que sais-je ?

LES RÉSEAUX DE NEURONES ARTIFICIELS

FRANÇOIS BLAYO
ET MICHEL VERLEYSEN

PRESSES UNIVERSITAIRES DE FRANCE

QUE SAIS-JE ?

Les réseaux de neurones artificiels

FRANÇOIS BLAYO

Docteur ès Sciences
Laboratoire de statistique appliquée
et de modélisation stochastique
Université Paris 1 - Panthéon - Sorbonne (France)
Ecole Polytechnique Fédérale de Lausanne (Suisse)

MICHEL VERLEYSEN

Docteur en Sciences Appliquées
Laboratoire de Microélectronique
Université catholique de Louvain (Belgique)

ISBN 2 13 047355 5

Dépôt légal — 1re édition : 1996, janvier

108, boulevard Saint-Germain, 75006 Paris

INTRODUCTION

Pour tout promeneur quelque peu attentif à son environnement, la nature est source d'inspiration. Le tissage d'une toile d'araignée, la construction d'une fourmilière, le chant d'un rossignol, ou la reptation d'une vipère sont autant de comportements naturels auxquels nous ne prêtons qu'une modeste attention. Or, les capacités de traitement de l'information manifestées par les êtres vivants, même les plus simples, sont actuellement hors de portée des systèmes informatiques traditionnels : un ordinateur ne sait pas « voir », « entendre » ou « reconnaître des odeurs ». Contrôler le vol d'un avion par ordinateur requiert des développements de programmes très complexes et très coûteux. Une mouche en fait autant, et contrôle son vol grâce à un volume cérébral microscopique.

Une approche du traitement de l'information consiste donc aujourd'hui à étudier les organismes vivants pour comprendre l'origine et le support de leurs capacités. Pour atteindre ce but, on retient l'hypothèse selon laquelle le comportement adaptatif et la faculté d'acquisition des connaissances sont pris en charge, chez les êtres vivants, par le cerveau et, plus généralement, par le système nerveux.

On espère ainsi, en mimant les structures des systèmes nerveux et les mécanismes de modification de ses constituants (neurones, synapses, etc.), développer de nouveaux outils de traitement de l'information. La technique qui en découle est indifféremment appelée « neuromimétique », « réseaux de neurones » ou

« connexionnisme ». Les réseaux de neurones font ainsi partie du domaine des sciences cognitives (cognitif est dérivé du latin *cognitio* : qui est relatif à la connaissance) qui cherche à développer des modèles de systèmes capables de manifester des capacités d'apprentissage (acquérir des connaissances) et d'adaptation à leur environnement.

On recense aujourd'hui des applications des réseaux de neurones dans des domaines très variés : la reconnaissance de caractères manuscrits, la classification de profilés d'aluminium, l'analyse de l'état de réseaux électriques, l'identification de textures de bitume, etc. Toutes ces tâches ont un point commun : elles sont complexes à modéliser, elles ne requièrent pas une solution unique et exacte, mais plutôt une estimation de la réponse la plus plausible, et enfin elles opèrent sur des données incertaines, toujours entachées de bruit.

Une approche par réseaux de neurones permet d'esquisser des réponses à ces problèmes, en réduisant beaucoup le temps consacré par des ingénieurs à leur analyse. Ce gain est important et son intérêt se situe bien au-delà d'une unique plus-value financière : la complexité de certaines tâches est telle qu'il n'existe pas d'outil pour modéliser leur fonctionnement. Esquisser une solution c'est déjà faire un immense progrès et l'application à certains problèmes a des retombées immédiates sur notre vie courante. La génétique, le soin des maladies du cerveau, les prothèses auditives intègrent partiellement des approches par réseaux de neurones et, demain, bénéficieront pleinement de leur apport.

La compréhension des mécanismes agissant dans le cerveau, comme pour tout système présentant un degré de complexité élevé, ne peut être abordée sous un seul angle de vue. Selon le *niveau de description* retenu, on tentera de comprendre la fonction de l'ensemble (c'est la tâche du psychologue), des différentes

parties (c'est ce qui intéresse un neurochirurgien), des multiples cellules (c'est l'échelle du neurophysiologiste) ou encore des échanges moléculaires (qui intéressent le biologiste moléculaire). Ainsi, pour un même organe, la description peut grandement varier, et le passage d'un niveau de description à un autre peut s'avérer même impossible : analyser la composition de l'encre ne donne rien sur le sens d'un texte écrit.

Mais chaque description est valide, et présente un intérêt en fonction de son *utilité*. Il est inutile et vain de vouloir décrire un cerveau complet à partir de l'échange moléculaire qui se produit entre les membranes des cellules. De même, il est illusoire de penser établir un lien entre un comportement d'une personne regardant un tableau, et les échanges moléculaires qui se produisent à cet instant dans la structure neuronale. En d'autres termes, le choix d'un niveau de description est indispensable, mais il constitue déjà un *a priori* sur ce que l'on cherche à montrer. Loin de le refuser, nous mettons l'accent sur ce choix qui est celui de la cellule nerveuse ou *neurone,* et de l'élément principal d'interaction entre neurones qui est la *synapse*. Notre point de vue sera donc de considérer le cerveau comme un ensemble de neurones et de synapses, ayant chacun leur comportement propre, et connectés selon des schémas variables. Les *cellules gliales*, qui constituent la deuxième classe de cellules présentes dans le cerveau, jouent un rôle fondamental dans la structure et l'entretien du système nerveux, mais ne paraissent pas traiter l'information électrique comme le fait un neurone. Elles ne font pas partie des modèles que nous présenterons par la suite.

Cet ouvrage n'a pas pour objet de décrire exhaustivement le domaine des réseaux de neurones. Il est destiné à donner un aperçu des principaux modèles actuels, et surtout de la démarche de modélisation

employée par les experts du domaine. Il débutera par une présentation des éléments et des structures qui composent le système nerveux. Cela nous permettra de passer en revue les principales fonctions connues, comme la perception, l'apprentissage, le raisonnement et l'action. Nous passerons alors dans le domaine de l'informatique où nous soulignerons la différence qui sépare la machine de l'être vivant, en termes de traitement de l'information. Dans une seconde partie, nous retracerons les grandes étapes de l'évolution des réseaux de neurones. Nous insisterons sur les origines de cette technique et sur quelques étapes qui ont jalonné son histoire. La troisième partie sera consacrée aux notions d'apprentissage et de raisonnement. Nous décrirons les modèles fondamentaux qui permettent de reproduire certaines fonctions telles que la mémoire des associations ou la catégorisation. Nous examinerons ensuite, dans la quatrième partie, les mécanismes de la perception et de la représentation, qui permettent d'acquérir les signaux émis par l'environnement, et de les transformer en une activité cérébrale représentative. Enfin, la cinquième et la sixième partie seront l'occasion de parler de quelques applications, et d'esquisser les futurs développements de ce domaine de recherche passionnant.

Remerciements

Nous tenons à remercier tout particulièrement le P[r] Marie Cottrell, directrice du Laboratoire de statistique appliquée et de modélisation stochastique de l'Université Paris 1, et le P[r] Christian Jutten, directeur du Laboratoire de traitement d'images et de reconnaissances de formes de l'Institut National Polytechnique de Grenoble. Ils ont tous deux relu ce document avec une grande attention et leurs remarques ont très largement contribué à l'amélioration de son contenu.

Tous nos remerciements vont également à nos collègues chercheurs qui participent activement à l'évolution de ce domaine,

ainsi qu'à nos proches qui, avec bienveillance et lucidité, en écoutent le récit : E. Amaldi, R. Berger, G. Baudat, J. Bullier, P. Comon, J. Cabestany, Y. Cheneval, D. del Corso, P. Demartines, J.-C. Fort, K. Goser, A. Guérin-Dugué, M. Hasler, J. Hérault, J. J. Hopfield, P. Jespers, C. et P. Lehmann, P. Levy, A. Marion, E. Mayoraz, J.-D. Nicoud, E. Oja, J. Pagès, N. Reeves, R. Reilly, L. Tettoni, P. Thiran, P. Thissen, C. Touzet, V. Tryba, E. Vittoz, J. L. Voz.

Il va de soi que nous ne pourrons ici nommer l'ensemble des personnes qui nous ont apporté leur soutien et leurs connaissances. Ils se retrouveront dans les lignes qui suivent, et sauront que nous leur sommes reconnaissants de toutes les discussions formelles et informelles partagées ensemble.

Chapitre I

Notions de base

Les motivations qui conduisent à étudier le système nerveux sont présentées ici. Elle permettent de dégager les concepts sur lesquels l'intelligence artificielle s'est bâtie, et de souligner ses limites. L'approche connexionniste est ainsi placée dans un cadre qui permet d'en saisir l'originalité et de préciser les postulats sur lesquels elle repose.

I. — Le neurone biologique

Le neurone est une cellule vivante, qui peut prendre des formes variables : pyramidale, sphérique ou étoilée. Sa forme est définie par une *membrane* qui sépare l'intérieur du neurone (ou cytoplasme) de l'extérieur. Il contient, dans son *soma*, un noyau détenteur des gènes, et son cytoplasme recèle des protéines. Il est également constitué de prolongements qui lui permettent d'établir des liaisons avec d'autres cellules. Les prolongements qui reçoivent les signaux en provenance d'autres cellules s'appellent des *dendrites*. Le prolongement, unique, qui diffuse le signal du neurone vers d'autres cellules est appelé *axone*. Il peut se diviser à son extrémité pour entrer en contact avec un grand nombre d'autres cellules. Le contact peut également être établi avec des muscles, par exemple, si l'on a affaire à un moto-neurone. Ce contact n'est pas une jonction directe entre la

membrane d'un neurone et les membranes de ses voisins, ou les tissus des muscles ; il est assuré par un élément de jonction, appelé *synapse*, qui joue un rôle essentiel dans la transmission, mais aussi dans la modulation des signaux qui transitent dans le système nerveux.

II. — La synapse

Lorsqu'on observe au microscope la jonction entre deux neurones, on constate que celle-ci n'est pas continue, sans rupture, comme l'entendrait un électricien ; au contraire, on remarque un espace entre les membranes de ces deux neurones. Un point de jonction est ainsi un lieu formé de la terminaison d'un neurone, de la surface de contact d'un autre neurone et de l'espace séparant les deux *(fente synaptique)* ; leur ensemble constitue une synapse. Le système nerveux n'est donc pas un magma de cellules fusionnées, comme le prétendaient les histologistes du XIX[e] siècle. C'est à l'Espagnol Santiago Ramon y Cajal que l'on doit cette observation exceptionnelle que chaque neurone est une entité cellulaire individuelle. Il peut être ainsi considéré comme une brique élémentaire du système nerveux, relié à ses semblables par des synapses. Plus généralement, on trouve dans le système nerveux des jonctions synaptiques entre axones et corps cellulaire, mais également entre axones et entre synapses et dendrites. Cette richesse de possibilités donne une idée de la complexité des structures pouvant se constituer au gré de l'évolution et de l'adaptation.

III. — Les cellules gliales

Les *cellules gliales* ne paraissent pas jouer un rôle dans le traitement de l'information nerveuse, mais sont essentielles pour plusieurs raisons. Elles assurent la

structuration du cerveau, s'offrant comme un soutien des réseaux de neurones. Dans cet échafaudage, les neurones s'insèrent et développent leurs jonctions. Ces micro-réseaux peuvent également être cloisonnés par les cellules gliales. Elles effectuent aussi des tâches *d'élimination* des rejets, qui évitent un empoisonnement des neurones. Enfin, les cellules gliales s'enroulent autour de l'axone de certains neurones, constituant ainsi une *gaine de myéline* qui accélère la conduction des signaux qui transitent sur l'axone.

IV. — Codage de l'information

L'ensemble des éléments que nous avons décrits constitue la base du système nerveux. Les cellules gliales et les neurones sont étroitement intriqués. Les neurones développent leur arborisation dendritique et axonale pour établir des chemins de communication afin d'échanger des signaux. Ces derniers sont de nature électrique, sauf à l'endroit de la jonction synaptique où le médiateur est de nature chimique. Le neurone, comme toutes les autres cellules du corps, crée une différence de charges ioniques entre l'intérieur et l'extérieur de sa membrane ; il est positivement chargé à l'extérieur et négativement à l'intérieur. Cette différence est à l'origine du *potentiel de repos* du neurone. Il est mesuré en millivolts (mV) et, prenant l'extérieur de la membrane comme référence à zéro, représente en moyenne — 60 mV.

Le système nerveux est en relation avec le monde extérieur par l'intermédiaire de récepteurs de différentes natures : lumière, pression, température, étirement... L'information reçue doit être transformée en un signal électrique qui sera ensuite traité pour l'action. Or, dans le système nerveux, on constate schématiquement qu'il y a deux types de codages : un codage en amplitude et durée *(potentiel de récepteur)* qui

concerne les stimulations provenant de l'extérieur du système nerveux, et un codage en fréquence et durée *(potentiel d'action)* réservé à la représentation de l'information dans le système nerveux.

Lorsqu'une stimulation est provoquée à la surface de la membrane d'un neurone récepteur, elle provoque une inversion de la polarisation (dépolarisation) qui va se propager de proche en proche sur la membrane ; c'est le potentiel de récepteur, qui peut atteindre jusqu'à 10 mV. Il se propage en s'atténuant en fonction de la distance parcourue : c'est une propagation purement *passive*, semblable à une onde. Par exemple, l'étirement d'un muscle est transformé en un signal électrique qui traduit conjointement la force et la durée de l'étirement. L'amplitude du potentiel de récepteur est proportionnelle à l'étirement, et sa durée est proportionnelle à celle de l'étirement. Lorsqu'il atteint le segment initial de l'axone, le potentiel de récepteur, s'il est suffisamment important, produit un potentiel d'action.

Le *potentiel d'action* est un signal qui, contrairement au potentiel de conduction, se propage sans atténuation le long de la membrane du neurone. C'est la propagation locale d'une dépolarisation qui atteint jusqu'à 100 mV d'amplitude. Lorsqu'un neurone reçoit un potentiel de récepteur, il le transforme en un *train de potentiels d'action* dont la fréquence est proportionnelle à l'amplitude et à la durée du potentiel de récepteur.

Lorsqu'un potentiel d'action parvient dans la région terminale d'un axone, il est transformé en un neurotransmetteur, diffusé dans la fente synaptique, qui se propage jusqu'à la membrane du neurone suivant. Il devient le vecteur de l'information à transmettre et provoque alors la génération d'un potentiel synaptique, qui dépend de la quantité de neurotransmetteur produite par le neurone précédent. On constate ainsi que le potentiel d'action ne se transmet pas tel quel, comme sur

un fil électrique, mais qu'il dépend des caractéristiques des synapses qui ont libéré le neurotransmetteur.

Le potentiel synaptique représente un codage similaire au potentiel de récepteur. Il se propage passivement à la surface de la membrane, et peut provoquer une excitation (dépolarisation) ou une inhibition (hyperpolarisation) du neurone. Finalement, le neurone reçoit, sur son soma, une importante quantité de signaux, sous forme de potentiels synaptiques, en provenance des neurones avec lesquels il est en liaison. Une dernière transformation s'opère dans la région de transition entre le soma et l'axone, qui est le siège de la génération du potentiel d'action. L'intégration de signaux qui a eu lieu sur la membrane du neurone produit un potentiel qui, selon son amplitude, provoquera la génération ou non d'un potentiel d'action. C'est une micro-décision *locale*, prise par le neurone, qui est la base du traitement *global* effectué par le système nerveux.

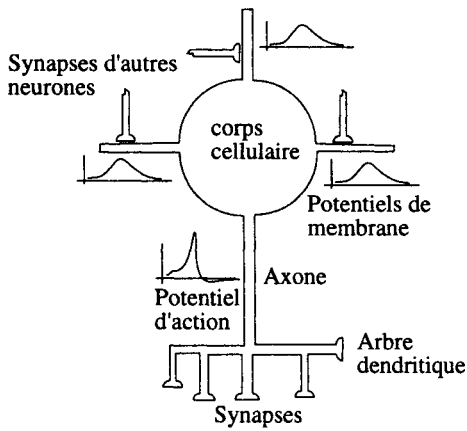

Fig. I.1. — Schéma symbolique du neurone et des potentiels

V. — **Assemblages de neurones**

Un neurone pris isolément, malgré la complexité du traitement qu'il effectue, ne peut prendre en charge la totalité du contrôle d'un être vivant évolué. Sa structure lui permet d'entrer en relation avec des milliers d'autres neurones (de 1 000 à 10 000 synapses selon le type de neurone), et sa seule voie de sortie, l'axone, se ramifie pour diffuser le résultat de la décision à d'autres neurones. Par des contacts multiples, c'est un réseau de connexions très dense qui se construit, et qui permet un échange riche entre des centaines de neurones différents. Il est ainsi possible de mettre en évidence, dans le système nerveux, des structures organisées, qui diffèrent selon la nature de l'information recueillie, et selon le traitement effectué.

VI. — **La perception**

Les structures mises en place par l'évolution et par l'apprentissage dépendent essentiellement de l'environnement. Notre système nerveux est ainsi façonné par l'expérience, ce qui suppose une capacité d'acquisition que l'on nomme *perception*. Celle-ci est donc, en première approche, définie comme la faculté d'acquérir des signaux émis en permanence par l'environnement. Néanmoins, se restreindre à la seule acquisition revient à ne s'intéresser qu'à la sensation, et non à la perception. Par extension, donc, la perception est conçue comme « une *conduite adaptative* qui, au cours de notre vie, se développe pour nous donner des phénomènes conformes aux attentes, aux motivations de l'organisme orienté vers un *but* »[1]. Il n'y a donc pas de

1. Cette définition est empruntée à R. Francès dans *La perception*, PUF, coll. « Que sais-je? », 1992, 8e éd.

perception sans but, mais aussi sans valeur attribuée aux informations recueillies et recherchées dans l'environnement. Le système nerveux oriente la perception, c'est-à-dire qu'il ne se contente pas de tourner le regard ou l'oreille dans une direction hasardeuse, espérant y trouver un indice pertinent. Il traque la stimulation signifiante, sous le filtre de la valeur qu'il y attribue. La perception est donc une faculté active, dynamique, qui n'a rien en commun avec la simple mesure d'une température, ou d'une longueur. Elle agit dès les premiers instants de la réception du signal, transformant les multiples fréquences qui assaillent l'être vivant en signaux électriques qui auront pour effet de *désynchroniser* les décharges cellulaires spontanées et rythmiques qui constituent l'activité permanente du système nerveux.

VII. — **L'apprentissage**

L'apprentissage est un processus d'acquisition de connaissances sur l'environnement. Il va de pair avec la *mémoire*, qui est un mécanisme de rétention de la connaissance et, naturellement, avec la *perception*. L'apprentissage se manifeste à travers un changement de comportement : ce peut être le raffinement d'un geste, l'acquisition d'un raisonnement ou encore la reconnaissance d'une situation. C'est dans le domaine de la psychologie que l'apprentissage a été le plus étudié, sujet dominant pendant plus de quatre-vingts ans : I. Pavlov a développé la notion de conditionnement, E. Tolman a développé l'explication en montrant qu'il n'y avait pas toujours une relation directe entre comportement et stimulation, B. Skinner a proposé les méthodes de conditionnement opératoire libre pour étudier les effets du renforcement, du contrôle discriminant, ou de la punition. Il ressort de ces études

que toutes les modifications comportementales postulent l'existence de la mémoire. Ce terme recouvre deux fonctions distinctes : l'acquisition d'un élément d'information et le rappel (ou l'extraction). Or, le lien, appelé aussi *engramme*, qui s'établit entre l'acquisition et le rappel, s'il se manifeste par un changement de comportement, se fonde probablement sur une modification des propriétés du système cérébral. Il y aurait donc une relation entre le changement de comportement et la modification des propriétés de la synapse et du neurone. Les travaux développés pendant les vingt dernières années dans le domaine de la psychologie et de la neurophysiologie vont dans ce sens. Grâce au développement de modèles formels de la mémoire, de nombreuses simulations informatiques ont permis l'avancée d'hypothèses séduisantes sur la mémoire. Néanmoins, l'engramme lui-même n'a pas été révélé explicitement car les recherches sur l'apprentissage ont essentiellement porté sur les stratégies de l'apprentissage : capacité de mémorisation, vitesse d'acquisition, durée de stockage. Ces dernières, comme nous l'avons souligné, sont mises en relation avec les processus biochimiques qui opèrent à l'échelle du neurone et de la synapse. Ainsi, les recherches actuelles visent à comprendre les mécanismes de modification qui accompagnent le changement de comportement relatif à un apprentissage. Néanmoins, cette démarche ne tient pas compte du fait que les modifications observées (par mesures électriques, chimiques, structurelles) peuvent être le résultat d'activités corrélées au changement de comportement, sans en être directement la cause. Tout en restant conscient de cette limite, l'expérimentateur l'écarte volontairement à cause de la complexité des processus étudiés.

On considère grossièrement que la mémoire repose sur deux processus, facilement identifiables : la mémoire

à long terme et la mémoire à court terme. La mémoire à court terme est très souvent sollicitée dans des jeux de société : par exemple, demander de se souvenir d'une série de mots ou de chiffres prononcés peu de temps auparavant. Le même sujet, sollicité quelques jours plus tard, ne saura généralement pas reproduire cette même série. Selon l'intérêt porté à ces chiffres (numéros de téléphone, date de naissance liée à l'affectif, noms de personnes clefs pour un contrat...), ces informations seront stockées dans la mémoire à long terme et pourront ainsi être rappelées plusieurs jours, voire plusieurs années après leur apprentissage. Cette différence peut se comprendre aisément : toutes les informations n'ont pas la même valeur pour celui qui les reçoit. Le stockage dans la mémoire à long terme repose donc sur un filtre qui sélectionne les stimulations selon la motivation qu'elles suscitent. Ici encore, on peut constater que la mémoire brute, comme on la conçoit pour un ordinateur, n'a rien à voir avec la mémoire biologique qui est subordonnée à *l'attention.* On en vient donc à poser plus pratiquement la question : quels sont les processus mis en jeu pour le codage de l'information dans la mémoire ? Un première réponse a été apportée par le physiologiste D. Hebb qui proposa, en 1949, un modèle simple pour expliquer ces possibles modifications. Nous développerons cette règle et ses dérivés dans la partie consacrée aux modèles de réseaux de neurones pour l'apprentissage. L'étape suivant le développement de l'exploitation de l'apprentissage concerne le mécanisme de *l'inférence* ; ce dernier est à la base de ce que l'on appelle le raisonnement. Il consiste à composer des faits et des règles abstraits afin de produire des conclusions susceptibles de guider le comportement dans des situations inconnues. Ces faits et règles sont issus de l'apprentissage, et semblent interagir avec la représentation de l'environnement élaborée par le système nerveux.

VIII. — **Le raisonnement**

Selon Piaget, le raisonnement est une capacité qui ne se manifeste que très modérément avant l'âge de 7 ans. Ce mécanisme requiert, selon lui, une coordination des instants successifs de la pensée, chacun d'eux étant trop dominant pour établir un courant homogène et fluide. Le jeune enfant aurait ainsi une grande difficulté à voir son propre point de vue comme élément d'un ensemble de points de vue possibles, sur lesquels s'établissent des relations variant selon chaque individu. Le raisonnement est donc considéré comme l'aboutissement de l'évolution des tâches cognitives. Il a ainsi été au centre des travaux de recherche sur l'intelligence artificielle, visant à reproduire le raisonnement logique. On peut d'ores et déjà souligner la différence entre les niveaux de modélisation : l'intelligence artificielle repose sur la modélisation des mécanismes permettant de remplir des tâches cognitives, et le connexionnisme sur la modélisation des structures corticales et de leurs mécanismes d'adaptation.

IX. — **L'action**

L'action est la concrétisation, dans le monde réel, des décisions élaborées par le système nerveux. Elle peut être réflexe (réaction de grattage chez le chien, éternuement, respiration, etc.) ou volontaire (prise d'un objet, déplacement, parole, etc.). L'action est évidemment l'expression nécessaire qui traduit le résultat des processus que nous avons brièvement décrits auparavant. Elle peut conduire le système à élargir son champ de perception, par le déplacement de la tête, des yeux, du corps en général. Elle peut conduire également à la traduction de sentiments : une expression du visage, la création d'une œuvre artistique, le contact

avec autrui. Enfin, l'action est en relation directe avec la perception : une réaction du genou se produit parce que le tendon est étiré. Les propriocepteurs subissent une traction, l'activité des neurones de la moelle épinière se modifie et conduit les muscles à une contraction opposée à la direction de traction. L'action est rarement prise en compte dans l'élaboration de systèmes cognitifs, en premier lieu parce qu'elle fait l'hypothèse d'un *corps*, rarement existant dans les simulations informatiques, ensuite parce que la vision traditionnelle des tâches cognitives commence après la perception, et s'arrête à l'instant de l'action. Cette vision éclatée traduit le schéma « mécaniste » hérité de la représentation de l'homme du XVIII[e] siècle, que l'on retrouve dans la robotique actuelle. Nous en venons donc à dresser un constat sur les limites de l'informatique en tant qu'outil de traitement de l'information. Cela nous permettra de commenter plus en détail le courant connexionniste, et de le placer dans un cadre précis, aux côtés de l'intelligence artificielle.

