du réseau sur x dépend uniquement des w_{ij} . Par conséquent, « le savoir » du réseau réside non pas dans sa règle de transmission mais plutôt dans la force des connexions inter-neuronales caractérisées par ces w_{ij} . Les valeurs de ces connexions peuvent s'acquérir aussi bien par des techniques d'apprentissage que par l'imposition a priori de valeurs de pondérations (Vemuri; 1988). L'absence de règles spécifiques de comportement est particulièrement avantageuse pour l'accomplissement de tâches où les règles de décision sont floues et ne peuvent donc pas être formulées explicitement.

- *Le modèle connexionniste.*

L'expression « modèles connexionnistes » sert généralement à regrouper une famille de modèles à laquelle appartiennent, entre autres, les modèles associatifs linéaires et non linéaires (Kohonen; 1989. Anderson et al.; 1977. Murdock; 1982), les modèles thermodynamiques (Hinton et Sejnowski; 1986), de même que les modèles basés sur la compétition (Grosberg; 1988), la propagation rétrograde et la règle générale delta (Rumelhart et McClelland; 1986).

Les RNA de type connexionniste sont constitués d'un grand nombre d'unités de traitement, appelées *neurones formels*, qui sont reliées entre elles par des connexions, *les synapses*, et qui sont caractérisées par un mode de traitement de l'information en parallèle. Chacune de ces unités est caractérisée à un temps t donné par un niveau d'activité qui peut être représenté par une valeur scalaire. Tout état correspond alors à un vecteur dont chaque composante indique le degré d'activité d'une des unités du réseau. Ainsi, des vecteurs corrélés décrivent une situation où les patrons d'activité sont similaires, tandis que des vecteurs orthogonaux indiquent que ces patrons sont différents.

Mais ce qui distingue plus particulièrement les modèles connexionnistes est sans doute leur capacité de rendre compte des changements adaptatifs qui caractérisent tout organisme. Dans ces modèles, l'adaptation possède deux caractéristiques: elle est associative et elle est distribuée. En effet, les connexions entre les unités du réseau se modifient en fonction des

relations spatio-temporelles qui existent entre les patrons d'activité. De plus, l'adaptation est distribuée parce que tout changement affecte un ensemble de connexions sans qu'aucune d'elles ne soit suffisante, ni nécessaire, pour que le changement soit mémorisé.

Les caractéristiques générales des modèles connexionnistes sont donc que :

- 1) Ils sont aptes à rendre compte des interactions qui existent entre les processus cognitifs, tels que la perception, la mémoire et l'apprentissage.
- 2) Ils s'inspirent généralement de connaissances acquises à propos du système nerveux (ils sont donc plus conciliables avec les données neuro-biologiques).
- 3) Ils ne dépendent pas de l'intervention d'un exécutant central qui gère l'ensemble des opérations à effectuer. De plus, il est montré que leur performance n'est que peu affectée par l'instabilité de leurs composantes.
- 4) Ils s'adaptent bien aux situations à contraintes multiples et mal définies qui sont caractéristiques des situations réelles.

En outre, ces modèles possèdent l'importante propriété de rendre compte de la capacité qu'ont les organismes de généraliser leurs apprentissages à des situations nouvelles. Cependant, cette propriété s'acquiert souvent aux dépens d'une perte de sélectivité, appauvrissant ainsi les capacités de catégorisation du réseau. Les modèles qui ont été proposés pour remédier à cette perte de sélectivité doivent, pour la plupart d'entre eux, recourir à un agent externe qui fournit les catégories à apprendre. Cependant, la présence d'un tel agent pose un problème puisqu'elle suppose soit une pré-catégorisation de l'environnement, soit l'action d'un autre organisme qui connaît déjà les catégories. Le premier cas est insoutenable car, même si cet environnement présente des régularités, il n'étiquette pas chacun des faits qui s'y produisent. Quant à la seconde hypothèse, elle est aussi paradoxale que celle qui invoque la présence d'un homuncule interne puisqu'il faudrait expliquer comment l'autre organisme, ou l'homuncule, fait pour catégoriser (Bégin; 1990).

En conclusion, seuls les systèmes massivement parallèles et distribués, basés sur les opérations de réseaux neuronaux non supervisés, semblent constituer la voie permettant d'échapper au problème du traitement de l'information en temps réel.

CHAPITRE II

Le modèle Eidos.

Issu des travaux sur le modèle BSB d'Anderson et al. (1977), le modèle Eidos (Proulx et Bégin; 1992) se distingue toutefois de ce dernier par l'utilisation d'une nouvelle règle d'apprentissage, de même que par l'introduction de deux paramètres généraux, lesquels permettent de régénérer constamment la dynamique du système.

- *L'architecture du modèle*

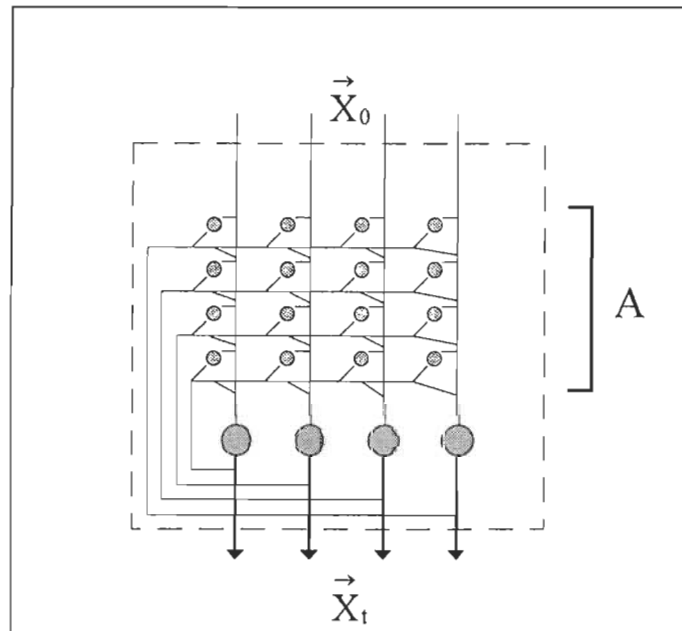


Figure 2. Architecture neuronale du modèle Eidos.

C'est un réseau mono-couche, totalement connecté, de type récurrent dont les signaux de sortie sont rebouclés vers les entrées via un processus de rétroaction. Il est défini par un système dynamique multidimensionnel à valeurs continues et à temps discrets, caractérisé à chaque

instant t par un vecteur d'état $\mathbf{x}[t]$ dont les composantes reflètent les activités des différentes unités. Chaque niveau d'activité est compris entre deux seuils $[-\gamma, \gamma]$. Par conséquent, on a

$$\mathbf{x}(t) = [x_1(t), x_2(t), \dots, x_n(t)]^T \in \{-\gamma, \gamma\}^n$$

où n est la dimension du réseau, et où T dénote l'opération de transposition vectorielle. On retrouve deux types de connexions distinctes. La première est directe et s'adapte en fonction de la corrélation temporelle entre les patrons d'activité de l'unité i et l'unité j . La seconde est réalisée par des inter-neurones qui inversent le signe de l'interaction entre l'unité pré-synaptique j et l'unité post-synaptique i .

- *La règle de transmission*

Le modèle utilise la même règle de transmission que celle du modèle BSB de Anderson et al.. La transmission est non linéaire et s'exprime par la règle suivante :

$$\mathbf{x}[t+1] = L(\mathbf{a}[t]) = L((\mathbf{A} + \Psi \mathbf{I}) \mathbf{x}[t]) \quad t = 0, 1, 2, \dots, T. \quad (1)$$

Dans cette équation, $\mathbf{a}[t]$ est le vecteur d'activation du réseau au temps t , $\mathbf{x}[t]$ est l'état du système au même temps ($\mathbf{x}[0]$ étant le stimulus initial), \mathbf{A} est la matrice synaptique résultante de l'apprentissage des stimuli, ψ est un paramètre de maintien de l'état précédant (limité par $0 < \psi \leq 1$) (Bégin; 1990) et \mathbf{I} est la matrice identité. L est une fonction de saturation dont le rôle consiste à maintenir l'activité des unités entre deux limites arbitraires et égales à $(\pm\gamma)$ et T est le moment où toutes les unités ont atteint un des seuils de saturation.

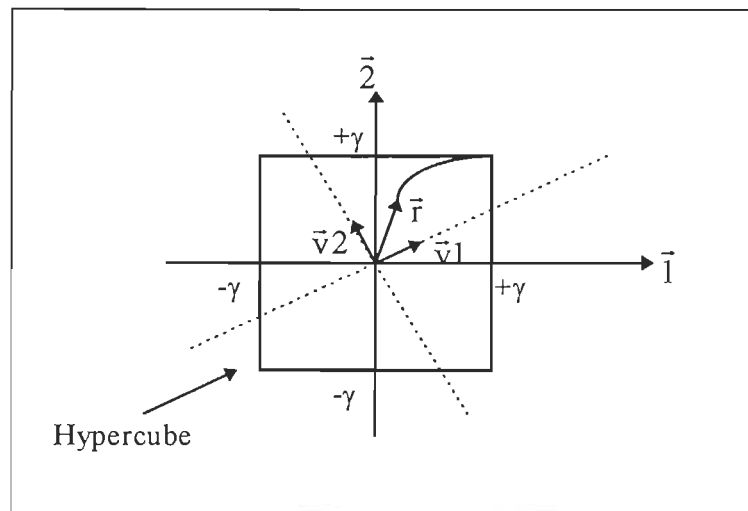


Figure 3. Illustration géométrique de la règle de transmission : \bar{r} représente la trajectoire suivie par un vecteur d'activation initial, $\{\bar{1}, \bar{2}\}$ est la base engendrée par l'espace stimuli et $\{\bar{v}_1, \bar{v}_2\}$ est la base initiale.

La matrice A produit un effet global de rétroaction positive et l'imposition des seuils de saturation a pour effet d'enfermer les vecteurs d'état dans un hypercube (figure 3) dont les coins stables constituent les réponses finales et invariantes du système suite à la présentation des stimuli.

- *La règle d'apprentissage*

Depuis les travaux de Hebb (1949), l'hypothèse la plus communément utilisée pour expliquer le mécanisme de la mémoire chez l'humain est qu'elle est due à une modification fonctionnelle qui se produit dans les synapses reliant les neurones, l'idée étant que des activités neuroniques corrélées augmenteraient l'efficacité du lien synaptique. Donc, les poids synaptiques w_{ij} augmentent chaque fois que les neurones i et j sont simultanément au même niveau d'activité et décroissent dans le cas opposé. En conséquence, pour mémoriser un vecteur prototype x conformément à l'hypothèse de Hebb, on modifiera le poids synaptique w_{ij} par une quantité

Δw_{ij} qui serait égale à $\eta x_i x_j$, où η est un facteur d'apprentissage positif. Pour une matrice synaptique qu'on aurait appelée A , la modification s'écrit donc sous la forme générale

$$\Delta A = \eta \mathbf{x}[k] \mathbf{x}^T[k] \quad (2)$$

où $\mathbf{x}[k]$ représente l'état après k itérations de l'état initial d'un prototype à apprendre $\mathbf{x}[0]$.

Proulx (1986) a montré que cette règle d'apprentissage (équation (2)) pose une difficulté majeure en présence de stimuli corrélés : elle a pour effet de favoriser une des valeurs propres de A au dépend des autres. Ce qui provoque la convergence du réseau vers une seule des directions permises dans l'hypercube formé par les limites d'apprentissage ; par conséquent, tous les stimuli se retrouvent catégorisés dans un seul coin. Pour éviter ce problème, Proulx a suggéré que la règle d'apprentissage devrait restreindre sélectivement le gain sur le plus grand vecteur propre de A de façon à permettre l'émergence des valeurs propres associées aux autres vecteurs propres. En d'autres termes, aucun des seuils ne devrait être atteint à l'intérieur du nombre d'itérations du vecteur état nécessaire à l'application de la règle d'apprentissage.

Proulx et Bégin (1988) ont alors proposé un nouveau modèle d'apprentissage qu'ils ont appelé « Eidos », et qui comprend l'addition d'un nouveau terme « anti-hebbien » au terme hebbien. Cet ajout permettrait la catégorisation de chacun des stimuli vers le coin stable qui lui est associé lors des apprentissages, et ainsi sous certaines conditions qu'on démontrera plus loin, il lui assurerait son développement dans l'espace-stimuli adéquat. Au niveau de l'adaptation des connexions, la transformation que subit la matrice A après chaque essai d'apprentissage, suite à la présentation d'un stimulus \mathbf{x} quelconque, devient

$$\mathbf{A}_{i+1} = \zeta \mathbf{A}_i + [\alpha \mathbf{x}[p] \mathbf{x}^T[p] + \beta \mathbf{x}[n] \mathbf{x}^T[n]] \quad (3)$$

où A_i représente la valeur de la matrice A après i essais d'apprentissage, ζ est un paramètre d'oubli ($0 < \zeta \leq 1$), $x[p]$ et $x[n]$ représentent l'état du réseau aux instants p et n , et α et β sont les poids respectifs des changements hebbien et anti-hebbien.

CHAPITRE III

Optimisation du modèle Eidos.

Les résultats obtenus par Proulx et Bégin (1990) montrent qu'il y a bien stabilisation des valeurs propres de la matrice synaptique (figure 4). Cependant, ils ont utilisé des valeurs arbitraires pour les paramètres de leur modèle. La question qui se pose alors est de trouver un intervalle de confiance dans lequel le choix de ces paramètres donne une réponse optimale sans pour cela qu'il y ait dégradation de performance : Augmentation du taux d'erreurs et/ou du temps de réponse lors d'une phase de rappel.

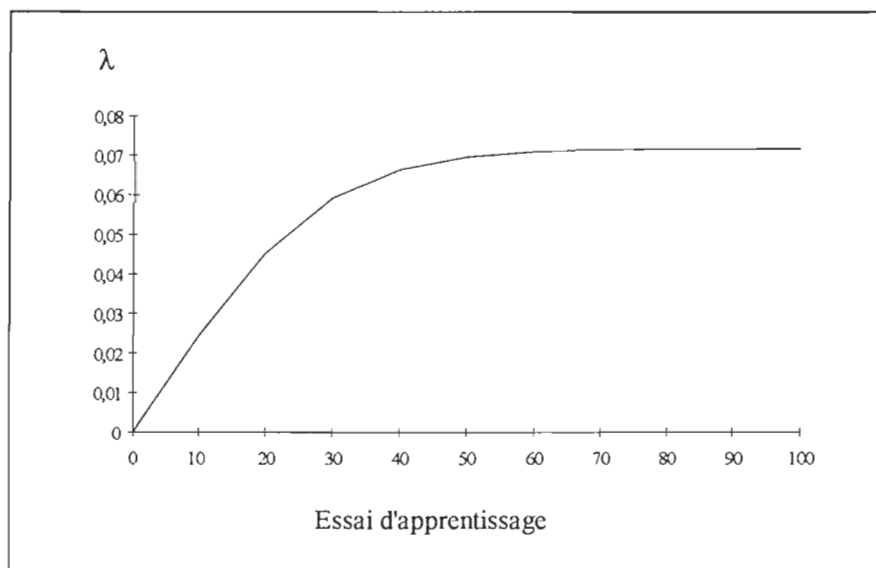


Figure 4. Évolution de la valeur propre de A en fonction du nombre d'essais d'apprentissage d'un seul prototype avec les valeurs de paramètres de Proulx et Bégin.

