UNIVERSIDAD AUTONOMA DE BAJA CALIFORNIA

FACULTAD DE CIENCIAS

APRENDIZAJE Y RECUPERACION DE INFORMACION EN UNA RED NEURONAL TIPO HOPFIELD

T E S I S

Que para obtener el título de:

F I S I C O

Presenta:

Joel Eduardo Rodríguez Ramírez

Ensenada, Baja California **Mayo de 1993**

UNIVERSIDAD AUTÓNOMA DE BAJA CALIFORNIA
Facultad de Ciencias

APRENDIZAJE Y RECUPERACIÓN DE INFORMACIÓN EN UNA RED NEURONAL TIPO HOPFIELD

Tesis profesional que como requisito
parcial para obtener el título de

FÍSICO

presenta:

JOEL EDUARDO RODRÍGUEZ RAMÍREZ

Aprobado por:

Dra. Laura Viana Castrillón
Presidente del Jurado.

Dr. Rafael Kelly
Primer Vocal.

M. en C. Armando Reyes Serrato
Secretario.

..... A Mi Padre Antonio y Madre Leonila.
..... A Mi Abuela Esperanza y Mi Hermana Yamina.
..... A Mis Hermanos Yadir e Ivan.

AGRADECIMIENTOS

Quisiera agradecer a las personas que hicieron posible que este trabajo en particular se llevara a cabo. Por principio quiero agradecer a mi padre Antonio Ríos, por haber proporcionado los medios necesarios incluyedo tiempo de computo, información bibliográfica, así como sustento económico, para que yo pudiera terminar ésta tesis, apoyo sin el cual ésta no hubiera sido terminada. A continuación quisiera agradecer a mi maestra la doctora Laura Viana Castrillón, por haberme introducido a este bello campo de la ciencia, las redes neuronales, así como por haberme orientado en la elaboración de ésta tesis, es justo decir que gracias a su orientación, así como a valiosas discusiones, es que ésta pudo ser concluída.

Gran parte del trabajo de esta tesis fue elaborado gracias a las facilidades otorgadas por el Instituto de Física de la U.N.A.M. Laboratorio de Ensenada, por lo que quisiera agradecer al Dr. Leonel Cota, quien ademas de haber sido parte importante de mi formación academica es el director del mismo. Quisiera agradecer al Instituto de Astronomía de la U.N.A.M. en Ensenada, especialmente al Dr. Luis Carrasco Bazúa, quien durante su gestion como jefe al frente del Observatorio Astronómico en San Pedro Martir, proporcionó los recursos para que parte de esta tesis pudiera ser completada. Asimismo, quisiera agradecer al personal del Observatorio Astronómico Nacional con sede en la Sierra de San Pedro Martir Baja California, por haber proporcionado la calma necesaria, durante la elaboración de una gran parte de esta tesis.

Quisiera agradecer a Claudia Talavera, Armando Reyes, Antonio Diaz, Hector Aceves, Raul Michel, Hector Velazquez, Sergio Carnalla, Alfonso Paredes, Citlali Martínez, Jorge Villavicencio, José Guadalupe Lepe, Alvaro Lepe, Silvia Woods y Gabriella Bianchini, por las valiosas sugerencias a este trabajo.

RESUMEN

Se hace un revisión bibliográfica del área conocida como Redes Neuronales, específicamente, del problema conocido como Memoria Asociativa (Hertz, 1991), se plantéa éste como un sistema dinámico de ecuaciones acopladas (Red Neuronal). Se efectúa una analogía entre el sistema de ecuaciones dinámicas y el modelo de Hopfield (Hopfield, 1982), asimismo, se efectúan simulaciones numéricas, utilizando el algoritmo de *Codificación Múltiple de Espines* desarrollado por Penna y Oliveira (Penna, 1989), se presentan algunos resultados computacionales generados por las simulaciones y se concluye con una discución de estos.

ABSTRACT

A bibliographic survey of the area known as Neural Networks is made, specifically of the problem known as Asociative Memory (Hertz, 1991), the model is proposed as a dinamic system of coupled equations (Neural Network). An Analogy is made between the dinamic equations and the Hopfield model (Hopfield, 1982). Some numerical simulations using the *Multy Spin Coding Algorithm* developed by Penna y Oliveira (Penna, 1989) are made, and some results due to the simulations are presented, as well as a discution of them.

INDICE

O. Introducción

Laboratorios de vanguardia alrededor del mundo efectúan en la actualidad investigación en lo que se ha dado a conocer como *computadoras de la sexta generación*, las cuales están inspiradas en la arquitectura del cerebro. Los objetivos de las computadoras de la sexta generación, son verdaderamente computación inteligente, medición, control, robótica y sistemas de información. Aunque no se puede esperar construir una computadora que emule el cerebro, sí se pueden emular muchas funciones lógicas e intelectuales de sistemas neuronales. Esta investigación es interdisciplinaria, pues involucra a las más diversas áreas de investigación tales como las ciencias computacionales, la ingeniería, la neurofisiología, la linguística, lógica, la psicología, la física y las matemáticas. Esta investigación, ha generado nuevas arquitecturas de computadoras, algoritmos, componentes, sistemas y aplicaciones, haciendo que la disciplina avance y abra nuevas perspectivas de empleos y negocios. Como fruto de la investigación en esta área una gran cantidad de periféricos, sistemas, aplicaciones y servicios están ahora disponibles al público consumidor. En el futuro, si se requiere un mayor avance de esta nueva área de investigación, será necesario una mayor interacción entre las ciencias del cerebro y las ciencias computacionales. De esta forma se espera un mejor entendimiento de las habilidades cognitivas, perceptivas y de memoria, relacionadas con el trabajo de aprender. Además, un mayor avance será alcanzado a través de una mayor cantidad de investigación que contemple arquitecturas diversas, dentro de este nuevo paradigma computacional. En la actualidad, la aplicación de lo que podría considerarse como un nuevo paradigma computacional es cada día mayor, debido a que con éste es posible resolver problemas que son practicamente imposibles de resolver con algoritmos computacionales convencionales.

La historia de la adaptación de estos nuevos paradigmas computacionales se puede remontar a tratar de simular de alguna forma el funcionamiento del tejido nervioso en el cerebro, haciendo una analogía entre las neuronas biológicas y sus correspondientes representaciones teóricas. Sin embargo, debe notarse que en lo sucesivo no se tratará de describir el funcionamiento del cerebro o de modelar éste y mucho menos de extrapolar resultados. Debe señalarse, sin embargo, que simular las funciones de ciertas regiones del cerebro o de ciertas funciones efectuadas por éste, representa una área interdisciplinaria de intensa actividad científica. En este sentido, el área comúnmente conocida como *redes neuronales o computación neuronal*, es un área científica que ha sido inspirada del conocimiento de las neurociencias, por lo que a la largo de la tesis se hará referencia continuamente a información con bases biológicas. Esta información es requerida ya sea por razones históricas,

como marco de referencia, como sugerencia para interpretar el comportamiento de alguna clase de tejido nervioso, etcétera.

El objetivo de esta tesis será el de hacer una revisión bibliográfica y explorar algunas de las propiedades físicas que emergen del comportamiento cooperativo de un conjunto muy grande de unidades simples, cada una de las cuales es capaz de efectuar cálculos numéricos sencillos, desde la perspectiva de la mecánica estadística. Las unidades, serán organizadas bajo una arquitectura, que será la que distinga al sistema.

El propósito del primer capítulo, será el de proporcionar las definiciones indispensables, de forma tal que éste sirva como normador del lenguaje que será utilizado durante toda la tesis. En el segundo capítulo se presentará el modelo de *memoria asociativa*. Este modelo es comúnmente conocido como el modelo de Hopfield y es utilizado para mostrar la forma en que una red neuronal artificial puede memorizar y recuperar imágenes, de igual forma se calcularán las propiedades que caracterizan el modelo (i.e. capacidad de la red, estabilidad de las imágenes almacenadas, etcétera). En el tercer capítulo se presentará la analogía matemática entre las redes neuronales y el modelo para un tipo de material magnético desordenado, conocido como *vidrio de espín*. La consideración de este nuevo esquema, permitirá hacer uso de las herramientas de mecánica estadística, originalmente desarrolladas para entender los sistemas magnéticos, y que ahora pueden ser utilizadas, para entender el comportamiento cooperativo de redes neuronales, y así establecer un marco teórico para el estudio sistemático de estos modelos. En el cuarto capítulo, se presentará una simulación computacional llevada a cabo en una computadora digital, con el objetivo de ejemplificar el problema de memoria asociativa. Se hará una explicación del algoritmo utilizado. El quinto capítulo, se presentarán los resultados obtenidos, de la simulación computacional y en el sexto capítulo se presentarán las conclusiones de la tesis. Finalmente, se incluye un apéndice (A), el cual contiene el código del programa utilizado para llevar a cabo una de las simulaciones computacionales.

La tesis concluye presentando la serie de referencias bibliográficas, utilizadas durante toda la tesis, así como lectura recomendada.

Ensenada, Baja California, México, a 22 Marzo de 1993.

Joel Eduardo Rodríguez Ramírez.

I. Redes Neuronales

Este capítulo se iniciará presentando algunos *elementos básicos acerca del cerebro* desde una perspectiva biológica. Posteriormente se mostrarán algunos modelos teóricos muy simplificados, de arreglos de neuronas los cuales han sido inspirados del estudio del cerebro, por parte de las neurociencias; estos modelos serán referidos, como modelos de *redes neuronales artificiales*. El capítulo continuará presentando el concepto de computación tipo Von Newmann, que es el esquema computacional que ha sido mayormante explotado, de forma tal que su consideración permita establecer un marco de referencia que posteriormente pueda servir para poner de manifiesto la estructura de lo que se conoce como computación neuronal.