X. — L'intelligence artificielle

L'intelligence artificielle est une technique qui découle d'un courant de pensée philosophique connu sous le nom de *cognitivisme*. Il postule que la cognition est le produit d'une opération sur des représentations symboliques. Ainsi, le monde réel est supposé exister en lui-même et, plus important encore, supposé être décomposable en une structure formée :

— d'entités codées sous forme de symboles représentant des objets du monde réel (par exemple le symbole C peut représenter un camion) ;
— de propriétés liées à ces entités (la couleur rouge, par exemple, est une propriété liée aux camions de

pompiers et au sang) permettant de les regrouper en catégories ;
— de relations entre les entités.

Sur cette structure, construite par un expert humain, on peut appliquer un mécanisme de déduction qui repose sur une logique préalablement définie. C'est ce que l'on appelle techniquement un *moteur d'inférence*. Toute déduction est alors le résultat de compositions logiques sur les entités, les propriétés et les relations, en respectant les règles qui décrivent la logique du système. En d'autres termes, la cognition est réduite à une manipulation de symboles, ne faisant appel à aucun *sens* prédéfini. Penser, c'est réaliser des transformations syntaxiques, le rapport avec le monde réel étant établi *a posteriori*, c'est-à-dire après la transformation des symboles. L'ordinateur, en tant qu'aboutissement de la logique booléenne, est l'outil idéal pour la composition d'expressions logiques. Il n'est donc pas surprenant de constater l'efficacité de cette approche dans le domaine des sciences techniques car elle est directement inspirée des mathématiques ensemblistes. Néanmoins, lorsqu'on s'éloigne des tâches facilement formulables sous forme d'expressions logiques, l'intelligence artificielle échoue : par exemple pour reconnaître un discours dans un bruit de fond, identifier des objets dans une scène foisonnant de détails, conduire un robot mobile dans un environnement inconnu, reconnaître une route ou la texture d'un cuir... Et pourtant, un être humain parvient à remplir ces fonctions, souvent sans même y penser. C'est à partir de ces quelques remarques et des échecs constatés en utilisant une technique d'intelligence artificielle, que s'est développée une autre technique qui se place dans le courant philosophique du *constructivisme*.

Le *constructivisme* consiste à saisir la cognition et le

langage dans leur genèse, en tant que phénomènes naturels. Il intègre ainsi plusieurs échelles de temps : l'échelle de l'évolution des espèces (apparition et disparition des dinosaures), l'échelle de l'évolution des individus (de la naissance à la mort) et l'échelle de l'évolution des processus neurophysiologiques (le temps d'une modification de l'efficacité synaptique). La cognition n'est plus considérée ici comme une fonction supérieure et indépendante de l'environnement, mais comme émergence, aboutissement d'interactions et d'échanges entre individus de toutes sortes, à des échelles de temps variables. La séparation entre un individu et le monde n'est donc plus valide et, *a fortiori,* le monde n'existe pas *en lui-même*, mais résulte également du même processus émergeant. Cette notion d'émergence est au cœur des réflexions sur la complexité et l'organisation. Pour l'illustrer, il suffit de se pencher sur le fonctionnement de sociétés d'insectes, comme les fourmis. Pris isolément, chaque individu est faible, manifeste un comportement élaboré mais élémentaire. Il est, par exemple, capable de se déplacer, d'échanger des messages par ses phéromones, mais il est incapable de construire à lui seul l'édifice de la fourmilière. Dix individus non plus. Une société en est capable, et en retour, la fourmilière, comme entité sociale, agit sur chaque individu en le spécialisant : un guerrier, un éleveur, une future reine... On comprend ainsi que le *tout* organisé est une entité qui est plus que la simple somme de ses parties. Les propriétés complexes de la fourmilière (son architecture composée de galeries à plusieurs étages, ses couvains, ses réserves de nourriture) ont émergé, et sont observées par un promeneur, sans que celui-ci puisse dire comment elles ont émergé ; il peut seulement les constater. Cette illustration n'a pas valeur de généralité. Elle permet de saisir le concept de l'émergence, mais elle ne donne pas la clef de la conscience.

Retenir le cadre du constructivisme conduit donc naturellement à une recherche des mécanismes de la cognition à un niveau différent, plus proche de la réalité biologique. C'est cette tendance qui s'est développée en donnant naissance au domaine des *réseaux de neurones*. Mais, comme nous l'avons souligné en introduction, il n'y a pas opposition entre une description symbolique des mécanismes du raisonnement, et une description des mêmes mécanismes sous forme d'activité de réseaux neuronaux[1]. C'est le niveau de description qui change, constituant ainsi un changement de point de vue.

XI. — **Les réseaux de neurones**

Nous en arrivons au point où il faut définir ce que sont les réseaux de neurones. Les réseaux de neurones sont une métaphore des structures cérébrales : des assemblages de constituants élémentaires, qui réalisent chacun un traitement simple voire simpliste, mais dont l'ensemble fait émerger des propriétés globales dignes d'intérêt. Chaque constituant fonctionne indépendamment des autres, de telle sorte que l'ensemble est un *système parallèle*, fortement interconnecté. L'information détenue par le réseau de neurones est *distribuée* à travers l'ensemble des constituants, et non localisée dans une partie de mémoire sous la forme d'un symbole. Enfin, un réseau de neurones ne se programme pas pour réaliser telle ou telle tâche. Il est entraîné sur des données acquises, grâce à un mécanisme *d'appren-*

1. « La logique classique est un outil rétrospectif, séquentiel et correctif qui nous permet de corriger notre pensée séquence par séquence, mais, dès qu'il s'agit de son mouvement même, de son dynamisme même et de la créativité qui existe dans toute pensée, la logique peut tout au plus servir de béquille, jamais de jambe » (E. Morin, De la complexité : *complexus, Les théories de la complexité*, Seuil, 1991).

tissage qui agit sur les constituants du réseau afin de réaliser au mieux la tâche souhaitée.

Parmi les tâches particulièrement adaptées au traitement par réseau de neurones, on trouve *l'association* (le texte de Proust sur les « madeleines » est un cas typique d'association entre une odeur et une expérience passée), la *classification* (décider que le sang, les camions de pompiers ou les coquelicots sont d'une même couleur rouge), la *discrimination* (suivre une conversation avec un interlocuteur dans une assemblée bruyante), ou *l'estimation* (le risque que représente la traversée à la nage d'une rivière, par temps froid, en ayant absorbé un copieux repas...). Toutes ces tâches sont très différentes de celles traitées par l'informatique traditionnelle : la comptabilité, le traitement de texte, le calcul scientifique, la gestion de bases de données. On tente alors par une approche neuromorphique de transposer dans les machines une infime partie de nos capacités de traitement de l'information. Nous avons présenté les constituants principaux du système nerveux, le neurone et la synapse, et nous avons constaté qu'ils pouvaient se connecter en réseaux, définissant ainsi une structure capable d'apprendre à réaliser un traitement de l'information. Tout *l'art* de la technique des réseaux de neurones consiste aujourd'hui à modéliser ces divers éléments. Il faut ainsi proposer des modèles mathématiques du neurone réel et de la jonction synaptique, des schémas d'interconnexion des neurones, et enfin des règles permettant d'agir sur les différents paramètres de cette modélisation. Cette démarche a été engagée en 1943 par W. McCulloch et W. Pitts[1] qui ont proposé un premier modèle de neu-

1. W. Mc Culloch, W. Pitts (1943), A logical calculus of the ideas immanent in nervous activity, *Bulletin of Mathematical Biophysiology*, 3, 115-133.

rone, et par D. Hebb[1] qui a énoncé une règle de modification empirique des connexions synaptiques. Depuis, de nombreux travaux ont été conduits dans ce domaine, dont les grandes étapes sont évoquées dans le chapitre suivant.

Néanmoins, ces outils sont parfois proches de méthodes de traitement d'information déjà éprouvées. Certains d'entre eux correspondent à des algorithmes originaux, mais d'autres sont simplement des réécritures commodes de méthodes classiques. Il n'est donc pas surprenant de constater un rapprochement sensible entre les réseaux de neurones et les méthodes statistiques traditionnelles.

1. D. Hebb (1949), *The organization of the behaviour*, New York, Wiley.

Chapitre II

HISTORIQUE

Un champ de recherche ne s'établit pas en un jour. Dans ce chapitre, nous passons en revue les grands moments qui ont jalonné l'évolution des réseaux de neurones artificiels. Nous introduisons dans ce contexte historique les concepts essentiels présents dans l'ensemble des modèles.

I. — Le modèle de McCulloch et Pitts

Le passage des observations neurophysiologiques et anatomiques au neurone formel a été proposé en 1943 par W. McCulloch et W. Pitts. Cette étude tente de comprendre le fonctionnement du système nerveux à partir d'éléments formels ayant les propriétés des neurones et synapses connues à cette époque. C'est un pas essentiel qui a été franchi avec ce travail car ce sont les *fonctions* remplies par le neurone et la synapse qui sont formalisées. Mais c'est aussi une approche binaire du traitement de l'information qui se trouve renforcée : les auteurs l'annoncent dès les premières lignes de l'article.

« Because of the "all-or-none" character of the nervous activity, neural events and the relations among them can be treated by means of propositional logic. It is found that the behavior of every net can be described in these terms, with the addition of

more complicated logical means for nets containing circles ; and that for any logical expression satisfying certain conditions, one can find a net behaving in the fashion it describes. »

Malgré la simplicité de cette modélisation, ou peut-être grâce à elle, le neurone formel dit de « McCulloch et Pitts » reste aujourd'hui un élément de base des réseaux de neurones artificiels. De nombreuses variantes ont été proposées, plus ou moins biologiquement plausibles, mais reprenant toujours les concepts présentés dans cette étude. On sait néanmoins aujourd'hui que ce modèle n'est qu'une approximation des fonctions remplies par le neurone réel et, qu'en aucune façon, il ne peut servir pour une compréhension profonde du système nerveux.

Le neurone de McCulloch et Pitts est un dispositif binaire, qui reçoit des stimulations par des entrées, et les pondère grâce à des valeurs réelles appelées *coefficients synaptiques*, *poids synaptiques* ou simplement *synapses*. Ces coefficients peuvent être positifs, et l'on parle alors de synapses excitatrices, ou négatifs pour des synapses inhibitrices. Le neurone calcule ainsi une somme de ses entrées pondérées par les coefficients, et prend une décision en la comparant à un seuil fixé : si la somme pondérée des entrées dépasse le seuil, la sortie produite vaut + 1, sinon la sortie vaut – 1. Un schéma du neurone formel est proposé sur la figure II.1.

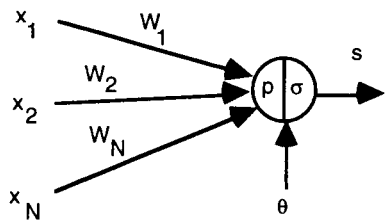

Fig. II.1. — Modèle formel de McCulloch et Pitts

Le neurone reçoit les entrées $x_1, x_2, \ldots, x_N$ et calcule le potentiel p comme la somme pondérée des entrées : $p = x_1 W_1 + x_2 W_2 + \ldots + x_N W_N$. Ensuite, une décision est prise pour calculer la sortie s en fonction du seuil θ :

$$\begin{aligned} &\text{si } p > \theta, \quad \text{alors } s = +1 \\ &\qquad\qquad \text{sinon } s = -1, \end{aligned}$$

ce qui revient à tester si la différence $(p - \theta) > 0$. Il est donc équivalent de remplacer le seuil par une entrée fixe, de valeur -1, avec un poids variable. Ceci est représenté par une fonction à seuil, appelée fonction de décision et notée σ (voir fig. II.2).

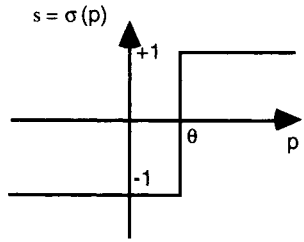

Fig. II.2. — La fonction de décision du neurone de McCulloch et Pitts

Les variations possibles de ce modèle viendront du choix du nombre d'entrées, de la valeur des poids, ou de la fonction de décision. Le nombre d'entrées dépend essentiellement des problèmes abordés : en reconnaissance de formes par exemple, on pourra avoir autant d'entrées que de pixels dans l'image, en classification de données on aura autant d'entrées que de mesures du processus observé, etc. La fonction de décision est un élément du modèle qui dépend en partie du problème : si la sortie du neurone doit délivrer une valeur

binaire, il faut utiliser une fonction à échelon ; mais, si la sortie doit être une valeur réelle, une fonction à échelon ne convient pas. Dans ce cas, on peut proposer d'autres fonctions qui répondent aux contraintes exprimées. Par exemple, la fonction sigmoïde, dont la caractéristique entrée/sortie est proche de celle du neurone réel, est couramment utilisée.

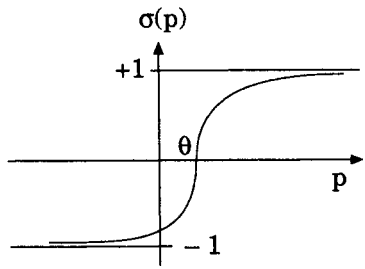

Fig. II.3. — Fonction de décision sigmoïde

Les derniers paramètres du modèle sont les coefficients « synaptiques » (ou « poids synaptiques ») W, qui dépendent également du problème à résoudre. Ce sont eux qui construisent le modèle de résolution, en fonction des informations données au réseau : il faut donc trouver un mécanisme qui permette de les calculer à partir des grandeurs que l'on peut acquérir sur le problème. C'est le principe fondamental de l'*apprentissage.* Dans un modèle de réseaux de neurones formels, apprendre, c'est d'abord calculer les valeurs des coefficients synaptiques en fonction d'exemples disponibles. A l'opposé des méthodes traditionnelles d'analyse, on ne doit pas construire un programme pas à pas pour résoudre le problème comme un programmeur l'a compris. On construit un réseau de neurones formels,

et l'on adapte les paramètres du modèle en fonction d'exemples de solutions. La plus grande difficulté viendra donc du choix du réseau, et de la représentation que l'on a du problème à résoudre.

II. — La règle de Hebb

Comment calculer les coefficients synaptiques en fonction des données disponibles sur un problème? Cette question au cœur des réflexions sur l'apprentissage a connu un début de réponse en 1949 grâce aux travaux de D. Hebb décrits dans don ouvrage *The Organization of Behavior*. Hebb a proposé une règle simple qui permet de modifier la valeur de coefficients synaptiques en fonction de l'activité des éléments qu'ils relient. Cette règle aujourd'hui connue sous le nom de « règle de Hebb » est presque partout présente dans les modèles actuels, même les plus sophistiqués.

« When an axon of cell A is near enough to excite cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased. »

Cette observation suggère donc d'accroître la valeur des coefficients synaptiques entre neurones formels qui ont une activité synchronisée, et de ne rien modifier si ce n'est pas le cas. C'est une règle purement qualitative, que l'on retrouve dans une grande partie des règles d'adaptation actuelles. Elle présente l'avantage d'être très générale et d'expliquer simplement comment une variation des poids synaptiques peut avoir lieu en ne disposant que d'informations locales. Elle n'est cependant pas suffisante et la compréhension des mécanismes biologiques de l'adaptation donne lieu aujourd'hui à de nombreux travaux de neurophysiologie.

III. — **Cadre de modélisation**

La notion clef du domaine des réseaux de neurones est que tout traitement d'information peut se réduire à la construction d'une fonction. Deux approches sont alors possibles : on explicite cette fonction formellement, et on essaye de trouver les valeurs optimales des paramètres qui modélisent au mieux le système observé. C'est la tâche essentielle du modélisateur, mais qui conduit parfois à des modèles très complexes, et dans certains cas impossibles à étudier. On peut alors proposer un modèle fonctionnel général, et définir une procédure de sélection automatique qui produise une fonction aussi proche que possible du phénomène observé : c'est le principe de base des réseaux de neurones.

Le modèle fonctionnel général s'appelle donc un *réseau de neurones,* par analogie avec les systèmes neuronaux biologiques. Il comprend des *éléments* (neurones ou synapses) dont la fonction est parfaitement définie, un schéma d'interconnexion appelé *architecture* ou structure du réseau et une procédure de sélection des paramètres appelée *règle d'apprentissage.*

Les *éléments* du réseau sont les neurones formels. Chacun d'eux est constitué de deux types de composantes : le soma et les coefficients synaptiques. Le soma intègre ses entrées, définissant ainsi son état interne, et applique une transformation pour produire une sortie. Les coefficients synaptiques reçoivent une entrée (on ne considérera pas le cas des connexions synapto-synaptiques connues sous le nom de « sigma-pi »), la multiplient par la valeur du coefficient et délivrent une sortie.

La *structure* est décrite par un graphe orienté dont les nœuds sont des soma et les arcs des coefficients synaptiques. On trouvera ainsi des structures à

connexions directes entre les entrées et les sorties, à connexions latérales entre neurones, à connexions totales entres tous les neurones, et parfois à des mélanges de schémas de connexions. Par exemple le réseau de Kohonen contient une partie à connexions directes et une partie à connexions latérales. C'est généralement le concepteur du réseau qui choisit la structure en fonction du problème à résoudre. Cela requiert une certaine expérience qui n'est que partiellement formalisée aujourd'hui.

Enfin, les *règles* d'adaptation peuvent agir en général sur l'ensemble des paramètres du modèle. Néanmoins, dans l'état actuel des recherches, les modifications ne portent que sur les valeurs des coefficients synaptiques (on parle alors de règle d'apprentissage ou *learning rule*), et parfois sur la structure du réseau (on parle alors de réseau à structure évolutive ou *incremental learning*). On ne modifie presque jamais la fonction de décision appliquée au potentiel. Elle est choisie *a priori* par le concepteur du réseau. Les règles d'adaptation actuelles reposent dans leur quasi-totalité sur la règle de Hebb et opèrent toujours à partir d'une base d'apprentissage, qui contient les mesures faites sur le processus à modéliser. Généralement, ces règles correspondent à la minimisation d'une fonction de coût associée au problème.

IV. — **L'évolution**

On admet que les premiers travaux sur les réseaux de neurones formels remontent à 1943, lors de la parution des résultats de McCulloch et Pitts, qui ont tenté de comprendre le fonctionnement du système nerveux à partir d'éléments abstraits ayant les propriétés des neurones biologiques connues à cette époque. Ces travaux furent suivis par ceux de D. Hebb en 1949, qui

sont un début de modèle comportant un mécanisme d'apprentissage. L'ensemble des éléments du modèle formel était ainsi défini : le modèle des éléments de base, la structure très simple d'un seul neurone, et une règle d'adaptation. En 1958, F. Rosenblatt propose le modèle du Perceptron, qui est une première tentative de neurone orienté vers le traitement automatique de l'information. Parallèlement à ces travaux, B. Widrow et M. Hoff proposent le modèle de l'Adaline qui sera repris par la suite comme le modèle de base des structures multicouches. En 1961, E. Caianiello développe une théorie du traitement de l'information à base d'équations neuroniques. Quelques années plus tard, en 1969, M. Minsky et S. Papert publient une analyse rigoureuse des propriétés du Perceptron qui éclaircit le champ d'application de ce modèle. En 1970, J. Hérault propose un modèle électronique de la transmission synaptique, et I. Aleksander développe une réalisation sous forme de microcircuits d'un modèle de cellule nerveuse. En 1972, T. Kohonen développe des travaux sur les mémoires associatives à base d'associateurs linéaires et propose des applications à la reconnaissance de motifs. En 1973, C. von der Marlsburg, très influencé par les travaux de D. Hubel et T. Wiesel sur les colonnes d'orientations visuelles, établit un lien entre une approche théorique et les mécanismes possibles de l'organisation corticale. En 1974, P. Werbos, dans le cadre de sa thèse, propose une méthode de calcul de gradient qui sera repris ultérieurement pour évaluer les valeurs des connexions de réseaux multicouches. En 1976, S. Grossberg publie ses travaux sur la théorie de la résonance adaptative, qui tente d'inclure des mécanismes d'attention dans des modèles de réseaux à deux couches rebouclées ; il applique sa théorie à la reconnaissance de motifs spatiaux et temporels. En 1977, S. I. Amari réalise une étude statis-

tique d'une population de neurones ; il énonce des propriétés utiles pour comprendre le comportement collectif d'un réseau. En 1981, J. McClelland et D. Rumelhart proposent le modèle d'Interactive Activation et Competition, basé sur le comportement antagoniste de populations de neurones. En 1982, J. Hopfield apporte un éclairage original par l'étude d'un réseau complètement rebouclé, dont il analyse la dynamique. Il établit ainsi une relation avec la théorie physique des verres de spins. En 1982, T. Kohonen, reprenant les travaux de C. von der Marlsburg, propose un modèle d'auto-organisation, qui manifeste des capacités de développement d'une organisation à partir de stimulations seules. Ce modèle sera très largement repris par la suite, et appliqué à des domaines aussi variés que la modélisation du cortex visuel, ou l'analyse de données économiques. De nombreux travaux théoriques, dont ceux de M. Cottrell et J.-C. Fort (1987), D. Ritter et K. Schulten (1988), seront également engagés sur ce modèle. En 1986, plusieurs chercheurs, Y. Le Cun, D. Rumelhart, G. Hinton, proposent une règle de calcul des connexions pour des réseaux multicouches appelée règle de rétro-propagation du gradient qui conduit à de très nombreuses applications et travaux théoriques. C. Jutten, J. Hérault et B. Ans proposent un modèle de séparation de sources, ou analyse en composantes indépendantes, dérivé de modèles biologiques des capteurs fusoriaux. Ce modèle a donné lieu également à des développements théoriques (P. Comon pour l'identification des conditions statistiques sur les signaux à garantir pour la séparation) dans le domaine du traitement du signal, ainsi qu'à des réalisations de circuits intégrés analogiques, par E. Vittoz et son équipe, pour la séparation de signaux indépendants.

Aujourd'hui le domaine des réseaux de neurones

artificiels est en pleine expansion. Il a connu un fort développement depuis les années 1940, avec des périodes de latence mais aussi avec une croissance exponentielle du nombre de chercheurs depuis les années 1980. Le développement de l'informatique y est pour beaucoup, permettant de nombreuses investigations expérimentales. L'effet de mode a également joué un rôle, tirant parti des termes biologiques qui, employés parfois sans prudence, font immédiatement penser à un cerveau artificiel. Nous verrons qu'il n'en est rien dans la réalité et que les réseaux de neurones artificiels sont bien loins de produire une machine intelligente, autonome, perceptive, dotée d'une identité et d'une histoire. Et même si cela peut paraître banal, il n'est pas inutile de dire que les inventions de la vie ne sont pas encore détrônées par l'intelligence artificielle.

Chapitre III

L'ASSOCIATION

Les concepts de base de la mémoire et de l'association de données furent déjà découverts et expliqués par Aristote. Il décrivit quatre conditions selon lesquelles deux événements peuvent être liés dans la mémoire : s'ils se déroulent simultanément, s'ils se déroulent successivement, s'ils sont similaires, ou s'ils sont contraires. Les deux premières conditions sont liées à l'apprentissage (relation spatiale dans le premier cas, temporelle dans le deuxième), tandis que les deux suivantes sont liées au rappel d'informations depuis la mémoire. L'information est stockée dans les mémoires sous forme de représentations internes qui peuvent être très différentes de l'image que nous avons de cette information. Des idées, des images, des sensations peuvent être enregistrées dans un cerveau humain ; comprendre la façon dont cet enregistrement est effectué est un des buts de la modélisation du système nerveux. Mais il est aussi important de comprendre comment on peut rappeler l'information, comment se différencient mémoires auto-associatives, hétéro-associatives et classifieurs, ou encore mémoires locales ou globales.

Ce chapitre est consacré à l'étude des mécanismes d'association, que nous présentons en préambule. Nous les détaillons en réalisant une association avec un seul neurone. Cela nous permet de présenter la

règle de Hebb. Nous étudions ensuite le cas de plusieurs associations avec un seul neurone : ce sont les méthodes algébriques et itératives. Nous généralisons à l'association avec plusieurs neurones dans une même couche. Enfin, nous analysons le cas général de l'association sur une structure à plusieurs couches de neurones, qui nous conduit à l'algorithme de rétro-propagation du gradient.

I. — Les mémoires

1. **Mémoires locales.** — La plupart des ordinateurs utilisent des mémoires adressées ; ceci signifie que l'unité centrale de traitement *(Central Processing Unit)* donne une adresse à la mémoire, et que cette mémoire répond en donnant l'information contenue à cette adresse. Il y a donc correspondance parfaite et non ambiguë entre une information stockée préalablement dans la mémoire, et cette même information lue par la suite.

2. **Mémoires adressables par le contenu.** — Un autre moyen de récupérer de l'information dans une mémoire est de l'adresser par son contenu, c'est-à-dire par un code de même nature que l'information stockée elle-même. Un exemple est la mise en correspondance, dans un dictionnaire, de mots anglais et français : la mémoire retient le lien entre les couples « chien-dog », « chat-cat », « cheval-horse »... ; lorsqu'une entrée correspondant à la première composante des couples est présentée à la mémoire (par exemple « chat »), celle-ci trouve la position du couple en question (son « adresse » dans la mémoire), et restitue la deuxième composante « cat ». Il n'est plus nécessaire, dans ce cas, de connaître l'ordre dans lequel les informations ont été rangées dans la mémoire ; seule l'absence d'am-

biguïté entre les premières composantes des couples est nécessaire.

Ce dispositif est appelé « mémoire adressable par le contenu », ou « mémoire associative ». Il s'agit en fait toujours d'une mémoire locale, mais dont le système d'adressage a été remplacé par une table de correspondances, et la localisation de l'information à récupérer n'est pas connue *a priori*. Cette forme de mémoire est utilisé dans nombre de problèmes rencontrés en intelligence artificielle, et peut être implantée de manière logicielle ou matérielle.