- *Phase d'apprentissage.*

Proulx et Bégin ont prouvé que pendant la phase d'apprentissage, l'usage de l'équation (3) permet d'obtenir une mémoire qui ne sature pas (aucune des unités n'a pu atteindre le seuil de saturation durant cette phase). Ils ont démontré l'existence d'un point critique où, sur présentation d'un prototype $\mathbf{x}[0]$, on obtient $\Delta\mathbf{A} = \mathbf{0}$ après un certain nombre d'essais d'apprentissage. Leur raisonnement est comme suit : si on pose $\psi=1$ dans l'équation (1), cette dernière devient, en l'absence de saturation des neurones :

$$\begin{aligned}\mathbf{x}[t+1] &= L((\mathbf{A}+\mathbf{I}) \mathbf{x}[t]) \quad \text{pour } t = 0, 1, \dots, T. \\ &= (\mathbf{A}+\mathbf{I}) \mathbf{x}[t] \\ &= (\mathbf{A}+\mathbf{I})^{t+1} \mathbf{x}[0]\end{aligned}\tag{4}$$

Comme la matrice synaptique \mathbf{A} est initialement nulle lors de la présentation d'un premier prototype $\mathbf{x}[0]$, elle ne peut qu'évoluer dans la direction de $\mathbf{x}[0]$. Donc $\mathbf{x}[0]$ est un vecteur propre de \mathbf{A} . Ceci permet d'écrire la relation (4) en fonction des valeurs propres de \mathbf{A} :

$$\mathbf{x}[t] = (1+\lambda)^{t+1} \mathbf{x}[0]\tag{5}$$

L'équation (3) devient alors, en posant que le paramètre d'oubli ζ vaut 1 :

$$\begin{aligned}\Delta\mathbf{A} &= \alpha(1+\lambda)^{p+1} \mathbf{x}[0] \{(1+\lambda)^{p+1} \mathbf{x}[0]\}^T + \beta (1+\lambda)^{n+1} \mathbf{x}[0] \{(1+\lambda)^{n+1} \mathbf{x}[0]\}^T \\ &= \{\alpha(1+\lambda)^{2(p+1)} + \beta(1+\lambda)^{2(n+1)}\} \mathbf{x}[0] \mathbf{x}^T[0]\end{aligned}$$

Par conséquent, on peut écrire $\Delta\mathbf{A}$ sous la forme

$$\Delta\mathbf{A} = \Delta\lambda \mathbf{x}[0] \mathbf{x}^T[0]$$

et, si on pose

$$f(\lambda) = \Delta\lambda$$

on obtient que la croissance des valeurs propres de la matrice synaptique est régie par la fonction

$$f(\lambda) = \alpha (\lambda+1)^{2(p+1)} + \beta (\lambda+1)^{2(n+1)} \quad (6)$$

qui dépend des paramètres α , β , p et n .

Pour que \mathbf{A} converge, Bégin (1990) a déterminé qu'il fallait que $f(\lambda)$ soit positive, qu'elle soit décroissante pour des valeurs de λ supérieures à un certain seuil et qu'elle admette un seul maximum dans l'intervalle de valeurs de λ comprises entre 0 et ∞ . Ces conditions imposent les contraintes suivantes sur les paramètres α , β , p et n :

$$\alpha > |\beta| \text{ pour } \beta < 0 \quad (7a)$$

$$p < n \quad (7b)$$

On assure alors la convergence de λ vers une valeur critique λ_c pour laquelle $f(\lambda) = 0$ (donc pour laquelle la matrice synaptique cesse de croître). Cette valeur propre critique est, d'après l'équation (6) :

$$\lambda_c = \left(\frac{\alpha}{-\beta} \right)^{\frac{1}{2(n-p)}} - 1 \quad (8)$$

Proulx et Bégin ont arbitrairement choisi des valeurs de α , β , p et n qui satisfont aux conditions des équations (7a) et (7b). La règle d'apprentissage qu'ils proposent d'utiliser a

pour paramètres $\alpha = 0.005$, $\beta = -\alpha/2$, $p = 5$ et $n = 2p = 10$, ce qui donne une valeur propre critique $\lambda_c = 0.0718$. La forme de l'équation 6 qui en découle est illustrée par la figure suivante :

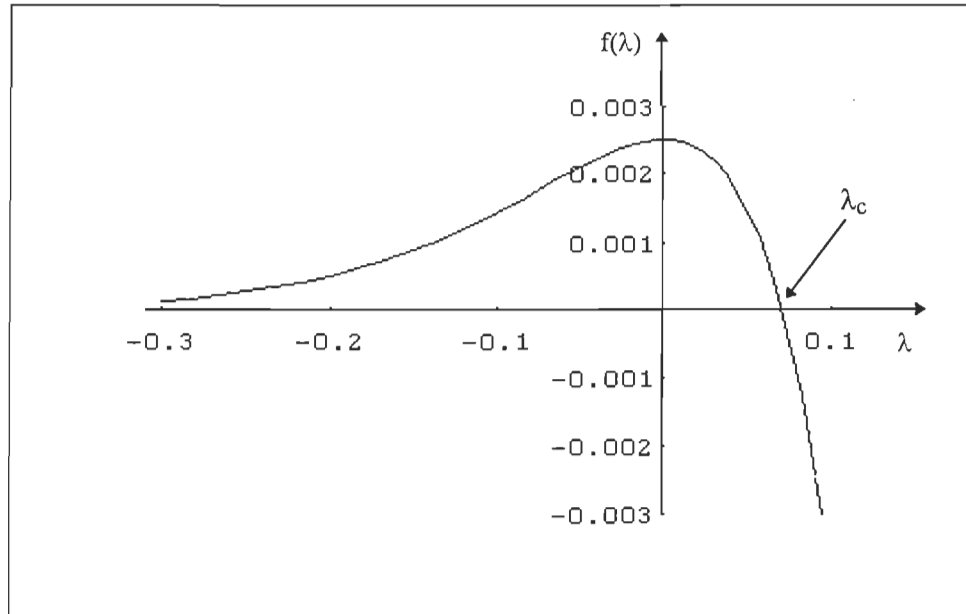


Figure 5. Évolution de $f(\lambda)$ pour les valeurs de paramètres utilisées par Proulx et Bégin.

- *Conséquence du choix des valeurs des paramètres sur la performance du réseau.*

L'idée première dans la présente étude était de chercher les valeurs des paramètres α , β , p et n qui permettraient d'obtenir un comportement optimal du réseau lors des phases d'apprentissage et de rappel. Malheureusement, une recherche théorique pour mettre en évidence chacun des paramètres indépendamment des autres s'est avérée impossible ; les conditions qui régissent ce système étaient insuffisantes pour tenir compte de tous ses degrés de liberté. On a donc essayé de procéder par simulation, en se basant sur les contraintes (7a) et (7b), afin de fixer la valeur de chacun des paramètres autrement que par le choix arbitraire de Proulx et Bégin. Les résultats obtenus lors de ces simulations ont montré la difficulté de trouver des valeurs qui pourraient correspondre à un choix définitif. Néanmoins, ils semblaient indiquer une plus grande sensibilité du réseau envers le paramètre β qu'envers les autres

paramètres. On a alors décidé, pour la suite de ce travail, de fixer les valeurs des paramètres α , p , et n à celles utilisées par Proulx et Bégin, et de varier seulement la valeur du paramètre anti-hebbien β , afin d'évaluer l'impact de ce changement sur la performance du système et de comparer nos résultats avec ceux trouvés par Bégin (1990).

Si on pose $\beta = -\alpha/C$, où C est une constante supérieure à 1, et si $n = 2p$, alors l'équation (6) devient

$$f(\lambda) = \alpha (\lambda + 1)^n \{ 1 - (1/C) (\lambda + 1)^n \} \quad (9)$$

La valeur propre critique pour laquelle $f(\lambda)=0$ devient donc

$$\lambda_c = C^{1/n} - 1 \quad \text{pour } \lambda_c > 0 \quad (10)$$

et la valeur propre pour laquelle $f(\lambda)$ atteint son maximum est donnée par

$$\lambda_{\max} = (C/2)^{1/n} - 1 \quad \text{pour } \lambda_c > \lambda_{\max} > 0 \quad (11)$$

On a alors :

$$f(\lambda_{\max}) = \alpha C/4. \quad (12)$$

On veut trouver une valeur de C pour laquelle la surface de la fonction $f(\lambda)$ soit la plus grande possible tout en restant finie. Cette approche permet de maximiser le nombre de boucles d'apprentissage pour un prototype donné tout en maximisant $f(\lambda)$ à chaque itération et en empêchant la saturation de la matrice synaptique A . La valeur optimale de C dans ce cas

correspond à l'intersection des courbes qui définissent λ_c (équation (10)) et $f(\lambda_{\max})$ (équation (12)); elle est représentée dans la figure 6.

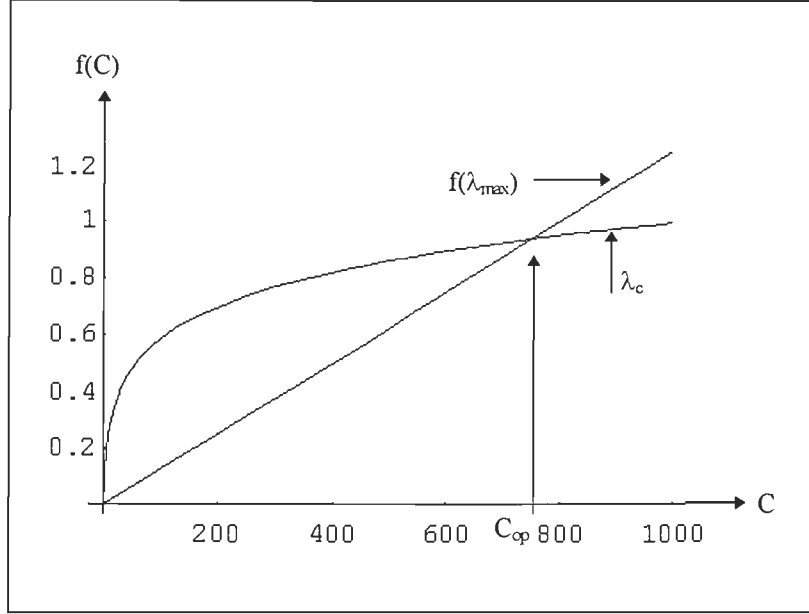


Figure 6. Valeurs de λ_c et de $f(\lambda_{\max})$ en fonction du choix de C .

La valeur optimale obtenue pour C , pour $\alpha = 0.005$, $p=5$ et $n = 2p$ est donnée par $C_{op} \cong 751$, cela correspond à $\lambda_c \cong 0.939$ et $f(\lambda_{\max}) \cong 0.939$. Malheureusement, cette valeur est trop grande pour être utilisée en pratique : La taille limitée des mots utilisés dans les ordinateurs présents fait que pendant les premières itérations, l'importance du terme hebbien dans l'équation 6 l'emporte sur le terme anti-hebbien et réduit l'importance de ce dernier. Il s'ensuit que la matrice synaptique explose avant que le terme anti-hebbien ait pu agir, révélant ainsi une contrainte supplémentaire, d'ordre pratique cette fois ci, sur le choix de C .

Puisque des contraintes pratiques empêchent l'utilisation de la valeur théorique $C = C_{op}$ résultant des choix numériques de α et n dans les éq. (10) et (12), des expériences de simulation se servant de l'éq. (9) avec les mêmes valeurs de α et de n ont été conduites afin de trouver une valeur de C qui soit optimale en tenant compte des contraintes pratiques. Le

Tableau 1 résume les résultats obtenus en essayant de faire l'apprentissage des 9 prototypes utilisés par Proulx et Bégin (1990) pour différentes valeurs de C. Les colonnes indiquent pour chaque valeur de C le nombre de présentations de prototypes requis durant la phase d'apprentissage avant que la matrice synaptique se stabilise. Les valeurs de C choisies ont été limitées à $C \leq 64$ parce que au-delà de cette valeur, on obtenait des erreurs de débordement lors des calculs effectués par le logiciel de simulation.

C	2	4	8	16	20	25	30	32	36	42	52	64
Nombre de présentations	10000	8000	5600	4400	4000	3600	4000	4000	4000	4000	4000	4400

Tableau 1. Nombre d'essais d'apprentissage requis pour neuf prototypes en fonction des valeurs de C.

La figure 7 qui donne une représentation graphique de ces résultats montre que le nombre de présentations requis passe par un minimum pour $C=25$ et tend à se stabiliser autour de cette valeur par la suite.

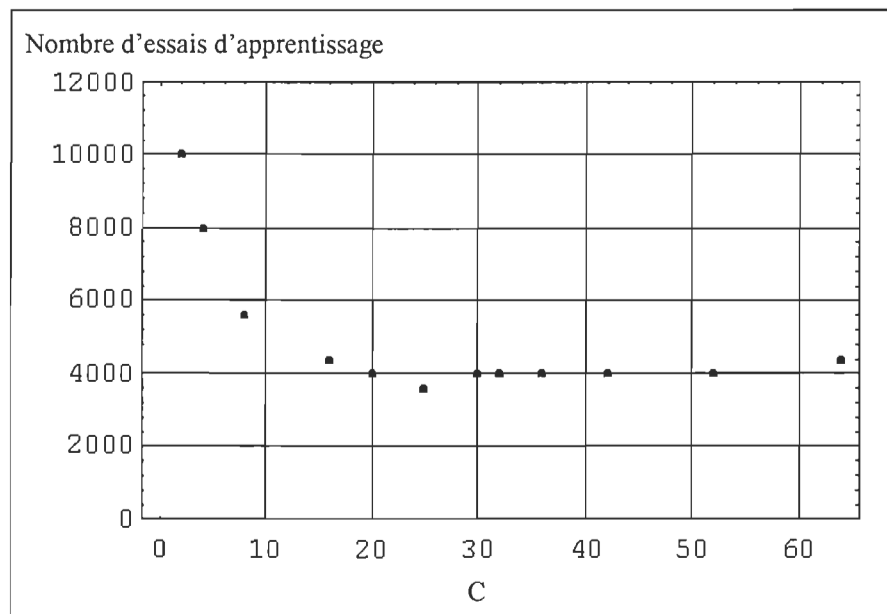


Figure 7. Nombre d'essais d'apprentissage requis pour neuf prototypes avant que la matrice synaptique se stabilise en fonction de la valeur de C.

Il apparaît donc que la valeur $C = 25$ est un choix optimal lorsque les valeurs des autres paramètres sont $\alpha = 0.005$, $p = 5$ et $n = 2p = 10$. On obtient alors $\lambda_c = 0.3797$, $\lambda_{\max} = 0.287$ et $f(\lambda_{\max}) = 0.03125$. La courbe qui suit montre la forme de $f(\lambda)$ obtenue avec ce choix de paramètres via l'éq. (9) :

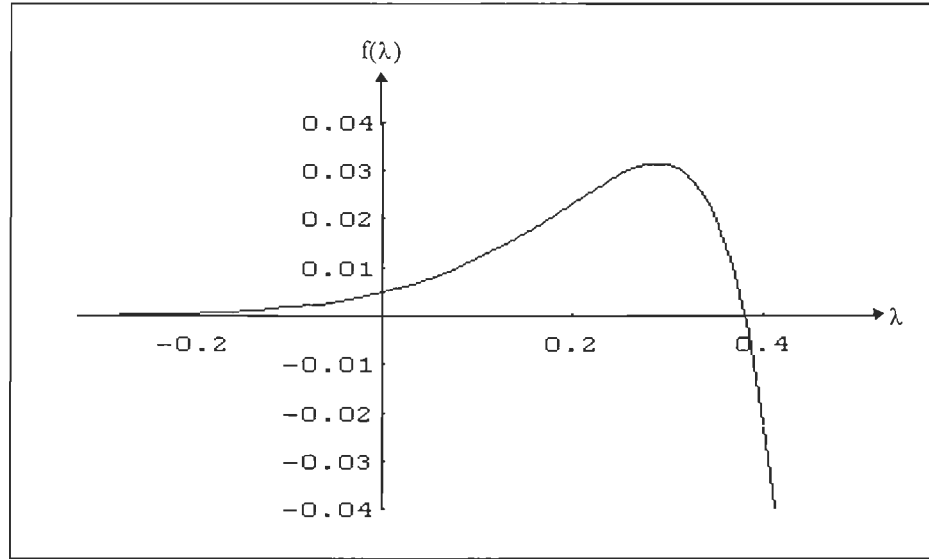


Figure 8. Courbe de $f(\lambda)$ pour $C=25$.