1.1 Elementos básicos acerca del cerebro. A fines del siglo XIX, el Histólogo Santiago Ramón y Cajal encontró los fundamentos para el estudio de sistemas nerviosos (Ramón y Cajal 1913). Este científico demostró que los sistemas nerviosos estaban conformados por densas redes de células discretas e interconectadas a las que llamó **neuronas**; entre otras cosas, Ramón y Cajal pudo medir el tiempo que tardan dos neuronas para comunicarse (por medio de impulsos eléctricos) y encontró que éste, era del orden de varios milisegundos. Desde entonces a la fecha se ha avanzado mucho en el entendimiento del funcionamiento del cerebro, asimismo, se han efectuando entre otras cosas, mediciones más precisas de las propiedades del tejido nervioso (Neher 1992, Alkon 1989).

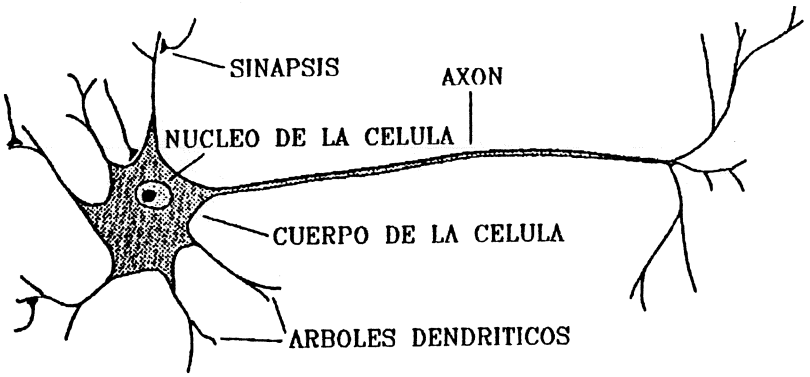


Fig. 1.1 DIBUJO DE UNA NEURONA TÍPICA

El cerebro, está compuesto aproximadamente por 10^{10} células nerviosas o neuronas (Hertz 1991, Amit 1989). Una neurona típica, está conectada con aproximadamente 10^4 otras neuronas en lugares de la célula llamados **sinapsis**. Conectadas al cuerpo de la

célula se encuentran ramificaciones en forma de árbol llamadas **dendritas**; por otro lado, extendiéndose del cuerpo de la célula, existe una fibra larga llamada **axón**. Eventualmente el axón se ramifica terminando en las junturas sinápticas de otras neuronas. Las sinapsis pueden ser encontradas en las dendritas, en el axón o en el mismo cuerpo de la célula (véase figura 1.1).

Cuando una señal (impulso eléctrico) es transmitida de una célula a otra, en la sinapsis que conecta a las dos neuronas ocurre un proceso químico complicado (véase por ejemplo (Amit 1989), en el cual la célula que envía la señal libera substancias químicas -neurotransmisores- en pequeños paquetes.

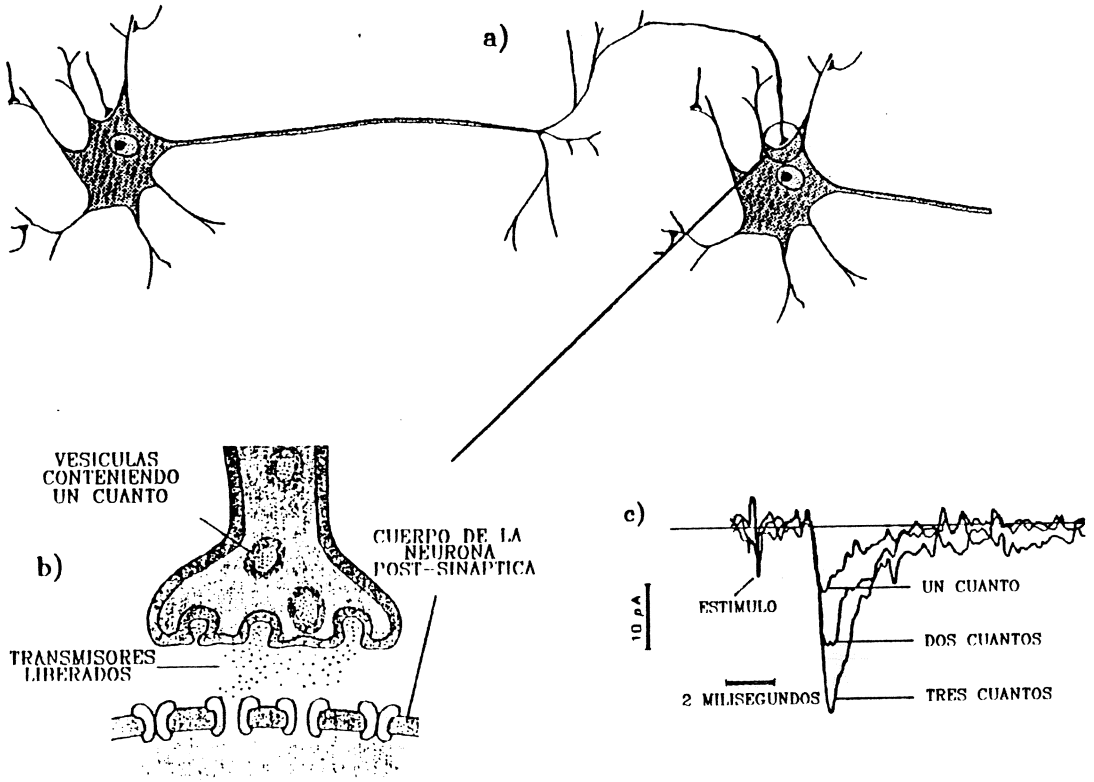


Fig. 1.2 (*a*) ARREGLO DE DOS NEURONAS CONECTADAS A TRAVÉS DE UNA SINAPSIS, (*b*) SINÁPSIS CON MATERIAL QUÍMICO 'CUANTIZADO', (*c*) CORRIENTE ELÉCTRICA EN LA SINAPSIS, PROPORCIONAL A LA LIBERACIÓN DE MATERIAL QUÍMICO CUANTIZADO.

La captación de substancias químicas en la juntura sináptica de la célula receptora

provoca la "subida" o "bajada" del potencial eléctrico en el cuerpo de esta célula (neurona post-sináptica). Si este potencial alcanza un cierto umbral, un paquete de pulsos de intensidad y duración fijos es transmitido a través del axón (Tolouse 1989), de esta forma la señal es conducida por las diferentes ramificaciones del axón llegando a las junturas sinápticas (unidades receptoras) de otras neuronas. Una vez que la célula ha disparado, ésta tiene que esperar un cierto tiempo ("periodo de recuperación"), para poder volver a disparar; así, una neurona emite pulsos con frecuencia variable (Singer 1987).

En redes biológicas se sabe que la señal transmitida a través de las sinapsis no es función únicamente de la señal recibida, ésto es, se sabe que en general las neuronas operan en un ambiente ruidoso (Neher 1992, Amit 1989). Este ruido puede ser atribuído al hecho de que la cantidad de **paquetes cuantizados de neuro-transmisor químico** que se libera en la sinapsis (véase figura 1.2) no es siempre la misma, por lo que se obtienen contribuciones al potencial de la neurona ***post-sináptica*** que en general tampoco son las mismas para todas las neuronas. Para el propósito de esta tesis bastará entender que las neuronas en la realidad operan en un ambiente ruidoso, y para efectos de modelaje, que este ruido está distribuido Gaussianamente.

Donald Hebb (Hebb 1949), descubrió que las propiedades químicas de las junturas sinápticas están directamente relacionadas con la correlación de "disparo" de la neurona *pre* y *post*-sináptica. Si se considera que una sinapsis entre dos neuronas, ajusta sus propiedades químicas de acuerdo a la correlación de disparo de éstas y que además estas propiedades químicas se mantienen por un determinado tiempo, entonces es posible (una vez modificadas las propiedades químicas de la sinapsis) efectuar un proceso inverso, e inferir el comportamiento de alguna señal en un determinado tiempo, entre las dos neuronas. Se dice entonces que las sinapsis, son capaces de **guardar información** referente al comportamiento de las neuronas (en términos de correlación de disparo de señales).

Como ya se mencionó, una sola neurona puede recibir del orden de 10^4 señales provenientes de otras células y todas estas señales pasan a través de sus respectivas sinapsis (ie. 10^4 sinapsis); de aquí se puede apreciar, que si se tiene un número muy grande de neuronas (por ejemplo 10^{10} neuronas como el cerebro humano), solamente un porcentaje muy pequeño de las propiedades químicas en las junturas sinápticas será modificado, permitiendose así, el que una cantidad muy grande de **información** pueda ser almacenada.

1.2 Modelo artificial de una neurona biológica. Como se pudo apreciar en la sección anterior el modelo biológico es complicado y representa hasta la fecha un área de intensa

actividad en las Neurociencias. El tratar de entender el funcionamiento **colectivo** de las neuronas desde una perspectiva física, matemática o computacional, no es menos complicado (Tolouse 1989).

El primer modelo intentando describir el funcionamiento colectivo de un conjunto de neuronas fue elaborado por McCoulloch y Pitts (McCulloch 1943). En este modelo, cada unidad o neurona formal debe calcular una suma de las señales de entrada y entregar como resultado un uno o un cero dependiendo de si la suma esta por "arriba" o por "abajo" de un cierto número o umbral. Más detalladamente, la forma como se llevó a cabo la analogía, fue la siguiente: Primero, la complejidad del estado de una neurona es simplificado hasta que ésta posea solamente dos estados posibles: se encuentra emitiendo una señal (1) o no (0). Segundo, las complejas propiedades químicas de cada sinapsis biológica, son simplificadas hasta ser representadas por un número real. Finalmente, la complejidad de la respuesta de una neurona[1] a las señales de entrada, es simplificada hasta convertirla en una función "escalón" (con umbral) de las entradas.