3. **Mémoires distribuées.** — Les processus observés dans le cerveau sont différents de ceux présentés précédemment : il n'y a pas d'endroit précis où chaque information est localisée. Au contraire, chaque élément d'information est réparti sur un vaste ensemble de neurones et synapses, chacun de ceux-ci ne mémorisant qu'une petite partie d'un grand nombre de ces éléments.

Le point important de ce conflit apparent est que *chaque information est distribuée sur l'ensemble des éléments de corrélation* que l'on appelle les *synapses*. Ceci a deux conséquences immédiates. Premièrement, comme chaque élément contient une partie de l'information de plusieurs données, les concepts de mémorisation et de rappel sont plus vagues que dans une mémoire localisée ; par conséquent, il y a un compromis entre la quantité d'information à stocker et la qualité du stockage. Deuxièmement, comme chaque information est répartie dans toute la mémoire, des défaillances de certains éléments n'entraînent pas nécessairement des sorties erronées : cela dépend de la redondance de la mémoire. Ceci explique l'effet bien connu de « tolérance aux fautes » des réseaux de neurones qu'on peut exploiter à condition de prendre toutes les précautions nécessaires.

4. **Mémoires auto-associatives.** — Les mémoires adressables par le contenu (locales ou distribuées) sont généralement groupées en trois classes : les mémoires auto-associatives, les mémoires hétéro-associatives, et les classifieurs. Les mémoires auto-associatives sont des dispositifs pour lesquels les entrées et sorties ont la même taille et la même nature ; leur intérêt réside dans leurs propriétés de généralisation, c'est-à-dire qu'il est possible d'y introduire comme entrée une version incomplète ou corrompue d'une des informations enregistrées, sans empêcher la sortie de donner l'information apprise à l'origine. Ceci définit la propriété de « généralisation », terme fort utilisé dans ce livre. Les mémoires auto-associatives peuvent être utilisées dans un ensemble d'applications de reconnaissance, en compression de données ou encore en traitement d'images. Le réseau de Hopfield est un exemple d'une telle mémoire dont nous reparlerons plus loin.

5. **Mémoires hétéro-associatives.** — Ce type de mémoire ressemble aux mémoires auto-associatives, mais en diffère par la nature et la dimension des entrées et des sorties : elles ne sont pas nécessairement identiques. Les dispositifs de reconnaissance de caractères en sont un exemple : il n'est pas nécessaire de reconstruire une version intacte d'une matrice de points à partir d'une version scannée corrompue, mais il est plus intéressant de trouver la représentation ASCII du caractère, pour l'utiliser dans un logiciel de traitement de texte. Dans ce cas, l'information d'entrée est une matrice de points, et celle de sortie un code ASCII. Ceci peut être réalisé en ajoutant une couche à une mémoire auto-associative, transformant la version intacte de l'image en un code par un tableau de correspondances. Une autre possibilité est de mesurer la distance de Hamming (nombre de bits différents) entre le

vecteur d'entrée et ceux stockés, et ensuite de choisir le minimum de ces distances : c'est le principe du réseau de Hamming.

6. **Classifieurs.** — Un dernier type de mémoire mentionné ici est le classifieur : son but est de grouper des classes d'information d'entrée en des codes communs à chaque classe. Il est évident que l'opération effectuée est identique à celle de la mémoire hétéro-associative : l'application de reconnaissance de caractères mentionnée ci-dessus n'attribue pas des codes séparés pour chaque version de la même lettre, mais bien un code commun pour l'ensemble d'entre elles. La différence entre mémoires hétéro-associatives et classifieurs est donc artificielle, et ne réside que dans le nombre d'exemples d'entrée (appelés aussi prototypes) pour lesquels le même code de sortie est attribué (un pour le cas de la mémoire hétéro-associative, plusieurs pour le classifieur). La mémoire auto-associative peut également être vue de la même manière ; la seule différence entre les deux modèles précédents réside dans le mode de représentation de l'information. Une simple transformation des données rend donc tous ces types de mémoires similaires ; c'est la raison pour laquelle on peut les grouper sous le vocable « mémoires associatives ».

La *mémoire associative* est donc considérée comme un outil informatique, algorithmique ou mathématique, selon le point de vue choisi, qui présente deux propriétés bien distinctes. D'une part, elle permet *d'associer* deux objets, que nous décrivons par la suite sous forme de vecteurs. En associant deux objets, elle crée des paires, et permet par exemple de retrouver un objet inconnu par son binôme ; aucune restriction n'est faite ici quant à la forme, la taille ou la façon de représenter les deux objets à associer. Une table de vérité d'une fonction logique est un exemple particulier de proces-

sus d'association : les deux objets sont d'une part le vecteur d'entrée de la table, pouvant prendre autant de valeurs binaires différentes qu'il y a de lignes à la table, et d'autre part le vecteur de sortie qui donne la réponse de la fonction logique pour chacun des vecteurs d'entrée.

La deuxième propriété de la mémoire associative réside dans sa capacité de *mémorisation* de l'information ; dans l'exemple ci-dessus, cette capacité peut être matérialisée sous forme d'une mémoire conventionnelle reprenant ligne par ligne, et dans un ordre bien précis, toutes les valeurs possibles du vecteur de sortie de la table.

Le cas général d'une mémoire associative va cependant bien au-delà de cet exemple. Supposons que nous voulions mémoriser des associations entre des vecteurs d'entrée binaires de taille N et des vecteurs de sortie de taille K. Supposons également que nous ne désirions utiliser qu'un petit nombre parmi les 2^K vecteurs de sortie possible. Supposons enfin qu'à chaque vecteur de sortie réellement utilisé, nous voulions non seulement associer un vecteur type d'entrée, mais aussi tous les vecteurs d'entrée « semblables » à ce vecteur type. En d'autres termes, un vecteur de sortie particulier sera associé à un groupe de vecteurs d'entrée possibles, groupe qui sera déterminé selon une certaine notion de similitude, ou de distance, entre vecteurs. Dans ce cas, on voit aisément que la matérialisation de cette mémoire sous forme de table de vérité d'une fonction logique conduirait à une sous-utilisation des ressources mémoires : on mémoriserait la même information dans un grand nombre de lignes de la table, alors que la notion même de distance entre vecteurs d'entrée suffit à caractériser un groupe auquel doit être associé un vecteur de sortie unique.

L'objet de la mémoire associative est donc de

réduire la quantité d'information à mémoriser par rapport à une mémoire de type classique, pour certaines tâches particulières telles que celles décrites plus haut.

II. — La règle de Hebb

Considérons tout d'abord le cas d'une mémoire où nous n'avons qu'une seule association à mémoriser, entre les vecteurs $x(1)$ (vecteur normé de R^N) et un scalaire $y(1)$. Soit un neurone unique à N entrées, de poids $W = (W_1, W_2, \ldots, W_N)$, $s(1)$ sa sortie calculée par

$$s(1) = W.x(1).$$

Construire une mémoire associative capable d'associer le vecteur $x(1)$ à la sortie $y(1)$, c'est trouver W qui réalise :

$$y(1) = W.x(1).$$

On voit que le vecteur W donné par :

$$W = y(1)\, x(1)$$

respecte la tâche d'association attendue car (en rappelant que $x(1)$ est normé) :

$$W.x(1) = y(1)\, x(1).x(1) = y(1).$$

Cette première façon de calculer la matrice W est apparentée à la règle qualitative de Hebb que nous avons vue plus haut. Par extension, cette règle utilisée dans le cas d'une mémoire associative sera également appelée règle de Hebb.

Bien entendu, mémoriser un seul vecteur n'est ni intéressant ni utile. On peut alors construire une mémoire associative dans le but d'enregistrer P couples de type $(x(j), y(j))$, $1 \leqslant j \leqslant P$, où les vecteurs $x(j)$ sont normés. Utilisant la même analogie avec la règle

de Hebb, mais cette fois-ci sommant les contributions de chacun des couples, on trouve :

$$W_i = \sum_{j=1}^{P} y(j)\, x_i(j).$$

Dans le cas d'une mémoire associative, ou dans le cas des réseaux de neurones décrits dans les sections suivantes, la phase de reconnaissance consiste à présenter un vecteur au réseau ainsi formé et à calculer les sorties, les valeurs des poids synaptiques W_i calculés par la formule ci-dessus étant figés. C'est ce que l'on appelle la phase de reconnaissance, par opposition à la phase d'apprentissage, qui consiste à choisir les coefficients synaptiques W_i en fonction des données d'apprentissage $(x(j), y(j))$. On aura alors :

$$s(r) = W.x(r) = \sum_{j=1}^{P} y(j)\, x(j).x(r)$$

où $s(r)$ est la sortie calculée de la mémoire associative lorsqu'on lui présente un vecteur $x(r)$ comme entrée. Si on suppose maintenant d'une part que les P prototypes $x(j)$ utilisés pour l'apprentissage, c'est-à-dire pour le calcul de la matrice W, sont orthogonaux 2 à 2 (le produit scalaire de vecteurs orthogonaux est nul), et d'autre part que le vecteur $x(r)$ présenté en phase de reconnaissance est normé et égal à un des prototypes $x(j)$, tous les termes de la somme dans la dernière équation deviennent nuls, excepté celui qui correspond au vecteur $x(r)$. La sortie devient :

$$s(r) = y(r)\, x(r).x(r).$$

On a ainsi réalisé une mémoire associative qui, à chaque présentation d'un vecteur $x(r)$ en entrée, fournit en sortie le vecteur $y(r)$, dont l'association avec $x(r)$

aura été préalablement « apprise » à la mémoire lors de la phase de calcul de la matrice W. La restriction sur l'orthogonalité des prototypes $x(j)$ est néanmoins sévère : il y a peu de cas réels de reconnaissance où l'on pourra observer strictement cette propriété. De plus, une mémoire associative telle que nous l'avons définie plus haut doit pouvoir « reconnaître » un vecteur d'entrée, c'est-à-dire donner une sortie exacte $y(r)$ même si le vecteur présenté à l'entrée n'est pas exactement égal à $x(r)$, mais lui en est quand même proche au sens d'une certaine mesure de distance. Cette propriété n'est pas strictement vérifiée dans le cas de la mémoire associative dont l'apprentissage a été réalisé par la règle de Hebb, même si les prototypes appris sont orthogonaux. Les réseaux de neurones décrits dans les sections suivantes peuvent, sous certaines conditions, pallier à cet inconvénient grâce à leurs non-linéarités, alors que la mémoire associative présentée ici est purement linéaire.

III. — **Méthodes algébriques**

Telle que nous venons de la présenter, la tâche de construction d'un associateur linéaire peut se ramener à un problème de résolution d'un système d'équations linéaires. Si nous considérons un neurone linéaire, qui calcule simplement la somme pondérée de ses entrées, pour une entrée $x(1) = [x_1(1), x_2(1), \ldots, x_N(1)]$, la sortie $s(1)$ est calculée par :

$$s(1) = \sum_{i=1}^{N} W_i \, x_i(1)$$

où W_i représente le coefficient synaptique qui relie l'entrée i au neurone. Le vecteur W de dimension N représente ainsi l'ensemble des connexions du réseau.

La tâche d'association consiste à construire un opérateur qui, pour un ensemble de vecteurs $X = [x(1), x(2), \ldots, x(P)]$ associe un ensemble de valeurs $Y = [y(1), y(2), \ldots, y(P)]$ sous la contrainte que $x(i)$ soit associé à $y(i)$, quel que soit i. On dispose ainsi de deux matrices d'exemples, X de dimension (N, P) et Y de dimension (1, P), pour lesquelles il faut construire un opérateur linéaire W, tel que :

$$Y = WX.$$

En pratique ceci est rarement possible, et l'on se contentera d'une contrainte plus faible en disant que WX doit être « aussi proche que possible » de Y. Néanmoins, en théorie, l'équation $Y = WX$ admet une solution unique dans un cas très particulier : celui où la matrice X est inversible. Ceci impose que X soit carrée et que les vecteurs $x(j)$ soient linéairement indépendants. Dans ce cas, on trouve W donnée par :

$$W = YX^{-1}.$$

Pratiquement, il faut que $N = P$, et donc que le *nombre d'exemples à associer soit exactement égal à la dimension de ces exemples* ; ceci est évidemment irréaliste. On se satisfait donc de la recherche d'une solution approchée de l'équation $Y = WX$ en définissant une erreur ε que l'on tente de minimiser. Cette fonction d'erreur est généralement une norme quadratique, définie de la manière suivante :

$$\varepsilon = \sum_{j=1}^{P} (y(j) - W.x(j))^2.$$

Ce critère est un critère classique des moindres carrés.

Si la matrice d'exemples X n'est pas carrée, elle n'admet pas d'inverse. Néanmoins, toute matrice A dont

les lignes sont linéairement indépendantes admet une pseudo-inverse unique, notée A^+, et définie de la façon suivante :

$$A^+ = A^T(AA^T)^{-1}$$

où ^{T} représente l'opérateur de transposition.

En effet, la matrice (AA^T) est carrée, et si les lignes de A sont linéairement indépendantes elle est inversible. On montre que, lorsque l'équation $Y = WX$ n'a pas de solution exacte, la matrice $W = YX^+$ est celle pour laquelle la fonction d'erreur ε définie précédemment est minimum. On obtient ainsi la matrice des coefficients W, qui s'écrit :

$$W = YX^T(XX^T)^{-1}$$

dès que les lignes de la matrice X sont linéairement indépendantes, ce qui correspond au fait que $N \leqslant P$ et que les P vecteurs d'entrée forment un système de rang N.

Cette question de l'association est équivalente à un classique problème de régression linéaire puisqu'on cherche à minimiser :

$$\varepsilon = \sum_{j=1}^{P} (y(j) - W.x(j))^2.$$

C'est une régression de y sur $(x_1, \ldots, x_N)$ pour des couples d'observations $(y(j), x(j)), j = 1, \ldots, P$.

Le calcul direct de cette matrice W peut s'avérer extrêmement coûteux. Nous verrons, dans la section suivante, qu'il est possible d'atteindre cette solution de manière itérative. Ce gain en simplicité est payé par un temps de calcul plus élevé, mais nous rapproche de mécanismes d'adaptation plus en accord avec une approche neuronale.

IV. — **Méthodes itératives et adaptatives**

La méthode de résolution algébrique de systèmes linéaires souffre d'un défaut majeur dans le cadre qui nous concerne : elle n'est pas biologiquement plausible. On imagine mal, en effet, que le système nerveux ait développé des structures qui mémorisent des suites d'événements sous forme de matrices, et qui trouve une solution en inversant ces matrices. En d'autres termes, il doit exister des méthodes de résolution qui n'exigent pas de mémorisation de séquences d'événements, mais qui s'approchent de la solution lorsqu'un événement survient. Idéalement, ces méthodes ne devront pas faire appel à l'ensemble des connaissances acquises par le système, mais simplement à celles disponibles localement à chaque élément de calcul. Il faudra donc qu'elles soient locales vis-à-vis du temps, et locales vis-à-vis de l'espace. De telles méthodes existent, et sont héritées des techniques développées dans le cadre de l'optimisation : ce sont les méthodes de gradient.

On se donne *a priori* une fonction f dont on recherche un minimum local x^* et on la choisit suffisamment bien pour que ce minimum soit précisément solution du problème que l'on se pose. Dans le cas d'une fonction f à une seule variable, pour savoir si le point x^* est un minimum, il suffit de savoir si la dérivée de la fonction f est nulle au point x^* (il y a un « creux » ou une « bosse » au point x^*), et si la dérivée seconde en ce même point est positive (c'est un creux, sinon ce n'est pas un minimum, c'est un maximum). On peut ainsi se servir de ces deux conditions pour rechercher « en aveugle » ce point x^*.

Cela se fait assez aisément en partant d'un point initial $x(0)$, et en cheminant pas à pas par approximations successives en se déplaçant suivant la règle suivante :

$$x(t+1) = x(t) - \alpha f'(x(t)).$$

Le choix de $x(0)$ et de α relève de l'art du numéricien[1]. Tout ceci n'est vraiment simple que lorsque f a un seul minimum global, et lorsque le pas α est correctement choisi.

Exemple

Soit la fonction $f(x) = x^2 - 1$. On recherche son minimum par la méthode du gradient. Sa dérivée est $f'(x) = 2x$. On modifie donc successivement les valeurs de x afin d'atteindre la minimum en suivant la règle :

$$x(t + 1) = (1 - 2\alpha)\, x(t) = (1 - 2\alpha)^{t+1}\, x(0).$$

Si la valeur de α est strictement comprise entre 0 et 1, $x(t)$ converge vers 0, qui est bien le minimum de f.

On généralise cette méthode de recherche du minimum d'une fonction à n'importe quelle fonction de R^N dans R qui dispose de dérivées partielles du second ordre continues. Dans ce cas, on détermine non plus la dérivée de la fonction, mais le vecteur gradient de la fonction, défini par :

$$\text{grad}\, f(x) = \left(\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \ldots, \frac{\partial f}{\partial x_n}\right).$$

Ainsi, la recherche du minimum de la fonction f peut se faire en appliquant l'algorithme suivant :

$$x(t + 1) = x(t) - \alpha\, \text{grad}\, f(x(t))$$

où $x(t) = [x_1(t), x_2(t), \ldots, x_N(t)]$. Néanmoins, cette méthode souffre d'une extrême lenteur de calcul et du fait qu'il serait préférable de choisir un pas de gradient α variable selon le temps, ce qui n'est pas le cas ici.

On peut maintenant appliquer cette méthode de gra-

1. M. Minoux (1983), *Programmation mathématique*, t. 1, Dunod.

dient à la minimisation de la fonction d'erreur définie plus haut, par rapport aux coefficients W_i.

$$\varepsilon = \frac{1}{2} \| Y - WX \|^2 = \frac{1}{2} \sum_{j=1}^{P} (y(j) - W.x(j))^2.$$

Pour rechercher le minimum de la fonction (qui est ici unique), on calcule son gradient par rapport au vecteur W :

$$\text{grad } \varepsilon = \left[\frac{\partial \varepsilon}{\partial W_1}, \frac{\partial \varepsilon}{\partial W_2}, \ldots, \frac{\partial \varepsilon}{\partial W_N} \right] = -(Y - WX)\, X^T.$$

La règle itérative de modification des coefficients s'écrit donc :

$$W(t+1) = W(t) - \alpha \text{ grad } \varepsilon = W(t) + \alpha(Y - WX)\, X^T.$$

Cette méthode n'impose pas d'inversion de matrice comme la méthode algébrique, mais elle reste globale dans le sens où elle requiert la connaissance et le stockage de l'ensemble des vecteurs exemples $x(j)$, qui doivent être connus à chaque itération.

La recherche des minima d'une fonction quelconque, *connue uniquement comme une succession d'observations*, peut également être réalisée par une méthode de gradient. Nous allons l'étudier sur un exemple pratique : le modèle de l'Adaline.

V. — L'adaline

Le modèle de l'Adaline (ADAptive LINar Element) a été proposé par B. Widrow[1], chercheur américain à Stanford, dans les années 1960. Il travaillait sur les systèmes adaptatifs, et cherchait comment on pourrait

1. B. Widrow, M. Hoff (1960), Adaptive switching circuits, *Proc. of the 1960 WESCON*, 4, 96-140.

construire un système capable de trouver une solution à un problème en ne connaissant, *à chaque instant, qu'une partie des données du problème.* Par exemple, construire un système qui apprenne à classer des pièces de voiture à partir d'une dizaine d'exemples, et qui soit capable de poursuivre le classement pour des pièces jamais rencontrées auparavant. Ce type de tâche est aisément réalisée par un être humain : nous savons facilement faire la différence entre un arbre, une voiture ou une maison. Pourtant, nous n'avons pas vu tous les arbres possibles, ou encore toutes les voitures ou toutes les maisons possibles. Nous n'avons pas non plus à réfléchir pour identifier à coup sûr ces objets. Cela se fait « naturellement ». Il est donc probable que notre système nerveux a été capable d'apprendre ce qui caractérise une famille d'objets, et de séparer ces objets entre eux. Il reste néanmoins que la limite est parfois discutable : une caravane peut se concevoir comme une « voiture-maison » !

1. **Structure.** — L'Adaline est un modèle de neurone formel, semblable à celui que nous avons vu précédemment. Sa sortie est calculée comme est une somme de ses entrées, pondérées par les coefficients synaptiques (fig. III.1). Les entrées sont à valeurs réelles, la sortie également et la fonction de décision est la fonction identité.

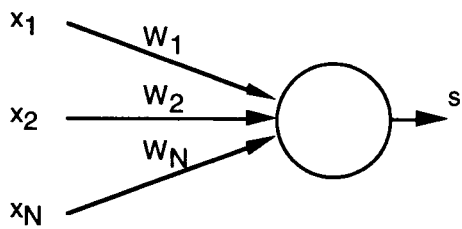

Fig. III.1. — Structure d'une Adaline

Il est possible, avec un tel élément, de réaliser une mémoire associative très simple, qui servira de base à des structures plus complexes. Pour une entrée $x(t)$ présentée à l'Adaline[1], la sortie $s(t)$ calculée vaut :

$$s(t) = \mathrm{W} . x(t).$$

Si l'on considère que les coefficients synaptiques W n'ont pas été calculés pour remplir la tâche souhaitée, il n'y a aucune raison *a priori* pour que $y(j) = s(j)$. Or, c'est précisément ce que l'on souhaiterait obtenir, et ceci pour tout j. Il faut donc construire une procédure qui permette d'associer aussi précisément que possible les entrées présentées et les sorties désirées, et ceci pour l'ensemble des exemples disponibles.

Comme nous venons de le montrer, la solution par calcul du gradient présente un inconvénient majeur dans le contexte des réseaux de neurones. La détermination de W impose la connaissance *a priori* de l'ensemble des exemples $x(j)$, ce qui est impossible en pratique, sauf à disposer d'une mémoire infinie. Et si cela était possible, il semble difficile de considérer une inversion de matrice comme opération de base du système nerveux.

Il faut donc, comme nous l'avons souligné en préambule, trouver une règle plus *locale* pour calculer les coefficients W, en ne connaissant qu'un seul exemple à la fois. Ceci est possible par la méthode dite du gradient stochastique, qui conduit directement à la règle de l'Adaline.

1. Ce modèle de neurone est strictement identique à ce que nous venons de présenter. Notons cependant que les exemples d'apprentissage sont maintenant supposés être connus séquentiellement par rapport au temps, ce qui justifie l'emploi de l'argument (t) dans les équations présentées.

2. **Règle de calcul des coefficients.** — Cette méthode consiste à approcher la solution W, en ne connaissant qu'une suite aléatoire d'observations successives. Dans ce cas, on chercher à minimiser non plus l'espérance ou la valeur moyenne de la fonction $\varepsilon(W)$, mais la fonction $\varepsilon(W, t)$ à chaque observation. On peut montrer, dans certaines conditions de régularité, que l'effet de « moyenne » permet d'affirmer que la solution que l'on obtient est celle qui serait atteinte par la méthode du gradient. Toutefois la trajectoire dans l'espace des coefficients W est plus erratique, conduisant ainsi à une vitesse de convergence plus lente.

L'étude théorique des problèmes liés à la méthode du gradient stochastique dépasse le cadre de cet ouvrage. En particulier, on doit admettre que cette règle n'est valable que pour des entrées stationnaires. Pour plus de détails, le lecteur pourra consulter l'ouvrage de M. Duflo[1].

Considérons donc l'erreur instantanée suivante :

$$\varepsilon(W, t) = \frac{1}{2}(y(t) - s(t))^2.$$

Le calcul du gradient de cette fonction nous donne directement :

$$\frac{\partial \varepsilon(W, t)}{\partial W_i} = \sum_{j=1}^{N} - (y(t) - W_j . x_j(t))\, x_i(t).$$

On peut ainsi écrire la règle de modification des poids, qui permet d'atteindre itérativement le minimum de $E[\varepsilon(W, t)]$:

$$W(t + 1) = W(t) + \alpha(y(t) - s(t))\, x(t).$$

Cette règle est également appelée « règle delta » dans la littérature américaine.

1. M. Duflo (1990), *Méthodes récursives aléatoires*, Masson.

Pour l'Adaline, le minimum de la fonction quadratique est unique. Donc, on obtient effectivement le même point $W^* = YX^T(XX^T)^{-1}$ si le rang de la matrice X est égal à N.

3. **Illustration.** — La figure III.2 montre la droite des moindres carrés trouvée, dans un plan, pour la séparation de deux classes qui ne se recouvrent pas. Les sorties $y(t)$ assignées aux prototypes des deux classes sont respectivement – 1 et 1. Le critère cherche à minimiser les écarts entre les prototypes et deux droites dont les équations sont respectivement $W.x = 1$ et $W.x = -1$; la droite « séparatrice » est bien sûr celle donnée par $W.x = 0$, et les deux régions de décision sont situées de part et d'autre de cette droite.