La figure 9 donne un exemple de l'évolution de la valeur propre de la matrice synaptique en réponse à l'apprentissage d'un seul stimulus. On remarque une vitesse de convergence de la valeur propre vers la valeur critique $\lambda_c = 0.3797$ plus rapide (~ 30 présentations) que celle de la figure 4, pour laquelle le nombre de présentations avant stabilisation est presque deux fois et demi plus grand. On constate aussi que la courbure de la figure 9 est plus prononcée, et que la progression de la valeur propre de A passe par trois phases distinctes avant de se stabiliser.

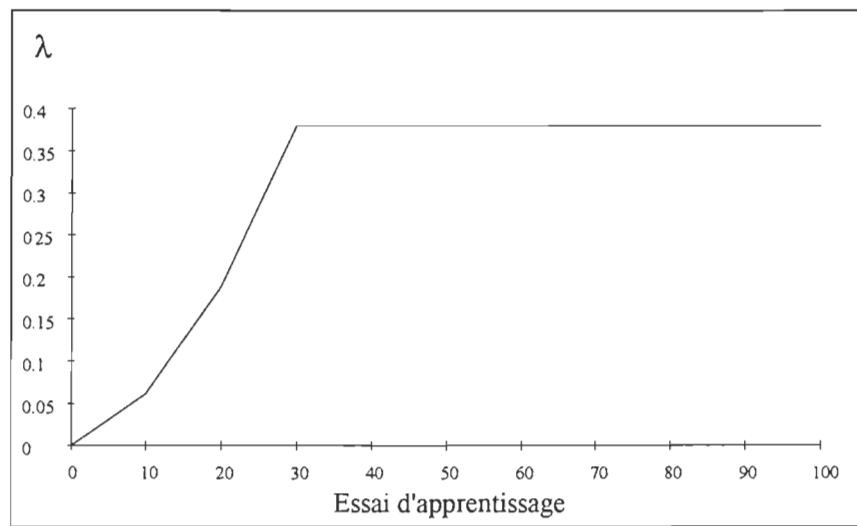


Figure 9. Évolution de λ_c la valeur propre de A en fonction du nombre d'essais d'apprentissage d'un seul prototype ($C=25$).

La mémoire et ses différents aspects.

Dans ce qui suit, on cherchera à comprendre le développement de la mémoire humaine et son organisation pour la conservation des informations afin de mieux percevoir le rôle joué par le paramètre de maintien ψ dans l'éq. (1) et son importance dans les modifications synaptiques.

Plusieurs résultats expérimentaux indiquent que la mémoire n'est pas homogène, mais qu'elle regroupe deux grandes catégories de mécanismes. La première est caractérisée par une capacité limitée et un oubli très rapide; on parle à ce sujet de mémoire à court terme. La deuxième catégorie de mécanismes est caractérisée par une grande capacité et par un oubli progressif, qui peut s'étendre sur des années; c'est la mémoire à long terme. Étant donné la relative stabilité des informations dans le cas de cette mémoire à long terme, le problème essentiel est celui de l'organisation de ces informations.

Un des phénomènes les plus frappant de la mémoire à court terme est sa capacité limitée (en rappel immédiat). Si l'on présente à un sujet une séquence d'éléments à mémoriser et qu'on lui demande un rappel immédiat, on observe qu'en moyenne, environ 7 éléments pourront être rappelés (Guyton; 1984). Le plus curieux est que le nombre des éléments rappelés est à peu près constant, que ces derniers soient des chiffres, des mots, des phrases significatives, ou toutes autres unités familières à la mémoire.

Un des facteurs importants de la transmission synaptique est son épuisement. Ce dernier correspond à une diminution progressive du nombre de décharges du neurone post-synaptique lorsqu'une synapse excitatrice est stimulée d'une façon répétitive. Or il est fréquent au niveau des synapses qu'après une série de stimulations afférentes à fréquence élevée, suivie d'une période de repos, le neurone post-synaptique soit plus sensible aux stimulations ultérieures. C'est le phénomène de la *facilitation* (Guyton 1984). Cette dernière peut persister pendant quelques secondes dans certains neurones, et même plusieurs heures dans d'autres, ce qui

pourrait être vu comme un support de stockage de l'information. La facilitation pourrait ainsi représenter un mécanisme de mémoire à court terme.

Plusieurs théories ont été proposées pour expliquer les mécanismes possibles de la mémoire à court terme. Parmi elles on trouve la théorie des circuits réverbérants (Hebb; 1945). Hebb suppose qu'une information pénétrant dans le système nerveux central y circulerait quelque temps grâce à des boucles de neurones, après quoi surviendraient des modifications synaptiques permanentes au niveau des neurones impliqués dans ces circuits. L'un des principaux arguments en faveur de cette théorie est que toute perturbation brutale du fonctionnement cérébral efface immédiatement les souvenirs à court terme. Ces souvenirs ne peuvent plus être évoqués, même après l'interruption du facteur de perturbation, sauf s'ils ont eu le temps d'être emmagasinés dans la mémoire à long terme.

La figure (10) représente la forme présumée des signaux de sortie d'un circuit réverbérant. La figure montre que l'intensité du signal de sortie s'accroît rapidement, puis diminue lentement jusqu'à un point critique pour cesser brusquement. La durée de cette post-décharge dépend du degré d'inhibition ou de facilitation au sein de l'ensemble de neurones.

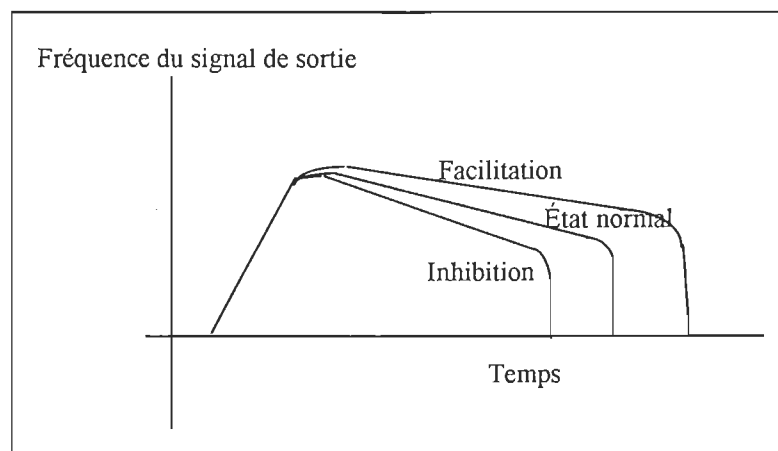


Figure 10. Aspect typique du signal de sortie d'un circuit réverbérant (Guyton; 1984).

Le rôle du paramètre de maintien a été défini par Bégin (1990) comme étant une atténuation à long terme des effets de bruit. Son effet est de faire disparaître progressivement et d'une manière itérative toute composante du vecteur état qui serait hors de l'espace stimuli. Cependant, au su des considérations plus haut, on peut aussi envisager que sa tâche n'est pas seulement limitée à éliminer le bruit, mais peut aussi refléter le/les mécanismes de formation de la mémoire à court et long terme. On verra au chapitre suivant que, dépendant de la valeur donnée à ce paramètre, on observera une augmentation ou une diminution du nombre de présentations pour que l'apprentissage de tous les prototypes soit complété. Ainsi, pour une valeur fixée du paramètre, le nombre de présentations requis pour que la matrice A se stabilise correspondra à un apprentissage à long terme. Ceci nous amène donc à penser que ce comportement peut être perçu comme représentant certaines caractéristiques du développement de la mémoire.

La facilitation, les circuits réverbérants et le rôle joué par le paramètre de maintien suggèrent tous que le développement des mécanismes de stockage de l'information par lesquels le cerveau passe pour faire son apprentissage dépendent de la durée de la transition de l'information entre les neurones avant leur adaptation. Un état d'activation temporaire est engendré lorsque cette durée est courte et correspondra à une mémoire à court terme qui peut devenir une mémoire à long terme si on augmente le nombre de présentations.

Caractéristiques des fonctions de transmission.

Des expériences au laboratoire ont montré les variations que subissait un potentiel d'action lors de la transmission synaptique. Lorsque, au niveau du début de l'axone, ce potentiel excède un certain seuil, le neurone émet un influx qui se propage sans atténuation jusqu'à l'arborisation terminale de l'axone. Cette propagation du potentiel d'action n'est pas continue mais se fait par une régénération successive le long de l'axone (Guyton; 1984). De plus,

l'information véhiculée par les potentiels d'action du neurone est codée par leur fréquence moyenne sur un temps court et non par leur amplitude.

Par le passé, plusieurs auteurs ont essayé de modéliser le comportement précédent à l'aide de fonctions non linéaires. Parmi les plus utilisées on retrouve :

1) La fonction linéaire à saturation définie par

$$L(x_i[t]) = \begin{cases} -1 & \text{pour } x_i[t] \leq -1 \\ x_i[t] & \text{pour } -1 < x_i[t] < 1 \\ 1 & \text{pour } x_i[t] \geq 1 \end{cases} \quad (13)$$

et représentée par la figure suivante :

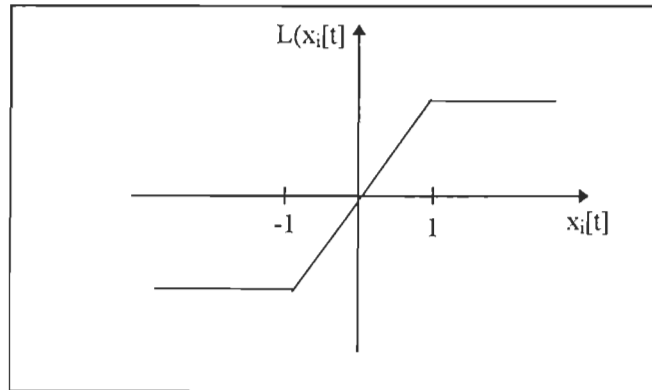


Figure 11. La fonction linéaire saturante.

2) La fonction sigmoïde à gain variable définie par

$$L(x_i[t]) = -1 + \frac{2}{1 + \exp(-\lambda x_i[t])} \quad (14)$$

et représentée par la figure suivante :

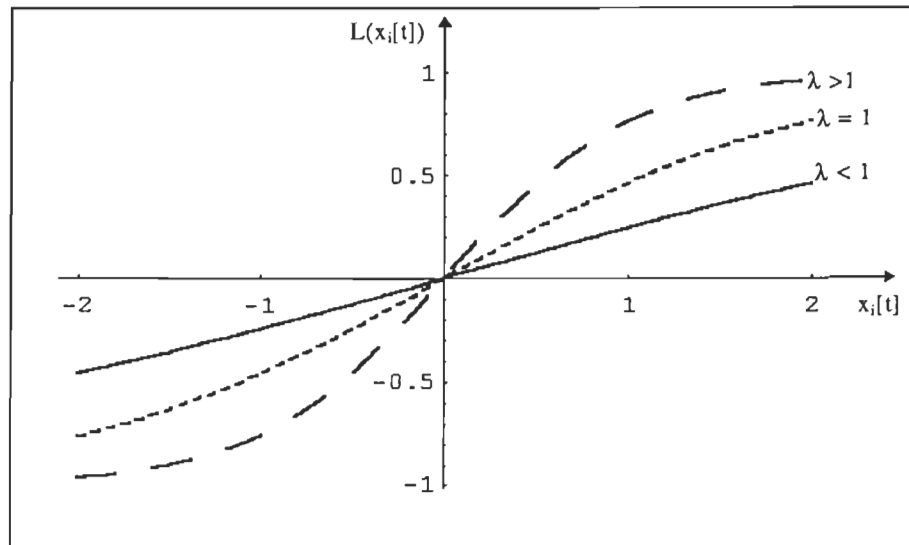


Figure 12. La fonction sigmoïde.

Dans ce travail nous avons aussi utilisé une nouvelle fonction définie par

$$L(x_i[t]) = \tanh[\kappa(A + \psi I)]x_i[t] \quad (15)$$

où la tangente hyperbolique agit sur chaque élément de son argument matriciel. Nous verrons au chapitre suivant que l'usage de cette fonction donne de meilleurs résultats que celui des autres fonctions de transmission. Cette fonction est représentée par la figure suivante :

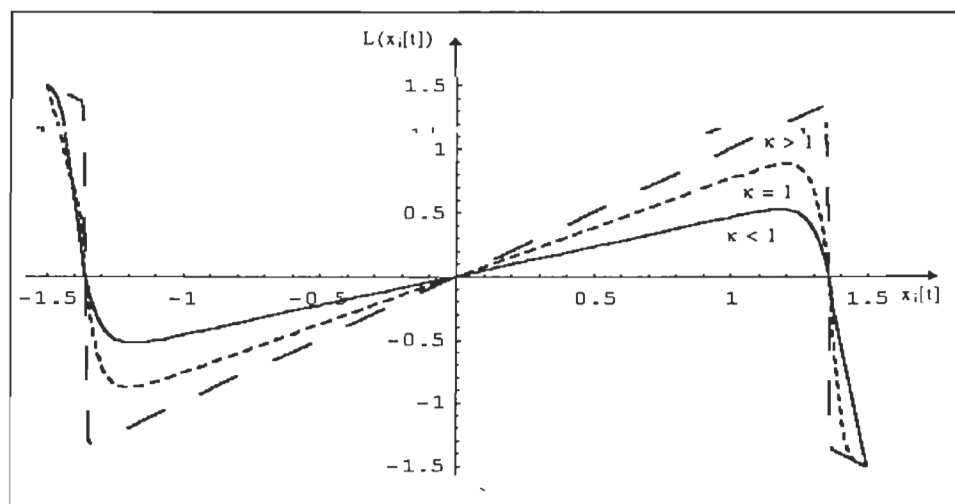


Figure 12a. La fonction tangente hyperbolique multipliée par la fonction linéaire.

CHAPITRE IV

Étude du modèle BSB/Eidos optimisé : Résultats de simulations.

Afin d'étudier l'impact du changement de la valeur du paramètre β sur le modèle Eidos, on soumet celui-ci à des simulations qui vont mesurer sa performance durant les phases d'apprentissage et de la reconnaissance des 9 caractères alphabétiques utilisés par Proulx et Bégin. Les simulations visent à mesurer le nombre d'essais d'apprentissage requis, ainsi que le taux d'erreurs de catégorisation et le temps de réponse du réseau.

- *Méthodologie générale des simulations.*

Les simulations ont été réalisées à l'aide de programmes écrits en langage C, qui appliquent les règles définissant le modèle BSB/Eidos. Les programmes ont été exécutés sur un micro-ordinateur PC 486/66Mhz équipé de 8 Mbyte de mémoire vive.

Le répertoire des stimuli se compose de vecteurs à 35 composantes obtenus à partir de la conversion numérique des neuf lettres majuscules A, E, H, I, N, O, R, S et T. Chaque lettre est décomposée en une grille de 5 x 7 pixels et est décrite par un vecteur dont les coefficients valent 1 ou -1 dépendant de l'état dans lequel se trouve le pixel correspondant.

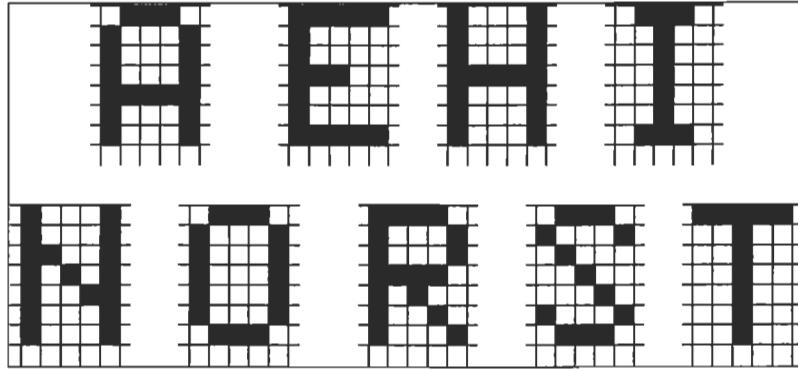


Figure 13. Représentation graphique des neuf prototypes utilisés pour les simulations.