La forma que adquirió la ecuación en el modelo de McCoulloch-Pitts para representar el estado de alguna neurona *i-ésima* η_i al tiempo $t + \Delta t$, como función del estado de las demás neuronas $\{\eta_i\}$ al tiempo, es

$$\eta_i(t + \Delta t) = \theta\Big(\sum_j \omega_{ij}\eta_j(t) - r_i\Big), \tag{1.1}$$

donde θ es la función escalón definida como

$$\theta(x) = \begin{cases} +1, & \text{si } x \geq 0 \\ 0, & \text{si } x < 0, \end{cases} \tag{1.2}$$

donde el valor del "peso" de la conexión entre la neurona i y la neurona j, asi como el valor del umbral de respuesta de la neurona j, son independientes del tiempo y están representados respectivamente por los números reales ω_{ij} y r_i. Así, con la ecuación (1.1) se obtiene el estado $\eta_i(t+\Delta t) = 1$ ó 0 (Emitiendo/No Emitiendo) de la *i-ésima* neurona.

Existen diversas formas de asignar valores a las constantes $\{\omega_{ij}\}$, en particular, en el

[1]En lo sucesivo y con el fin de evitar confusiones, se utilizará indistintamente, el término **unidad** y el término **conexión** para referise a la "analogía artificial" de neurona y sinapsis respectivamente.

capítulo II, se mostrará una forma de determinar estos valores.

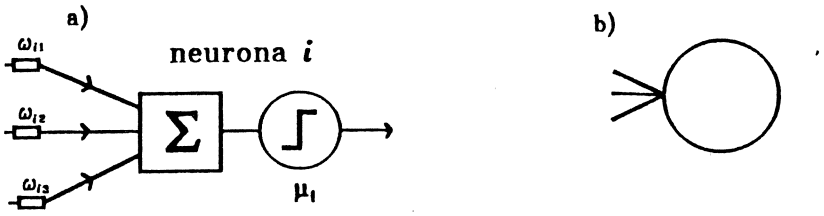


Fig. 1.3 *a*) DIAGRAMA ESQUEMÁTICO DE UNA NEURONA TIPO MCCULLOCH-PITTS, CON TRES ENTRADAS, *b*) REPRESENTACIÓN GRÁFICA DE LA MISMA.

De la simplificación de una neurona biológica se obtuvo la ecuación (1.1). Sin embargo, debe notarse que este modelo teórico no contempla, o más bien difiere de su contraparte biológica en los siguientes puntos, entre otros:

-Las neuronas reales, no son estrictamente dispositivos de umbral.

-Las neuronas reales, desempeñan una suma no-lineal, algunas veces de las señales recibidas.

-Los efectos del ruido debido a la liberación aleatoria de material químico cuantizado, en las sinapsis de neuronas reales, no están contemplados en este modelo.

Sin embrargo, estudiar las propiedades de *redes neuronales* suponiendo que el funcionamiento de una neurona está descrito por la ecuación (1.1), puede servir como marco de referencia teórico, de forma tal que se permita el estudio **sistemático** del comportamiento **cooperativo** observado en algunos grupos de neuronas en el sistema nervioso.

1.3 Computación Tradicional. En la actualidad, las computadoras digitales son algunas veces llamadas "máquinas tipo von Neumann".

Los puntos esenciales de este diseño son (Viana 1990):

-Unidad central de procesamiento, la cual coordina la ejecución de los programas. Esta unidad es mejor conocida como "CPU".

-Existencia de una unidad de memoria, separada del CPU, en la cual se almacena

la información por dirección. Esta memoria sirve para almacenar las instrucciones de los cálculos que deben de ser efectuados, conjuntamente con los datos que deberán de ser procesados y los resultados de dichos cálculos.

-La búsqueda-procesamiento-almacenamiento de información se hace de manera secuencial, en la cual una instrucción es tomada de la unidad memoria (junto con cualquier dato necesario) y la unidad central de procesamiento ejecuta la operación. El resultado de la operación es puesto en una localidad de memoria específica.

Una computadora efectúa sus cálculos sobre números que han sido convertidos a una base digital binaria, cuya unidad básica, se conoce como bit de información y esta representada por un 0 o un 1. Una vez que el número real ha sido digitalizado, se tiene que el valor de los incrementos entre un número y el siguiente, puede ser en la práctica muy pequeño, sin embargo, debe notarse que aunque un número real expresado por 64-bit's es extremadamente preciso, los incrementos discretos permanecen.

La arquitectura tipo Von Neumann ha servido muy bien por 40 años, sin embargo hay tareas que no hace muy bien. De esto hablaremos más adelante.

1.4 Computación Neuronal. A un conjunto de unidades o neuronas formales (unidades que podrían estar descritas por la ecuación (1.1)) interconectadas con una cierta arquitectura , se le denomina *computadora neuronal*[2]; al hecho de efectuar operaciones con esta clase de computadora se le denomina **computación neuronal**. Este nuevo paradigma computacional alternativo al de Von Neumann, es también conocido como "*red neuronal*", "*red asociativa*", "*computación colectiva*", "*conexionismo*" y probablemente de muchas otras formas. En este trabajo se utilizará libremente el término *Redes Neuronales* (por razones históricas), para representar la colección de unidades o neuronas formales, que agrupadas bajo una cierta arquitectura y con una regla de procesamiento, efectúa alguna operación.

[2]Debe notarse que una computadora neuronal, podría estar conformada por miles de unidades.

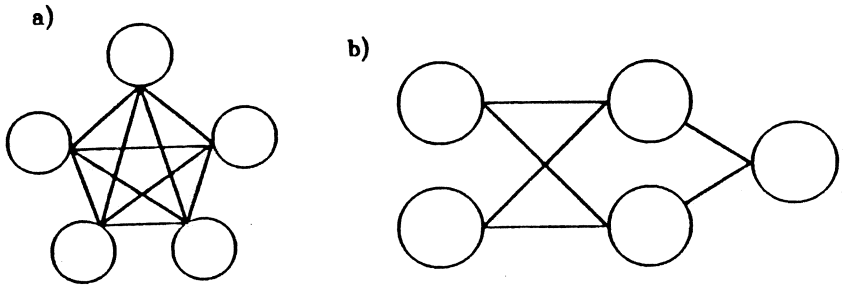


Fig. 1.4 REPRESENTACIÓN GRÁFICA DE LO QUE SERÍA UNA COMPUTADORA O *red neuronal* DE CINCO UNIDADES: *a*) TOTALMENTE CONECTADA *b*) PARCIALMENTE CONECTADA (CONOCIDA COMO PERCEPTRÓN)

La Computación Neuronal ha demostrado (Hertz 1991), que puede ofrecer mejores alternativas de solución a cierta clase de problemas que las que podría ofrecer su contraparte tradicional; por ejemplo, en problemas de reconocimiento y filtrado de señales y en problemas de optimización. En general, las redes neuronales han mostrado su habilidad para solucionar problemas en los cuales está implícita una cierta cantidad de incertidumbre y ambigüedad.

En una red neuronal capaz de procesar información, existen básicamente dos etapas que distinguen al sistema, éstas son, la fase de aprendizaje o entrenamiento, y la fase de recuperación de información. A continuación se explican de una forma *muy general*, estas dos fases.

-En la fase de **aprendizaje**, se tiene que la red neuronal aprende por ejemplos, esto es, modifica los valores de las interconexiones entre las unidades tomando como guía las señales de entrada disponibles, conjuntamente con alguna regla o algoritmo de aprendizaje. Los valores iniciales de las interconexiones no necesitan tener algún valor específico, de hecho, éstos pueden ser escogidos como valores aleatorios pequeños. Posteriormente, estos valores son modificados utilizando la regla de aprendizaje. Una vez terminada la fase de aprendizaje los valores de las interconexiones estarán almacenados en el conjunto de todas las ω_{ij}'s.

-La fase de **recuperación** o de desempeño consiste principalmente en proporcionar

información de entrada a la red y dejar que ésta la procese de acuerdo con una regla de recuperación (como podría ser la ecuación (1.1)), en la cual se utiliza el conjunto de los valores entre las interconexiones que fue definido durante la fase de aprendizaje. De esta forma se obtiene lo que se conoce como **dinámica de la red** (i.e. la ecuación de evolución de los estados de las neuronas a través del tiempo).

Como se puede apreciar por la forma en que una red neuronal puede *aprender* y *recuperar* información (procesar información), se tiene que este nuevo paradigma computacional viola todos los principios de la arquitectura Von Neumann, esto es:

-Una red neuronal no tiene unidad central de procesamiento. La información es procesada por muchas unidades igualmente importantes, las cuales están altamente interconectadas. Cada unidad recibe muchas señales de entrada que provienen a través de las interconexiones de otras unidades. Cada elemento de la red puede efectuar una suma como la descrita por la ecuación (1.1) por ejemplo, y generar una señal de salida, la cual será transmitida a las otras unidades a través de la interconexión de salida.

-Una red neuronal no tiene memoria para el almacenamiento de datos e instrucciones en forma explícita y localizada. En su lugar, la memoria de una red neuronal está distribuida en toda la red, esto es, está almacenada en en forma implícita en el conjunto de valores que toman las interconexiones de las unidades.