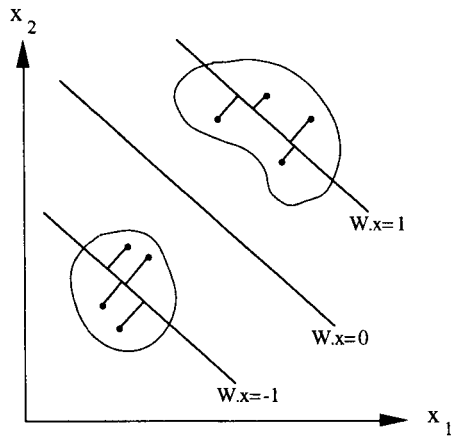

Fig. III.2. — Critère des moindres carrés

Le principal problème du critère des moindres carrés réside dans le fait que la droite $W.x = 0$ ne sépare pas nécessairement les deux classes, même si celles-ci sont linéairement séparables. L'exemple de la figure III.3

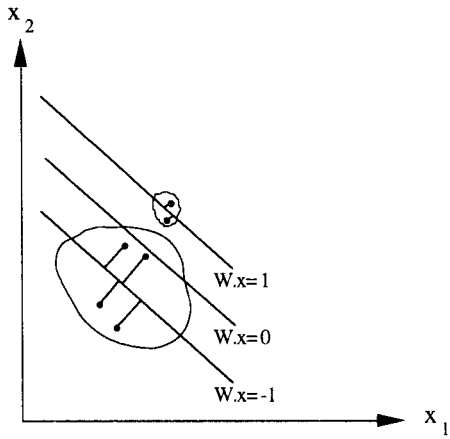

Fig. III.3. — Classes linéairement séparables

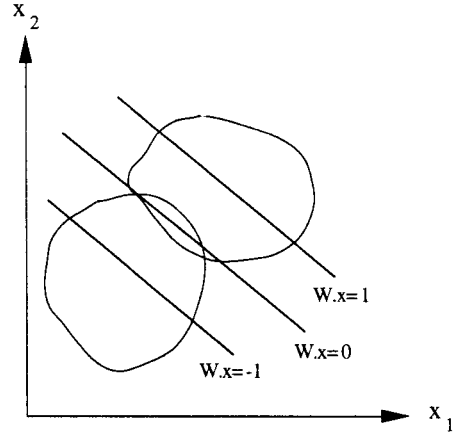

Fig. III.4. — Classes non linéairement séparables

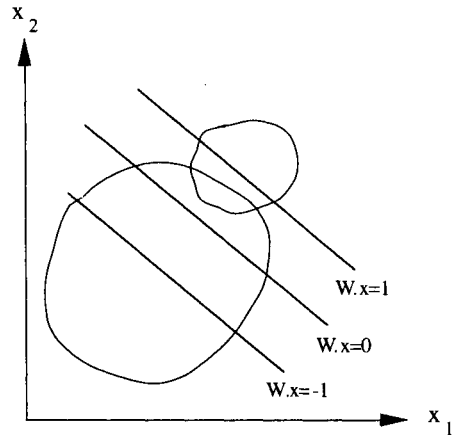

Fig. III.5. — Classes non linéairement séparables

illustre ce fait. Le même genre de constatation peut être fait lorsque les classes se recouvrent, et ne sont donc pas linéairement séparables. Les exemples des figures III.4 et III.5 illustrent le critère des moindres carrés dans ce cas.

4. **Le modèle non linéaire.** — Nous avons considéré jusqu'à présent un modèle de neurone très simple : il ne réalise qu'une somme pondérée de ses entrées, et ne prend pas de « décision », comme semble le faire un neurone réel. Il est donc tout naturel d'enrichir le modèle en adjoignant au neurone formel une fonction de décision dont la caractéristique se rapproche d'une sigmoïde. Nous allons donc revoir la règle d'adaptation des poids qui doit tenir compte de cet enrichissement. La sortie $s(t)$ du neurone est maintenant calculée comme la somme pondérée des entrées, transformée

par une fonction de décision non linéaire $\sigma(.)$, qui est souvent une tangente hyperbolique : $\sigma(p) = \text{th}(p)$.

$$p(t) = \mathbf{W}.x(t)$$
et $$s(t) = \sigma(p(t)).$$

On rappelle que $p(t)$ est le potentiel du neurone formel. En reprenant le raisonnement précédent, et pour une mesure de l'erreur instantanée (règle du gradient stochastique), nous obtenons l'expression du gradient de $\varepsilon(\mathbf{W}, t)$ suivante :

$$\frac{\partial \varepsilon(\mathbf{W}, t)}{\partial \mathbf{W}_i} = (\sigma(\mathbf{W}.x(t)) - y(t))\, x_i(t)\, \sigma'(p(t))$$

et la règle de modification des coefficients s'écrit donc :

$$\mathbf{W}(t+1) = \mathbf{W}(t) + \alpha(y(t) - s(t))x(t)\sigma'(p(t)).$$

Par exemple, si $\sigma(p) = \text{th}(p)$, la dérivée de $\text{th}(p)$ est $(1 - \text{th}^2(p))$, et la règle de modification devient :

$$\mathbf{W}(t+1) = \mathbf{W}(t) + \alpha(y(t) - \text{th}(\mathbf{W}(t).x(t)))x(t)(1 - \text{th}^2(p(t))).$$

Cette règle est connue sous le nom de « règle delta généralisée », et servira par la suite pour la construction de la règle de rétro-propagation du gradient.

5. **Illustration.** — On peut illustrer l'effet de la non-linéarité par la figure III.6. Les lignes pleines ($\mathbf{W}.x = 0, \mathbf{W}.x = -1$ et $\mathbf{W}.x = 1$) correspondent approximativement à la solution du critère des moindres carrés (avec ou sans fonction non linéaire) pour l'ensemble des prototypes représentés par des cercles pleins pour la classe 1 et par des cercles creux pour la classe – 1 ; cette solution ne prend en compte que les deux amas de prototypes, sans tenir compte du prototype de la classe 1 ajouté au bas de la figure III.6. Lorsqu'on ajoute ce prototype, un classifieur linéaire voit sa droite de décision sensiblement écartée, l'effet

du prototype ajouté dans le calcul du critère des moindres carrés étant important. Le résultat est représenté par les lignes discontinues (notons que pour la facilité celles-ci ont été représentées parallèles aux premières, mais que ceci n'est pas vrai dans le cas général). Par contre, l'effet non linéaire de la fonction σ est d'atténuer l'importance de prototypes situés loin des droites $W.x = 1$ ou $W.x = -1$, suivant le cas. Les droites pointillées, correspondant au critère des moindres carrés dans le cas non linéaire, sont donc plus proches des droites initiales, visiblement mieux adaptées à notre problème où le point ajouté est un « élément étranger » dont l'importance doit être minimisée.

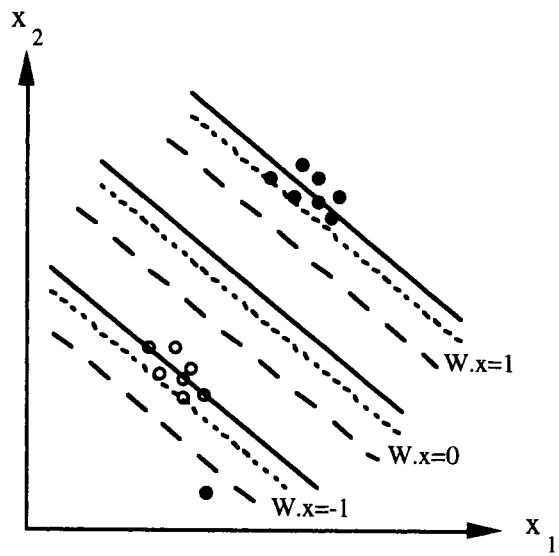

Fig. III.6. — Effet de la non-linéarité

VI. — Le Perceptron

1. **Introduction.** — La règle du Perceptron[1], à titre historique, mérite d'être mentionnée. Le principe des équations de calcul du potentiel reste inchangé, mais cette fois la fonction de décision non linéaire utilisée est abrupte :

$$\sigma(p) = \text{Sign}(p).$$

Bien entendu, il devient impossible d'utiliser une méthode de descente de gradient, la fonction n'étant pas dérivable. La règle du Perceptron est alors une règle adaptative destinée à modifier les poids W dans la bonne direction lorsqu'une mauvaise classification survient. Cette règle peut s'énoncer par :

$$W(t + 1) = W(t) + \alpha(y(t) - \sigma(W.x(t))\, x(t).$$

On voit que cette règle s'apparente fortement à celle de l'Adaline ; seul le terme correspondant à la dérivée de la fonction non linéaire a été supprimé.

L'effet de la règle du Perceptron est très différent des règles de minimisation de gradient. Ici, il faut le souligner, lorsque deux classes sont linéairement séparables, on démontre que la règle converge vers une droite séparatrice correcte. On constate en effet qu'en utilisant une fonction de type « signe », la règle d'adaptation au temps t est sans effet sur les coefficients W si le prototype $x(t)$ est déjà bien classé. Elle provoque un déplacement de la frontière W « dans la bonne direction » si le vecteur $x(t)$ est mal classé. Dans le cas de classes linéairement séparables, il y a néanmoins une infinité de solutions possibles, et le Perceptron peut converger vers n'importe laquelle de celles-ci (fig. III . 7).

1. F. Rosenblatt (1958), The Perceptron : A probabilistic model for information storage and organization in the brain, *Psychological Review*, 65, 368-408.

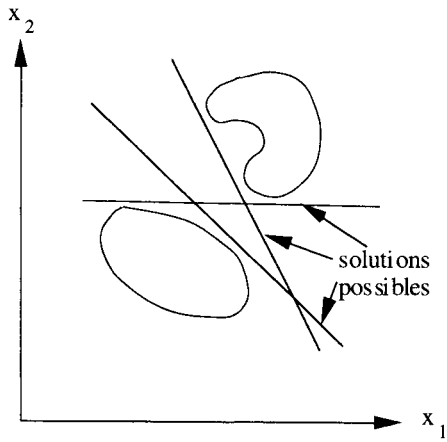

Fig. III. 7. — Solutions possibles de la règle du Perceptron

Un point intéressant, sinon essentiel, noté par Minsky et Papert est que la règle du Perceptron ne converge tout simplement pas lorsque les classes ne sont pas linéairement séparables, ce qui limite considérablement ses possibilités d'utilisation.

En résumé, lorsque les classes sont linéairement séparables, les règles delta et delta généralisée (conduisant au critère des moindres carrés sur l'erreur globale) convergent, mais pas nécessairement vers une droite séparatrice ; la règle du Perceptron, elle, converge vers une droite séparatrice correcte lorsqu'elle existe, mais vers n'importe laquelle parmi les différentes solutions possibles (cela dépend des conditions initiales, du pas d'adaptation α, et de l'ordre de présentation des prototypes).

Mais, lorsque les classes ne sont pas linéairement séparables, les règles delta et delta généralisée convergent vers la solution des moindres carrés, qui ne corres-

pond pas nécessairement à celle qui minimise la probabilité d'erreur d'attribution de classes au sens de Bayes, tandis que la règle du perceptron ne converge pas.

On peut démontrer le résultat suivant, connu sous le nom de « théorème de convergence du Perceptron ».

Enoncé : Si les 2 classes sont linéairement séparables (au sens fort) et si les vecteurs à classer sont de norme bornée, l'algorithme d'apprentissage du Perceptron est convergent en un nombre fini d'étapes.

Convergence — (communiquée par M. Cottrell). On peut montrer qu'il est équivalent de supposer que les vecteurs à classer sont normés. Ensuite, on remarque que l'on peut remplacer tous les vecteurs x qui ont comme sortie attendue -1 par $-x$. Ainsi, si W^* est solution du problème, on aura $W^* . x > 0, \forall x$.

L'hypothèse de séparabilité forte est qu'il existe δ et W^* (qu'on peut prendre normé) tels que pour toute entrée x, on ait $W^* . x > \delta$.

On pose alors :

$$\beta(W(t)) = \frac{W^* . W(t)}{| W(t) |} = \cos (W^*, W(t))$$

et on a $\beta(W(t)) \leqslant 1$.

Pour toute modification de W, lors d'une présentation d'un exemple $x(k)$ mal classé, le numérateur sera modifié de la façon suivante :

$$W^* . W(t+1) = W^* . W(t) + \alpha W^* . x(t) \geqslant W^* . W(t) + \alpha\delta.$$

Après M changements, on a :

$$W^* . W(M) \geqslant W^* . W(0) + \alpha M \delta.$$

De même, au dénominateur,

$$| W(t+1)|^2 = | W(t)|^2 + 2\alpha W(t)\, x(t) + \alpha^2 | x(t)|^2 \leqslant | W(t)|^2 + \alpha^2$$

puisqu'il y a changement quand $W(t) . x(t) < 0$. D'où, après M changements, $| W(M)|^2 \leqslant | W(0)|^2 + \alpha^2 M$.

On pose, ce qui ne change rien, $W(0) = 0$, et donc :

$$\frac{\alpha M \delta}{\alpha \sqrt{M}} \leqslant \beta(W(M)) \leqslant 1$$

d'où l'on déduit que M est fini, majoré par $1/\delta^2$.

Donc si W^* existe, au bout d'un nombre fini d'étapes, $W(t)$ reste fixe et il y a donc convergence.

VII. — **Modèle à une couche**

Les modèles étudiés dans les parties précédentes ne comportent qu'un seul neurone. Les applications possibles avec de tels modèles sont donc très restreintes. Cependant on peut rassembler plusieurs neurones dans une même structure, ayant en commun les mêmes entrées, et que l'on nomme une *couche* de neurones. Au sein d'une couche, chaque neurone agit indépendamment des autres, et en particulier, il ne reçoit aucune connexion en provenance des neurones de cette couche. Cette structure est une généralisation des précédentes, qui permet de traiter des problèmes pour lesquels la sortie attendue n'est pas une seule valeur scalaire, mais un vecteur de K valeurs scalaires.

Un exemple de réseau à une couche de neurones est donné figure III.8.

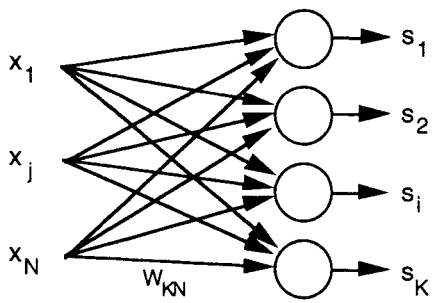

Fig. III.8. — Structure d'une couche de neurones

Chaque neurone reçoit N entrées, et calcule sa sortie s_k en fonction du modèle de neurone retenu : linéaire, non linéaire avec fonction sigmoïde, ou à échelon. Les connexions entre neurones sont représentées par une matrice W, W_{ik} représentant la

connexion de l'entrée i vers le neurone k. Le résultat du calcul délivré par le réseau est un vecteur d'activités $s = [s_1, s_2, \ldots s_K]$. Les règles d'adaptation des coefficients W sont strictement identiques à celles présentées dans les parties précédentes. En effet, pour chaque neurone, on connaît exactement la différence entre la sortie calculée et la sortie attendue. Il faut simplement, pour K neurones, effectuer le calcul K fois pour les K différentes sorties. Les méthodes algébriques ou itératives peuvent être utilisées, selon l'intérêt qu'elles présentent pour l'application visée. Néanmoins, la capacité de traitement présentée par une couche de neurones ne diffère pas de celle présentée par un seul neurone : c'est un séparateur linéaire. La seule différence vient du fait qu'une couche de neurones est une collection de séparateurs linéaires. Pour traiter des problèmes qui ne sont pas linéairement séparables, il faut enrichir la structure du réseau, et en particulier permettre la composition de couches.

VIII. — **Modèle à couches multiples**

Après avoir construit des couches de neurones, on généralise encore une fois en construisant des modèles à composition de couches. C'est une succession de couches de neurones, reliées entre elles par des coefficients synaptiques. Le modèle du neurone est non linéaire, de type sigmoïde ou, plus généralement, n'importe quelle fonction continue et dérivable[1]. Une structure à composition de couches est représentée sur la figure III.9.

1. Si l'on ne prend que des fonctions linéaires, le réseau se réduit à une seule couche linéaire et l'on perd la richesse d'une structure multicouches.

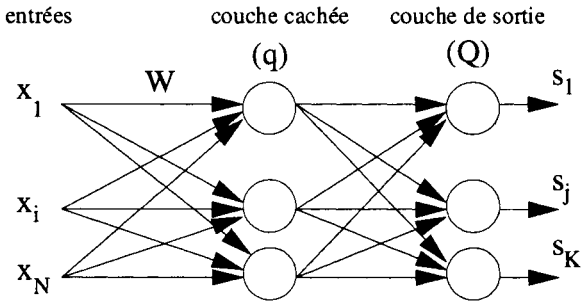

Fig. III.9. — Structure d'un réseau en couches

IX. — Le Perceptron multicouches

1. **Introduction.** — Les règles locales d'adaptation que nous venons d'étudier, fondées sur une optimisation par gradient stochastique, visent à déterminer la meilleure séparation linéaire entre classes selon un critère défini. Il va de soi qu'une simple classification linéaire limite fortement les applications que l'on peut envisager. Il faut donc la dépasser pour prétendre construire un système adaptatif capable de traiter des problèmes complexes comme la reconnaissance de caractères ou le contrôle de processus. Les *Perceptrons multicouches* sont une approche possible, qui connaît un grand succès dans le domaine des applications industrielles.

2. **Structure.** — Un Perceptron multicouches est une généralisation du modèle de l'Adaline. Il consiste en une succession de couches d'unités avec des fonctions de décision différentiables, reliées entre elles par des coefficients synaptiques comme nous l'avons vu sur la figure III.9.

Les neurones fonctionnent tous comme celui de l'Adaline avec une fonction non linéaire. Ainsi, pour autant qu'il y ait K_{q-1} neurones dans la couche $(q-1)$, le potentiel $p_j^{(q)}$ d'une unité j de la couche (q) vaut :

$$p_j^{(q)} = \sum_{i=1}^{N_{q-1}} W_{ij}^{(q)} x_i^{(q-1)}$$

où $W_{ij}^{(q)}$ représente le poids entre le neurone i de la couche $(q-1)$ et le neurone j de la couche (q), et $x_i^{(q-1)}$ est la sortie de l'unité i dans la couche $(q-1)$. La sortie de l'unité j de la couche q est alors donnée par :

$$x_j^{(q)} = \sigma(p_j^{(q)})$$

où σ est une fonction différentiable non linéaire, par exemple une tangente hyperbolique (th).

Tout le problème de l'apprentissage sur une telle structure vient du fait que la sortie que l'on souhaite associer à une entrée n'est disponible que pour les unités de sortie. Les unités dites internes, c'est-à-dire celles qui sont placées entre les entrées et la couche de sortie, n'ont aucune information sur le but à atteindre. Or, nous l'avons vu pour les règles précédentes, c'est essentiellement grâce à l'écart que l'on mesure entre la sortie calculée et celle désirée que l'on arrive à estimer la correction à appliquer pour s'approcher de l'erreur minimale. Sans mesure de l'erreur effectuée à l'intérieur du réseau, il a longtemps paru difficile d'appliquer une correction des coefficients. Une méthode a donc été proposée par P. Werbos puis par D. Rumelhart[1] et Le Cun, qui consiste à fixer un but aux unités internes à

1. D. Rumelhart, D. McClelland (1986), *Parallel Distributed Processing. Explorations in the microstructure of cognition*, vol. I, Cambridge, MIT Press.

partir de l'erreur mesurée en sortie du réseau, et véhiculée par les poids qui relient les couches entre elles. En réalité cette méthode est une simple application de la méthode de gradient stochastique, et comme on constate que le calcul du gradient de l'erreur instantanée se fait de proche en proche dans le sens rétrograde des connexions, on a appelé cette méthode *rétro-propagation du gradient.*

3. **Règle de la rétro-propagation du gradient.** — Pratiquement, l'on cherche à minimiser par une méthode de gradient stochastique la mesure de l'erreur quadratique instantanée. Ainsi, en omettant la variable W pour simplifier la notation, on écrit l'erreur :

$$\varepsilon(t) = \frac{1}{2} \| s(t) - y(t) \|^2$$

où $s(t)$ est le vecteur de sortie calculé par le réseau et $y(t)$ est le vecteur de sortie souhaité.

Calcul explicite des composantes du gradient. — Le terme du gradient peut se décomposer en deux termes qui sont évalués séparément, d'après la règle des dérivées de fonctions composées.

$$\frac{\partial \varepsilon}{\partial W_{ij}^{(q)}} = \frac{\partial \varepsilon}{\partial p_j^{(q)}} \frac{\partial p_i^{(q)}}{\partial W_{ij}^{(q)}}.$$

Pour alléger la notation, on omet volontairement l'indice temporel (t). On appelle *signal d'erreur*, noté $\Psi_j^{(q)}$, l'opposé du premier terme de cette dérivation.

On pose donc :

$$\Psi_j^{(q)} = -\frac{\partial \varepsilon}{\partial p_j^{(q)}}.$$

Le deuxième terme est évalué directement :

$$\frac{\partial p_j^{(q)}}{\partial W_{ij}^{(q)}} = \frac{\partial}{\partial W_{ij}^{(q)}} \left(\sum_{k=1}^{K_{q-1}} W_{kj}^{(q)} x_k^{(q-1)} \right) = x_i^{(q-1)}.$$

Le signal d'erreur est également décomposé en deux termes d'après la règle de dérivation composée :

$$\Psi_i^{(q)} = -\frac{\partial \varepsilon}{\partial p_j^{(q)}} = -\frac{\partial \varepsilon}{\partial x_j^{(q)}} \frac{\partial x_j^{(q)}}{\partial p_j^{(q)}}.$$

Or, comme

$$x_j^{(q)} = \sigma(p_j^{(q)})$$

on a immédiatement :

$$\frac{\partial x_j^{(q)}}{\partial p_j^{(q)}} = \sigma'(p_j^{(q)}).$$

Pour terminer le calcul du gradient de l'erreur il faut distinguer deux cas : i / (q) est la couche de sortie, numérotée Q ou ii / (q) est une couche intermédiaire.

i / Le neurone de sortie appartient à la couche Q :

$$\frac{\partial \varepsilon}{\partial x_j^{(Q)}} = \frac{1}{2} \frac{\partial}{\partial x_j^{(Q)}} \sum_{k=1}^{K_Q} (s_k - y_k)^2$$

$$\frac{\partial \varepsilon}{\partial x_j^{(Q)}} = (s_j - y_j).$$

Le signal d'erreur pour une unité j de la couche de sortie est donc :

$$\Psi_j^{(q)} = (y_j - s_j)\, \sigma'(p_j^{(q)}).$$

ii / Le neurone j appartient à une couche intermédiaire $q < Q$. On peut alors écrire :

$$\frac{\partial \varepsilon}{\partial x_j^{(q)}} = \sum_{k=1}^{K_{(q+1)}} \frac{\partial \varepsilon}{\partial p_k^{(q+1)}} \frac{\partial p_k^{(q+1)}}{\partial x_j^{(q)}}$$

où k numérote les unités qui « suivent » l'unité j, sur la couche $(q+1)$. On a donc :

$$\frac{\partial \varepsilon}{\partial x_j^{(q)}} = \sum_{k=1}^{K_{(q+1)}} \frac{\partial \varepsilon}{\partial p_k^{(q+1)}} \frac{\partial}{\partial x_j^{(q)}} \sum_{l=1}^{K_{(q)}} W_{lk}^{(q+1)}\, x_l^{(q)}$$

$$= -\sum_{k=1}^{K_{(q+1)}} \Psi_k^{(q+1)}\, W_{jk}^{(q+1)}$$

Le signal d'erreur Ψ_j pour les couches internes vaut donc :

$$\Psi_j^{(q)} = \left(\sum_{k=1}^{K_{(q+1)}} \Psi_k^{(q+1)} W_{jk}^{(q+1)} \right) \sigma'(p_j^{(q)}).$$

On voit que le signal d'erreur de la couche (q) se calcule à partir des signaux d'erreur des unités de la couche suivante, pondérés par les poids des connexions de j vers k, utilisées « à l'envers ». D'où la règle de modification des coefficients synaptiques, proportionnelle à l'opposé du gradient de l'erreur.

$$\Delta W_{ij}^{(q)} = \alpha \Psi_j^{(q)} x_i^{(q)}$$

où $\Psi_j^{(q)}$ est donné par l'une ou l'autre des deux expressions précédemment calculées.

4. **Propriétés.** — Le Perceptron multicouches connaît de nombreuses applications, essentiellement grâce à sa propriété d'approximation. Il a été montré qu'un réseau à une seule couche cachée est capable d'approximer n'importe quelle fonction définie sur un compact de R^n avec la précision souhaitée, à condition que le nombre de neurones cachés soit suffisant. C'est la propriété essentielle de ce réseau, qui lui confère la capacité d'approximateur universel. Néanmoins, cette propriété ne permet pas de choisir, pour un problème donné, le nombre d'unités optimal dans la couche cachée. Cette question est encore l'objet de recherches théoriques.

L'autre propriété importante présente un avantage dans le contexte de la classification. Il a été montré que les sorties d'un Perceptron multicouches tendent vers les probabilités bayesiennes lorsque la taille de l'ensemble d'apprentissage tend vers l'infini[1]. Ici encore, ceci est impossible à obtenir en pratique car la base d'apprentissage contenant les couples d'exem-

1. P. Comon (1992), Classification supervisée par réseaux multicouches, *Traitement du signal*, vol. 8, n° 6, p. 387-407.

ples n'est jamais infinie. On ne peut donc espérer atteindre qu'une valeur approchée des probabilités bayesiennes.