Les exemples d'un prototype sont générés en ajoutant à ce dernier un vecteur bruit aléatoire \mathbf{b} dont les composantes sont indépendantes et identiquement distribuées suivant une loi normale $N(0, \sigma)$. La proportion de bruit ajoutée peut être interprétée comme une distance angulaire entre le prototype et l'exemple (Bégin; 1990). La valeur de l'écart-type σ est donnée par

$$\sigma = \frac{\pi}{\sqrt{35}} \quad (16)$$

où π est la quantité de bruit désirée; ceci permet d'obtenir un vecteur \mathbf{b} normé. Les proportions de bruit retenues dans ce travail varient entre 0% (pas de bruit) et 100% de l'amplitude du prototype.

Le choix des lettres a été fait de manière à englober plusieurs situations de corrélation possibles. Pour l'ensemble proposé, le produit scalaire entre les vecteurs représentant les différentes lettres donne un indice de leur rapprochement (Tableau 2) : plus cet indice est grand en valeur absolue, plus grande est la corrélation entre les vecteurs.

	A	E	H	I	N	O	R	S	T
A		0.143	0.371	-0.2	0.486	0.543	0.429	0.143	-0.2
E			0.314	0.2	0.2	0.371	0.486	0.314	0.2
H				-0.486	0.771	0.257	0.486	-0.143	-0.257
I					-0.486	0.143	-0.086	0.429	0.771
N						0.257	0.257	0.086	-0.257
O							0.2	0.486	-0.086
R								0.029	0.029
S									0.2

Tableau 2. Indice de proximité entre les lettres utilisées lors des simulations.

Phase d'apprentissage. Tout d'abord, la phase d'apprentissage des caractères est faite sans ajout de bruit, suivant la procédure suivante :

- 1) Piger au hasard, avec remise et selon une distribution uniforme, l'un des prototypes du répertoire.
- 2) Calculer $x[p]$ selon la formule $(A+I)^p x[0]$.
- 3) Calculer $x[n]$ selon la formule $(A+I)^n x[0]$.
- 4) Ajuster la matrice A par l'ajout de la matrice ΔA calculée suivant l'équation (3) et répéter à partir de l'étape (1) jusqu'à obtenir $\Delta A = 0$. La matrice A de départ vaut 0 .

L'apprentissage se termine lorsque la matrice A est stabilisée, c'est-à-dire lorsqu'il n'y a plus de modification du spectre des valeurs propres entre deux essais d'apprentissage successifs.

Phase de rappel. Un essai de rappel consiste à essayer d'identifier un prototype à partir d'un exemple qu'on présente au réseau, en utilisant la matrice synaptique A obtenue à la fin de la phase d'apprentissage dans l'équation (4). L'exemple est formé à partir d'un prototype du répertoire, modifié par l'ajout d'un vecteur bruit. Le vecteur obtenu subit plusieurs itérations suivant l'équation (4) jusqu'à ce qu'il atteigne un état stable, c'est à dire qu'il ne se modifie plus entre l'itération T et l'itération $T+1$. Cet état constitue alors la réponse catégorique et la valeur T est la mesure du temps de réponse.

- *Simulation 1 : Apprentissage des prototypes.*

Dans cette première simulation, l'apprentissage a été fait directement à partir des prototypes (fig. 13). La figure ci-dessous montre le spectre des valeurs propres obtenu après 10, 100, 1000 et 4000 essais d'apprentissage, en utilisant les valeurs de paramètres définies dans le chapitre précédent. Pour illustrer le processus de l'évolution de la matrice **A** ainsi que des valeurs propres, un exemple sera donné en annexe avec des états plus petits.

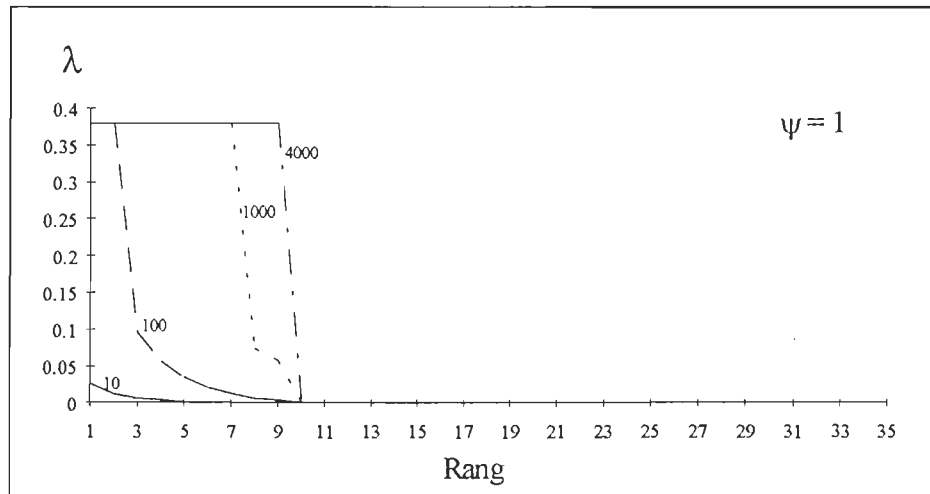


Figure 14. Évolution du spectre des valeurs propres de **A** obtenues après apprentissage par présentation de neuf prototypes. (Les nombres à côté de chaque figure indiquent le nombre d'essais d'apprentissage)

On remarque qu'après 100 essais d'apprentissage, deux valeurs propres ont atteint leurs valeurs maximales, et après 1000 essais, sept valeurs propres ont atteint la même valeur. L'apprentissage cesse après 4000 essais, puisque les neuf valeurs propres définissant l'espace stimuli ont toutes atteint la valeur pour laquelle $\Delta \mathbf{A}$ devient nulle. La matrice **A** obtenue a alors été retenue pour les tests de rappel. On trouve $\lambda_i = \begin{cases} 0.3797 & \text{pour } i = 1, \dots, 9 \\ 0 & \text{pour } i = 10, \dots, 35 \end{cases}$, où i représente le rang de la matrice **A**.

- *Simulation 2 : Exemple d'un essai de rappel.*

Dans cette seconde simulation, on a utilisé la matrice synaptique générée durant la phase d'apprentissage précédente. Le rappel a été réalisé sous différentes conditions. Dans un premier temps, on a essayé de faire le rappel des prototypes appris (fig. 13) sans les corrompre avec du bruit additif, ceci afin de connaître les temps de réponse du réseau dans des conditions idéales d'opération. Ensuite, il s'est agi de produire des exemples en ajoutant à chaque prototype un bruit gaussien dont l'écart-type vaut 10% à 100% de l'amplitude du prototype et de laisser au réseau le soin de les reconnaître. Un exemple de rappel est donné dans la figure 15 où la lettre A est « bruitée » à 100% et où elle est reconnue après neuf itérations.

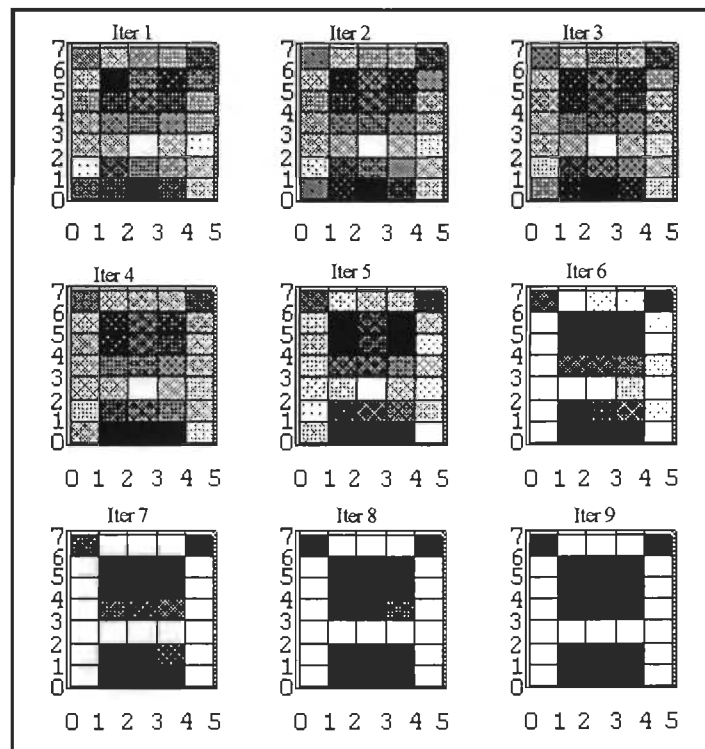


Figure 15. Exemple de rappel avec bruit de 100% pour la lettre A.

Lors d'un essai de rappel, la performance du réseau est mesurée par les erreurs de catégorisation et par les temps de réponse. Les erreurs de catégorisation sont obtenues à l'aide de la distance de Hamming, qui est donnée par

$$D = 0.5 \sum |x_i[0] - x_i[t]| \quad (17)$$

où $x_i[0]$ représente la valeur initiale du $i^{\text{ème}}$ vecteur de test et où $x_i[t]$ représente le vecteur d'état final qui constitue la réponse. Il y a erreur de catégorisation lorsque D est différent de zéro.

La figure 16 donne les résultats obtenus pour les erreurs de catégorisation en fonction des lettres lors d'un essai de rappel avec des quantités de bruit de 25%, 50% et 100%.

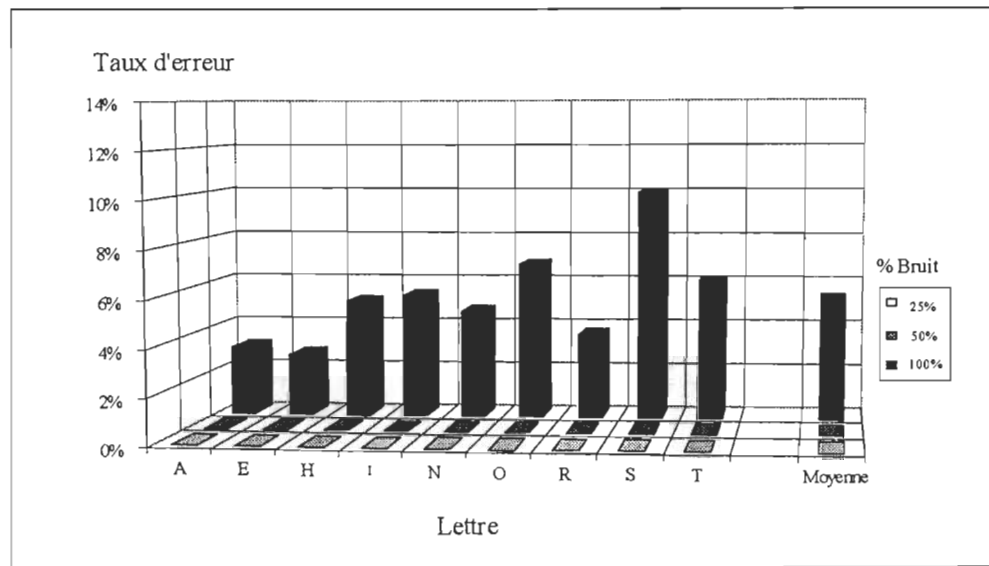


Figure 16. Erreurs de catégorisation pour la règle de transmission $x[t+1]=(A+I)x[t]$.

On constate que le réseau offre une performance parfaite pour des quantités de bruit inférieures ou égales à 50% et un taux moyen de 5.3% pour des quantités de bruit de 100%.

La figure 17 donne le temps de réponse du réseau pour chacune des lettres. La figure montre que le temps de réponse croît en fonction de la quantité de bruit. Sans bruit, le temps de réponse pour que chaque lettre soit reconnue est de 7 itérations. Pour une quantité de bruit de 100%, le temps de réponse moyen est de 13 itérations.

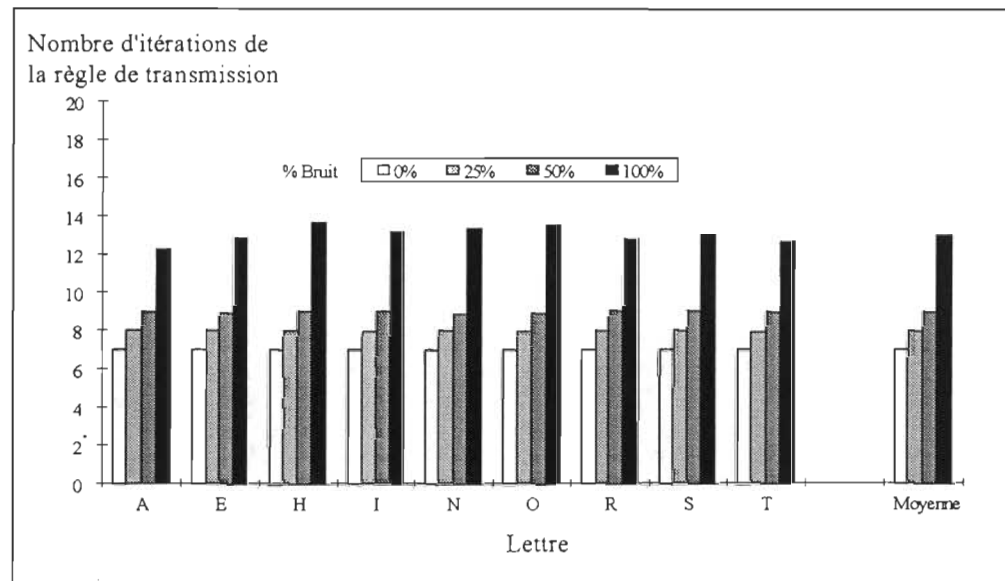


Figure 17. Temps de réponse pour la règle de transmission $x[t+1]=(A+I)x[t]$.

- *Simulation 3 : Apprentissage à partir d'exemples.*

Dans cette simulation, l'apprentissage se fait à partir d'exemples obtenus en ajoutant différentes quantités de bruit aux vecteurs prototypes. La figure 18 donne un exemple des spectres de valeurs propres obtenus pour une quantité de bruit de 25% et pour un nombre d'essais d'apprentissage de 10, 100, 1000 et 4000. On constate l'émergence de nouvelles valeurs propres dues à la contribution du bruit dans la création de l'espace stimuli, ce qui donne un espace de dimension supérieure au nombre de prototypes à apprendre. On conçoit facilement qu'avec des essais d'apprentissage plus importants, les magnitudes des valeurs propres résiduelles vont augmenter jusqu'à atteindre la valeur propre critique λ_c . Le réseau devient alors incapable de reconnaître les prototypes qui forment son répertoire.

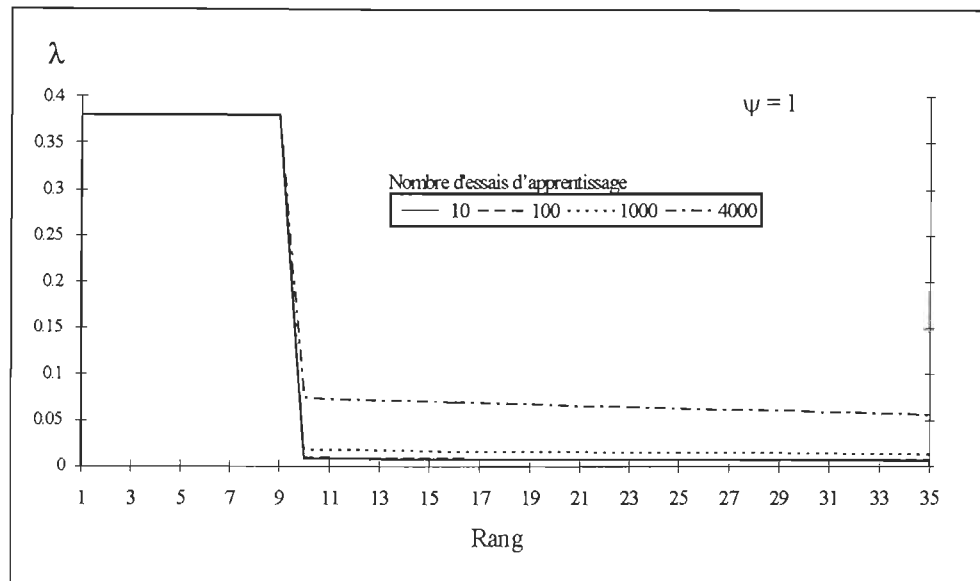


Figure 18. Évolution du spectre des valeurs propres de A pour une proportion de bruit de 25%. (Apprentissage de neuf prototypes)

Les spectres de la figure 18 montrent que durant les phases d'essais d'apprentissage 10, 100 et 1000, neuf valeurs propres émergent etaturent plus rapidement que les 26 autres ($\lambda_c = 0.3797$). Ces neuf valeurs propres correspondent à l'espace-stimuli désiré.