-Una red neuronal no ejecuta instrucciones por medio del ciclo encuentra-procesa-almacena, como lo hacen las computadoras tradicionales; en su lugar, las unidades de la red reaccionan a un conjunto particular de señales de entrada, generando en respuesta un conjunto particular de señales de salida. Tampoco existe un programa que controle la actividad de cada una de las unidades, sólamente existe un mapeo funcional entre la entrada total que recibe cada unidad y su salida correspondiente. Finalmente, las computadoras tradicionales[3] funcionan efectuando operaciones en serie, ya que hacen cálculos en base a un programa; en contrapartida una red neuronal es un sistema que procesa información en **paralelo** y que, más que *calcular*, lo que hace es *reaccionar* a alguna señal de entrada.

Como un ejemplo del potencial computacional de un conjunto de neuronas formales, considerese el caso en que se quiere calcular una operación binaria lógica, como una red

[3]Una computadora digital efectúa una sola instrucción a la vez, ya sea ésta una operación de lectura o almacenamiento de información en la memoria o una operación en la unidad central de procesamiento

neuronal constituida por una capa de entrada compuesta por dos unidades, y una capa de salida compuesta por una unidad (véase figura 1.5).

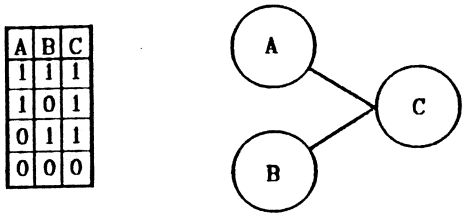


Fig. 1.5 CÁLCULO DE LA OPERACIÓN BINARIA (LÓGICA) OR

Se puede demostrar que existe un conjunto de valores para las ω_{ij}'s, de forma tal que casi cualquiera de las operaciones binarias lógicas, pueden ser efectuadas (a excepción de la operación XOR, (Minsky 1968))[4].

En el siguiente capítulo se presenta la forma como una red neuronal, con un tipo de arquitectura de la forma mostrada en la figura 1.4(a), puede resolver el problema de *memoria asociativa*. Este problema pondrá de manifiesto de una manera sencilla, el potencial de la computación neuronal.

[4]Se puede demostrar (Hertz 1991, Talavera 1992), que una arquitectura de neuronas, como la descrita por la figura 1.4(b), es capaz de efectuar cualquier operación binaria lógica.

II. Memoria Asociativa

El problema de memoria asociativa consiste basicamente, en almacenar un conjunto de P patrones o "fotografías", de forma tal que si al sistema se le presenta una de las fotografías que fueron originalmente almacenadas, pero incompleta o con ruido agregado, el sistema utilizado debe ser capaz de completar o filtrar la fotografía, según sea el caso. El problema de memoria asociativa es considerado como el más fundamental del campo de redes neuronales; éste a su vez, ilustra de una forma sencilla el potencial de la computación neuronal. El problema de memoria asociativa será el tema central de esta tesis.

2.1 Modelo Digital. El problema de memoria asociativa puede ser resuelto en una computadora convencional, pará ésto, simplemente se almacena en memoria una lista de P patrones definidos como un conjunto compuesto de N bits de información ζ_i^μ (en donde $i = 1, \ldots, N; \mu = 1, \ldots, P$). Si ahora, a la computadora se le presenta un nuevo patrón η_i con $i = 1, \ldots, N$ (abreviado $\{\eta\}$), entonces sera "recordado" el patrón ζ_i^λ en donde $i = 1, \ldots, N$ y $\lambda \in \mu$ (abreviado $\{\zeta^\lambda\}$) que más fuertemente se asemeje al patrón presentado. Esto significa que $\{\zeta^\lambda\}$ y $\{\eta\}$ deberán diferir en el menor número posible de bits. Una forma de medir esta diferencia, es por medio de la función

$$H_\mu = \frac{1}{N} \sum_{i=1}^{N} (\eta_i - \zeta_i^\mu)^2. \tag{2.1}$$

Esta medida entrega como resultado lo que podría ser denominado "*fracción de coincidencias*" entre el estado presentado y el μ-*ésimo* patrón almacenado. Así, se encuentran las P distancias H_μ y el patrón que se encuentra a menor distancia, es tomado como el resultado final de este problema. Más explicitamente, H_μ, deberá ser mínima para $\mu = \lambda$. H_μ es llamada la **distancia de Hamming** entre el patrón $\{\eta\}$ y $\{\zeta^\mu\}$ donde μ es alguno de los P patrones almacenados. Es importante mencionar, sin embargo, que una computadora convencional compara el patrón inicial $\{\eta\}$, con cada uno de los otros P patrones almacenados para obtener la respuesta, lo cual consume mucho tiempo de cómputo.

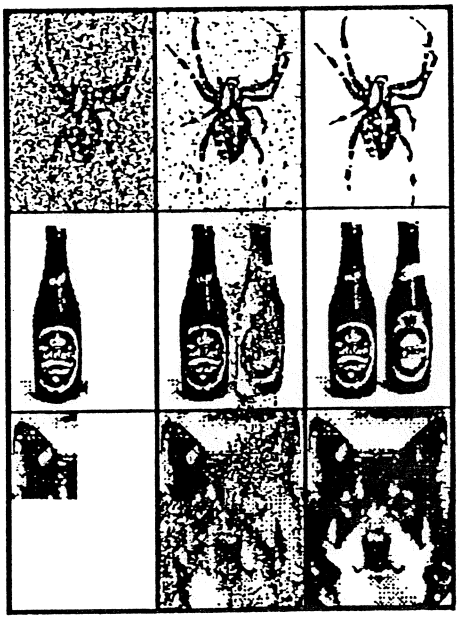


Fig. 2.1 EJEMPLO DE COMO UNA MEMORIA ASOCIATIVA PUEDE RECONOCER IMAGENES

2.2 Modelo Neuronal.

2.2.1 Dinámica. En esta sección se analizará la forma como cada una de las unidades que componen una red neuronal, actualiza su estado, de acuerdo con una variante del modelo de McCulloch-Pitts (McCulloch 1943, véase ecuación (1.1)). Se empezará por redefinir, por conveniencia matemática, los valores que cada elemento puede tomar, como $S_i(t) = +1$ (disparando) o $S_i(t) = -1$ (no disparando) en lugar de $\eta_i = 1$ y $\eta_i = 0$ en el modelo de McCulloch-Pitts. La variable $\xi_i^{\mu} = \pm 1$, denotará el valor de los bits en los patrones que se desean almacenar, de igual forma, se utilizará la variable J_{ij} para denotar el valor de la interacción sináptica, que en este caso será un número racional positivo o negativo. La variable correspondiente al umbral será eliminada (igualada a cero) por simplicidad matemática.

En general, la forma como estará definida la dinámica de la red, será por medio de la siguiente ecuación

$$S_i(t + \Delta t) \equiv signo\left(\sum_{j=1}^{N} J_{ij} S_j(t)\right), \tag{2.2}$$

en donde

$$signo(x) = \begin{cases} +1, \text{ si } x > 0 \\ sin\ cambio, \text{ si } x = 0 \\ -1, \text{ si } x < 0 \end{cases} . \tag{2.3}$$

La ecuación (2.2) será la ecuación por medio de la cual será actualizado el estado de cada una de las neuronas; la forma como se organizarán las neuronas (i.e. la arquitectura), será la descrita por la figura 1.4(a) del primer capítulo, ésto es, todas las unidades conectadas con todas.

El objetivo de esta tesis es analizar el comportamiento del modelo descrito, simulado en una computadora digital, por lo que se tomaran en cuenta algunas consideraciones particularmente, la forma como se actualizan los estados de las neuronas. Estos estados pueden actualizrse de tres formas diferentes:

1º) **Dinámica secuencial**. En esta forma de actualizar los estados de las neuronas, lo que se hace es tomar la primera neurona y calcular el estado al que debe de evolucionar al tiempo $(t + \Delta t)$ de acuerdo con (2.2); se prosigue actualizando de esta misma forma de una por una, y en un orden, al resto de las neuronas. La escala de tiempo que se utilizará será proporcional al tamaño de la red, esto es, $\Delta t = \frac{1}{N}$ por cada vez que una neurona es actualizada (i.e. al tiempo t=1 todos los estados de las neuronas ya fueron actualizados una sola vez).

2º) **Dinámica secuencial aleatoria**. En esta forma de actualizar los estados de las neuronas, lo que se hace es tomar una neurona al azar y calcular el estado al que debe de evolucionar al tiempo $(t + \Delta t)$ de acuerdo con (2.2); se prosigue tomando otra neurona al azar y actualizandola de la misma forma. El proceso se repite un determinado número de veces hasta que, en promedio, todas las neuronas ya fueron actualizadas. La escala de tiempo que se utilizará, será proporcional al tamaño de la red, esto es, $\Delta t = \frac{1}{N}$ por cada vez que una neurona es actualizada. A esta forma de actualizar las neuronas también se le conoce como dinámica de Glauber (Glauber 1963) en otros modelos similares.

3º) **Dinámica en paralelo**. En esta forma de actualizar los estados de las neuronas, lo que se hace es tomar la primera neurona y calcular el estado al que debe de evolucionar al tiempo $(t + \Delta t)$ de acuerdo con (2.2). La única diferencia entre la simulación de esta dinámica y la de la dinámica secuencial, será que en este caso el estado de la neurona no

será actualizado inmediatamente, sino que este valor será guardado en memoria. Una vez que todos los valores de todos los estados han sido calculados, se procederá a actualizar los estados de todas las neuronas a la vez; ésto se hace tomando los nuevos valores de los estados en memoria y substituyéndolos por los estados anteriores. La escala de tiempo que se utilizará, será $\Delta t = 1$ (i.e. al tiempo t=1 todos los estados de las neuronas ya fueron actualizados una sola vez). A esta forma de actualizar las neuronas, también se le conoce como actualización sincrónica.