L'algorithme de rétro-propagation du gradient souffre d'un inconvénient majeur : le temps de convergence est extrêmement élevé, et peut conduire à des temps d'apprentissage irréalistes dans le cas de problèmes réels. Pour pallier ce problème, il existe des procédures plus rapides, mais d'une complexité de calcul plus élevée. Nous citerons pour mémoire les méthodes quasi newtowniennes et la méthode du gradient conjugué, qui sont des méthodes de minimisation très utilisées.

X. — **Réseaux à fonctions radiales de base**

Les réseaux à fonctions radiales de base (RBF, « Radial Basis Functions ») sont avant tout des réseaux multicouches, à une couche cachée (fig. III.10). Ils présentent néanmoins une différence fondamentale par rapport aux réseaux multicouches décrits plus haut : dans la couche cachée, on utilise des fonctions qui ne dépendent que de la norme de la différence entre le vecteur d'entrée $x = (x_1, \ldots, x_N)$ et un vecteur « centre » propre à chaque neurone, $c_i = (c_{i1}, \ldots, c_{iN})$; les réseaux à fonctions radiales de base tirent leur appellation de cette particularité. Dans la couche de sortie, la fonction d'activation est linéaire, le réseau se bornant à sommer les sorties des fonctions radiales multipliées par des poids W_k.

La figure III.10 représente un réseau à une sortie, c'est-à-dire destiné à approximer une fonction d'un compact de R^N dans R ; on peut néanmoins ajouter d'autres sorties au réseau RBF, en multipliant par d'autres vecteurs de poids les sorties des fonctions radiales. Comme nous le verrons, ceci n'influence pas la

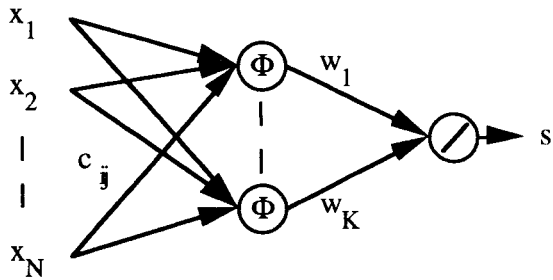

Fig. III.10. — Structure d'un réseau à fonctions radiales de base

méthode d'apprentissage destinée à calculer les différents paramètres des réseaux RBF.

La fonction s réalisée par un réseau RBF peut s'exprimer sous la forme d'une combinaison linéaire de fonctions radiales :

$$s(x) = \sum_{j=1}^{K} W_j \, \Phi(\| x - c_j \|).$$

En pratique, la fonction radiale la plus couramment utilisée est le noyau Gaussien :

$$\Phi(\| x - c_j \|) = \exp - \left(\frac{\| x - c_j \|}{\sigma_j} \right)^2$$

où σ_j est le facteur de largeur associé au noyau j.

Une fois la forme générale des noyaux choisie, l'apprentissage d'un réseau RBF consiste à choisir les paramètres W_j, σ_j et c_j de manière à apprendre le mieux possible les P couples $(x(i), y(i))$ d'une base d'apprentissage, les $y(i)$ correspondant aux sorties désirées du réseau RBF lorsque l'entrée $x(i)$ est présentée. La qua-

lité de l'apprentissage se mesure de façon classique selon un critère des moindres carrés :

$$\varepsilon = \frac{1}{2} \sum_{i=1}^{P} (y(i) - s(x(i)))^2.$$

Une autre différence fondamentale entre un réseau RBF et un Perceptron multicouches réside dans la façon dont l'apprentissage est réalisé. Dans le Perceptron multicouches, l'ensemble des paramètres libres du réseau était considéré simultanément, et une méthode de descente de gradient vise à adapter tous ces paramètres en même temps. Dans un réseau RBF par contre, et afin de lutter contre la lourdeur excessive des calculs, les trois types de paramètres sont calculés successivement et indépendamment les uns des autres ; l'avantage de cette méthode est bien entendu la simplicité et le nombre réduit d'opérations, le désavantage est qu'il n'y a pas de garantie explicite de convergence vers un minimum de la fonction d'erreur définie par rapport à l'ensemble des paramètres.

1. **Emplacement des centres des noyaux.** — L'idée utilisée pour la phase de calcul des centres de noyaux c_i est que pour minimiser l'erreur des moindres carrés définie plus haut, la qualité de l'approximation doit être d'autant meilleure que l'on est en présence d'une zone où le nombre de données dans la base d'apprentissage est important ; de plus, on suppose également que la qualité de l'approximation est d'autant meilleure que l'on dispose de plus de centres, et donc de fonctions noyaux, dans la zone considérée. Partant de cette constatation, on applique une quantification vectorielle sur les vecteurs x(*i*), afin d'obtenir une distribution dans l'espace des centres c_i similaire à la distribution des vecteurs de données $x(i)$, le nombre de vecteurs étant néanmoins réduit de P à K (fixé *a priori*).

N'importe quelle méthode de quantification vectorielle peut alors être envisagée, telle que la méthode classique LBG (Linde, Buzo, Gray), encore appelée GLA (Generalized Lloyd Algorithm). Une méthode plus simple néanmoins et qui peut être utilisée de façon itérative est celle de l'apprentissage compétitif : après une initialisation aléatoire ou quelque peu « dirigée » des K centres c_i, l'apprentissage compétitif consiste, à chaque présentation d'un vecteur d'entrée $x(i)$, à déterminer le centre c_k le plus proche de $x(i)$ (habituellement au sens d'une distance euclidienne), et à rapprocher le centre c_k du vecteur x(i) selon la règle :

$$c_k(t + 1) = c_k(t) + \alpha(t)\,(x(i) - c_k(t))$$

où t représente l'indice temporel de l'évolution de la position des centres en fonction des présentations successives des éléments de la base d'apprentissage. Le scalaire $\alpha(t)$ est un paramètre d'adaptation, choisi entre 0 et 1, et généralement décroissant au cours du temps pour assurer la convergence de la méthode.

Après une présentation répétée de l'ensemble des vecteurs $x(i)$ de la base d'apprentissage, la distribution des centres c_k sera similaire à celle des vecteurs $x(i)$, assurant dès lors une plus grande concentration des centres dans les zones de l'espace où le nombre de données d'apprentissage est plus important.

2. **Largeur des noyaux.** — Déterminer la largeur σ_i des noyaux revient à déterminer comment des noyaux voisins vont se recouvrir. Dans cet esprit, Moody et Darken[1] ont proposé de fixer les σ_i en minimisant une

1. J. Moody, C. Darken (1989), *Learning with localized receptive fields*, D. Touretzky *et al.* (eds), *Proceedings of the 1988 Connectionist Models Summer School*, San Mateo, CA, Morgan Kaufmann.

fonction d'erreur définie afin de tenir compte d'un paramètre de recouvrement Q :

$$\varepsilon_Q = \frac{1}{2} \sum_{r=1}^{K} \left[\sum_{s=1}^{K} \exp\left(-\left(\frac{\| c_s - c_r \|}{\sigma_r} \right)^2 \right) \left(\frac{\| c_s - c_r \|}{\sigma_r} \right)^2 - Q \right].$$

Les paramètres σ_i sont trouvés par une minimisation de cette fonction, par exemple à travers une méthode de descente de gradient. Une autre méthode consiste à estimer l'écart type de chaque noyau sur les vecteurs de la base d'apprentissage et à l'utiliser dans l'estimation de la largeur des noyaux (les σ_i de noyaux Gaussiens sont fixés au double de cet écart type). Là encore, plutôt que de calculer les largeurs de façon globale une fois que les centres sont fixés, il est possible de les estimer de façon itérative, en commençant avec des valeurs initiales nulles et en adaptant de façon itérative le σ_k du noyau dont le centre est le plus proche de chaque vecteur présenté $x(i)$:

$$\sigma_k(t + 1) = (1 - \alpha(t))\, \sigma_k(t) + \alpha(t)^2 \| x(i) - c_k(t) \|.$$

L'utilisation d'une méthode itérative pour le calcul de la position des centres et des largeurs des noyaux permet, en présence de bases d'apprentissage comprenant un grand nombre de vecteurs, de ne pas mémoriser ceux-ci, mais de les utiliser successivement au fur et à mesure de leur génération.

3. **Coefficients synaptiques.** — On peut remarquer que les deux premières étapes d'estimation des centres des noyaux et de leurs largeurs ne fait pas intervenir les sorties désirées $y(i)$ des vecteurs de la base d'apprentissage, mais se base uniquement sur les entrées $x(i)$. Bien entendu, le calcul des coefficients synaptiques du réseau fait intervenir ces sorties désirées : c'est en fait la fonction d'erreur des moindres carrés définie plus haut

qui sera minimisée pour trouver les coefficients synaptiques W_j. La fonction d'erreur est quadratique par rapport aux coefficients synaptiques et la simple annulation de ses dérivées partielles conduit à la valeur optimale des coefficients W_j :

$$W_j = \sum_{k=1}^{K} (\Phi^{-1})_{kj} \left[\sum_{i=1}^{P} \Phi_k (x(i))\, y(i) \right]$$

où $$(\Phi)_{kj} = \sum_{l=1}^{P} \Phi_k (x(a))\, \Phi_j(x(a))$$

et $$\Phi_k(x(a)) = \Phi(\| x(a) - c_k \|).$$

4. **Optimisation des paramètres.** — La procédure décrite ci-dessus pour le calcul des trois types de paramètres dans les réseaux RBF n'assure l'optimalité que des seuls coefficients W_j, à paramètres c_j et σ_j fixés. Pour obtenir des valeurs de l'ensemble des trois types de paramètres optimaux au sens de la fonction ε, il aurait été possible d'utiliser une simple méthode de descente de gradient sur cette fonction, par rapport cette fois à l'ensemble des paramètres W_j, c_j et σ_j. Cette méthode a néanmoins été évitée pour des raisons de complexité des calculs, mais également parce que la fonction ε est fortement non linéaire par rapport aux paramètres c_j et σ_j ; une méthode de descente de gradient aurait inévitablement convergé vers un minimum local de la fonction, sans aucune garantie que celui-ci donne une erreur petite. Par contre, si on considère que les coefficients obtenus après la méthode en trois étapes définie ci-dessus sont une assez bonne approximation des coefficients « idéaux », il est tout à fait adéquat de raffiner leurs valeurs par une procédure de descente de gradient sur la fonction ε, pour trouver un minimum de cette fonction qui est cette fois définie par rapport aux trois types de

paramètres du réseau. Cette procédure de raffinement permet d'augmenter considérablement la qualité de l'approximation tout en gardant la complexité des calculs dans des limites raisonnables.

Signalons enfin que pour la construction de réseaux RBF à plusieurs sorties, les étapes de calcul des positions c_j et largeurs σ_j des noyaux sont en tous points identiques (puisqu'elles ne font pas intervenir les sorties désirées $y(i)$), et que le calcul des coefficients synaptiques se fait également de façon identique, en considérant séparément les ensembles de poids raccordés aux différentes sorties.

Notes théoriques. — Plusieurs auteurs ont étudié ces réseaux sous l'angle théorique de l'approximation universelle de fonctions : Cybenko[1], Funahashi[2], Hornik, Stinchcombe et White[3] ont montré que toute fonction continue définie sur un compact peut être approximée par un perceptron multicouches avec seulement une couche interne d'unités non linéaires à fonction sigmoïde. Leshno[4] a ajouté à cela que cette propriété d'approximation universelle était également garantie avec n'importe quelle fonction d'activation non linéaire (sauf des fonctions polynomiales) et qu'elle peut être réalisée avec une précision choisie, pourvu que la couche cachée ait un nombre suffisant de neurones. Néanmoins, une différence a été soulignée entre les Perceptrons multicouches et les réseaux RBF. Pour ces derniers, la capacité de « meilleure approximation » a été montrée par Girosi et Poggio[5]. Enfin, d'un point de vue plus pratique, Wray[6]

1. G. Cybenko (1989), Approximation by superposition of sigmoidal functions, *Mathematics of Control, Signal and System*, 2, 303-314.
2. K. Funahashi (1989), On the approximate realization of continuous mappings by neural networks, *Neural Networks*, 2, 183-192.
3. K. Hornik, M. Stinchcombe, H. White (1989), Multilayer feedforward networks are universal approximators, *Neural Networks*, 2, 359-366.
4. M. Leshno, V. Ya. Lin, A. Pinkus, S. Schocken (1993), Multilayer feedforward networks with a nonpolynomial activation function can approximate any function, *Neural Network*, 6, 861-867.
5. F. Girosi, T. Poggio (1990), Networks and the best approximation property, *Biological Cybernetics*, 63, 169-176.
6. J. Wray, G. Green (1995), Neural Networks approximation theory and finite precision computation, *Neural Networks*, 8, 31-37.

a montré que toutes les preuves d'existence avancées par les travaux précédents ne sont pas justifiées si on considère les limitations imposées par une simulation numérique sur un ordinateur. En particulier, la différence entre Perceptrons Multicouches et réseaux RBF sur la propriété de meilleure approximation n'est plus pertinente.

XI. — **Réseaux à structure évolutive**

Dans les chapitres qui précèdent, sauf pour le RBF que nous venons de voir, les différents types de réseaux de neurones envisagés ont une structure fixe, c'est-à-dire un nombre de neurones, de synapses et un schéma de connexions entre ces éléments connus *a priori*. Par contre, d'autres réseaux possèdent des degrés de liberté supplémentaires. Le Perceptron multicouches en est un exemple ; même si le nombre de neurones dans les couches d'entrée et de sortie est fixé par le problème, le nombre de couches cachées et le nombre de neurones dans chacune de celles-ci restent des paramètres libres. Plutôt que de fixer ces paramètres de façon arbitraire, ou simplement en se basant sur l'expérience, ce qui est souvent le cas à cause du manque de critères objectifs, il peut être intéressant de concevoir des algorithmes d'apprentissage qui modifient la structure même du réseau au fur et à mesure des besoins et des contraintes ; on parle alors de réseaux à structure évolutive.

Un exemple typique d'un tel réseau est le RCE[1] (« Restricted Coulomb Energy »). Mentionnons tout d'abord une différence fondamentale dans les algorithmes de classification entre les réseaux décrits dans les chapitres qui précèdent et cet algorithme RCE. Les premiers appartiennent à la catégorie des algorithmes

1. D. Reilly, L. Cooper, C. Elbaum (1982), A neural model for category learning, *Biological Cybernetics*, 45, 35-41.

PLS (« Piecewise Linear Separation ») ; le principe est que l'espace d'entrée tout entier est divisé en un certain nombre de classes, la plupart du temps par des hyperplans séparateurs. Suivant la configuration des hyperplans, les régions qu'ils définissent peuvent donc être de taille finie ou infinie, mais dans tous les cas, il y a au moins certaines régions de taille infinie qui font en sorte que tout l'espace d'entrée soit couvert ; de cette façon, chaque vecteur d'entrée en phase de reconnaissance du réseau est attribué à une classe, même si ce vecteur est « loin » de tout vecteur utilisé lors de l'apprentissage. L'algorithme RCE appartient lui à la classe des algorithmes ROI (« Region-Of-Influence »), dont le principe est, durant l'apprentissage, de construire successivement des régions de taille limitée dans l'espace autour des vecteurs d'apprentissage. Dans ce type d'algorithme donc, la couverture de l'espace par le réseau n'est assurée que dans les régions où l'on trouve des vecteurs d'apprentissage. Les deux types d'algorithmes offrent bien sûr des avantages ; les premiers seront utilisés dans le cas général où chaque vecteur d'entrée doit obligatoirement être classé par le réseau, tandis que les seconds seront préférés lorsqu'un rejet est préférable à une classification douteuse dans une région de l'espace vide de tout vecteur d'apprentissage.

Sans entrer dans les détails de l'algorithme RCE, son principe peut être décrit de manière très simple. Au début de l'apprentissage, aucune région n'est définie dans l'espace. Chaque fois qu'un vecteur d'apprentissage est présenté au réseau et n'est pas classé dans la bonne classe, ce qui est forcément le cas au début de l'apprentissage, une « région d'influence » sphérique (ou hypersphérique dans le cas d'une dimension supérieure à 3) est créée autour du vecteur présenté, avec un rayon initial fixé *a priori*. Dans la suite, tout vec-

teur de la même classe présenté au réseau, et dont la position se situe à l'intérieur de cette région d'influence, est considéré comme étant correctement classé, et donc ne donne plus naissance à une nouvelle région d'influence. Par contre, un vecteur d'apprentissage d'une classe différente peut être mal classé, c'est-à-dire tomber dans la zone d'influence d'un ou de plusieurs vecteurs de classes différentes de la sienne ; dans ce cas, le rayon de ces régions est diminué jusqu'à ce que ces dernières ne contiennent plus le vecteur en question. Les deux processus (création d'une région et diminution du rayon d'une autre) peuvent être utilisés simultanément le cas échéant lors de la présentation d'un nouveau vecteur, et le processus est répété jusqu'à convergence complète du réseau, c'est-à-dire dire jusqu'à ce que tous les vecteurs d'apprentissage soient correctement classés.

C'est justement à cause de cette exigence que des problèmes peuvent se poser. En cas de recouvrement important entre classes, un algorithme constructif tel que le RCE, qui apprend jusqu'à ce que le taux de classification correct soit de 100 % sur la base d'apprentissage, peut créer un nombre trop important de régions d'influences, chacune de petite taille, centrées sur la plupart des vecteurs d'apprentissage. En résumé, dans un cas limite, on peut obtenir un nombre de régions d'influence égal au nombre de vecteurs d'apprentissage, chaque région n'englobant qu'un seul de ces vecteurs. Même si les performances de classification sur la base d'apprentissage sont toujours de 100 %, on voit que, dans ce cas, le phénomène de généralisation, c'est-à-dire la capacité du réseau à classer correctement des nouveaux vecteurs proches de vecteurs d'apprentissage, est inexistant, et donc que les performances du réseau sur des nouvelles données (ensemble de test) peuvent être mauvaises.

Ce phénomène se rencontre pour l'algorithme RCE, mais également pour d'autres méthodes constructives, comme celles qui essaient de construire un Perceptron multicouches en augmentant le nombre de neurones dans les couches cachées au fur et à mesure des besoins ; c'est le phénomène de sur-échantillonnage rencontré en approximation de fonctions. Pour combattre le phénomène tout en gardant l'idée de faire varier les structures des réseaux, il est possible d'utiliser des méthodes d'élagage, qui consistent à supprimer, soit au cours de l'apprentissage, soit après celui-ci, des neurones ou plus souvent des connexions inutiles, ou peu utiles, entre neurones.

Pour élaguer un réseau au cours de l'apprentissage, l'idée est de transformer l'habituelle fonction d'erreur qui est minimisée pour trouver les poids du réseau, par exemple par une méthode de descente de gradient, en lui ajoutant un terme tenant compte de la complexité du réseau. Différents termes peuvent être utilisés, les plus simples consistant en des sommes de carrés ou de valeurs absolues des poids ; après convergence de l'algorithme d'apprentissage, ces méthodes conduisent à avoir certains coefficients synaptiques de petites valeurs, coefficients qui peuvent alors être supprimés sans influencer de manière trop importante le comportement du réseau. L'inconvénient de ce type de méthode réside dans la pondération qu'il faut utiliser dans la fonction d'erreur, entre le terme d'erreur classique et celui ajouté pour tenir compte de la complexité du réseau ; une pondération trop importante en faveur du premier est sans effet en vue d'un élagage, tandis que le contraire modifie de façon trop importante la forme de la surface d'erreur, ce qui peut conduire à un minimum fort éloigné du minimum initial.

Une autre méthode possible appelée « robust lear-

ning »[1] consiste à essayer d'accorder une importance similaire à chacun des poids du réseau. Pour ce faire, chaque fois qu'un vecteur est présenté au réseau, l'apprentissage est effectué non pas en utilisant le réseau tout entier, mais en choisissant aléatoirement un sous-réseau, composé d'une partie seulement des connexions du réseau initial. Par cette méthode, on force les connexions qui n'auraient pas eu beaucoup d'importance dans le réseau à en avoir plus, ce qui permet lors d'une étape ultérieure d'élagage de supprimer certaines connexions de façon purement aléatoire, puisqu'elles seront toutes supposées avoir la même importance. Précisons toutefois que donner la même importance aux connexions ne veut pas dire leur donner des valeurs semblables : ce sont les variations des activations de neurones lorsque des vecteurs de différentes classes sont présentés au réseau qui sont importantes dans un problème de classification, et l'amplitude de ces variations n'est pas uniquement déterminée par la valeur absolue des poids, mais bien par une subtile combinaison de ceux-ci !

Les méthodes d'élagage après apprentissage illustrent parfaitement cette caractéristique. Elles visent à éliminer les connexions dans le réseau dont l'influence sur la fonction d'erreur est négligeable. Pour cela, on calcule la variation d'erreur due à la suppression de chaque connexion indépendamment, et on supprime celle(s) dont cette variation est la plus faible. Diverses hypothèses sur les développements en série associés au calcul de cette variation d'erreur conduisent aux méthodes d'OBD (« Optimal Brain Damage ») ou OBS (« Optimal Brain Surgeon »), qui conduisent toutes deux à une variation d'erreur proportionnelle non seu-

1. P. Kerlirzin, F. Vallet (1993), Robustness in multilayer perceptrons, *Neural Computation*, 5, 473-482.

lement au poids associé à la connexion elle-même, mais aussi à des termes du second ordre de la fonction d'erreur par rapport à ces poids. Une méthode[1] (Statistical Stepwise Method) fondée sur une démarche statistique analogue à la régression descendante généralise les méthodes citées ci-dessus, en permettant de pratiquer des tests d'hypothèses sur la nullité des connexions. Signalons enfin que toutes ces méthodes d'élagage, et d'autres basées sur les méthodes classiques appliquées aux arbres de décision binaires ou sur les algorithmes génétiques font encore actuellement l'objet de recherches intensives[2].

XII. — Le modèle de Hopfield

Le modèle de Hopfield[3] a été proposé au début des années 1980. La présentation de ce modèle, dans un article devenu célèbre, avait ceci d'original qu'elle utilisait un langage compréhensible tant par les physiciens que par ceux qui avaient posé, quelques années auparavant, ce qui allait devenir le domaine des réseaux de neurones artificiels. Le modèle a la particularité d'être dynamique ; contrairement à l'Adaline, au Perceptron simple et autres modèles similaires, dont les sorties sont calculées en une seule passe après présentation des entrées, le modèle de Hopfield réinjecte les sorties calculées à chaque itération dans le réseau, afin d'ini-

1. M. Cottrell, B. et Y. Girard, M. Mangeas, C. Muller (1994), SSM : a statistical stepwise method for weight elimination, *Proceedings of ICANN94,* Sorrento, Springer Verlag, vol. 1, p. 681-684.

2. Pour une revue détaillée de ces méthodes, le lecteur pourra consulter l'article C. Jutten, O. Fambon (1995), Pruning methods : A review, *Proceedings of the European Symposium on Artificial Neural Networks,* Bruxelles, M. Verleysen Ed., D facto publications, p. 129-140.

3. J. J. Hopfield (1982), Neural networks and physical systems with emergent collective computational abilities, *Proceedings of the National Academy of Science,* 79, 2554-2558.

tier une itération supplémentaire, jusqu'à ce que l'état interne du réseau devienne stable.

1. **Structure.** — Le modèle de Hopfield comporte, en toute généralité, N entrées i, N sorties σ_i, et un état interne composé de N valeurs x_i. Aussi paradoxal que cela puisse paraître, les entrées ne sont en général pas utilisées ; elles peuvent être vues comme un moyen d'imposer, avant une première itération des calculs propres au réseau, un état interne initial. Nous pouvons donc immédiatement écrire les équations du réseau comme :

$$s_i(t+1) = \sigma(x_i(t+1)) = \sigma\left(\sum_{j=1}^{N} W_{ij}\, s_j(t)\right)$$

où t représente l'indice temporel et W_{ij} repésente le poids de la connexion de l'unité j vers l'unité i. Dans ce modèle, la fonction non linéaire σ est une fonction seuil, qui donne la valeur + 1 lorsque son argument est plus grand que le seuil et – 1 lorsqu'il est plus petit ; on est donc en présence d'un réseau binaire, dont les valeurs de sortie sont restreintes à + 1 et – 1. En pratique, on prend ce seuil égal à 0.

Pour un modèle dynamique tel que le réseau de Hopfield, il faut déterminer comment les valeurs sont successivement mises à jour. La dynamique est bien évidemment fixée par l'ensemble des poids W_{ij} ; on peut cependant choisir dans quel ordre les sorties sont calculées et si, pour un même cycle du réseau, on utilise les nouvelles valeurs déjà calculées pour mettre à jour les autres. Dans la pratique, on retiendra trois schémas de mise à jour : la méthode *synchrone*, pour laquelle les valeurs de tous les neurones sont modifiées au même instant (en d'autres termes, on calcule les nouvelles valeurs dans un ordre sans importance, mais on n'utilise aucune de celles-ci avant d'avoir effectué

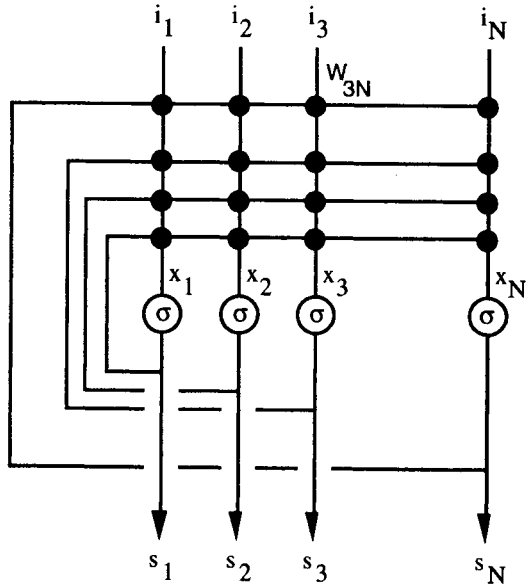

Fig. III.11. — Structure d'un réseau d'Hopfield

les calculs sur la totalité du réseau), la méthode *asynchrone* séquentielle, pour laquelle les valeurs des neurones sont mises à jour séquentiellement, et la méthode *asynchrone aléatoire*, pour laquelle l'ordre de remise à jour est aléatoire. Dans les deux derniers cas, un seul neurone voit donc son état changer à chaque itération : la nouvelle valeur ainsi calculée est utilisée pour la mise à jour consécutive d'un autre neurone, ceci pouvant donc être considéré comme une autre itération. L'asynchronisme des mises à jour des neurones est une condition essentielle de la convergence du réseau détaillée plus loin.