- *Simulation 4 : Modification de la valeur du paramètre de maintien.*

Cette simulation reprend la règle de transmission utilisée précédemment, mais en changeant la valeur du paramètre de maintien ψ de 1 à 0.95 suivant l'exemple de Bégin (1990); ce choix est arbitraire.

La règle de transmission devient :

$$\mathbf{x}[t+1] = (\mathbf{A} + 0.95 \mathbf{I}) \mathbf{x}[t] \quad (18)$$

Phase d'apprentissage avec des prototypes. La figure 19 montre les résultats obtenus lors de la phase d'apprentissage. On constate que le nombre d'essais d'apprentissage

requis pour former les neuf vecteurs propres de l'espace stimuli a augmenté de 4000 à 5000, ainsi que la magnitude de la valeur propre critique ($\lambda_c = 0.4297$).

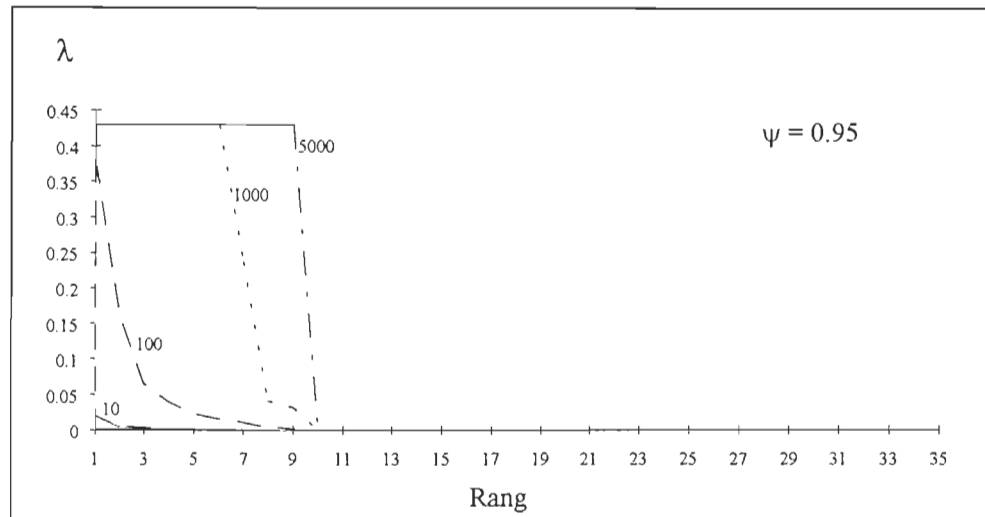


Figure 19. Évolution du spectre des valeurs propres de A lors d'une modification de la règle de transmission. (Les nombres à côté de chaque figure indiquent le nombre d'essais d'apprentissage)

Des simulations avec des valeurs de plus en plus petites de ψ ont révélé que le nombre d'essais requis pour l'apprentissage des neuf lettres augmente à mesure que la valeur de ψ diminue (Tableau 3).

ψ	1	0.95	0.8
Nombre d'essais d'apprentissage	4000	5000	21000
λ_c	0.3797	0.4297	0.5797

Tableau 3. Effets de varier la valeur du paramètre de maintien.

Phase de rappel. On a modifié de même façon la valeur du paramètre de maintien en rappel. Les résultats obtenus pour $\psi=0.95$ sont donnés dans les figures 20 et 21. On constate des temps de réponse plus rapides ainsi que des taux moyens d'erreur plus faibles

qu'auparavant. Par exemple, avec 100% de bruit, le temps de réponse moyen est de 12 itérations et l'erreur moyenne est de 3.5% (par rapport à 13 et 5.3% quand $\psi=1$).

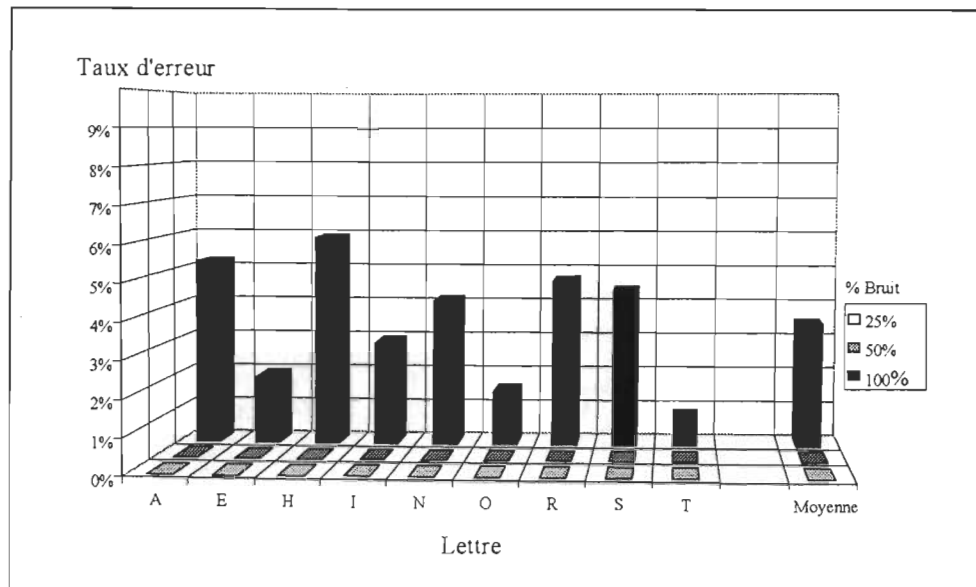


Figure 20. Erreurs de catégorisation pour la règle de transmission $x[t+1]=(A+0.95 I)x[t]$.

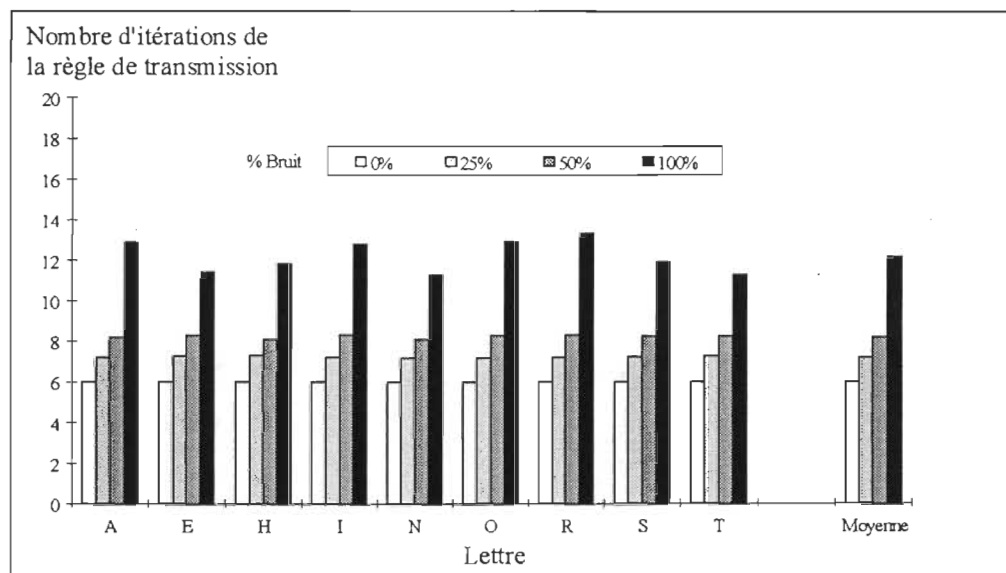


Figure 21. Temps de réponse pour la règle de transmission $x[t+1]=(A+0.95 I)x[t]$.

Phase d'apprentissage avec des exemples. La figure 22 montre que pour une quantité de bruit de 25%, on observe une baisse de la magnitude des valeurs propres

résiduelles introduites par ce dernier en comparaison avec les résultats illustrés dans la figure 18. Cependant, l'importance des valeurs propres résiduelles augmente suivant le nombre d'essais d'apprentissage comme auparavant pour se stabiliser à λ_c .

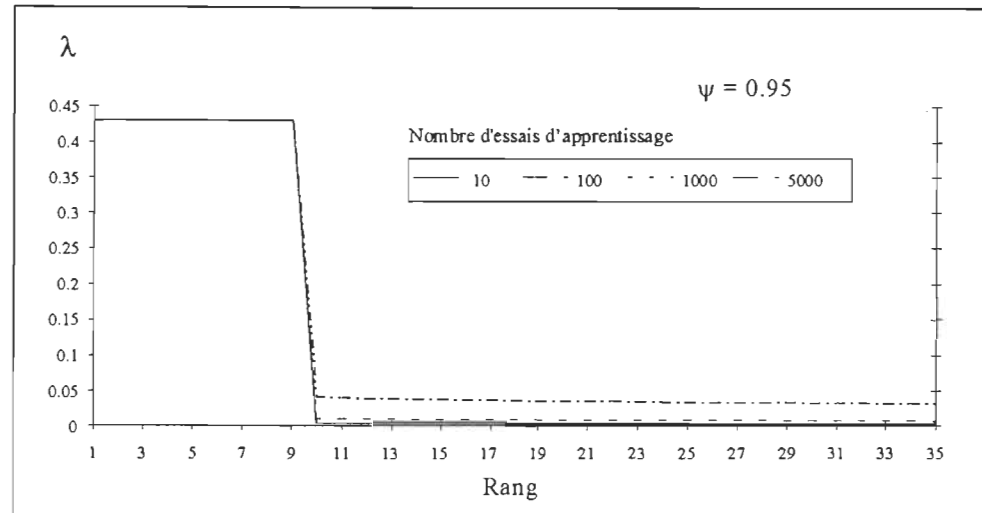


Figure 22. Évolution du spectre des valeurs propres de A pour une proportion de bruit de 25%. (Apprentissage de neuf prototypes)

Phase de rappel après apprentissage par des exemples. La matrice utilisée lors de cette phase est celle obtenue après 5000 essais d'apprentissage avec une proportion de bruit de 25%. Les résultats sont donnés par les figures 23 et 24. Ces dernières représentent les erreurs de catégorisation et les temps de réponse respectivement. On constate que pour 100% de bruit, le taux d'erreur moyen a augmenté à 4.7% (fig. 23), comparativement à 3.5% trouvé lors de l'apprentissage des prototypes (fig. 20), et que le temps de réponse est passé de 12 (fig. 21) à 13 (fig. 24) itérations.

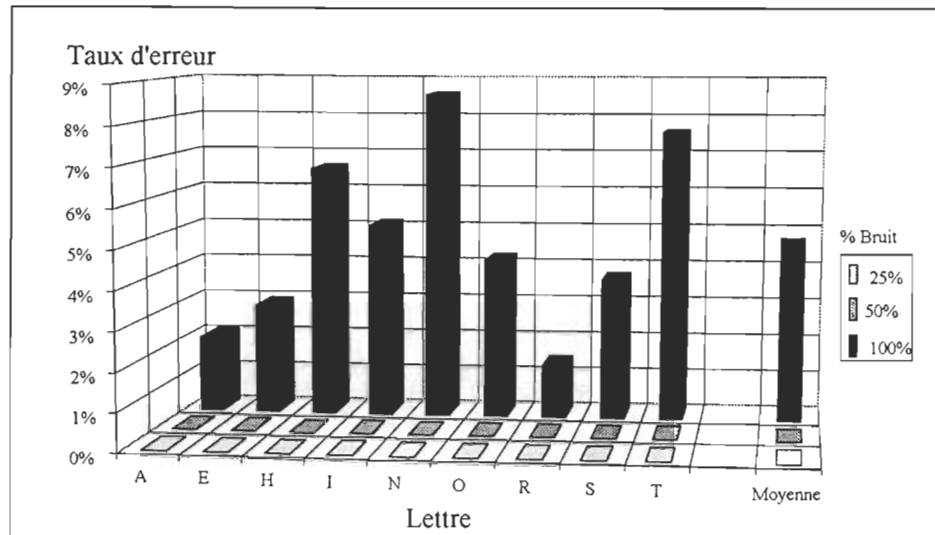


Figure 23. Erreurs de catégorisation pour la règle de transmission $x[t+1]=(A+0.95 I)x[t]$.

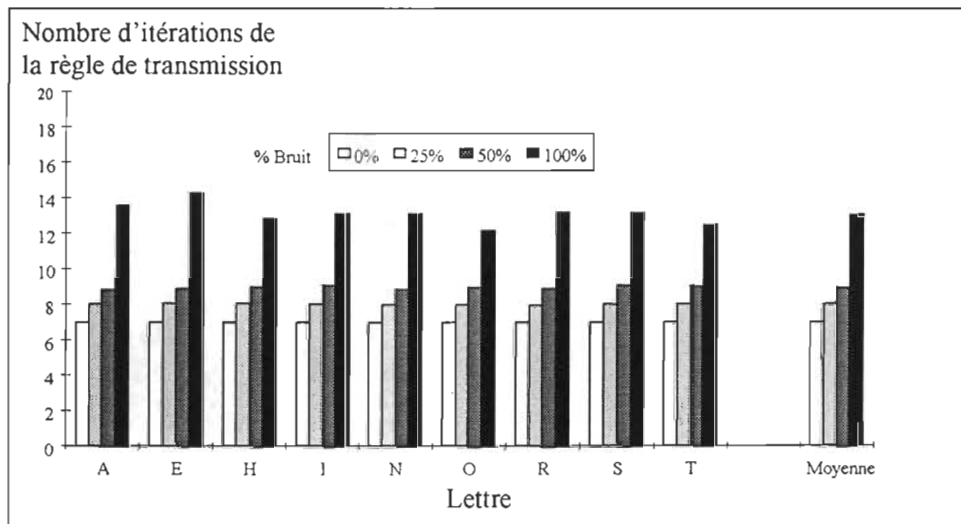


Figure 24. Temps de réponse pour la règle de transmission $x[t+1]=(A+0.95 I)x[t]$.

- *Comparaison des résultats obtenus avec et sans paramètre de maintien.*

Le tableau 4 résume les résultats obtenus. On y compare les données pour une règle de transmission avec $\psi = 1$ aussi bien durant la phase d'apprentissage que durant la phase de

rappel, et avec $\psi = 0.95$. Dans les deux cas, l'apprentissage a été effectué à l'aide des mêmes prototypes.

Règle de transmission	$\psi = 1$	$\psi = 0.95$
Nombre d'essais d'apprentissage	4000	5000
Valeur propre critique	0.3797	0.4297
Temps de rappel sans bruit/100% de bruit	7 / 13	6 / 12
Taux moyen d'erreur pour 100% de bruit	5.3%	3.5%

Tableau 4. Résumé des résultats obtenus lors des simulations.

Utilisation d'une nouvelle fonction de transmission.

Pour analyser l'impact de différentes fonctions de transmission sur le comportement du réseau, plusieurs nouvelles simulations ont été effectuées au niveau des phases d'apprentissage et de rappel. Ces simulations ont utilisé des fonctions de transmission autres que la fonction linéaire à saturation mais ont été réalisées de la même façon que précédemment. Les critères utilisés pour comparer l'impact de ces fonctions sur le comportement du réseau étaient la vitesse de convergence de la matrice synaptique durant la phase d'apprentissage, et le temps de réponse du réseau et le taux d'erreur de catégorisation moyen du réseau durant la phase de rappel.

- *Phase d'apprentissage.*

On a effectué des essais avec des fonctions comme la sigmoïde, la tangente hyperbolique et l'exponentielle décroissante. Les résultats de ces simulations nous ont indiqué l'impossibilité de la stabilisation du réseau. Les réponses obtenues montraient que les valeurs propres ne convergeaient pas et que la dimension de l'espace stimuli augmentait jusqu'à concurrence de celle de l'espace réseau. Comme la non-linéarité de ces fonctions pourrait expliquer la raison pour laquelle le réseau n'arrivait pas à catégoriser, on a décidé de continuer avec la fonction linéaire à saturation comme fonction de transmission durant la phase d'apprentissage, et d'essayer à nouveau des fonctions de transmission non-linéaires durant la phase de rappel.