Nótese que $t = 0$ representa el tiempo en el que el conjunto de estados $\{S(t = 0)\}$ representa la configuración inicial o el patrón a restaurar[1]; de esta forma, con las ecuaciones (2.2) y (2.3), y utilizando alguna de las tres formas de actualizar los estados de los espines, es como la dinámica de la réd ha quedado definida.

2.2.2 Aprendizaje de la red, con un sólo patrón. A continuación se presenta la forma en que la dinámica descrita en la sección anterior por las ecuaciones (2.2) y (2.3), conjuntamente con la forma de actualizar los estados de las unidades, es capaz de resolver el problema de memoria asociativa, cuando se desea almacenar y recuperar un sólo patrón correctamente. Este problema, aunque no es de utilidad práctica debido a su trivialidad, nos permitirá ver cual es el mecanismo de evolución en una red neuronal tipo Hebb. Dividiremos la discución en dos partes: primero veremos si el estado almacenado corresponde a un punto fijo del sistema, y después analizaremos si dicho punto fijo corresponde a un atractor dinámico.

Se puede efectuar un análisis sencillo de las ecuaciones que definen la dinámica de los estados de las unidades en la red neuronal (ecs. (2.2) y (2.3)), de forma tal que se encuentren las características que las J_{ij}'s y $S_i(t)$'s, deben cumplir, para que un patrón quede almacenado de alguna forma en los valores de las J_{ij}'s. Para tal efecto, la metodología a seguir, consistirá en encontrar las características de los parámetros en las ecuaciones (2.2) y (2.3) (i.e. J_{ij}'s), tales que, los estados o valores finales de cada una de las unidades sean **independientes** del tiempo; cuando ésto sucede, se dice que la dinámica de la red ha evolucionado hacia un **punto fijo**. *El que la dinámica de la red permanezca en un punto fijo será el criterio con el que se calificará la estabilidad de la red neuronal.* Para que la red neuronal pueda ser útil en la solución del problema de memoria asociativa, se pedirá, además, que el punto fijo de la dinámica de la red (i.e. estado final, del conjunto de estados

[1]En lo sucesivo, se referirá al patrón a restaurar, como condición inicial de la dinámica de la red.

de las neuronas), **represente al patrón almacenado**, con lo que el problema quedará resuelto.

Más específicamente, se definirá $S_i(t + \Delta t) = S_i(t)$, como la condición de punto fijo. Lo que de buscará a continuación, serán las condiciones que las J_{ij}'s deben cumplir, de forma tal, que cuando la dinámica de la red se inicializa con el patrón sin errores (i.e. el punto fijo.), la dinámica de la red, no cambie el estado del sistema; estas condiciones para las J_{ij}'s definen una regla de aprendizaje ó algoritmo de aprendizaje. Nótese, que si ésta forma óptima de aprendizaje existe, **se puede efectuar un proceso inverso**, calculando las J_{ij}'s primero, con la nueva regla de aprendizaje, y posteriormente utilizar las ecuaciones que describen la dinámica de la red, para resolver el problema de memoria asociativa.

Utilizando las ideas anteriores, se obtiene, que utilizando la ecuación (2.2), se puede ver que el patrón que se quiere almacenar $\xi_i = \pm 1$ para $i = 1 \ldots N$, (definirá un punto fijo en la dinámica de la red), si y sólo si, se cumple la condición

$$S_i(t + \Delta t) = S_i(t) = \xi_i, \tag{2.4}$$

para todas las unidades i. Más explícitamente, utilizando (2.2) se obtiene

$$\xi_i = signo\left(\sum_j^N J_{ij}\xi_j\right), \tag{2.5}$$

para todas las unidades. De la ecuación (2.5), se puede observar que esta condición se cumplirá siempre y cuando las J_{ij}'s, cumplan con la condición

$$J_{ij} \propto \xi_i\xi_j, \tag{2.6}$$

ya que substituyendo (2.6) en (2.5) se obtiene

$$\xi_i \propto signo\left(\xi_i \sum_j^N \xi_j^2\right) = signo(\xi_i), \tag{2.7}$$

y dado que $\xi_j^2 = +1$. Esto dará como resultado el signo de ξ_i, siempre y cuando la constante de proporcionalidad sea positiva; la constante de proporcionalidad en (2.7) será tomada como $1/N$ (Müller 1991), en donde N es el número de neuronas, convirtiendo (2.6) en

$$J_{ij} = \frac{1}{N}\xi_i\xi_j. \tag{2.8}$$

De (2.8), se pueden apreciar, cuatro casos diferentes

$$\begin{aligned}
&\text{si} \quad \xi_i = +1 \; y \; \xi_j = +1, \text{se obtendrá } J_{ij} = +1/N, \\
&\text{si} \quad \xi_i = +1 \; y \; \xi_j = -1, \text{se obtendrá } J_{ij} = -1/N, \\
&\text{si} \quad \xi_i = -1 \; y \; \xi_j = +1, \text{se obtendrá } J_{ij} = -1/N, \\
&\text{si} \quad \xi_i = -1 \; y \; \xi_j = -1, \text{se obtendrá } J_{ij} = +1/N.
\end{aligned} \tag{2.9}$$

Una vez calculadas todas las eficiencias sinápticas de acuerdo con (2.8), se puede apreciar, que es posible representar el conjunto de todas las interacciones (eficiencias sinápticas), en una matriz simétrica de N por N elementos, con $\frac{+1}{N}$'s en la diagonal. Esta matriz será denominada como **matriz de interacciones o conocimiento.** Es importante señalar que debido a la simetría existente en la matriz de interconecciones ó de aprendizaje, descrita por la ecuación (2.8), son de hecho, dos puntos fijos los que son almacenados a la vez, estos son $\{\xi\}$ y $\{-\xi\}$. Así, si el estado de la red se inicializa en cualquiera de estos dos puntos fijos, los estados de las unidades, permanecerán invariantes durante actualizaciones sucesivas (i.e. para $t > 1$).

De (2.9), se puede apreciar, que si el estado de dos neuronas es el mismo, el valor que obtendrá la eficiencia sináptica es de $\frac{+1}{N}$ (sinápsis excitadora), mientras que si alguna de las dos neuronas tiene signo diferente, el valor de la eficiencia sináptica obtiene el valor de $\frac{-1}{N}$ (sinápsis inhibidora). Esto es, la eficiencia sináptica se modifica de acuerdo con la correlación del valor de las neuronas *pre* y *post-sinápticas*, lo cual está de acuerdo con la tesis biológica del neurofisiólogo Donald Hebb (Hebb 1949, véase sección (1.1)), hecho por el cual a una generalización de esta forma de aprendizaje, se le conoce como **regla de aprendizaje de Hebb**. Sin embargo, de (2.9) se puede apreciar también, que las eficiencias sinápticas se ven incrementadas aún cuando las dos neuronas no están activadas, cosa que no es biológicamente razonable.

2.2.3 Estabilidad del patrón almacenado. Como hemos visto, una vez almacenado el patrón por medio de la ecuación (2.8), y sabiendo que si el estado de la red se inicializa con este patrón (i.e. en el punto fijo.), los estados de las unidades permanecerán en el punto

fijo a través de todas las actualizaciones sucesivas ($t > 0$). A continuación veremos si el punto fijo corresponde a un atractor de la dinámica del sistema definida por las ecuaciones (2.2) y (2.3). La metodología a seguir será la siguiente: una vez almacenado el patrón, se puede inicializar el estado de la red en un estado que no sea exactamente el patrón almacenado, ésto es, inicializar el estado de la red en un estado cercano al punto fijo, y ver su evolución temporal. Una forma de situar el estado inicial de la red en un estado cercano al definido por el punto fijo, puede hacerse tomando algunos de los elementos del patrón almacenado y "volteando" intencionalmente algunos de sus estados (-1, en lugar de $+1$ y viceversa), a continuación se utiliza este nuevo patrón como condición inicial de la red. Sin pérdida de generalidad se puede asumir que estos bits equivocados son los primeros n elementos del patrón, matemáticamente

$$S_i(t=0) = \begin{cases} -\xi_i, \text{ para } i = 1, \ldots, n \\ +\xi_i, \text{ para } i = (n+1), \ldots, N. \end{cases} \tag{2.10}$$

Por lo que las contribuciones para la unidad i, al tiempo $t = 0$, definidas como $h_i(t=0)$, pueden ser re-escritas como

$$h_i(t=0) = \sum_{j=1}^{N} J_{ij} S_j(t=0) = \frac{1}{N} \xi_i \sum_{j=1}^{N} \xi_j S_j(t=0). \tag{2.11}$$

Utilizando (2.10) en (2.11), se obtiene

$$h_i(t=0) = \frac{1}{N} \xi_i \big[(N-n) - n\big] = \left(1 - \frac{2n}{N}\right) \xi_i, \tag{2.12}$$

así, para $n < \frac{N}{2}$ la red efectúa una transición correcta hacia el patrón almacenado, en un sólo paso, ésto es, en un tiempo $t = 1$. Más explicitamente, de (2.12) se obtiene:

$$S_i(t + \Delta t) = signo\big[h_i(t=0)\big] = signo\big(1 - \frac{2n}{N}\big) signo(\xi_i) = \xi_i. \tag{2.13}$$

En el caso en el cual $\frac{2n}{N} > 1$, se tendrá que existen ($n > \frac{N}{2}$). Esto es, existirán más diferencias que coincidencias con los bits del patrón almacenado, por lo que la red ciertamente reconoce lo que podría ser considerado como el "negativo" de la fotografía o $\{-\xi\}$, ya que este patrón, también está definido como punto fijo de la dinámica de la red.