La propriété essentielle du modèle de Hopfield est sa convergence vers des états stables. Son utilisation peut en effet être résumée par la présentation d'un vecteur, utilisé pour fixer l'état interne initial du réseau, et des calculs successifs de cycles du réseau avant de converger vers un état stable, qui a préalablement été défini et mémorisé par l'intermédiaire des connexions W_{ij}. Une utilisation judicieuse passe donc par le choix d'une matrice W telle qu'un certain nombre d'états désirés soient stables et, si possible, qu'aucun autre état ne le soit.

2. **Convergence.** — Pour prouver la convergence du réseau de Hopfield vers des états stables, on pose généralement trois hypothèses : la mise à jour est asynchrone, la matrice W est symétrique et sa diagonale est nulle. Devant le nombre de paramètres (N^2) de la matrice W, les deux dernières hypothèses, qui réduisent ce nombre à $(N^2 - N)/2$, ne sont pas trop restrictives. L'essentiel est qu'on peut alors définir une « fonction d'énergie » :

$$\varepsilon(t) = -\frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} W_{ij}\, s_i(t)\, s_j(t).$$

Deux cas sont alors possibles lors d'une itération du réseau ; soit aucun des neurones ne change d'état, il est immédiat que l'énergie associée au réseau reste alors constante et l'état $s(i)$ est stable ; soit un seul neurone (selon l'hypothèse d'une mise à jour asynchrone) change d'état. Soit p l'indice de ce neurone. Nous allons montrer que dans ce dernier cas, l'énergie du réseau ne peut que diminuer. Comme la valeur de cette énergie est bornée inférieurement par définition (en supposant les poids W_{ij} eux-mêmes bornés en valeur absolue), on a alors la preuve que le réseau ne peut que

converger vers un état stable correspondant à un minimum de ε. Calculons la différence d'énergie entre les instants $t+1$ et t. On a :

$$\varepsilon(t+1) - \varepsilon(t) = -\frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} W_{ij}\, s_i(t+1)\, s_j(t+1) + \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} W_{ij}\, s_i(t)\, s_j(t).$$

En simplifiant tous les termes qui ne dépendent pas de l'indice p, et en utilisant les hypothèses sur la matrice W, on trouve :

$$\varepsilon(t+1) - \varepsilon(t) = -\sum_{i \neq p} W_{ip}\, s_i(t)\, s_p(t+1) + \sum_{i \neq p} W_{ip}\, s_i(t)\, s_p(t)$$

$$= (s_p(t) - s_p(t+1)) \left(\sum_{i \neq p} W_{pi}\, s_i(t) \right).$$

Enfin, en rappelant que les sorties $s(t)$ des neurones sont binaires ($+1$ ou -1), que le neurone p change d'état, et que les deux termes du produit sont toujours de signes opposés, nous avons dans tous les cas :

$$\varepsilon(t+1) - \varepsilon(t) < 0$$

ce qui complète la preuve.

3. **Apprentissage et états stables.** — Le choix des éléments de la matrice W permet de fixer les états vers lesquels tout processus de convergence à partir d'un vecteur d'état à l'instant 0 peut converger. Néanmoins, quelle que soit la méthode choisie pour calculer cette matrice, on est en présence d'états stables parasites, c'est-à-dire d'états stables qui se sont créés automati-

quement par l'enregistrement des états stables désirés ; nous en verrons des exemples plus loin.

Différentes possibilités peuvent être envisagées pour calculer la matrice W afin de fixer les états stables du réseau. Les conditions de stabilité d'un vecteur y sont, pour tout i :

$$\text{Sign} \left(\sum_{j=1}^{N} W_{ij} y_j \right) = y_i.$$

Si nous voulons enregistrer un seul vecteur $y(1)$ dans le réseau, on voit facilement que les conditions ci-dessus sont réalisées si on prend chaque W_{ij} proportionnel au produit $y_i(1)\, y_j(1)$, par exemple :

$$W_{ij} = \frac{1}{N} y_i(1)\, y_j(1).$$

Le vecteur $y(1)$ est donc stable en choisissant une matrice W selon la règle ci-dessus ; on peut néanmoins voir immédiatement que le vecteur $-y(1)$ est stable également, ce qui constitue un premier cas d'état stable parasite.

Si le vecteur $y(1)$ est stable, cela veut dire qu'en le présentant au réseau (c'est-à-dire en imposant ce vecteur comme état initial du réseau), la dynamique de celui-ci sera telle que l'état interne n'évoluera plus. En revanche, en initialisant le réseau à un autre vecteur quelconque, il se peut que l'évolution conduise également au vecteur $y(1)$ comme résultat final après convergence. Cette propriété est utilisée pour définir le « bassin d'attraction » du vecteur $y(1)$: il s'agit de l'ensemble des vecteurs initiaux qui conduisent au vecteur $y(1)$ après convergence du réseau. La principale utilité du modèle de Hopfield réside dans son utilisation en tant que mémoire auto-associative : on présente au réseau une version corrompue, ou incomplète,

du vecteur $y(1)$ (ou d'un autre vecteur correspondant à un état stable préenregistré), et on souhaite que la dynamique converge précisément vers cet état $y(1)$, qui peut être vu comme la version « correcte » du vecteur corrompu de départ. Le but d'un réseau de Hopfield est donc non seulement d'imposer un certain nombre d'états stables au réseau, mais aussi d'obtenir des bassins d'attraction aussi larges que possible autour de ces états, au sens d'une certaine mesure de distance. Dans le cas simple qui nous occupe (enregistrement d'un seul état stable), et en supposant N impair, on peut montrer qu'il n'existe que deux états stables ($y(1)$ et $-y(1)$), et que leurs bassins d'attractions sont tels qu'on peut inverser jusqu'à $(N-1)/2$ éléments d'un de ces deux vecteurs présentés tout en assurant la convergence vers le vecteur lui-même. Il n'y a donc que deux bassins d'attraction de tailles identiques.

Un cas plus intéressant est évidemment celui où l'on désire enregistrer P vecteurs comme états stables du réseau. Une extension immédiate du calcul de W vu ci-dessus est :

$$W_{ij} = \frac{1}{N} \sum_{k=1}^{P} y_i(k)\, y_j(k).$$

Bien que la règle d'apprentissage ci-dessus n'implique aucune notion temporelle dans la présentation des « vecteurs d'apprentissage » $y(k)$ au réseau, l'analogie avec la règle de Hebb est frappante : la connexion entre les neurones i et j est augmentée lorsque les valeurs des composantes i et j d'un vecteur d'apprentissage sont identiques, et diminuée dans le cas contraire. Cette règle est la plus utilisée et c'est celle qui a été étudiée par J. Hopfield. Cependant, la règle de Hebb rend délicate l'utilisation d'un réseau de Hopfield ; de nombreux résultats théoriques montrent en effet ses limites, tant du

point de vue de la capacité à stocker un nombre important de points fixes (états stables) que du point de vue de la taille des bassins d'attraction qui y sont associés. Citons deux de ces résultats les plus connus. Le premier montre que, lorsque N est grand, si trois vecteurs $y(1)$, $y(2)$ et $y(3)$ sont enregistrés dans le réseau et sont donc stables, n'importe quelle des huit combinaisons $\pm y(1) \pm y(2) \pm y(3)$ sera stable (ce qui, avec les trois vecteurs $-y(1)$, $-y(2)$ et $-y(3)$, fait déjà 11 états parasites pour 3 états désirés !). Le second prouve que, toujours en grande dimension, le nombre d'états stables désirés ne peut dépasser environ 0,15 N vecteurs, qui sont de plus soumis à des conditions très sévères d'orthogonalité ; ce chiffre est bien entendu à comparer aux N^2 poids (donc éléments mémoires) du réseau. Le résultat le plus négatif est donc que le quotient du nombre d'états stables mémorisables par N unités tend vers 0 quand N tend vers l'infini !

Pour pallier les limitations de la règle de Hebb, on peut utiliser les autres méthodes d'apprentissage vues dans le cas d'un neurone isolé. Chaque neurone du modèle de Hopfield peut en effet être considéré comme un neurone isolé, à N entrées s_i, et à une sortie s_j. On peut suivant cette méthode utiliser la règle de la projection, la règle du perceptron, les règles basées sur les gradients, les règles de Ho-Kashyap, la règle d'optimisation,... Notons cependant que pour les règles nécessitant habituellement une fonction non linéaire continue et dérivable, comme celle de la projection, on pose l'hypothèse qui consiste à utiliser une fonction linéaire lors de l'apprentissage, et à la remplacer par la fonction signe lors de l'utilisation du réseau, afin de ne plus avoir que des sorties binaires.

4. **Performances.** — Les performances d'un réseau d'Hopfield peuvent être présentées sous deux aspects

bien différents. Tout d'abord, il est important de voir si les vecteurs qui ont été utilisés lors de l'apprentissage sont correctement mémorisés ; ensuite, la taille des bassins d'attraction peut être évaluée en présentant au réseau des vecteurs proches de ceux qui ont été utilisés lors de l'apprentissage, et en calculant le pourcentage de ceux-ci qui convergent vers le « bon » état stable, c'est-à-dire celui qui est le plus proche du vecteur présenté. Pour procéder à cette évaluation on utilise la distance de Hamming entre vecteurs : il s'agit simplement du nombre de composantes (+ 1 et – 1) qui diffèrent entre deux vecteurs ; la distance de Hamming peut donc varier entre 0 et la dimension de ceux-ci.

Pour combiner les deux mesures de performances mentionnées, la première peut être vue comme un test de vecteurs d'entrée dont la distance de Hamming avec un des vecteurs utilisés lors de l'apprentissage est 0, tandis que la seconde peut être vue comme le même test, mais en utilisant une distance de Hamming différente de 0. Les deux mesures peuvent alors être représentées sur un diagramme qui donne le pourcentage de convergences correctes de vecteurs d'entrée, en fonction de la distance de Hamming entre chaque vecteur d'entrée et le vecteur le plus proche ayant servi à l'apprentissage ; une convergence correcte d'un vecteur d'entrée est définie comme la convergence de celui-ci vers le vecteur le plus proche utilisé lors de l'apprentissage, après un nombre indéterminé de cycles de calcul. La première valeur de ce diagramme (pour une distance de Hamming nulle) sera de 100 % si l'ensemble des vecteurs d'apprentissage a été correctement mémorisé, ce qui peut constituer une première façon de vérifier le bien-fondé d'un apprentissage.

Signalons enfin que, pour comparer différentes méthodes d'apprentissage sous des conditions différentes (nombre de vecteurs d'apprentissage, orthogo-

nalité...), il peut être utile d'obtenir une valeur unique pour caractériser les performances, plutôt qu'un diagramme. La méthode couramment utilisée est alors d'intégrer le diagramme précédent, en pondérant les valeurs par une fonction décroissante de la distance de Hamming ; comme nous savons que les opposés de vecteurs enregistrés le sont également de manière automatique, les diagrammes de performances ci-dessus ne sont utilisés que jusqu'à une distance de Hamming valant la moitié de la dimension des vecteurs, et la fonction de pondération pour intégrer le diagramme sera choisie de telle manière qu'elle s'annule pour cette distance.

Chapitre IV

LA PERCEPTION

Dans cette partie, nous présentons deux modèles qui se rapportent particulièrement à des tâches liées à la perception. Le modèle de Kohonen, inspiré de structures biologiques, est utilisé pour mettre en évidence des rapports entre des objets représentés par un grand nombre de caractéristiques. Il révèle des ressemblances difficiles à percevoir directement dans les mesures. De plus, le modèle de Kohonen permet d'aborder de façon pratique la question de l'auto-organisation. Le modèle de Hérault-Jutten, appelé aussi Analyse en composantes indépendantes, en révélant les composantes indépendantes d'une mesure, permet ainsi d'accéder aux grandeurs fondamentales d'un signal. Tous deux sont utilisés dans des applications industrielles.

I. — Le modèle de Kohonen

Les réseaux biologiques présentent une organisation des cellules qui dépend souvent de leur spécialisation. Dans le cortex, par exemple, les zones réceptrices sont localement ordonnées selon un ordre identique à celui des unités de l'organe sensoriel lui-même. De plus, on remarque une correspondance topologique entre ces zones : deux zones proches dans le cortex visuel correspondent à deux zones également proches dans la rétine, comme cela a été montré par Hubel et Wiesel en 1947.

Cette observation est également valable pour le cor-

tex auditif, avec un arrangement tonotopique des neurones, c'est-à-dire que des fréquences proches sont détectées par des neurones proches dans le cortex auditif. La surface sensorielle superficielle du corps est également représentée par une carte somatotopique dans le cortex moteur.

Plusieurs expériences mettent en évidence le fait que cette organisation n'est pas génétique, mais qu'elle se met en place au cours d'une phase d'apprentissage. En d'autres termes, la représentation interne du corps se construit partiellement par l'expérience.

Ainsi, dans le cerveau, il semble exister des zones qui répondent spécifiquement à des stimuli, en fonction de leur modalité (auditif, visuel, musculaire, etc.) et de leur destination (parole, détermination de trajectoire, etc.). Ces zones, après analyse électrophysiologique, révèlent une organisation interne, qui leur permet de répondre spécifiquement à des conditions d'excitation déterminées. On les appelle des *champs récepteurs*. Ils produisent une réponse spécifique lors de la présentation de motifs particuliers (cortex visuel), ou de fréquences (cortex auditif). Ils jouent donc un rôle de *détecteurs de caractères* de l'environnement, et sont construits par les stimulations elles-mêmes.

Comment une telle représentation peut-elle se construire dans le cortex ? C'est pour apporter un début de réponse à cette question que C. Von der Marlsburg et T. Kohonen[1] ont proposé un modèle qui tente de reproduire une carte d'organisation topologique à partir de stimuli appliqués à un système artificiel. Il faut noter que ce processus ne dépend que des entrées du système et que l'organisation en carte n'est pas construite à partir des indications d'un professeur. En ce sens, on dit que la

1. T. Kohonen (1982), Self-organized formation of topologically correct feature maps, *Biological Cybernetics*, 43, 59-69.

carte est le résultat d'une auto-organisation, et on parle alors *d'apprentissage non supervisé.*

1. **Structure du réseau.** — Suivant le symbolisme que nous avons employé jusqu'à présent, le modèle de Kohonen peut être représenté sur la figure IV.1 (espace d'entrée à deux dimensions et 5 unités) :

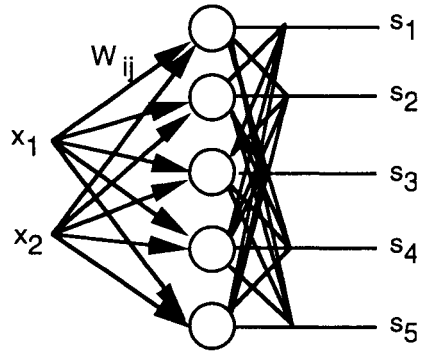

Fig. IV.1. — Structure du réseau

Le réseau est constitué de deux couches d'interconnexions et d'une seule couche de neurones. La première couche d'interconnexions est dite plastique, car c'est elle qui mémorise les modifications de poids. La seconde couche d'interconnexions fixes joue le rôle d'un réseau compétitif destiné à renforcer sélectivement l'activité du réseau.

2. **Fonction de la première couche.** — La première couche d'interconnexions est utilisée pour déterminer l'activité de chaque neurone de sortie, en fonction d'un vecteur de stimuli présenté en entrée. Si l'on note K le nombre de neurones et N le nombre d'entrées de chaque

neurone du réseau, x un vecteur de stimuli de composantes $x = [x_1, x_2, \ldots, x_N]^T$, W la matrice (K, N) des coefficients et s le vecteur d'activité des neurones, $s = [s_1, s_2, s_K]^T$, alors pour une entrée $x(t)$, l'activité des neurones se calcule par :

$$s(t) = Wx(t).$$

On retrouve donc le modèle du neurone linéaire tel que nous l'avons étudié dans le chapitre sur les mémoires associatives. Si les poids W_i et les entrées $x(t)$ sont normés, le produit scalaire calculé entre l'entrée et chaque ligne de la matrice W_i est maximum quand $x(t)$ et Wi sont colinéaires et de même sens. Puisque alors le produit scalaire mesure le cosinus de l'angle entre les deux vecteurs, si les vecteurs ne sont pas convenablement normés, le produit scalaire ne mesure plus exactement la proximité ; il faut utiliser une normalisation adéquate pour comparer les directions des vecteurs. La fonction de la première couche est donc de produire une mesure de similarité entre les vecteurs poids associés à chaque neurone et l'entrée présentée.

3. **Fonction de la seconde couche.** — La seconde couche d'interconnexions implante un réseau à inhibitions latérales récurrentes. Chaque neurone est en relation avec ses voisins selon une fonction d'interaction telle que les poids associés aux connexions entre des neurones physiquement voisins sont élevés. La valeur du lien entre neurones diminue avec la distance à laquelle ils se trouvent. Il a été montré par Kohonen que la fonction de cette couche consiste à renforcer l'activité d'un ensemble de neurones qui répondent préférentiellement à un stimulus présenté en entrée. C'est grâce à elle que se forme une réponse localisée sur un ensemble de neurones qui réagissent à une entrée. Large au début de la stimulation, le rayon de la

réponse varie au cours du temps pour devenir de plus en plus petit autour de l'unité d'activité maximale. Ce type de structure existe dans le cortex, par exemple chez le primate, où l'on observe une excitation latérale entre cellules dans un rayon de 50 à 100 microns, suivie de connexions inhibitrices dans un rayon de 200 à 500 microns.

4. **Fonction du modèle.** — Les deux couches du réseau de Kohonen réalisent ainsi deux tâches distinctes. La première calcule un ensemble d'activités : distances entre l'entrée présentée et les vecteurs W_i attachés à chaque neurone. La seconde établit une zone d'activité stable, mais de rayon variable au cours du temps, autour de l'unité qui présente l'activité la plus élevée. On peut donc formaliser ces deux opérations en utilisant une mesure de distance $d(x(t), W_i)$ et une recherche de maximum (ou de minimum selon la mesure de distance utilisée) dans un ensemble de valeurs réelles.

Le dernier élément du modèle concerne la règle d'adaptation des coefficients. Elle opère un renforcement de similarité entre l'entrée présentée et les vecteurs attachés aux neurones qui répondent préférentiellement.

Formellement, pour une entrée $x(t)$, et pour une mesure de distance choisie, l'activité du réseau est calculée par :

$$p_i(t) = d(x(t), W_i), \quad 1 \leqslant i \leqslant K.$$

Ensuite, on recherche parmi les composantes $p_i(t)$, celle qui traduit la plus grande similarité entre $x(t)$ et W_i. Dans le cas d'un produit scalaire, une similarité maximum correspond à la plus grande valeur ; pour une distance euclidienne c'est la plus petite. Pour simplifier le calcul, afin d'éviter des normalisations de vecteurs, on retient généralement cette dernière. Ainsi,

l'activité de sortie d'un réseau de K neurones est calculée par :

$$s_k(t) = 1 \quad \text{si} \quad p_k(t) = \min(p_i(t)), \; s_i(t) = 0 \quad \text{si} \quad i \neq k.$$

On définit ainsi le neurone gagnant comme le neurone d'indice k. Si l'on souhaite modifier les coefficients W du réseau, on utilise la règle d'adaptation suivante :

$$W_i(t+1) = W_i(t) + \alpha(t)\,(x(t) - W_i(t)) \quad \text{si} \quad i \in V_k(t)$$
$$W_i(t+1) = W_i(t) \quad \text{si} \quad i \notin V_k(t)$$

où $V_k(t)$ est le voisinage du neurone k (le plus proche de $x(t)$) au temps t.

Dans cette règle apparaît une notion de voisinage entre neurones. Nous l'avons évoquée dans la description du modèle, mais elle prend ici toute son importance. En effet, contrairement à ce que nous avons vu pour les autres modèles, il existe ici une notion de *topologie* entre les neurones eux-mêmes, c'est-à-dire que leurs places relatives ont une importance. On parlera par exemple de réseau linéaire, pour lequel chaque neurone est encadré par deux autres neurones, sauf bien sûr le premier et le dernier. On parlera aussi de réseau bi-dimensionnel carré, où chaque neurone est encadré physiquement par 4 voisins, comme cela est montré sur la figure IV.2, ou hexagonal si chaque neurone a 6 voisins.

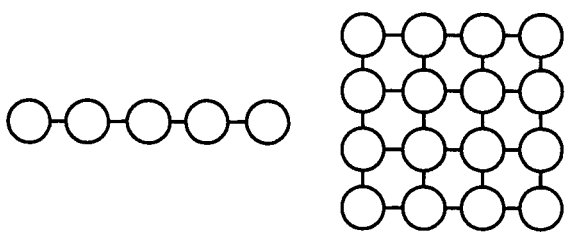

Fig. IV.2. — Formes de réseaux mono-dimensionnel et bi-dimensionnel

Le *voisinage d'adaptation* d'un neurone est alors une zone qui respecte la relation qui définit le *voisinage physique*, mais qui peut s'étendre sur une région plus vaste. On parlera donc de 1er, 2e ou *n*-ième voisins, suivant la distance à laquelle on va sélectionner les neurones, tout en respectant cette relation. Un exemple est donné sur la figure IV.3.

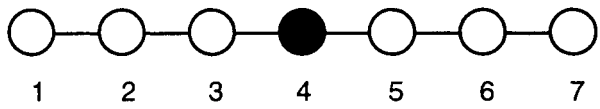

Fig. IV.3. — Voisinages : les premiers voisins du neurone 4 sont les 3 et 5, les seconds voisins sont les neurones 2 et 6, les troisièmes voisins sont les neurones 1 et 7

Ce voisinage peut varier au cours du temps, et permettra de contrôler le nombre de vecteurs modifiés autour du neurone gagnant pour une entrée $x(t)$. Pour une entrée donnée on rapproche donc « un peu » les coefficients synaptiques du neurone sélectionné et ceux qui font partie de son voisinage, et on laisse les autres inchangés. Ceci est réalisé pour l'ensemble des entrées disponibles. L'amplitude de la modification dépend du coefficient $\alpha(t)$, qui décroît au cours du temps.

Le moyen le plus simple pour percevoir l'effet de l'algorithme consiste à représenter les neurones dans l'espace des coefficients. En deux dimensions par exemple, chaque neurone i est placé aux coordonnées de son vecteur de coefficients (W_{i1}, W_{i2}). Modifier ce vecteur se traduit par un déplacement du neurone dans cette représentation. Les vecteurs d'entrée sont de même dimension que les vecteurs de coefficients associés à chaque neurone ; ils peuvent donc être

représentés sur un même diagramme, sans chercher à donner une signification aux axes. Deux exemples de représentations sont donnés sur la figure IV.4. La première montre un exemple de structure physique d'un réseau mono-dimensionnel à deux entrées : chaque neurone porte un numéro, et se trouve relié à ses deux entrées par ses poids synaptiques. La seconde montre le même réseau, représenté cette fois dans l'espace des coefficients. On remarque que l'emplacement des neurones dépend de la valeur de ses coefficients, et que la relation physique entre voisins est rappelée par les lignes qui relient les neurones.

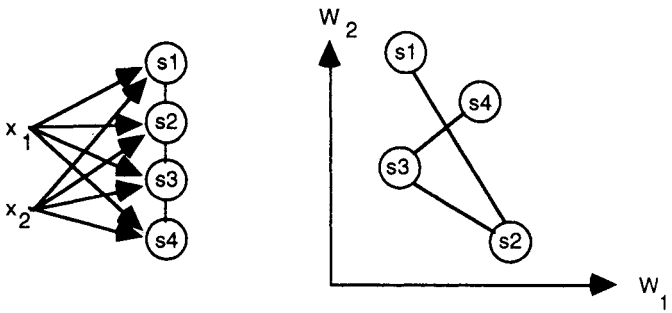

Fig. IV.4. — Représentations du réseau

C'est grâce à cette seconde représentation qu'on visualise l'organisation qui apparaît au cours de l'apprentissage. Considérons maintenant un réseau bi-dimensionnel de 10 par 10 neurones à voisinage carré avec des entrées de dimension 2 et pour lequel les coefficients sont initialisés aléatoirement. La représentation choisie produit le diagramme de la figure IV.5.