- *Phase de rappel.*

Lors de cette phase, on a essayé de la même façon que précédemment de remplacer la fonction linéaire à saturation avec différentes fonctions de saturation non linéaires, afin de déterminer le comportement du réseau face à ces changements. On a effectué plusieurs simulations pour lesquelles on a comparé les temps de rappel et les erreurs de catégorisation. On a gardé alors la

fonction de saturation pour laquelle on obtenait les meilleurs résultats. Cette nouvelle fonction est définie comme suit :

$$\mathbf{x}[t+1] = \tanh[\kappa (\mathbf{A} + \psi \mathbf{I})] \mathbf{x}[t] \quad (19)$$

Où $\kappa = 2, 10, 25$ représente le gain, ou pas, de la fonction.

Le choix de cette fonction a été motivé par le désir de changer le comportement de la matrice synaptique afin que ses coefficients interviennent différemment lorsqu'ils sont multipliés par le vecteur d'état. Suivant les valeurs des composantes de ce vecteur par rapport aux valeurs de seuil, la vitesse de saturation sera plus ou moins rapide. Ceci est dû au fait que la tangente hyperbolique agit de manière qu'une valeur qui se trouve loin du seuil augmente plus rapidement que si elle en était proche.

Les figures qui suivent montrent les résultats obtenus pour cette nouvelle règle de transmission avec les valeurs des paramètres trouvés au chapitre précédent. Dans un premier temps, les résultats sont donnés pour ψ valant 1, ensuite lorsque ψ est égal à 0.95.

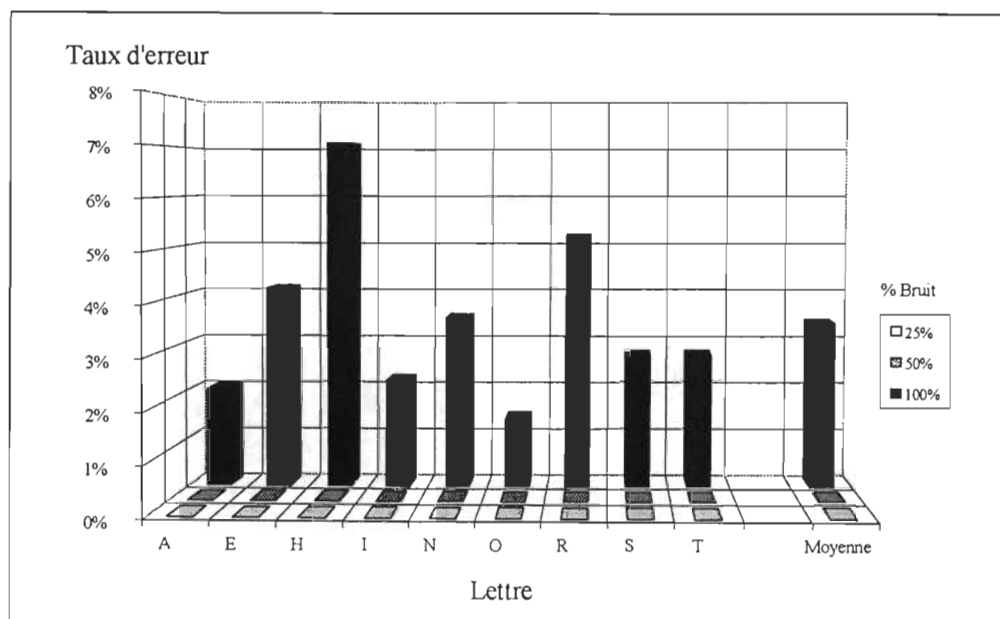


Figure 25. Erreurs de catégorisation pour $x[t+1] = \tanh[2(A+I)] x[t]$.

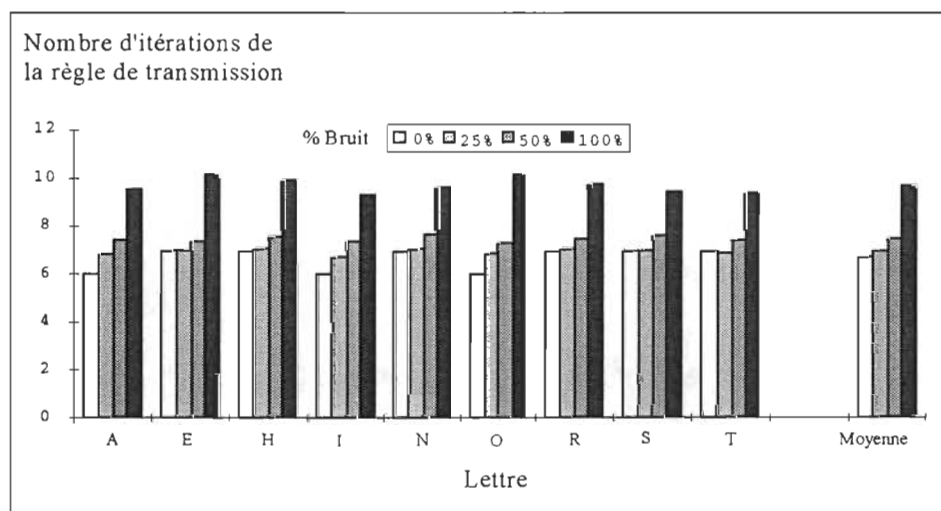


Figure 26. Temps de réponse pour $x[t+1] = \tanh[2(A+I)] x[t]$.

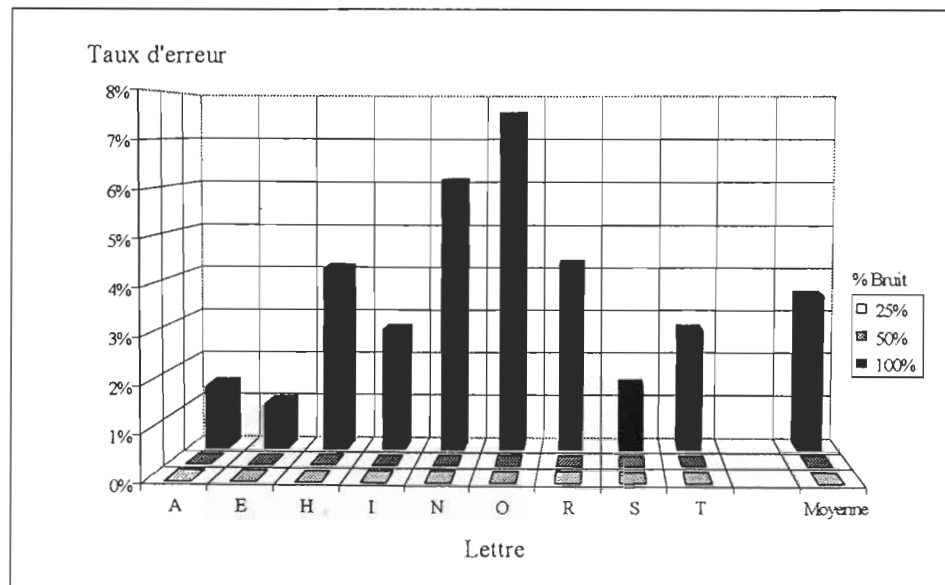


Figure 27. Erreurs de catégorisation pour $x[t+1] = \tanh[10(A+I)]x[t]$.

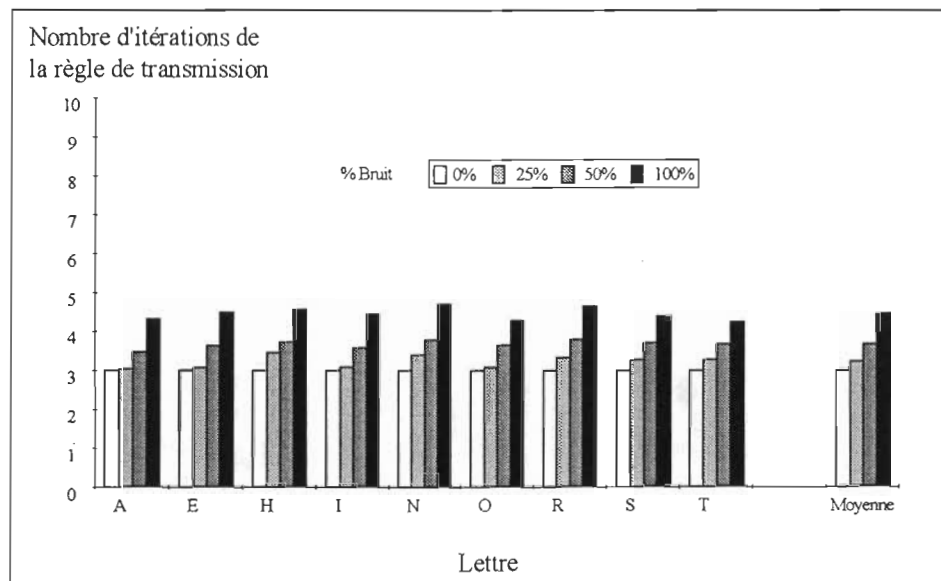


Figure 28. Temps de réponse pour $x[t+1] = \tanh[10(A+I)]x[t]$.

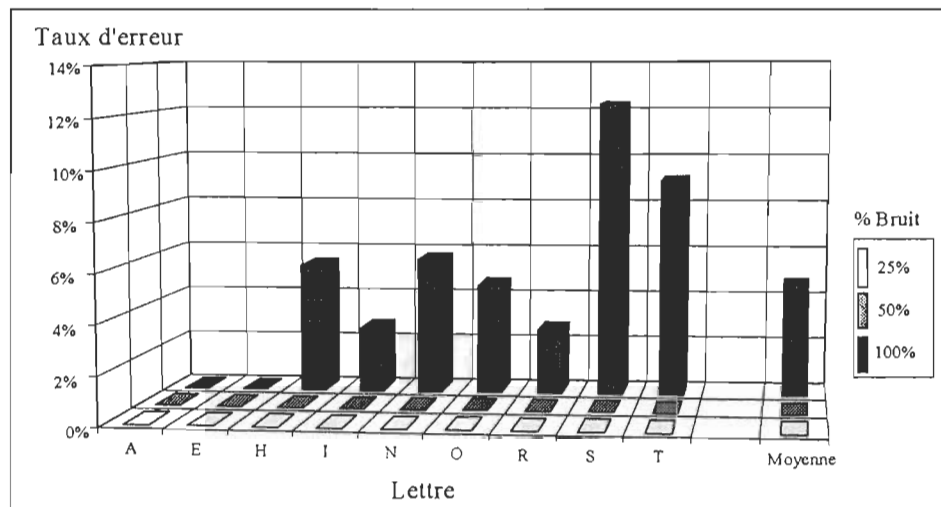


Figure 29. Erreurs de catégorisation pour $x[t+1] = \tanh[25 (A+I)] x[t]$.

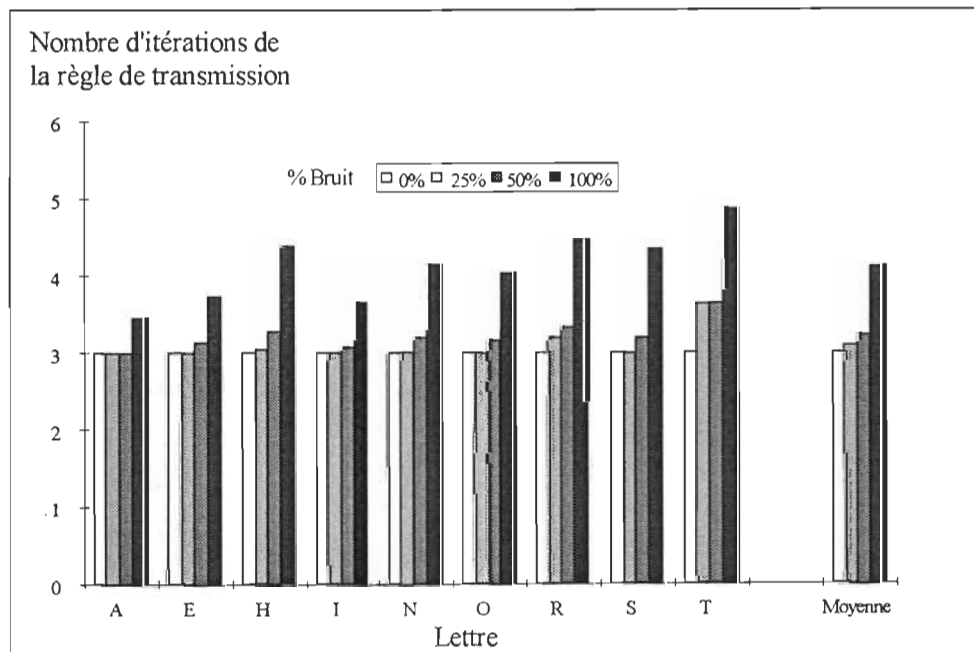


Figure 30. Temps de réponse pour $x[t+1] = \tanh[25 (A+I)] x[t]$.

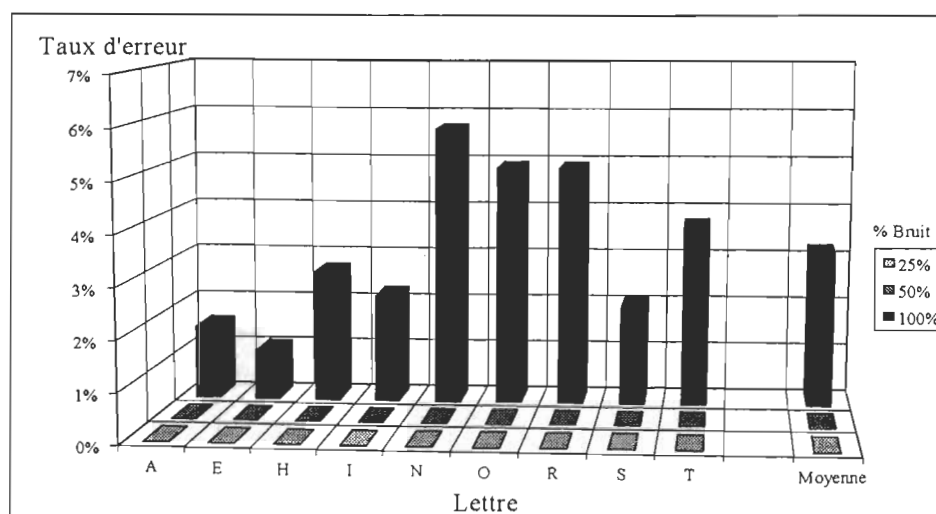


Figure 31. Erreurs de catégorisation pour $x[t+1] = \tanh[2(A+0.95I)]x[t]$

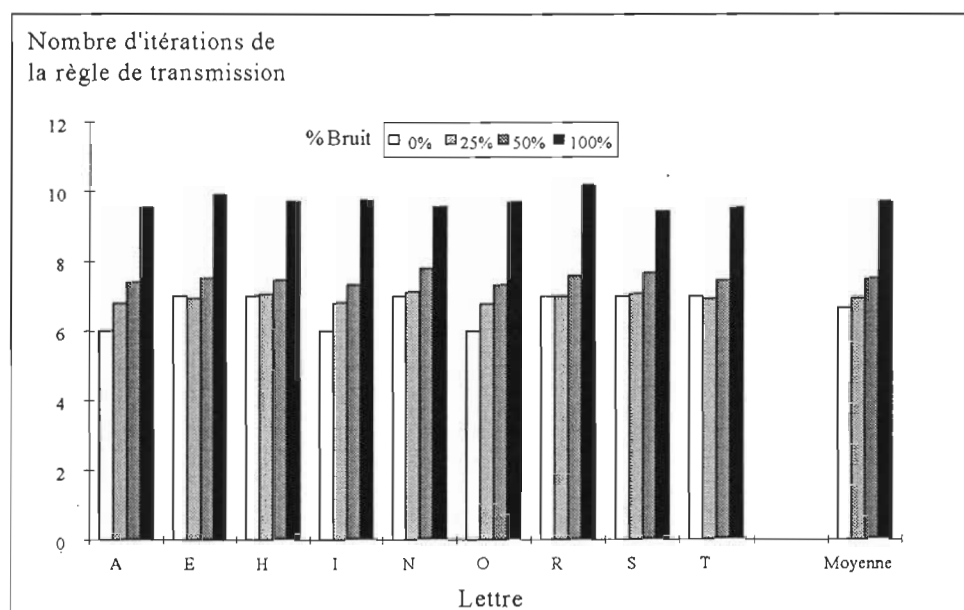


Figure 32. Temps de réponse pour $x[t+1] = \tanh[2(A+0.95I)]x[t]$.