2.2.4 Regla de aprendizaje de Hebb. En esta sección, se verá la regla de aprendizaje de Hebb que consiste en una generalización de la ecuación (2.8), para cuando se quiere almacenar un número arbitrario de patrones. Una vez definida la nueva regla de aprendizaje, se proseguirá a estudiar la estabilidad de los puntos fijos de la dinámica (patrones almacenados), que fueron almacenados por ésta. La dinámica estará descrita por la misma ecuación (2.2).

La forma como se generalizará la regla de aprendizaje para que la red aprenda más de un sólo patrón, será repitiendo el proceso descrito por la ecuación (2.8), un número de veces igual al número P de patrones que se quiere almacenar; de forma tal que se obtenga una matriz de interacciones o conocimiento por cada patrón que se quiere almacenar. Una vez generadas estas matrices, la forma como esta información será relacionada entre sí, de forma tal que los patrones que se desea almacenar representen puntos fijos de la dinámica de la red, será mediante la operación de **adición de matrices**. Esto es, una vez que todas las matrices han sido sumadas, generarán como resultado una sóla matriz simétrica de N por N elementos. En la siguiente sección, se mostrará que efectivamente, esta forma de relacionar las matrices (almacenar los patrones), "genera" puntos fijos de la dinámica de la red, descrita por (2.2), que coinciden con los patrones almacenados para ciertos valores de P, como se mostrará más adelante.

Tomando en consideración las ideas anteriores, se obtiene que la nueva regla de aprendizaje, para cuando se quiere almacenar un número arbitrario P de patrones, estará descrita por

$$J_{ij} = \frac{1}{N} \sum_{\mu=1}^{P} \xi_i^\mu \xi_j^\mu, \tag{2.14}$$

en donde $\mu = 1, 2, \ldots, P$ representa el número de patrones a aprender y en donde el factor de normalización seguirá siendo $\frac{1}{N}$. Esta es la comúnmente llamada **Regla de aprendizaje de Hebb**.

2.2.5 Estabilidad de más de un patrón almacenado. En esta sección se efectuará un análisis de estabilidad basado en métodos estadísticos (análisis *Señal a Ruido*), de forma tal que quede de manifiesto la estabilidad de los patrones almacenados (puntos fijos de la dinámica), cuando se utiliza la regla de Hebb (2.14), conjuntamente con (2.2).

Utilizando (2.2) y (2.14) se obtiene la nueva ecuación que describe la dinámica de los

estados de las unidades, quedando ésta expresada como:

$$S_i(t+\Delta t) = signo\left(\frac{1}{N}\sum_{j=1}^{N}\sum_{\mu=1}^{P}\xi_i^\mu\xi_j^\mu S_j(t)\right). \tag{2.15}$$

Para llevar a cabo el análisis, y siguiendo con la misma estrategia que en la sección (2.2.3), se considerará que el estado inicial $\{S(t=0)\}$, coincide exactamente con uno de los patrones que ya fueron almacenados, por ejemplo el patrón ν, además se considerará que no existen autointeracciones o sea que $J_{ii}=0$, en términos de las matrices, este proceso es equivalente a hacer cero los elementos de la diagonal en todas las matrices de interacciones (o hacer cero los elementos de la diagonal, en la matriz de aprendizaje de Hebb).

Utilizando la condición de punto fijo para la dinámica de la red, con $\{S_i(t)=\xi^\nu\}$ para cada i como condición inicial, se obtiene de la ecuación (2.15)

$$\xi_i^\nu = signo\left(\frac{1}{N}\sum_{j=1}^{N}\sum_{\mu=1}^{P}\xi_i^\mu\xi_j^\mu\xi_j^\nu\right), \tag{2.16}$$

lo cual puede reescribirse como

$$\xi_i^\nu = signo\left(\frac{1}{N}\sum_{j=1}^{N}\xi_i^\nu\xi_j^\nu\xi_j^\nu + \frac{1}{N}\sum_{j=1}^{N}\sum_{\mu\neq\nu}\xi_i^\mu\xi_j^\mu\xi_j^\nu\right) \tag{2.17}$$

$$\xi_i^\nu = signo\left(\frac{1}{N}(N\xi_i^\nu) + \frac{1}{N}\sum_{j=1}^{N}\sum_{\mu\neq\nu}\xi_i^\mu\xi_j^\mu\xi_j^\nu\right) \tag{2.18}$$

$$\xi_i^\nu = signo\left(\xi_i^\nu + \frac{1}{N}\sum_{j=1}^{N}\sum_{\mu\neq\nu}\xi_i^\mu\xi_j^\mu\xi_j^\nu\right), \tag{2.19}$$

multiplicando ambos lados de esta ecuación por ξ_i^ν, obtenemos

$$+1 = signo\left(1+\xi_i^\nu\frac{1}{N}\sum_{j=1}^{N}\sum_{\mu\neq\nu}\xi_i^\mu\xi_j^\mu\xi_j^\nu\right). \tag{2.20}$$

De esta última ecuación se puede apreciar claramente que el estado del elemento *i-ésimo* de la red $S_i(t) = \xi_i^\nu$ para cada i será estable a menos de que el argumento de la ecuación (2.20) sea negativo, tomando en cuenta estas ideas, considérese la cantidad

$$C_i^\nu = -\xi_i^\nu \frac{1}{N} \sum_j^N \sum_{\mu \neq \nu}^P \xi_i^\mu \xi_j^\mu \xi_j^\nu, \tag{2.21}$$

que podría ser considerado como un término de **interferencia** en (2.19). Si C_i^ν es negativa el término de interferencia tiene siempre el mismo signo que el término deseado ξ_i^ν y por lo tanto no afecta al campo en nada, pero si la cantidad C_i^ν es positiva y mayor que $+1$, entonces el signo del lado derecho de la ecuación (2.19) sería diferente del signo de ξ_i^ν, por lo que el valor del estado de **esa** neurona en particular, sería inestable. Se puede decir entonces, que el estado del elemento *i-ésimo* cambiará a un estado no deseado. Si repetimos este análisis para cada uno de los elementos de la red, veremos que un cierto porcentaje de "bits" no serán recuperados correctamente. Esto es, la condición inicial, no permanecerá invariante durante la actualización de los estados, y entonces se dice que el punto fijo de la dinámica de la red (definido por el patrón $\{\xi^\nu\}$), es inestable.

Debe notarse que lo único que impide que $S_i(t + \Delta t) = \xi_i^\nu$ en la ecuación (2.17), es función solamente de los patrones almacenados, por lo que la estabilidad de los patrones almacenados dependerán en este caso, de la magnitud de la interferencia generada por éstos (i.e. de las C_i^ν's). La magnitud de las C_i^ν's depende **sensiblemente** del número de patrones almacenados, y de la clase de información de la que está compuesta cada patrón, ésto es, de si se trata de patrones generados al azar, ortogonales, o correlacionados, ya que como se mencionó anteriormente, éste es el factor determinante en la distribución de las C_i^ν's.

Por el momento se considerará el caso en que los patrones fueron generados al azar, con igual probabilidad, para $\xi_i^\mu = +1$ y para $\xi_i^\mu = -1$, independientemente para cada i y μ. Para este caso es posible estimar la probabilidad P_{error} de que algún bit sea inestable. Expresando matemáticamente las ideas anteriores, se obtiene

$$P_{error} = Prob(C_i^\nu > +1). \tag{2.22}$$

Para calcular la expresión de P_{error} debe notarse que C_i^ν depende del número de unidades N y del número de patrones P, se asumirá, que ambos N y P son grandes

comparados con 1. Si C_i^ν es $\frac{1}{N}$ veces la suma de alrededor de NP números aleatorios independientes[2], cada uno de los cuales puede ser ± 1 con igual probabilidad. El problema es equivalente al problema conocido como del camino aleatorio, o el caminante que da pasos de tamaño $\frac{1}{N}$, hacia la derecha y hacia la izquierda con igual probabilidad. En este problema se quiere encontrar la probabilidad de que el caminante recorra una cierta distancia total (i.e. desde el punto de partida) después de NP pasos. Se puede demostrar (Reichl 1980, Bhattacharya 1990) que esta probabilidad está distribuída binomialmente con media cero y varianza $\sigma^2 = \frac{P}{N}$. En el límite termodinámico, esto es, cuando $N \to \infty$, esta última distribución puede aproximarse por una distribución Gaussiana continua con la misma media y la misma varianza, como se puede apreciar en la figura 2.2.

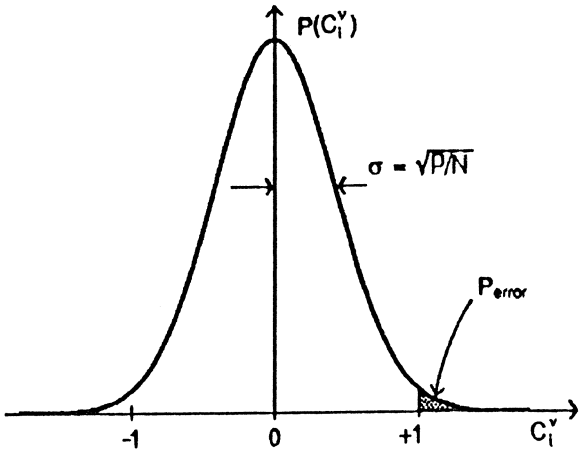


Fig. 2.2 DISTRIBUCIÓN DE PROBABILIDADES PARA EL TÉRMINO DE INTERFERENCIA C_i^ν, DADO POR (2.21), EL ÁREA SOMBREADA, REPRESENTA P_{error} Ó LA PROBABILIDAD DE ERROR POR UNIDAD i.