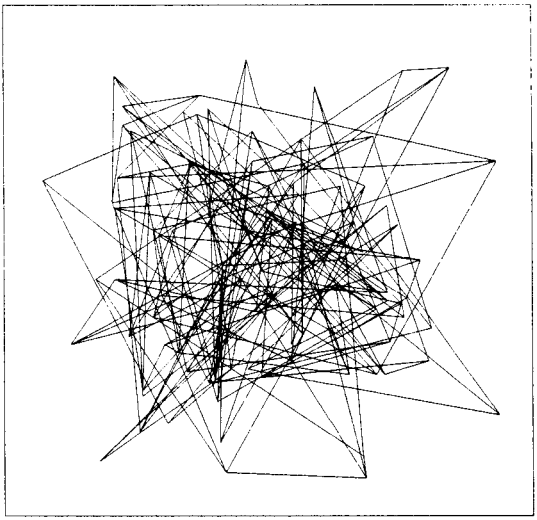

Fig. IV.5. — Répartition initiale des coefficients

Si maintenant on fournit au réseau des entrées uniformément distribuées dans un carré, la position des neurones dans l'espace des coefficients aboutira généralement à la répartition montrée sur la figure IV.6. Les trois diagrammes correspondent à des photographies de l'évolution des positions aux instants $t = 100$, 200 et 1 000 itérations. On remarque l'apparition d'une « organisation » du réseau, qui était initialement totalement désordonné.

Que traduit cette organisation ? Il faut se rappeler que la dimension des entrées ainsi que la dimension des poids liés à chaque neurone sont les mêmes. D'autre part, par la règle de calcul des coefficients, il est clair que le domaine de variation des coefficients est confondu

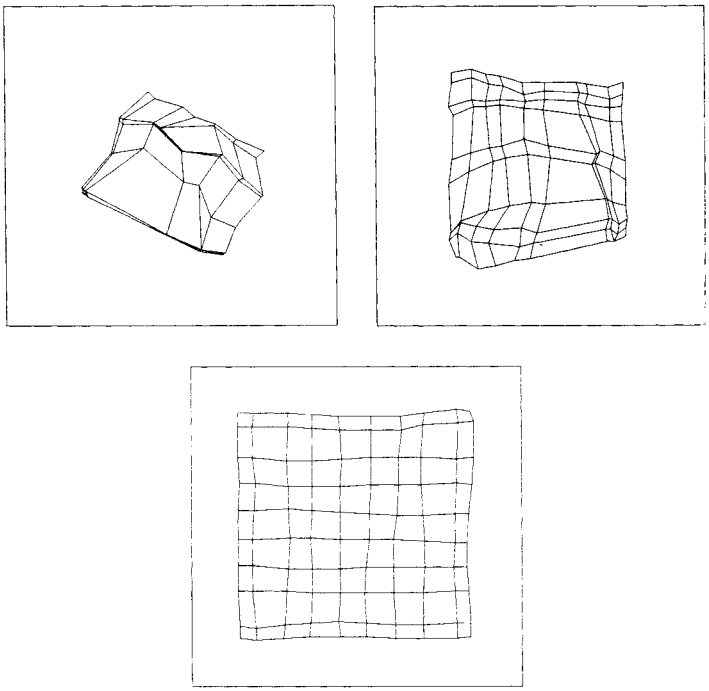

Fig. IV.6. — Adaptation des coefficients pour une distribution d'entrée uniforme sur un carré, et pour un réseau de 10×10 neurones

avec celui des entrées (puisque au temps $t + 1$ les vecteurs sont des combinaisons linéaires convexes des vecteurs au temps t et de l'entrée précédente) et, par conséquent, la distribution des entrées et des poids peut être représentée dans le même espace. Ainsi, la correspondance entre les deux apparaît-elle naturellement, comme le montre la figure IV.7.

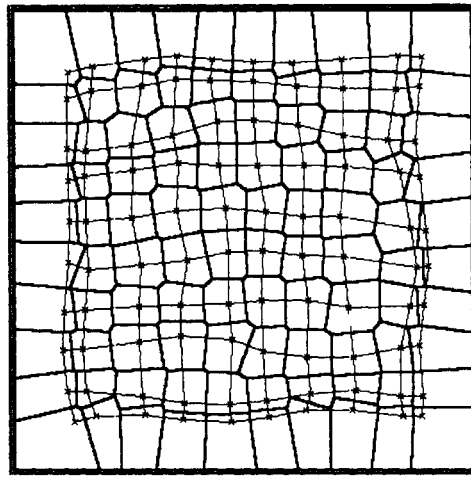

Fig. IV.7. — Correspondance entre distribution des entrées et position des neurones

Chaque neurone est représenté par une croix, et la relation entre neurones par les liens entre les croix. D'autre part, on remarque autour de chacun d'eux une zone polygonale ; elle représente la région de l'espace d'entrée qui est « prise en charge » par chaque neurone. Le forme précise de cette région est liée au choix de la mesure de distance euclidienne, et de la fonction de sélection qui choisit le neurone qui a la plus petite distance euclidienne entre une entrée et ses poids. Ceci conduit à tracer ce que l'on nomme les zones de Voronoï associées aux neurones, et dont l'union couvre la totalité de l'espace d'entrée. En d'autres termes, l'algorithme présenté ici effectue une quantification vectorielle de l'espace d'entrée, car il réduit la totalité de cet espace à un nombre fini de vecteurs W_i. Cette quantification a

une autre propriété : elle respecte la topologie de l'espace, c'est-à-dire que deux points voisins, suffisamment proches dans l'espace d'entrée, sont représentés par le même vecteur ou par deux vecteurs de coefficients voisins, comme le montre l'organisation obtenue à la convergence. C'est une des propriétés-clefs de ce modèle.

Jusqu'à présent, nous avons mis en évidence le fonctionnement du réseau sur des distributions uniformes carrées. Cependant, une propriété importante du réseau est que la quantification s'adapte à des distributions non uniformes, et sur des masques variables. Prenons par exemple une distribution de points bi-dimensionnelle uniforme triangulaire et un réseau dont les relations physiques entre neurones sont bi-dimensionnelles. Dans ce cas, l'adaptation conduit à l'approximation d'une surface triangulaire par un maillage carré, qui est représentée sur la figure IV.8.

Cette organisation ne peut évidemment pas conduire à une maille parfaitement régulière, mais elle

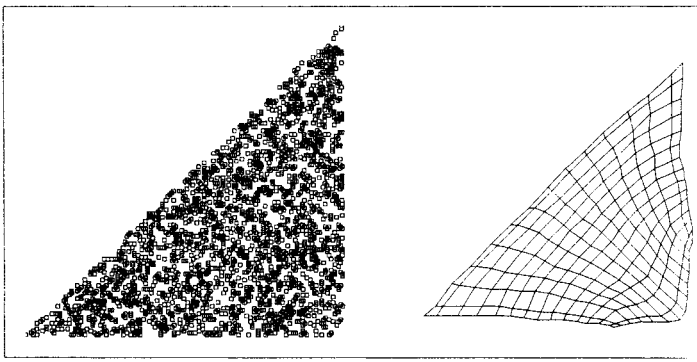

Fig. IV.8. — Auto-organisation
sur une distribution triangulaire uniforme

minimise les déformations afin que la probabilité de tirage soit asymptotiquement équivalente pour chacun des neurones lorsque le nombre de neurones tend vers l'infini. Ainsi, si l'on choisit maintenant une distribution non uniforme, cette propriété conduit à une densité des neurones plus grande dans les régions de l'espace où la probabilité d'avoir des stimuli est grande, de façon à garder l'approximation équivalente pour chaque neurone. Ce comportement est illustré par la figure IV.9. L'algorithme effectue donc une approximation discrète de la fonction de densité de la distribution d'entrée.

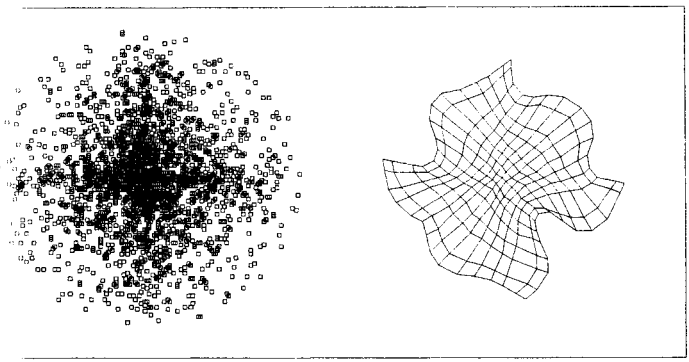

Fig. IV.9. — Auto-organisation
sur une distribution circulaire quadratique

Enfin, pour généraliser la description de l'algorithme, il faut souligner le fait que la structure du réseau, définie par les relations de voisinage physiques entre neurones, définit la dimension d'une pseudo-variété sur laquelle on effectue la quantification. Pour des stimuli d'entrée tri-dimensionnels, si la structure du réseau est bi-dimensionnelle, ce dernier représente

la distribution en plaçant les neurones de sorte à réduire la dimension de l'espace d'entrée, tout en conservant le mieux possible sa topologie. Ce comportement est mis en évidence sur la figure IV.10.

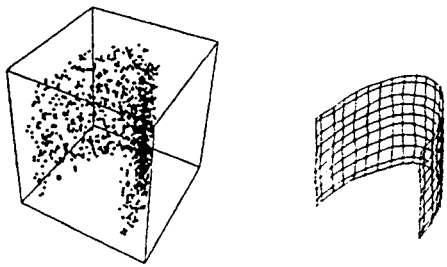

Fig. IV.10. — Approximation d'une distribution 3D par un réseau 2D
(1) Distribution ; (2) Forme du réseau 2D dans l'espace d'entrée

Ce principe de réduction de dimension est connu en analyse de données. En particulier, l'Analyse en composantes principales réalise un traitement comparable à celui effectué par le réseau de Kohonen. Néanmoins, de nombreuses différences notables ne permettent pas de les confondre : l'ACP est une méthode linéaire, qui réalise une réduction de dimension par projection orthogonale sur un plan, ou une droite. Le réseau de Kohonen effectue une double opération de quantification vectorielle en maintenant une métrique suivant une relation d'ordre définie *a priori*.

Note théorique. D'un point de vue théorique, l'étude de l'algorithme de Kohonen est difficile[1], en particulier quand la distribution des entrées suit une loi continue en dimension supérieure

1. M. Cottrell, J.-C. Fort, G. Pagès (1994), Two or three things that we know about the Kohonen algorithm, *Proceedings of the ESANN94 conference*, M. Verleysen Ed., D facto publications, 235-243.

à 1. Cependant, dans le cas où la loi des entrées est discrète, ce qui est généralement le cas en pratique, il a été montré que l'algorithme de Kohonen peut être considéré comme un algorithme de type gradient stochastique, qui minimise une fonction de potentiel définie par :

$$V(W_1, \ldots, W_{N \times N}) = \sum_{i=1}^{N \times N} \sum_{x_j} \| x_j - W_i \|^2$$

où x_j appartient à C_i ou $C_{i'}$ i' voisin de i.

Dans cette formule, x_j représente l'ensemble des vecteurs d'entrée attribués à la même unité gagnante de coefficients W_i. C'est une généralisation de la variance intra-classe, en prenant en compte les vecteurs de référence voisins. Cependant, cette fonction n'est pas partout dérivable si V est continue, ce qui complique beaucoup son étude.

II. — Le modèle de Hérault-Jutten

La plupart des modèles de réseaux de neurones trouvent leur utilité et leur puissance dans le grand nombre d'éléments de calcul, neurones et synapses, dont l'action conjointe est exploitée pour obtenir les résultats désirés. Le modèle de Hérault-Jutten va à l'encontre de ce principe : il se base sur un petit nombre d'éléments de calcul. Néanmoins, les principes d'apprentissage et d'adaptation sont similaires à ceux des autres modèles, ce qui lui donne l'appellation de « réseau de neurones ».

Le modèle de Hérault-Jutten[1] du nom de ses auteurs, encore appelé « séparation de sources indé-

1. J. Hérault, C. Jutten, B. Ans (1985), *Détection de grandeurs primitives dans un message composite par une architecture de calcul neuromimétique en apprentissage non supervisé,* Actes du X[e] Colloque GRETSI, Nice, 1017-1022.

pendantes » ou « analyse en composantes indépendantes », tente de trouver une solution au problème bien connu de la *cocktail party*. Supposons que deux signaux indépendants soient connus uniquement à travers plusieurs de leurs mélanges différents ; le cas peut se poser par exemple lorsque deux sources sonores se trouvent dans une même pièce, et que des microphones placés en différents endroits captent des mélanges *différents* de ces sources. Le problème qui se pose alors est de retrouver les signaux correspondant aux sources de départ, à partir des signaux mélangés recueillis aux microphones.

Dans la suite, nous allons supposer que les mélanges de signaux sont linéaires et additifs ; dans la pratique, cette condition est évidemment sévère, mais un bon nombre de situations peuvent néanmoins être approximées de la sorte au premier ordre. Précisons également que le modèle de Hérault-Jutten fait l'objet de nombreuses extensions à des mélanges ne respectant pas ces conditions ; leur description sort néanmoins du cadre de cet ouvrage.

Supposons donc que deux signaux inconnus et indépendants $e_1(t)$ et $e_2(t)$ soient mélangés de façon linéaire et additive pour donner deux mélanges *différents* $x_1(t)$ et $x_2(t)$. Les mélanges peuvent donc s'écrire respectivement $x_1(t) = a_{11}\, e_1(t) + a_{12}\, e_2(t)$, et $x_2(t) = a_{21}\, e_1(t) + a_{22}\, e_2(t)$, ou encore, sous forme matricielle, $x(t) = \mathrm{A}.e(t)$ avec $x(t) = (x_1(t), x_2(t))^{\mathrm{T}}$. La matrice A du mélange est bien entendu inconnue ; la seule information disponible est l'indépendance supposée des signaux $e_1(t)$ et $e_2(t)$. L'algorithme de Hérault-Jutten se base sur l'hypothèse qu'un réseau tel que celui de la figure IV.11, avec deux « coefficients synaptiques » W_{12} et W_{21}, pourra reproduire sur ses sorties $s_1(t)$ et $s_2(t)$ les signaux inconnus de départ $e_1(t)$ et $e_2(t)$.

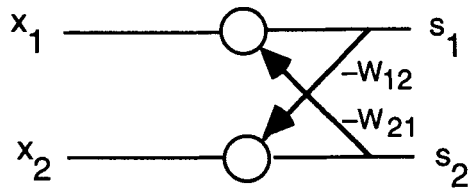

Fig. IV.11. — Réseau de Hérault-Jutten à deux neurones

Les équations liant les sorties aux entrées du réseau sont donc $s(t) = x(t) - \mathrm{W}.s(t)$, où W est une matrice dont la diagonale principale comporte des éléments nuls et la diagonale secondaire les poids W_{12} et W_{21}, ou encore $s(t) = (\mathrm{I} + \mathrm{W})^{-1}.x(t)$, où I est la matrice identité. En combinant les équations du mélange avec celles du réseau, on trouve :

$$s_1(t) = \frac{(a_{11} - \mathrm{W}_{12}\, a_{21})\, e_1(t) + (a_{12} - \mathrm{W}_{12}\, a_{22})\, e_2(t)}{1 - \mathrm{W}_{12}\, \mathrm{W}_{21}}$$

$$s_2(t) = \frac{(a_{21} - \mathrm{W}_{21}\, a_{11})\, e_1(t) + (a_{22} - \mathrm{W}_{21}\, a_{12})\, e_2(t)}{1 - \mathrm{W}_{12}\, \mathrm{W}_{21}}.$$

Deux solutions peuvent donc être trouvées : la première en choisissant les coefficients W_{12} et W_{21} de manière à annuler les termes $(a_{12} - \mathrm{W}_{12}\, a_{22})$ et $(a_{21} - \mathrm{W}_{21}\, a_{11})$, et dans ce cas les sorties $s_1(t)$ et $s_2(t)$ sont respectivement proportionnelles aux signaux initiaux $e_1(t)$ et $e_2(t)$, la deuxième en choisissant ces mêmes coefficients de manière à annuler $(a_{11} - \mathrm{W}_{12}\, a_{21})$ et $(a_{22} - \mathrm{W}_{21}\, a_{12})$, ce qui entraîne une permutation des deux sorties. Toutefois, en raison de la configuration en boucle du réseau de la figure IV.11, une seule de ces deux solutions est stable par rapport aux coefficients de la matrice W. Tout le problème consiste maintenant à trouver les deux coeffi-

cients W_{12} et W_{21} par une méthode ne faisant pas intervenir les paramètres a_{ij} inconnus.

Pour ce faire, et sans entrer dans les détails de la dérivation de l'algorithme, on peut, en supposant les signaux $e_i(t)$ centrés, considérer que la puissance des signaux de sortie $s_1(t)$ et $s_2(t)$ sera d'autant plus grande qu'ils seront corrélés ; en cherchant à minimiser cette puissance, par une méthode de descente de gradient par rapport aux paramètres W_{12} et W_{21}, on trouve des signaux $s_i(t)$ décorrélés, et donc reproductifs des signaux inconnus $e_i(t)$. En faisant plusieurs hypothèses de calcul, notamment le fait qu'on part de conditions initiales proches de la solution et que les coefficients W_{ij} sont petits, on trouve comme règle d'adaptation :

$$\frac{\partial W_{ij}}{\partial t} = \alpha s_i(t)\, s_j(t).$$

Ce résultat montre qu'à la convergence, c'est-à-dire lorsque les coefficients W_{ij} ne varient plus en moyenne, le produit $s_i(t)\, s_j(t)$ est également en moyenne nul, ce qui est bien la condition de non-corrélation des signaux $e_i(t)$ et $e_j(t)$. Néanmoins, *la non-corrélation ne signifie pas l'indépendance*, qui est une condition plus forte ; de plus, la règle ci-dessus impose aux coefficients W_{ij} et W_{ji} des évolutions symétriques, ce qui est une condition trop restrictive pour accéder à la solution attendue. On peut alors montrer que ces deux problèmes se résolvent en modifiant la règle d'apprentissage en :

$$\frac{\partial W_{ij}}{\partial t} = \alpha f(s_i(t))\, g(s_j(t))$$

où $f()$ et $g()$ sont deux fonctions non linéaires, différentes et impaires. De cette façon, la règle est rendue dissymétrique, et on peut montrer que tous les produits des moments d'ordres impairs des signaux $s_i(t)$

et $s_j(t)$ sont nuls, ce qui est bien une approximation acceptable du test de la condition d'indépendance des deux signaux. Dans la pratique, on peut prendre par exemple une fonction cube pour $f()$ et une tangente hyperbolique pour $g()$.

Les applications de cette méthode sont nombreuses et dépassent largement le cadre du problème de la *cocktail party* en présence de sources sonores ; citons par exemple la séparation de signaux transmis sur des lignes avec diaphonie, le redressement d'écriture manuscrite (les deux coordonnées de chaque point du texte étant considérées comme des valeurs instantanées de deux signaux $e_1(t)$ et $e_2(t)$). Notons cependant pour terminer que la méthode n'est applicable qu'à condition d'avoir au moins autant de mélanges différents $x_i(t)$ qu'il y a de sources $e_i(t)$ inconnues ; si on dispose de plus de mélanges, un nombre correspondant de sorties $s_i(t)$ convergeront vers une valeur nulle, tandis que si on dispose de moins de mélanges, les signaux $e_i(t)$ les plus énergétiques se retrouveront sur les sorties $s_i(t)$, entachées cependant d'un bruit correspondant aux autres signaux $s_i(t)$.

Chapitre V

VERS UN COGNITIVISME STATISTIQUE ?

Après ces analyses rigoureuses, qui nous ont conduits à travers des modèles de réseaux de neurones formels, nous souhaitons consacrer quelques pages à une réflexion plus ouverte sur un sens à donner aux recherches en réseaux de neurones artificiels. En particulier, au-delà des applications techniques possibles (assistance médicale, reconnaissance de caractères, simulation de mémoires associatives ou encore prévision de consommation électrique) les réseaux de neurones artificiels permettent de réaliser en action des principes qui relevaient jusqu'à présent de la spéculation théorique : l'apprentissage, la représentation distribuée et l'auto-organisation. Ces propriétés placent naturellement le connexionniste au cœur de la théorie générale des systèmes auto-organisateurs. Elle donne quelques clefs pour répondre à des questions simples mais qui cachent une complexité très élevée. Par exemple : quel est le mécanisme qui donne forme aux étoiles ? Quel est l'origine du temps qu'il fait aujourd'hui ? Qu'est ce qui donne forme à un récif de corail, à un flocon de neige, à la ramure d'un arbre ou à une tornade ? Qu'est est le mécanisme qui conduit à une organisation sociale, à l'échelle d'un pays ou à celle d'une entreprise ? A

l'échelle microscopique, comme celle du cortex, la question s'est également posée : comment les détecteurs d'orientation se sont-ils organisés ? A cela, Hubel et Wiesel ont répondu : par la perception. Plus généralement, c'est le mécanisme d'auto-organisation qui agit dans chacun de ces cas. Selon Henri Atlan[1], « l'organisation vivante apparaît ainsi comme un état intermédiaire entre la stabilité, la persistance immuable du minéral, et d'autre part la fugacité, l'imprévisible, le renouvellement de la fumée. D'un côté, le solide, de l'autre, le gaz ; et au milieu, se trouve le plan fugace du tourbillon liquide ». Deux des modèles que nous avons étudiés sont capables de conduire à une organisation à partir des stimulations reçues : le réseau de Kohonen et celui de Hérault-Jutten.

Le modèle de Kohonen, par ses interactions latérales, est un système en équilibre qui reçoit des perturbations, et s'adapte pour aboutir à un ordonnancement de ses unités révélant une organisation. Malgré un artifice qui permet de stabiliser artificiellement le système[2], on observe une auto-organisation faisant apparaître un ordre global à partir d'interactions locales et de stimulations successives au cours du temps. De même, le modèle de Hérault-Jutten évolue vers des états d'équilibre dynamique, adaptant ses paramètres lorsque les propriétés statistiques des entrées, ou les valeurs du mélange, varient. Dans ces deux cas, l'information extraite des signaux devient pertinente pour un observateur extérieur. On passe donc d'un ensemble de stimulations et de modifications locales, à un ordre global, porteur de sens pour

1. H. Atlan, *Entre le cristal et la fumée*, Seuil, 1979, p. 281.
2. Le contrôle des paramètres de gain et de voisinage assure une convergence en loi vers une distribution stationnaire.

un observateur. Nous sommes proches d'une « machine à fabriquer du sens » (pour reprendre l'expression de Atlan), mais également proches de traitements statistiques qui remplissent une fonction similaire. Cette propriété est clairement illustrée par l'exemple suivant[1].

On dispose d'un tableau de valeurs qui représentent des relevés de variables économiques faites sur 52 pays en 1984[2] : le pourcentage de croissance annuelle (%), le taux de mortalité infantile (‰), le taux d'illettrisme (%), la fréquentation scolaire au second degré (%), le produit intérieur brut par habitant et la croissance annuelle du produit intérieur brut (%). On donne dans le tableau suivant quelques exemples de ces valeurs :

France	0,4	9,1	1,2	86,0	11 326,0	0,5
Kenya	4,0	85,0	52,9	59,3	376,0	3,6
Pérou	2,8	85,0	19,3	72,0	997,0	– 12,0

L'application de la méthode d'Analyse en Composantes Principales[3], après normalisation des données et calcul de la matrice de covariances, permet de tracer la carte factorielle des variables dans le plan factoriel engendré ici par les deux premiers vecteurs propres. Cette méthode consiste à projeter orthogonalement les points origines de dimension 6 (car chaque pays est représenté par 6 variables) en dimension 2 (car le plan factoriel est engendré par les deux vecteurs propres). Le résultat est montré sur la figure V.1.

1. Cet exemple est issu de recherches présentées dans F. Blayo et P. Demartines, Data Analysis : how to compare Kohonen neural network to other techniques?, *Artificial Neural Networks,* Lecture Notes in Computer Science, 540, A. Prieto Editor, Springer-Verlag, 1991.

2. Extrait de « L'état du monde en 1984 », Éditions La Découverte.

3. Le lecteur trouvera une excellente introduction à l'Analyse en composantes principales dans J.-M. Bouroche et G. Saporta, *L'analyse des données,* PUF, « Que sais-je? », n° 1854, 1980.

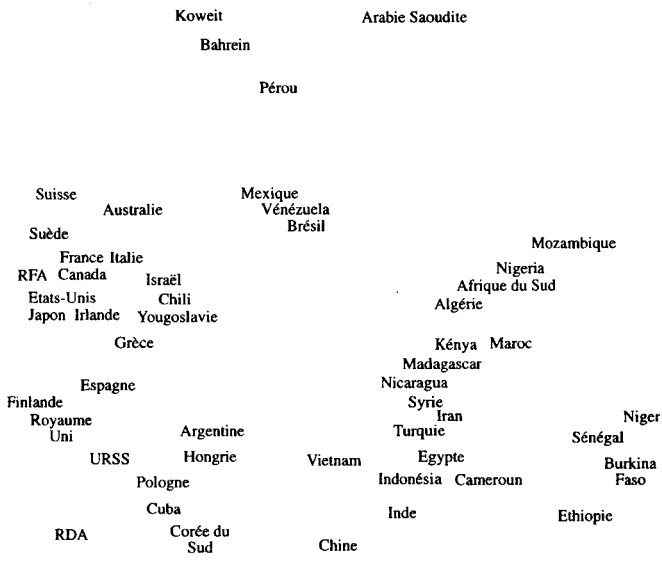

Fig. V.1. — Résultat de l'ACP

Le même problème, abordé avec un réseau de Kohonen, requiert le choix de plusieurs éléments du modèle. Le premier est la structure et la taille du réseau : elles sont arbitrairement choisies carrées pour la structure et la taille fixée à 8 par 8 neurones. C'est l'expérience de l'utilisateur du réseau de neurones artificiels qui est reflétée par ce choix. Le nombre d'entrées est évidemment fixé à 6, car chaque pays est représenté par un vecteur de 6 composantes. Enfin, la règle d'adaptation des poids du réseau est la règle de Kohonen, avec une mesure de distance euclidienne.