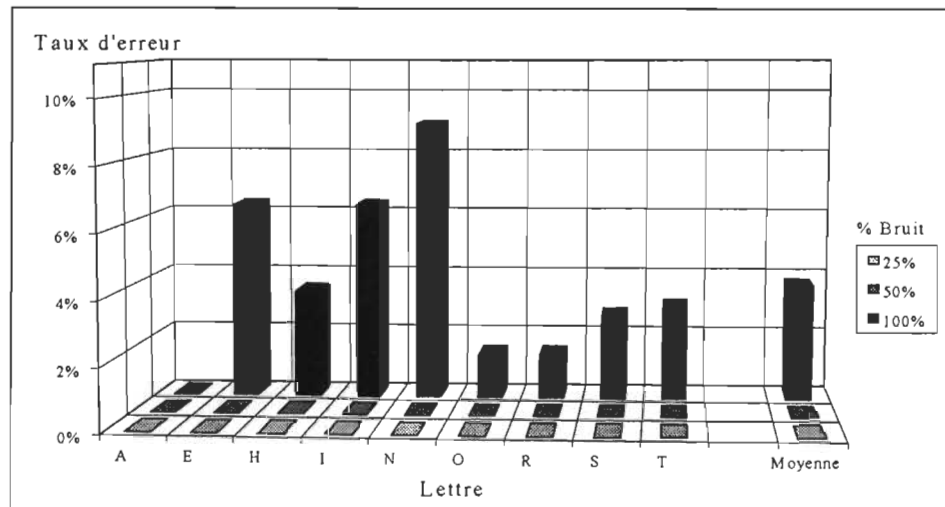


Figure 33. Erreurs de catégorisation pour $x[t+1] = \tanh[10 (A+0.95 I)] x[t]$.

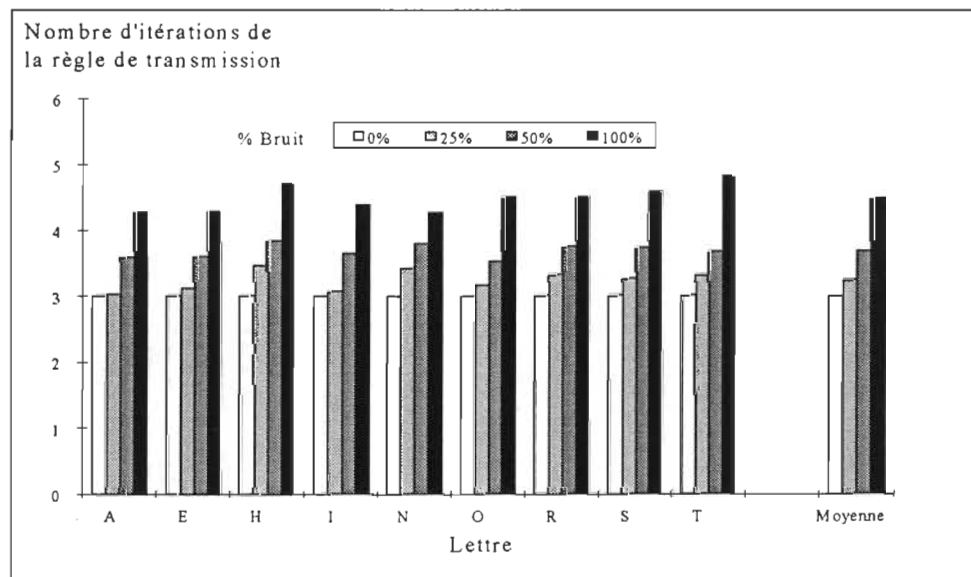


Figure 34. Temps de réponse pour $x[t+1] = \tanh[10 (A+0.95 I)] x[t]$.

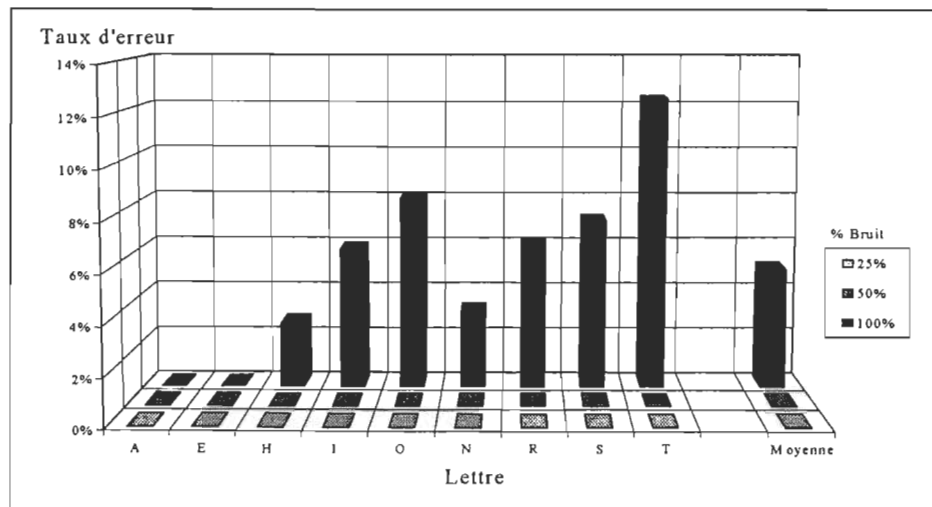


Figure 35. Erreurs de catégorisation pour $x[t+1] = \tanh[25 (A+0.95 I)] x[t]$.

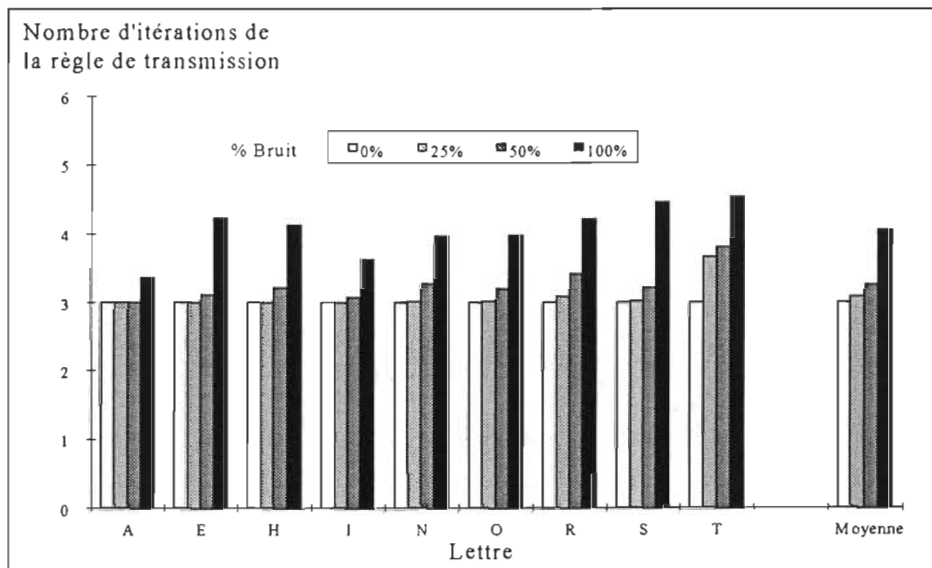


Figure 36. Temps de réponse pour $x[t+1] = \tanh[25 (A+0.95 I)] x[t]$.

On constate que l'utilisation d'une nouvelle fonction de saturation en rappel a amélioré la performance du modèle. Les erreurs de catégorisations moyennes sont plus faibles que celles obtenues lors des simulations précédentes avec la fonction linéaire saturante. Dans le cas où ψ

vaut 1, pour une quantité de bruit de 100%, les taux moyens sont respectivement 3.4%, 3.5% et 4.7% pour κ valant 2, 10 et 25. De même dans le cas où ψ est égal à 0.95 on obtient des taux moyens de 3.2%, 3.7% et 5%. Ces résultats montrent que la possibilité d'employer des nouvelles fonctions de transmission en rappel, autres que la fonction linéaire saturante utilisée en apprentissage peut améliorer la performance du réseau.

De même, on obtient de meilleurs temps de réponse lorsqu'on utilise la tangente hyperbolique en comparaison avec la fonction linéaire saturante. Cette convergence vers des états stables est de plus en plus rapide à mesure que le pas κ augmente ; le temps de réponse moyen sans bruit atteint un seuil de 3 itérations pour des valeurs de κ supérieur à 6, et sa variation moyenne est relativement faible pour des quantités de bruit supérieures.

Le tableau suivant donne une comparaison entre les résultats obtenus pour différentes valeurs de κ et de ψ pour le temps de réponse (en nombre d'itérations) et les erreurs de catégorisation pour une proportion de bruit de 100%.

κ	2	10	25
$\psi = 1$	10 / 3.4%	5 / 3.5%	4 / 4.7%
$\psi = 0.95$	10 / 3.2%	5 / 3.7%	4 / 5%

Tableau 5. Temps de réponse et erreurs de catégorisation pour une proportion de bruit de 100%. (Apprentissage et rappel de neuf prototypes)

On constate que le nombre d'itérations pour reconnaître un caractère dépend de la valeur de κ . Les résultats obtenus (tableau 5) montrent que plus la valeur de κ est grande, moindre est le nombre d'itérations. Cependant, ceci se traduit par un impact négatif sur le taux d'erreurs de catégorisation qui devient de plus en plus important.

CHAPITRE V

Discussion.

Cette recherche avait pour objectif d'augmenter la performance du système BSB avec la règle d'apprentissage Eidos. Les critères de performance retenus étaient le nombre requis d'essais d'apprentissage, le temps de réponse (nombre d'itérations) et les erreurs de catégorisation pour reconnaître un prototype durant la phase de rappel. À cette fin, on a gardé α , p et n aux valeurs arbitraires de Proulx et Bégin et on a essayé de déterminer la valeur optimale de β définie par la règle d'apprentissage Eidos. Par la suite on a évalué l'impact d'utiliser de nouvelles fonctions de saturation.

Les résultats obtenus montrent que l'utilisation d'une nouvelle règle de transmission en rappel est un élément déterminant sur la rapidité de la convergence des exemples vers les vecteurs d'état représentant les prototypes. Ces résultats confirment aussi la robustesse et la pertinence du modèle BSB\Eidos pour des tâches de classification et de catégorisation.

Dans la discussion qui suit, les résultats de simulation sont comparés avec ceux de Bégin (1990).

- *Développement de l'espace stimuli.*

Le comportement du modèle Eidos face à la modification de la valeur du coefficient du terme anti-hebbien n'a pas affecté sa cohérence interne. Le réseau est toujours en mesure de catégoriser les exemples déformés de ses prototypes, mais avec des taux d'erreurs plus faibles ou nuls et cela, même avec des quantités de bruit de 100%.

On a aussi analysé le problème de la capacité du réseau en augmentant le nombre de prototypes à mémoriser. On a remarqué que dans le cas où les 26 lettres de l'alphabet sont utilisées, l'espace stimuli se développe moins rapidement que précédemment. La figure 37 montre le développement des spectres des valeurs propres après 2000, 10,000, 50,000 et 150,000 essais d'apprentissage avec $\psi = 0.95$.

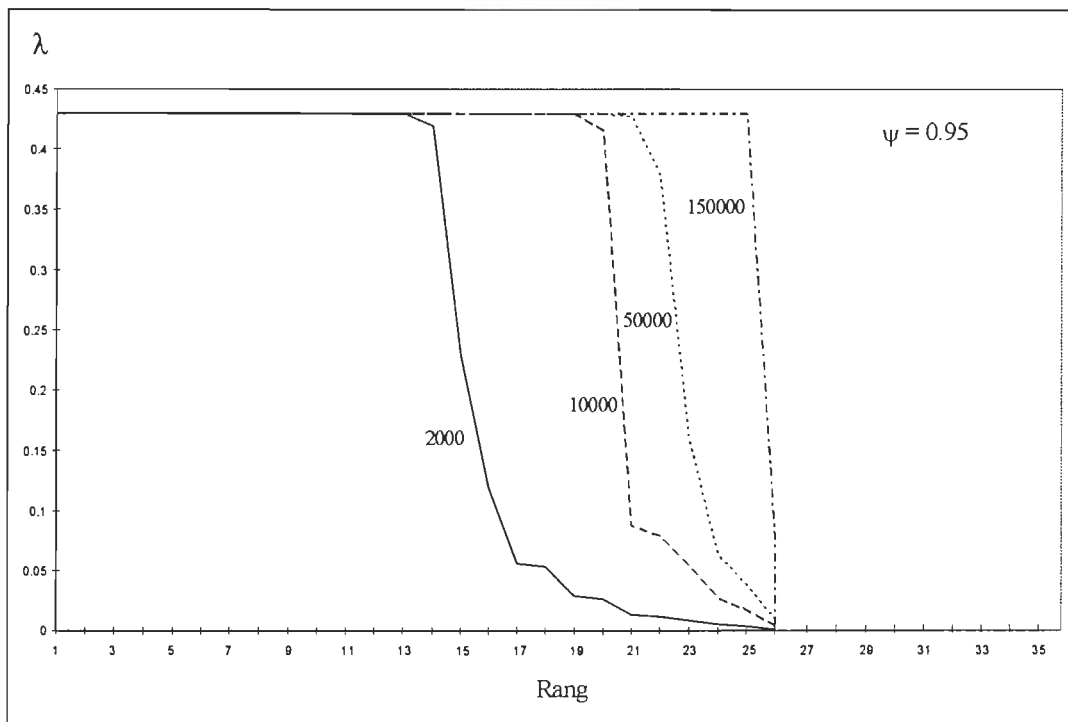


Figure 37. Évolution du spectre des valeurs propres de A pour les 26 lettres de l'alphabet. (Les nombres à côté de chaque courbe indiquent le nombre d'essais d'apprentissage)

On remarque que l'espace stimuli n'a pas fini de se développer après 150,000 essais d'apprentissage. Certains prototypes n'ont pas pu atteindre la valeur propre critique $\lambda_c = 0.4297$. Afin de retrouver les lettres qui n'ont pas pu être mémorisées, on a effectué un test de rappel pour connaître les temps de réponse ainsi que les taux d'erreur pour chacune des lettres.

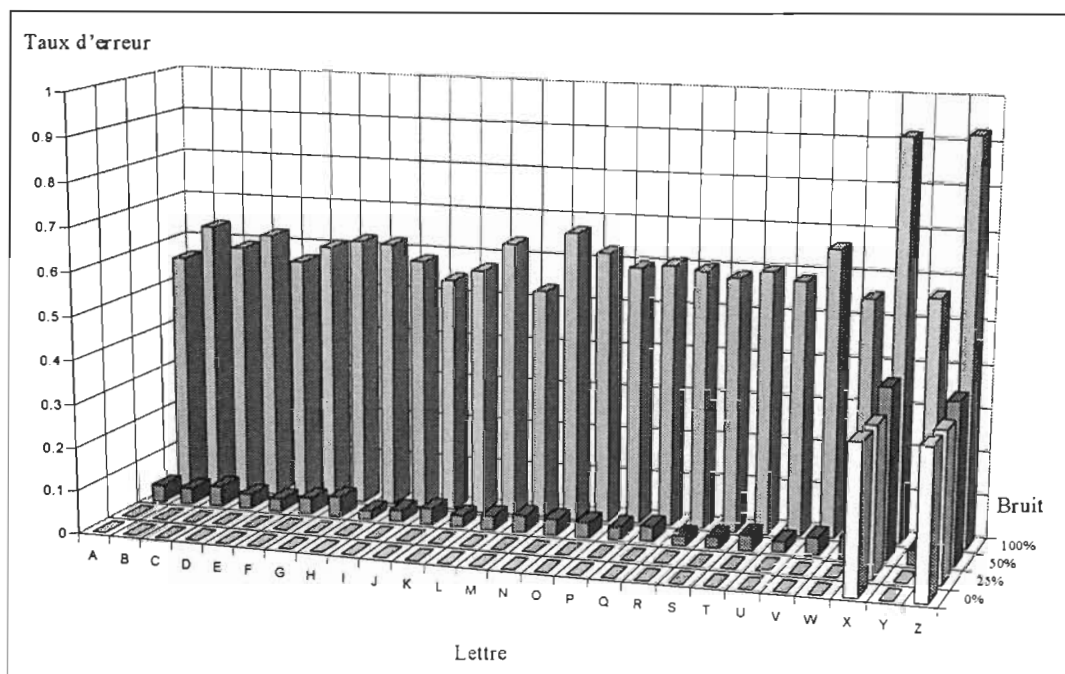


Figure 38. Erreurs de catégorisation pour les 26 lettres de l'alphabet.