La probabilidad P_{error} de que C_i^ν exceda $+1$ es simplemente el area sombreada bajo la curva en la figura (2.2), por lo que se obtiene que

[2]De hecho existen $N(P-1)$ términos, si se incluyen los términos $i = j$ o $(N-1)(P-1)$ términos si éstos no se incluyen. Cuando N y P son grandes, el número de términos es aproximadamente NP.

$$P_{error} = \frac{1}{\sqrt{2\pi\sigma^2}} \int_{+1}^{\infty} e^{\frac{-x^2}{2\sigma^2}} dx, \tag{2.23}$$

si ahora utilizamos la función de error $erf(x)$, definida como

$$erf(x) \equiv \frac{2}{\sqrt{\pi}} \int_0^x dt\, e^{-t^2}, \tag{2.24}$$

se obtiene

$$P_{error} = \frac{1}{2}[1 - erf(1/\sqrt{2\sigma^2})] = \frac{1}{2}[1 - erf(\sqrt{N/2P})]. \tag{2.25}$$

La tabla (2.1), presenta la probabilidad de error por bit P_{error}, para diferentes valores de P/N. Es importante señalar, que los valores obtenidos en esta tabla, son estrictamente válidos en el límite en que se ha considerado un número infiníto de unidades.

P_{error}	P/N
0.001	0.105
0.0036	0.138
0.01	0.185
0.05	0.37
0.1	0.61

Tabla 2.1 ERRORES OBTENIDOS EN LA FASE DE RECUPERACIÓN COMO FUNCIÓN DEL NÚMERO DE PATRONES Y EL NÚMERO DE UNIDADES.

Esta tabla muestra, que si N se mantiene constante, P_{error} se incrementa al incrementarse el número de patrones P que se intenta almacenar, lo cual es un resultado intuitivamente correcto. Eligiendo un criterio aceptable para el desempeño de alguna unidad, por ejemplo[3] $P_{error} < 0.01$. Se obtiene que la **cantidad máxima de patrones** P_{max} que pueden almacenarse en la red, ésto es, el máximo número de patrones que pueden almacenarse sin errores inaceptables. Si se exige un valor de $P_{error} < 0.01$, por ejemplo P_{max} sería de $0.15N$. La razón (P/N), se utiliza mucho en estudios más elaborados (véase

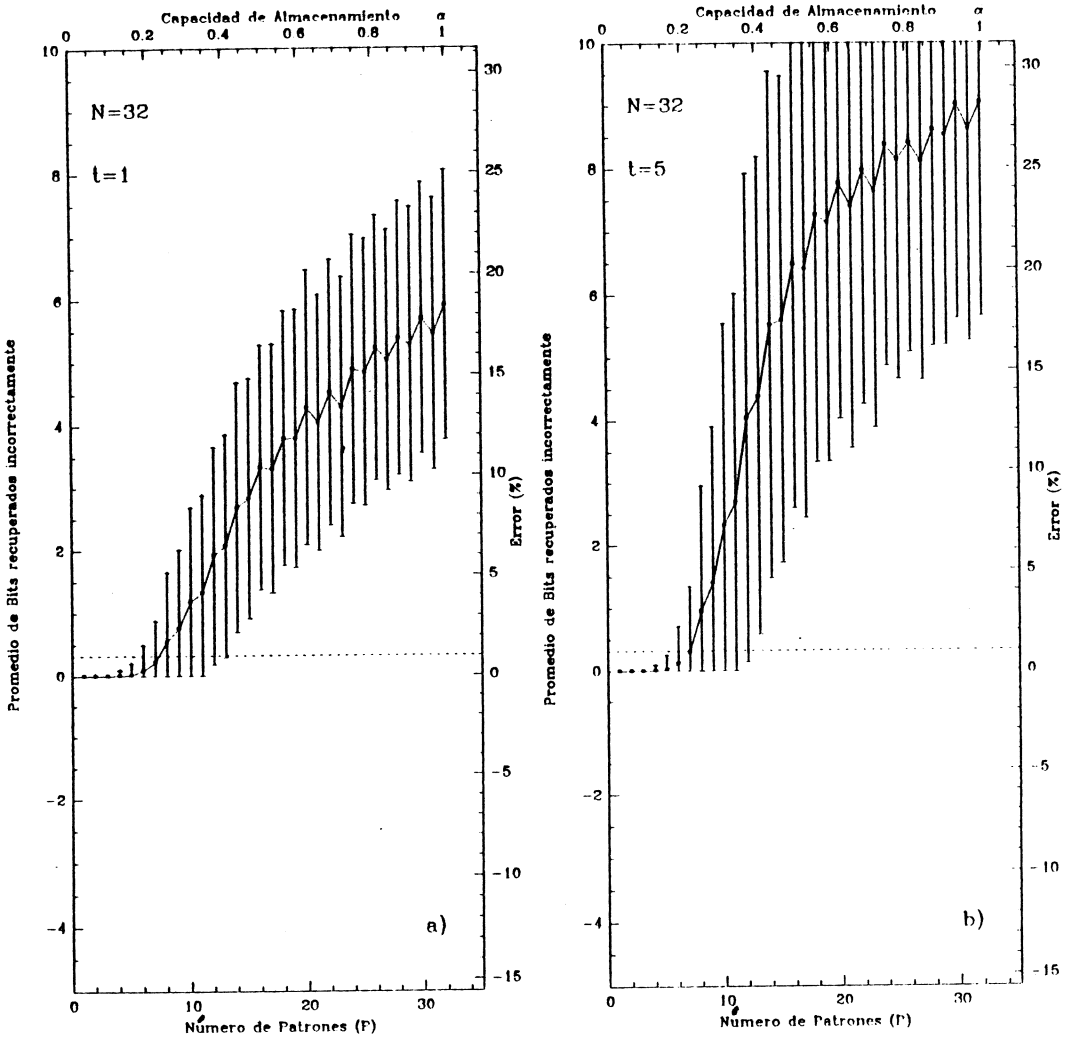
[3]Este valor, por ejemplo, podría equivaler a una red neuronal de 32 unidades N, la cual ha aprendido entre 5 y 6 patrones P.

Capítulo III), y se le conoce con el nombre de **razón o capacidad de almacenamiento** y se identifica con la letra α.

Nótese que hasta el momento solamente se ha tratado el problema de la estabilidad de patrones almacenados que se han utilizado como condiciones iniciales. Como se puede apreciar de la expresión anterior, si se requiere que $P < 0.138N$ por ejemplo, la expresión (2.25) diría que no más del 1% de los bits en los patrones serían inestables una vez alcanzado el punto de equilibrio o estado atractor.

Como se mencionó anteriormente, el resultado teórico obtenido es válido para cuando se está simulando un número infinito de unidades, por lo cual en redes de tamaño finito van a aparecer efectos relacionados con el tamaño del sistema. Por otro lado, estos resultados son válidos para el punto fijo, ésto es el estado de equilibrio al cual evolucionará el sistema.

La gráfica (2.1), muestra el número de bits equivocados en una red neuronal con 32 unidades, con P patrones almacenados, a dos diferentes tiempos t. Esta simulación fué llevada a cabo por el autor.



Grafica 2.1 GRAFICA MOSTRANDO EL NÚMERO PROMEDIO DE BITS EQUIVOCADOS A DOS DIFERENTES TIEMPOS. *a*) $t = 1$, *b*) $t = 5$, PARA UNA RED NEURONAL DE 32 UNIDADES, LA CUAL HA SIDO ENTRENADA, CON UN NÚMERO P DE PATRONES. EL ENSEMBLE DE SISTEMAS, SOBRE LOS QUE SE HA EFECTUADO EL PROMEDIO, CONSTA DE 1024 MUESTRAS.

En la gráfica (2.1), el eje y representa el valor promedio de los bits equivocados, durante la fase de recuperación, como función de P para dos tiempos diferentes *a*) $t = 1$ y *b*) $t = 5$. El promedio se efectuó para un total de 1024 simulaciones por punto. En el eje superior se ha graficado el valor de α, que para este caso es proporcional al número de

patrones, ya que el número de unidades permaneció constante ($N = 32$), durante todas las simulaciones. En el eje lateral derecho, se ha graficado el error obtenido expresado en porcentaje de bits incorrectos; este error es proporcional al número de bits recuperados incorrectamente. Las barras de error, fueron calculadas tomando en cuenta, los 1024 estados de las simulaciones, y lo que representan es una medida de que tan dispersos estaban los 1024 valores con respecto al valor promedio en unidades de bits incorrectos (desviación estándar).