La partie d'apprentissage consiste à présenter séquentiellement en entrée du réseau chaque ligne du tableau, et à appliquer la règle de modification des

poids. Ceci est répété jusqu'à la convergence du réseau, c'est-à-dire lorsque les paramètres de gain et de voisinage sont nuls (l'apprentissage s'arrête). Notons que chaque ligne du tableau sera présentée plusieurs fois. On passe alors en phase d'identification, qui consiste à établir une correspondance entre chaque ligne du tableau (un pays), et le neurone qui sera le plus actif pour celle-ci. On décide de représenter le pays à l'emplacement de ce neurone. Cette étape sera

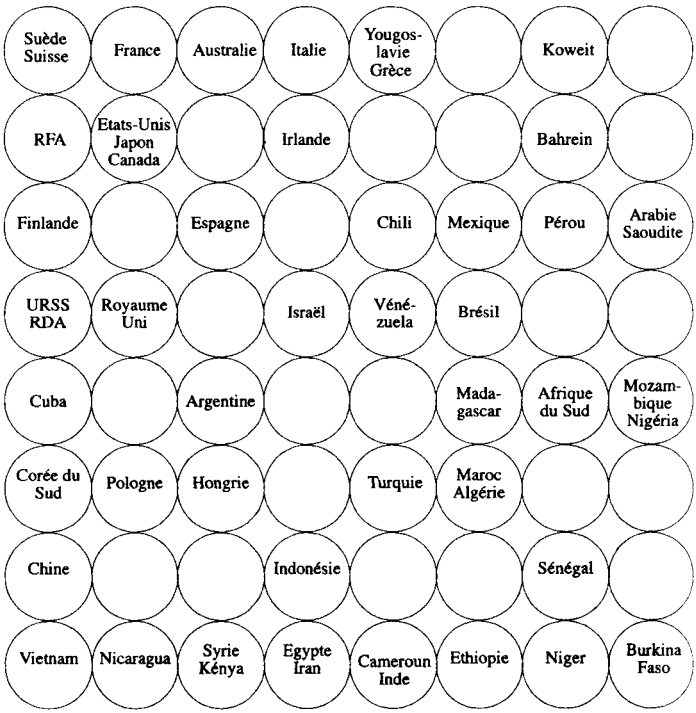

Fig. V.2. — Résultat du modèle de Kohonen avec 8×8 neurones; chaque cercle représente un neurone

répétée exactement 52 fois, puisqu'il y a 52 pays. On obtient à la fin une carte (voir fig. V.2) qui reflète la correspondance établie entre l'espace de dimension 6 et la carte de dimension 2.

Malgré une méthode de calcul très différente, on constate évidemment une certaine similarité dans les représentations, entre ACP et Kohonen, mais le modèle de Kohonen conduit à une répartition plus uniforme des pays sur la carte établie. Ainsi, les distances entre pays ou groupes qui apparaissent dans l'ACP disparaissent avec le modèle de Kohonen. Enfin, en analysant en détail les transformations effectuées par ces deux modèles, il ressort que l'ACP est une méthode linéaire, alors que le modèle de Kohonen réalise une transformation non linéaire. Ceci peut être intéressant dans certains cas, la représentation linéaire n'étant pas toujours adaptée à un nuage quelconque de points dans l'espace.

Notons enfin que l'interprétation des résultats obtenus est du ressort de l'expert du domaine traité. Plus précisément, les facteurs, ou les axes, des représentations peuvent avoir une signification qui donne toute sa richesse à la représentation. Mais, elle est souvent difficile à dégager et requiert une parfaite connaissance de la technique d'analyse, tout comme du domaine sur lequel elle porte.

Pour le modèle de Hérault-Jutten (appelé Analyse en Composantes Indépendantes dans le contexte de la statistique), le traitement d'information effectué se rapproche également d'un traitement statistique : alors que l'Analyse en Composantes Principales cherche à maximiser un critère de non-corrélation, l'ACI cherche à maximiser un critère d'indépendance. On retrouve dans ce modèle les principes fondamentaux de l'auto-organisation : construire un système dont l'état d'équilibre dépend des seules stimulations, et qui fait émerger un ordre global à partir d'interactions locales.

Il est donc naturel de considérer que les méthodes statistiques et les réseaux de neurones artificiels ont des points communs. Delacour[1] fait d'ailleurs remarquer que « certaines formes de l'activité cognitive sont équivalentes à une réduction de la dimension des données » (nous venons de le voir dans l'exemple précédent), « à une *compression* de celles-ci, comme le font de manière explicite des techniques statistiques d'analyse multivariée : elles donnent de façon en grande partie intuitive, imagée... une représentation simplifiée de données complexes, multidimensionnelles, en conservant le maximum d'information ». De nombreuses recherches font également état d'une ressemblance entre le traitement d'information cérébral et les traitements statistiques, en particulier dans le domaine de la vision ou de la perception des odeurs.

Les recherches en réseaux de neurones artificiels se rapprochent donc de la statistique, en y apportant des éléments originaux : la non-linéarité, la localité du traitement et la possibilité d'implanter facilement les algorithmes neuronaux sur des machines parallèles. Dans ce contexte, le système nerveux pourrait être envisagé comme un dispositif de traitement d'information alliant statistique et optimisation. Nous l'avons vu, les simulations de mémoire associative font appel à des méthodes d'optimisation, et en particulier à la technique du gradient stochastique. Nous voyons également que la représentation fait appel à l'auto-organisation, comme la décision repose sur la théorie bayésienne, toutes deux proches de méthodes statistiques. Sans considérer le mécanisme de l'action, que nous n'avons pas abordé dans cet ouvrage, nous avons déjà trois techniques mathématiques qui nous renseignent

1. J. Delacour, *Le cerveau et l'esprit*, PUF, « Que sais-je? », n° 2938, 1995.

sur des types de traitements effectués par le système nerveux. Ce ne sont probablement pas les seuls, mais ils remplissent des tâches capitales. Il est cependant frappant de constater que le cerveau réalise des traitements qui permettent de discerner des régularités, des formes, dans des espaces de dimensions élevées grâce à des méthodes de réduction de dimension, qui sont bâties par des processus d'auto-organisation. En d'autres termes, que ce soit pour la vision, l'olfaction, ou l'audition, les formes et les modèles sont indissociables, car ils s'établissent mutuellement. Comme le fait remarquer Atlan, l'élément le plus important dans ces phénomènes d'auto-organisation, c'est l'auto-création du sens, c'est-à-dire la création de significations nouvelles de l'information transmise d'une partie à l'autre ou d'un niveau d'organisation à une autre. C'est ce que nous constatons dans l'exemple que nous venons de traiter.

En dehors des notions de perception, certains modèles de réseaux de neurones peuvent être utilisés dans des domaines généralement couverts par des techniques de calcul numérique. En particulier, les Perceptrons multicouches, grâce à leur propriété d'approximation universelle, sont utilisés dans de très nombreuses applications d'approximation de fonctions, en petites ou grandes dimensions d'entrées ou de sorties. Bien entendu, il existe de nombreuses autres méthodes pour approximer des fonctions, comme les splines ; néanmoins, les Perceptrons multicouches apportent une facilité d'utilisation que ne possèdent généralement pas les autres méthodes.

Dans l'éventail des applications réellement fonctionnelles des réseaux de neurones, il serait critiquable d'omettre celles qui utilisent des Perceptrons multicouches. Elles sont en effet très largement majoritaires à l'heure actuelle, non pas parce qu'il s'agit du seul

modèle intéressant, mais bien parce qu'il bénéficie d'une antériorité qui lui a fait obtenir un nombre d' « adeptes », surtout aux Etats-Unis. Parmi les applications, il faut néanmoins faire la part des choses, entre celles qui ne trouvent pas de réels avantages par rapport à l'application de méthodes plus classiques, et celles pour lesquelles l'apport des réseaux de neurones artificiels est indéniable. Comme nous allons le voir dans l'exemple ci-dessous, la bonne compréhension du problème est également un prérequis *indispensable* à l'utilisation d'un réseau de neurones dans un réel cadre industriel.

L'exemple suivant[1] est celui d'un Perceptron multi-couches exploité dans une application industrielle de contrôle de procédé pour la production de pâte à papier par réaction chimique, dans laquelle du bois est dissous dans une solution dite « de cuisson ». La pâte est produite dans des récipients hermétiques dont le volume est approximativement de 250 m^3. Chaque récipient est un réacteur chimique qui est rempli avec un mélange de bois et de solution de cuisson composée de magnésium bisulfate et de dioxyde de soufre. L'ensemble est porté à une température de 132 °C. La lignine, substance organique qui imprègne les cellules, fibres et vaisseaux du bois, est ainsi dissoute par réaction chimique. Ce procédé conduit à la création de pâte à papier, qui est une matière fibreuse entrant dans la fabrication industrielle du papier et des fibres textiles. Il est donc important de contrôler la qualité de production de la pâte, car elle influe directement sur la qualité des produits dérivés.

1. Cette description est extraite de la publication Neural Networks for Industrial Process Control : Applications in Pulp Production, D. Obradovic, G. Deco, H. Furumoto, C. Fricke, *Proceedings of the Sixth International Conference « Neural networks and their industrial and cognitive applications »*, Nîmes, France, octobre 1993.

On sait par expérience que la qualité de la pâte à papier est liée à la concentration en permanganate mesurable dans le bain de cuisson. Or, cette concentration dépend elle-même de 6 classes de paramètres qui sont :

— la qualité du bois (humidité, structure, etc.) ;
— l'évolution de la pression durant le processus de production ;
— l'évolution de la température ;
— la concentration des produits chimiques ;
— le temps de chauffe ;
— le temps de réaction (processus de dissolution de la lignine).

Parmi eux, la qualité du bois et les concentrations chimiques sont connues avant le démarrage du procédé. Le temps de chauffe, la pression désirée et les profils d'évolution de température sont, quant à eux, fixés par avance. Il est également possible de connaître leur dérive pendant le procédé, ce qui est particulièrement important pour la température qui doit être maintenue en moyenne autour de 132 °C. Cependant le paramètre de qualité, c'est-à-dire la concentration en permanganate, ne peut être connu qu'en fin de procédé. On ne peut donc connaître la qualité de la pâte qu'*a posteriori*. Néanmoins, on sait que vers la fin de la réaction, tous les paramètres se stabilisent vers des valeurs moyennes, et qu'à ce moment la concentration en permanganate décroît rapidement en fonction du temps de cuisson, mais on ne connaît pas exactement cette relation. Ainsi, établir cette relation, c'est pouvoir fixer un temps de cuisson précis, suivant une concentration choisie *a priori*. Ce problème industriel a été posé par les producteurs de Cellulose do Caima au Portugal.

Jusqu'à présent, les meilleurs modèles capables de

réaliser la transformation des variables disponibles et de la concentration souhaitée en un temps de réaction reposaient sur un modèle analytique dérivé d'équations de cinématique chimique et très insuffisant pour assurer une production de qualité. La société Siemens a proposé de mettre en place un contrôle par réseau de neurones, qui s'est avéré répondre parfaitement aux besoins du client, en obtenant un accroissement de 30 % sur la qualité de la production de pâte à papier.

Les données disponibles sont constituées de 209 mesures, divisées en 166 pour l'apprentissage des paramètres du réseau et 43 pour le test en généralisation. Elles portent sur deux mois de mesures (décembre 1992 et janvier 1993).

Le réseau est un Perceptron multicouche, avec 11 entrées correspondant aux variables physiques des 6 groupes présentés auparavant et de la concentration en permanganate souhaitée, 10 unités dans l'unique couche interne et 1 sortie correspondant au temps de cuisson estimé.

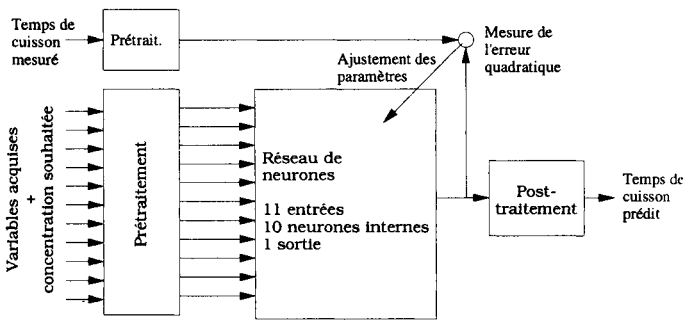

Fig. V.3. Principe du système d'apprentissage neuronal. Pour l'exploitation, le temps de cuisson mesuré et l'ajustement des paramètres sont retirés

Ce réseau a subi une modification des poids par apprentissage suivant deux méthodes, afin de comparer les résultats obtenus. La première méthode est un algorithme d'optimisation quasi newtonien, et la seconde méthode est un algorithme stochastique, incluant une minimisation de la structure du réseau par élagage des paramètres (algorithme proche de l' « Optimal Brain Damage »). Initialement, le réseau contenait 131 paramètres (10×11 connexions entre les 11 entrées et les 10 neurones de la couche interne, 10 paramètres de seuil pour les 10 neurones, 10 poids pour les connexions entre la couche interne et le neurone de sortie, et un seuil pour ce dernier). La méthode d'élagage a permis de ne retenir que 50 paramètres significatifs.

Les données disponibles ont été pré-traitées afin de les faire varier dans un intervalle [0, 1]. D'autre part, les auteurs de ce travail font remarquer que, malgré des données disponibles de juillet 1992 jusqu'à janvier 1993, ils n'ont retenu que décembre 1992 et janvier 1993 à cause de la non-stationnarité des mesures antérieures. Cette remarque est capitale pour une bonne utilisation du modèle de Perceptron multicouches. On sait aujourd'hui qu'appliquer sans précautions un modèle neuronal à des données brutes conduit sûrement à des résultats non interprétables. Il faut étudier en détails le pré-traitement, et également le post-traitement de données, afin de faire fonctionner le réseau dans des conditions qui sont correctes d'un point de vue théorique, et qui fournissent des résultats pratiques interprétables.

Après apprentissage des paramètres du réseau sur un ordinateur conventionnel (quelques centaines d'itérations pour la méthode quasi newtonienne et plusieurs milliers pour la méthode stochastique), le réseau a été implanté sur le site de production. Notons encore

que les algorithmes d'optimisation utilisés ici servent à minimiser une erreur quadratique (voir à ce sujet le chapitre 3). Or, minimiser l'erreur peut conduire à un sur-apprentissage sur la base d'exemples, et à une mauvaise généralisation sur la base de tests. Les auteurs ont donc choisi de tester le réseau à chaque nouvelle itération d'apprentissage, et d'arrêter l'optimisation des paramètres lorsque les performances en généralisation se dégradent. L'accroissement de l'erreur quadratique sur la base de test est donc le critère d'arrêt de l'apprentissage.

Pour des questions de sécurité de production, et d'évaluation du risque lié à l'implantation d'une nouvelle méthode de contrôle de procédé, le système neuronal a été implanté en parallèle avec le système de contrôle opérationnel. Cela a permis de comparer les résultats produits par les deux systèmes sur un même procédé. Pour chacun d'eux, le temps de réaction estimé a été mis en rapport avec l'erreur mesurée entre la concentration en permanganate attendue pour ce temps de réaction et la concentration obtenue effectivement. La campagne de mesure s'est étendue sur une semaine à raison de 3 réactions par jour dans 4 réacteurs. Les résultats obtenus montrent que le modèle prédictif par réseau de neurones était plus précis que le modèle analytique dans 30 % des cas pratiques.

Ce réseau est opérationnel et implanté dans le système d'automatique « Teleperm M » de Siemens AG.

Cette application montre que des réseaux de neurones peuvent se substituer à des modèles de contrôle existant, en apportant un accroissement de performances sensible pour un risque réduit. Ceci vient essentiellement de la capacité d'approximation universelle, telle que nous l'avons décrite dans le chapitre sur les mémoires associatives, et qui apporte une possibilité de construction automatique d'un modèle non

linéaire du procédé observé. La facilité de mise en œuvre d'un réseau de neurones ne doit pas être sous-estimée. Dans cet exemple, personne ne prétend avoir trouvé la solution idéale unique à un problème complexe d'approximation tel que celui décrit ci-dessus ; d'un point de vue industriel cependant, le meilleur compromis est celui entre les performances (et donc le gain) atteintes par la méthode utilisée, et la complexité (et donc le coût) de sa mise en œuvre.

L'exemple permet de comprendre également comment on parvient à mettre en place un système neuronal au sein d'une chaîne de production, et quelles sont les étapes suivies :

— analyse du problème ;
— sélection de variables et prétraitement ;
— choix du modèle de réseau et apprentissage. Cette étape peut se faire hors du site d'implantation car il ne requiert qu'un enregistrement des données acquises pendant une campagne de mesures ;
— éventuellement optimisation du modèle par des méthodes avancées ;
— implantation dans le système informatique existant, ou adjonction d'un contrôleur *ad hoc* en parallèle avec l'existant ;
— test et évaluation de performances, sans interférer sur la production actuelle. Si la validation est favorable au réseau, on peut éventuellement envisager une période de test plus longue pour consolider les résultats, et parvenir à un remplacement du système de contrôle actuel par le système neuronal.

De tels exemples donnent également toute sa signification à l'appellation « réseaux de neurones artificiels ». Celle-ci paraît en effet parfois, à première vue, « usurpée ». Quel rapport y a-t-il entre les processus chimiques et électriques entre neurones et synapses, et

un algorithme de rétro-propagation de gradient ou de calcul de matrice pseudo-inverse? Le rapport au niveau des algorithmes ou des équations est bien entendu inexistant. Par contre, c'est dans l'utilité des méthodes, et dans les principes qui s'en dégagent, tels l'auto-organisation, l'apprentissage, l'évolution et le parallélisme qu'il faut chercher le lien entre réseaux de neurones biologiques et artificiels.

Il ne faut pas chercher non plus à fixer de façon exacte les limites du domaine des réseaux de neurones artificiels. Plus qu'un nouveau domaine de la science, ils forment un ensemble de techniques dont l'origine (biologique), le développement (algorithmique), la validation (mathématique) et les multiples utilisations contribuent largement à leur « pluridisciplinarité ». Certaines techniques de calcul adaptatif et à apprentissage étaient néanmoins connues bien avant que l'on ne parle de réseaux de neurones artificiels ; de même, d'autres, développées plus récemment, ne sont vraisemblablement classées sous cette appellation qu'à cause d'un effet de mode, alors que rien n'empêcherait de les considérer comme des méthodes statistiques de traitement d'information. En vue d'un développement normal du domaine, sans en vouloir ni exagérer ni sous-estimer les possibilités, c'est donc dans un rapprochement entre réseaux de neurones et statistiques, et surtout dans l'absence d'une frontière qui ne peut être qu'artificielle entre les deux domaines, que l'on peut établir une démarche constructive qui fera sans aucun doute encore évoluer considérablement cet ensemble de techniques dans les prochaines années.

Chapitre VI

CONCLUSION

Il n'a pas été possible dans le cadre de cet ouvrage de donner une vue exhaustive de l'ensemble des techniques regroupées sous le nom de « réseaux de neurones artificiels » ; nous avons néanmoins voulu en donner un aperçu en partant des notions de base pour aboutir à des modèles plus élaborés, actuellement utilisés dans les domaines de traitement de l'information et du signal.

Il existe actuellement plusieurs tendances pour le développement de ces nouveaux modèles. Certains travaux de recherche portent sur l'élaboration d'algorithmes comme ceux décrits dans cet ouvrage, adaptés au traitement non linéaire de données, que ce soit dans les domaines de l'approximation ou de la classification, et dont les liens avec la biologie se limitent aux principes directeurs : calculs collectifs, distribution de l'information, apprentissage... D'autres, par contre, accordent une plus grande importance à une modélisation plus proche des processus biologiques, en rapprochant les structures et les opérations des réseaux artificiels de celles rencontrées dans les systèmes nerveux. Si la collaboration entre les différentes tendances n'est pas toujours aisée, principalement en raison de différences de buts et de langages utilisés, un rapprochement ne peut qu'être bénéfique à l'essor du domaine.

Après tout, l'explosion de l'intérêt envers les réseaux de neurones artificiels n'est-elle pas due, au moins en partie, au célèbre article de J. J. Hopfield en 1982, article dont le principal mérite était de traiter un problème d'ingénieur avec un langage de physicien, contribuant ainsi au caractère pluridisciplinaire de ce domaine de la science en pleine expansion ?

Les modèles utilisés pour le traitement de données et de signal ne doivent néanmoins pas être considérés comme la solution universelle à tous les problèmes difficiles ou impossibles à résoudre par des méthodes plus classiques. Trop souvent, on a tendance à confondre les propriétés asymptotiques des réseaux, comme la propriété d'approximation universelle de certains d'entre eux, avec leur capacité à traiter efficacement des problèmes avec des contraintes au niveau de la taille des réseaux, de la complexité des calculs...

Enfin, il faut bien souligner qu'un réseau de neurones artificiels est une méthode non linéaire de traitement d'information. A ce titre, certaines méthodes neuronales constituent un apport réellement nouveau par rapport à des méthodes classiques ; la comparaison des cartes auto-organisatrices de Kohonen et de l'analyse en composantes principales en est une flagrante illustration. D'autres ne sont purement et simplement que des réécritures commodes et parfois simplifiées de méthodes connues. Les réseaux de neurones et les statistiques se complètent donc, mais ont leurs particularités qui les rendent différentes : par exemple certaines méthodes neuronales sont bien adaptées à des flux de données qui ne peuvent pas être accumulées avant d'être traitées.

Les techniques neuronales, pour atteindre leur maturité et confirmer l'intérêt qu'elles suscitent auprès de l'industrie, devront obligatoirement engager ce rapprochement.

BIBLIOGRAPHIE

Nous indiquons ici quelques ouvrages de référence qui peuvent aider le lecteur dans un élargissement des notions présentées.

J. Anderson, E. Rosenfeld, *Neurocomputing, Foundations of Research*, Cambridge, The MIT Press, Massachusetts, Third Printing, 1988. Ouvrage historique, recueil des publications scientifiques séminales.

Les mécanismes de la vision, Bibliothèque pour la Science, Belin, 1990. Document interdisciplinaire, très fouillé, sur la vision biologique et les modèles formels qui en sont issus.

J.-P. Changeux, A. Connes, *Matière à pensée,* coll. « O. Jacob », 1989. Ouvrage de réflexion sur les liens entre modèles formels et cerveau.

G. Chapouthier, *La biologie de la mémoire,* PUF, coll. « Que sais-je? », 1994.

J. Delacour, *Biologie de la conscience,* PUF, coll. « Que sais-je? », 1994.

R. Francès, *La perception,* PUF, coll. « Que sais-je? », 8e éd., 1992.

J. Hérault, C. Jutten, *Réseaux neuronaux et traitement du signal,* Hermès, 1994.

La recherche en neurobiologie, Ouvrage collectif, Editions du Seuil, « Points Sciences », 1989.

TABLE DES MATIÈRES

Imprimé en France
Imprimerie des Presses Universitaires de France
73, avenue Ronsard, 41100 Vendôme
Janvier 1996 — N° 42 092

Que sais-je ?

COLLECTION ENCYCLOPÉDIQUE
fondée par Paul Angoulvent

Derniers titres parus

3011 **Le soleil et la peau**
L. Rossant

3012 **Les institutions du tourisme**
J.-L. Michaud

3013 **L'affichage**
M. Fitoussi

3014 **Les grands arrêts de droit communautaire**
J.-C. Masclet

3015 **L'information scientifique et technique**
F. Jacobiak

3016 **La sociologie du risque**
D. Le Breton

3017 **L'insécurité**
J.-L. Mathieu

3018 **Orphée et l'orphéisme**
R. Sorel

3019 **Le Midrach**
D. Banon

3020 **La peinture italienne du maniérisme au néoclassicisme**
S. Costa

3021 **Napoléon III**
T. Lentz

3022 **Jung**
C. Gaillard

3023 **L'Union de l'Europe occidentale**
P. Van Ackere

3024 **Histoire de la recherche sur le SIDA**
B. Seytre

3025 **Les groupes financiers français**
H. Bonin

3026 **Les politiques nucléaires**
H. Pac

3027 **La délégation de service public**
J.-F. Auby

3028 **Géologie des planètes**
R. Dars

3029 **Néron**
G. Achard

3030 **Staline**
J.-J. Marie

3031 **La science du judaïsme**
M.-R. Hayoun

3032 **L'environnement spatial**
J.-C. Boudenot

3033 **La politique de la concurrence au Royaume-Uni**
F. Souty

3034 **Le processus de paix au Moyen-Orient**
M. Konopnicki et S. Petermann

3035 **Les Nations Unies. Textes fondamentaux**
A. Pellet

3036 **L'environnement graphique windows**
P. Fabre

3037 **La réalité virtuelle**
B. Jolivalt

3038 **Histoire de l'Union soviétique de Khrouchtchev à Gorbatchev**
N. Werth

3039 **La communication politique locale**
M. Souchard et S. Wahnich

3040 **Victimes et victimologie**
G. Filizzola et G. Lopez

9 782130 473558