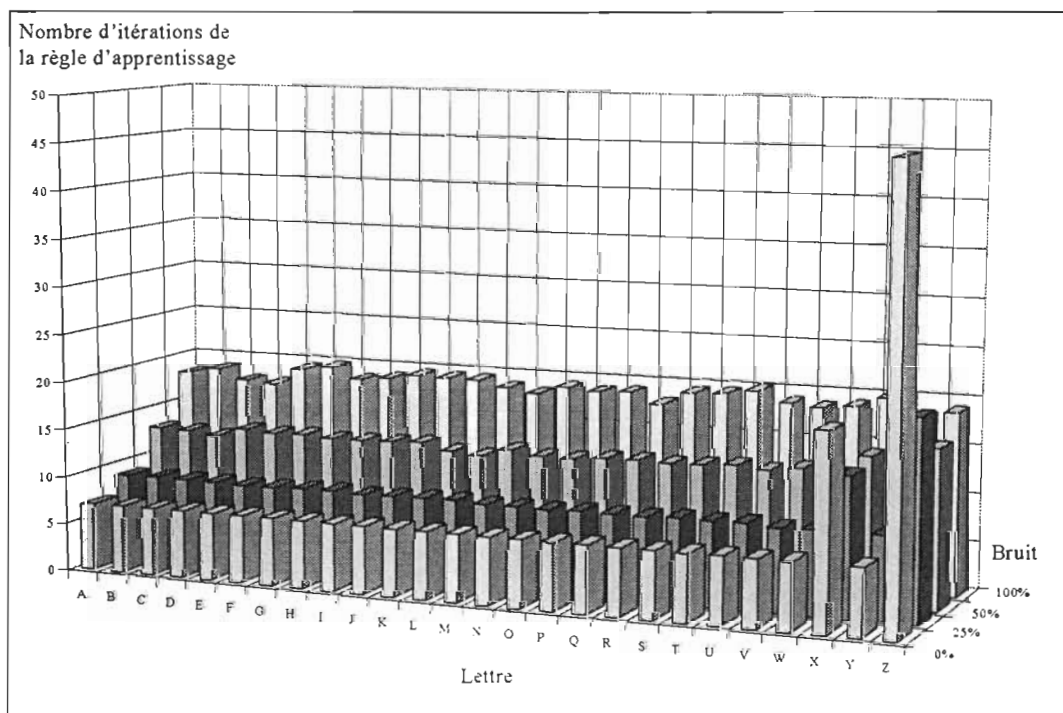


Figure 39. Temps de réponse pour les 26 lettres de l'alphabet.

D'après les figures 38 et 39, les taux d'erreur et les temps de réponse en phase de rappel semblent indiquer que se sont les lettres X et Z que le réseau n'est pas arrivé à mémoriser. Donc, sur les 26 lettres présentées au réseau, seulement 24 ont pu être mémorisées dans un espace réseau de dimension 35. Ce résultat suggère que le nombre de prototypes mémorisables avec le modèle Eidos soit inférieur à la dimension de l'espace réseau. Il révèle aussi des capacités de mémorisation meilleures ($\cong 0.69 n$) que celles trouvées par Hopfield (1982) qui sont limitées par $0.15 n$, où n représente la dimension de l'espace réseau.

Des simulations avec $\psi = 1$, ont donné des résultats similaires. On retrouve les mêmes taux d'erreur pour les lettres X et Z qui sont de 34%, et des temps de réponse qui sont respectivement de 20 et 37 itérations.

- *Performance en simulation.*

On a effectué un grand nombre de simulations afin de comprendre les rôles joués par les paramètres β et ψ utilisés dans les phases d'apprentissage et de rappel. Comme espéré, le système est robuste, et la plupart des variantes fonctionnent d'une manière similaire.

En apprentissage, le problème principal était de connaître le nombre de présentations d'exemples requis avant que la matrice synaptique cesse de se modifier. Pour cela, on a insisté pour que l'erreur commise sur la valeur propre résultante soit sur la sixième décimale. La précision de la matrice utilisée en rappel dépendait de cette limitation.

En rappel, les résultats de simulation étaient conformes à nos attentes. La réponse du réseau était fiable jusqu'à des taux relativement élevés de bruit. La comparaison de ces résultats avec ceux de Bégin (1990) et les performances obtenues sont données par les tableaux suivant :

	Résultats des simulations	Résultats de Bégin et Proulx	Performances
Nombre de présentations en phase d'apprentissage	3600	10000	64%
Valeur propre critique	0.3797	0.0718	---
Temps moyen de réponse sans bruit / avec 100% de bruit	7 / 13	27 / 54	74% / 76%
Erreurs moyenne de catégorisation pour 100% de bruit	5.3%	5.1%	-4%

Tableau 6. Comparaison des résultats pour $\psi = 1$.

	Résultats des simulations	Résultats de Bégin et Proulx	Performances
Nombre de présentations en phase d'apprentissage	5000	10000	50%
Valeur propre critique	0.4297	0.1218	---
Temps moyen de réponse sans bruit / avec 100% de bruit	6 / 12	27 / 49	78% / 76%
Erreurs moyenne de catégorisation pour 100% de bruit	3.5%	4.3%	19%

Tableau 7. Comparaison des résultats pour $\psi = 0.95$.

L'application d'une nouvelle fonction de transfert en rappel augmente la rapidité et la fiabilité de la réponse par comparaison à l'utilisation de la fonction linéaire à saturation. Le tableau suivant résume les résultats pour 100% de bruit par rapport au choix de la fonction.

	Temps moyen de réponse	Erreurs moyennes de catégorisation
$(A + I) x[t-1]$	13	5.3%
$\tanh[10 (A + I)] x[t-1]$	4.2	3.5%
Amélioration de Performance	68%	34%

Tableau 8. Comparaison des performances pour différentes fonctions de transmission. (Cas de neuf prototypes)

- *Développements futurs.*

Plusieurs recherches restent à être développées afin de pouvoir appliquer ce modèle à des situations réelles. Celles-ci devront porter sur un ensemble de problèmes non résolus jusqu'à maintenant. Certains points à élucider ont été déjà décrits par Bégin (1990) et sont :

- 1) Les problèmes de la catégorisation dans des espaces où les prototypes sont arbitraires (Les valeurs des composantes du vecteur prototype sont définies autrement que par 1 et -1).
- 2) Les problèmes de l'invariance aux changements de coordonnées.
- 3) Les problèmes du traitement en temps réel.

D'autres points à éclaircir incluent le concept du paramètre d'oubli (équation (3)). Ce dernier n'a pas été utilisé dans le cadre de cette recherche. Néanmoins, des simulations et une étude préliminaire ont révélé que la règle d'apprentissage devient difficile à analyser lorsque ζ est

différent de 1. Finalement, l'utilisation d'une nouvelle règle de transmission applicable seulement en phase de rappel, et non en phase d'apprentissage, pose plusieurs interrogations au niveau de la validité biologique du modèle.

Conclusions

Le modèle Eidos s'inspire des observations effectuées en neurophysiologie. Ses particularités principales concernent sa forte connectivité, la non-linéarité de ses opérateurs et l'adaptivité de ses connexions. Par ailleurs, aucune forme organisatrice n'est donnée au réseau ; celui-ci s'organise en fonction d'essais successifs. Ce modèle peut ainsi simuler une genèse qui possède beaucoup des caractéristiques fonctionnelles de l'apprentissage par conditionnement : progression du taux de réponses correctes, résistance du système en cas d'erreur, possibilité de réapprendre après détérioration.

L'analyse mathématique de ses propriétés est difficile, la démarche est heuristique et s'appuie sur des simulations. Il n'en reste pas moins qu'il se prête à de nombreuses applications dans des domaines tels la reconnaissance des formes et la compression de données.

Pour conclure, l'exploration du comportement du modèle Eidos au moyen de tests de simulation démontre ses capacités de catégorisation et de classification. Elle nous a permis aussi de vérifier ses caractéristiques telles :

- 1) La stabilisation du réseau après un nombre suffisant d'essais lors d'un apprentissage.
- 2) Le développement progressif des valeurs propres associées aux vecteurs propres de la matrice synaptique suivant les caractéristiques distinctives du répertoire stimuli.
- 3) La possibilité d'apprentissage de nouveaux prototypes, jusqu'à concurrence de $0.69 n$, où n est le nombre de neurones, et ce, même après un apprentissage préalable d'anciens items.
- 4) La capacité du réseau d'opérer en présence de bruit et de reconstruire de façon associative les parties manquantes d'un objet.

Bien entendu, d'autres études pourraient le compléter, permettant ainsi une généralisation du modèle qui résoudrait certains problèmes tels que la reconnaissance des objets indépendamment de leur position ou de leur orientation.

Références

- Anderson, J. A., Silverstein, J.W., Ritz, S. A. et Jones, R. S. (1977).
Distinctive features, categorical perception, and probability learning : Some application of neural model. Psychological Review, 84, 413-451.
- Anderson, J. A. et Mozer, M. C. (1981). Categorization and selective neurons. Dans G. E. Hinton et J. A. Anderson (Eds), *Parallel models of associative memory* (pp 213-236). Hillsdale : Lawrence Erlbaum.
- Bégin, J (1990).
Élaboration d'un nouveau modèle de mémoire associative appliqué au problème de la catégorisation. Thèse de doctorat, Université du Québec à Montréal.
- Cavanagh, P. (1978).
Size and position invariance in the visual system. Perception, 7, 167-177.
- Guyton, A. C. (1984).
Neurophysiologie. (3^{ième} ed.) Masson, Paris.
- Grossberg, S. (1988).
Neural Networks and Natural Intelligence. Cambridge, MIT Press.
- Hebb, D. O. (1949).
The organization of behaviour. J. Wiley & sons, New York.
- Hinton, D. E. et Sejnowski, T. J. (1986).
Learning and relearning in Boltzmann's machines. In D. E. Rumelhart et J. L. McClelland (Eds.). *Parallel distributed processing*. Cambridge, MIT Press.
- Hopfield, J. J. (1982).
Neural Network and physical system with emergent collective computational abilities. Proc. Natl. Acad. Sc., USA 79, 2554-2558.

- Kohonen, T. (1989).
Self organization and associative memory. Springer Verlag, Berlin.
- Murdock, B. B. (1982).
A theory for the storage and retrieval of item and associative information.
 Psychological Review, 89 (6), 609-626.
- Proulx, R. (1986).
Étude des propriétés de sélectivité d'un modèle parallèle de mémoire associative appliqué au problème de la reconnaissance des caractères alphabétiques.
 Thèse de Doctorat, Université de Montréal.
- Proulx, R. et Bégin J. (1988).
Sélectivité et catégorisation dans le modèle « Brain states in a box ». [Résumé].
 Actes du XI^e congrès de la S.Q.R.P, 141.
- Rumelhart, D. E. et McClelland, J. L. (Eds.) (1986).
Parallel distributed processing. Vol. 1, Foundations. Cambridge, MIT Press.
- Vemuri, V. (1988).
 Artificial neural networks : An introduction, in V. Vemuri (Ed.),
Artificial Neural Networks, Theoretical Concepts. Computer Society Press of IEEE,
 pp 1-11.

Annexes

Liste des programmes.

```

/*****
/*          Programmes sources utilisés pour le mémoire :          */
/*          Étude et tentative d'optimisation des paramètres d'un   */
/*          réseau de neurones de type auto-associatif.             */
/*                                                                    */
/*                                                                    */
/* Présenté par : Jamal Lamrani                                     */
/* Date : Juin 1994.                                              */
*****/

```

/* Ces programmes ont été compilés par Turbo C ver. 2.0 sous Dos avec modèle LARGE, et sont fournis sous disquette 1.44Meg.*/

/* Fonctions utilisées pour créer la librairie mathlibl.lib */

```

mxcopy.c
mxdeterm.c
mxdup.c
mxalloc.c
mxfree.c
mxident.c
mxinit.c
mxinv.c
mxinv22.c
mxinv33.c
mxmaxval.c
mxminval.c
mxmul.c
mxmul1.c
mxmul2.c
mxadd.c
mxscale.c
mxsub.c
mxtrace.c
mxtransp.c

```


vadd.c
vcopy.c
vcross.c
vdot.c
vdup.c
valloc.c
vectomat.c
vfree.c
vinit.c
vmag.c
vmaxval.c
vminval.c
vmxm.c
vmxmull.c
vscale.c
vsub.c
vread.c
vwrite.c
math_err.c
mathfree.c
matherr_.c
alloc_li.c
variable.c
eigsrt.c
gasdev.c
house.c
tqli.c

/* Programmes utilisés pour la simulation du modèle Eidos durant la phase d'apprentissage.*/

cpumat.c
readmat.c
eigenlk.c
learn2.c
learnbs.c

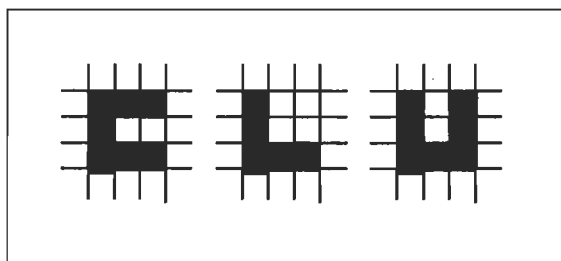
/* Programmes utilisés pour la simulation du modèle Eidos durant la phase de rappel.*/

recall.c
recall2.c

Exemple illustrant l'évolution de la matrice A.

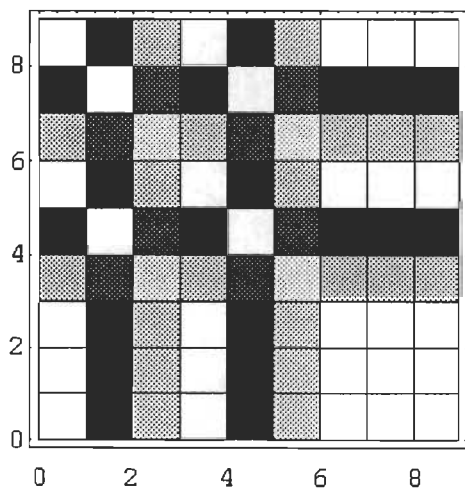
Pour montrer comment évoluent les éléments de la matrice synaptique **A** durant la phase d'apprentissage, et comment se dégagent ses valeurs propres, au fil des itérations, on a procédé avec un exemple de dimension plus restreinte des vecteurs d'état, et cela pour la simplification de cette illustration.

Le répertoire des stimuli se compose de vecteurs à neuf dimensions obtenus à partir de la conversion numérique (-1,1) des trois lettres majuscules C, L et U. Chaque lettre est décrite sur une grille de 3 x 3. La représentation graphique de la matrice **A** et de ses valeurs propres est donnée à chaque 10 itérations. Après 240 itérations, ces derniers se sont stabilisés.



Représentation graphique des trois prototypes utilisés pour l'illustration.

Évolution de **A** après chaque 10 itérations.



Évolution des valeurs propres.