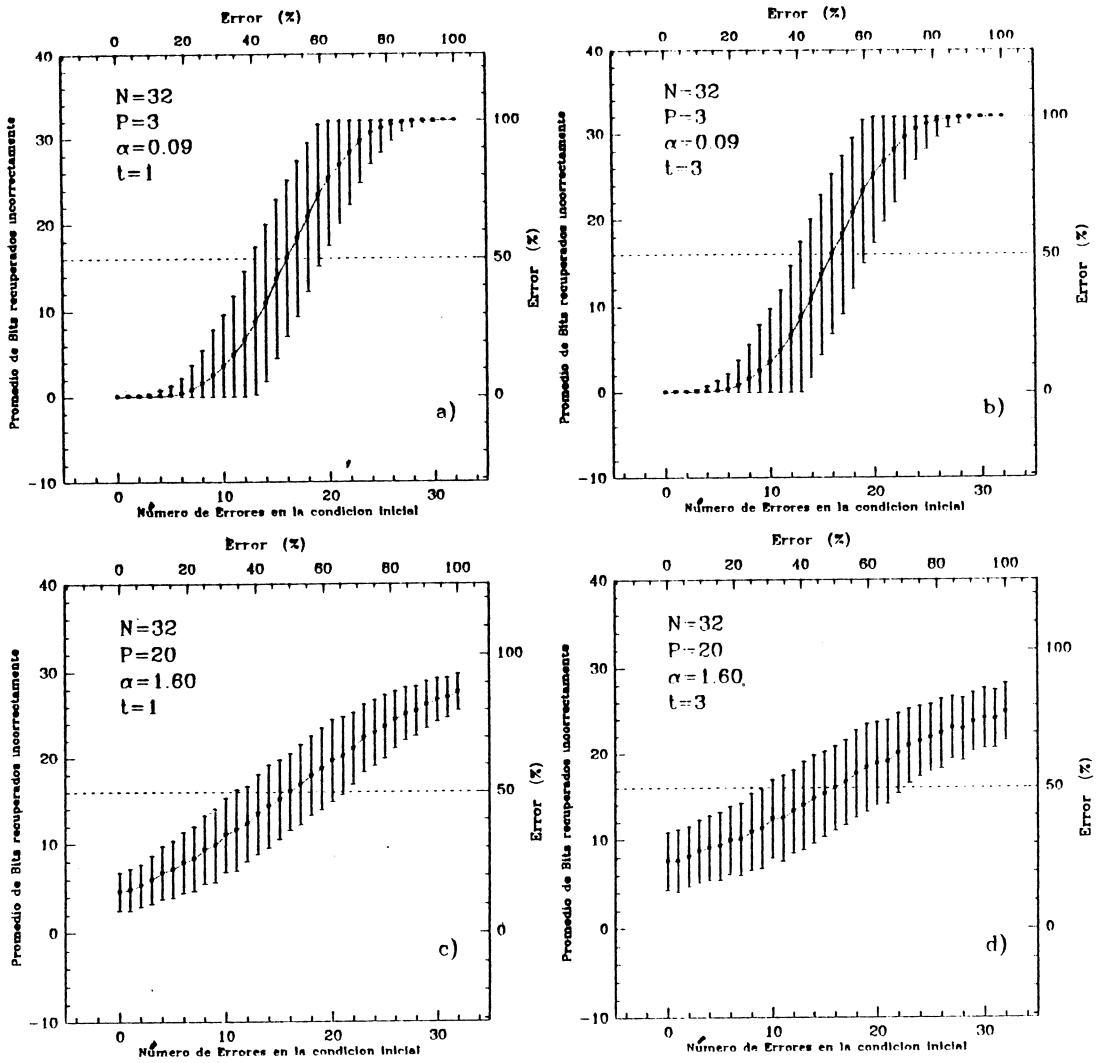
La linea punteada horizontal, que aparece en la gráfica representa el valor del 1% de error en la recuperación, así para el caso de 32 unidades, éste estará ubicado en el valor de 0.32 bits equivocados.

De la gráfica (2.1a), se puede apreciar que efectivamente, si se pide que no más del 1% de los bits estén equivocados, se tendrá que no más de $P_{max} = 5$ ($\alpha = 0.156$) patrones pueden ser almacenados y recuperados en promedio, mostrando así que el análisis señal a ruido utilizado y las simulaciones numéricas concuerdan, considerando que los resultados experimentales obtenidos no contemplan un número infinito de unidades. La gráfica (2.1b), muestra la misma red de la gráfica (2.1a) pero para la cual se han seguido actualizando los estados de las unidades, 4 veces más. Esto es, se ha utilizado el resultado de la iteración anterior, como condición inicial en la actualización posterior, este proceso se ha repetido 4 veces, el resultado, se muestra en la gráfica (2.1b). Como se puede apreciar de esta segunda gráfica, si se siguen actualizando los estados de la red, ocurre un fenómeno de avalancha de errores, en el cual después de un cierto tiempo, y para un cierto número de patrones, el estado inicial no guarda casi nada, ó nada de parecido con el patrón almacenado. En este caso no se comprobó si los estados del tiempo t corresponderían a puntos fijos, esto es, si $S_i(t = 4) = S_i(t = 5)$ para toda i, sin embargo, se intuye que α_c será del orden de 0.15 ($P_{max} < 5$).

Las observaciones arriba mostradas, pueden hacer que el problema pueda ser apreciado desde otro punto de vista. Por ejemplo, uno se puede preguntar qué es lo que pasa, si se utiliza como condición inicial el patrón $\{\xi^\nu\}$, pero con algunos bit's de información incorrectos. Lo que puede pasar en los primeros pasos de actualización es, que más y más bit's se volteen incorrectamente, causando el fenómeno de avalancha, en el cual después de un cierto número de actualizaciones, el estado final no guardará nada, ó casi nada de parecido con el patrón almacenado. También puede suceder que los bits incorrectos se orienten correctamente, recuperando así el patrón almacenado, o que al menos una porción grande de éstos lo haga, la forma en que ocurrira una de estas dos posibilidades, estará

determinada por el valor de $\alpha = \frac{P}{N}$.

La serie de gráficas (2.2), muestra los resultados de simulaciones de una red neuronal de 32 unidades, en la cual se ha considerado un número constante de patrones almacenados $P = 3$. En donde se ha considerado el número de bits equivocados, como la variable independiente. Las gráficas (2.2a) y (2.2b), muestran los resultados obtenidos, para $t = 1$ y $t = 3$ respectivamente.



Grafica 2.2 ACTUALIZACIÓN DE LA RED DE LA GRAFICA 2.1, CONSIDERANDO N Y P CONSTANTES, PERO CON UN NÚMERO VARIABLE DE ERRORES EN LAS CONDICIONES INICIALES. *a)* $t = 1$ Y *b)* $t = 3$, PARA $P = 3$, $N = 32$, *c)* $t = 1$ Y *d)* $t = 3$, PARA $P = 20$, $N = 32$. EL ENSEMBLE DE SISTEMAS SOBRE EL QUE SE HA EFECTUADO EL PROMEDIO, CONSTA DE 512 MUESTRAS.

Como se puede apreciar de la gráfica (2.2a), aunque la red sea inicializada fuera del punto fijo, cuando los errores en las condiciones iniciales son pocos, la red tiene la capaciad de corregirlos, orientando los valores de los estados en las direcciones correctas,

siempre y cuando $\frac{P}{N}$ sea menor que un cierto valor crítico α_c. Se puede apreciar que para $\alpha < \alpha_c$ los estados del sistema corresponden a puntos fijos de la dinámica, ya que son estables ante actualizaciones sucesivas de las unidades, como lo demuestra la gráfica (2.2b). Nótese que todas estas gráficas son simétricas con respecto a la línea que denota 50% de bits incorrectos, debido a que los patrones negativos también han sido almacenados. El comportamiento arriba observado, sucederá, siempre y cuando se trabaje dentro de los límites de operación (i.e. parametros máximos y críticos), para éste caso $\alpha = 0.09 < 0.14$.

La gráfica (2.2c), muestra la **incapacidad** de la red, para retener y orientar correctamente los bits en la red neuronal, cuando $\alpha = 1.60 > 0.14$, ésta incapacidad que se observa en (2.2c), se hace más evidente para tiempos de actualización sucesivos, como lo demuestra la gráfica (2.2d) para $t = 3$, en donde la red sí es capaz de distinguir el patrón en actualizaciones iniciales, sin embargo la incapacidad de retener la información es evidente.

Se pueden efectuar cálculos más sofisticados utilizando el método de replicas simétricas (Hopfield 1982, Amit 1985a, Amit 1985b, véase capítulo III), para mostrar que efectivamente el fenómeno de avalancha ocurre cuando P es aproximadamente mayor que $0.138N$ ($\alpha_c \simeq 0.138$), haciendo la memoria inútil. De esta forma los valores encontrados por (2.25) deben ser considerados como límites superiores.

III. Mecánica Estadística de Redes Neuronales

Este capítulo iniciará presentando un modelo de red neuronal, que contempla fuentes de ruido externo, como podría ser la liberación al azar de material químico cuantizado o neuromodulador (véase capítulo 1) en las sinapsis. La inclusión de ésta nueva variable, permitirá establecer una analogía, con sitemas físicos, más específicamente, con *sistemas magnéticos desordenados* también conocidos como **vidrios de espín**. Una vez hecha la analogía, se hablará indistintamente de espines magnéticos, de neuronas o de unidades en la red neuronal. Como consecuencia de la analogía, se podrá hacer uso de una gran cantidad de trabajo teórico desarrollado para entender los sistemas magnéticos, y que ahora, puede ser utilizada para entender mejor el comportamiento de una red neuronal, cuando esta es utilizada para resolver el próblema de memoria asociativa.

3.1 Efectos del ruido externo.

Esta sección se inicia estudiando el problema de memoria asociativa, presentado en el capítulo anterior, pero combinando ideas de modelación biológica. El objetivo, será establecer una analogía con sistemas magnéticos desordenados, que resultará como consecuencia de la inclusión del ruido externo en el modelo.

El potencial h_i que alguna neurona S_i siente debido a la interacción con las demás neuronas $\{S_j\}$ puede ser definido como la suma de las contribuciones hacia ésta, expresado matemáticamente

$$h_i(t) = \sum_{j=0}^{N} J_{ij} S_j(t). \tag{3.1}$$

En donde los valores de las J_{ij}'s están determinados de acuerdo con la regla de aprendizaje de Hebb, por lo que solamente se estudiarán los efectos en la dinámica, debidos a la influencia del ruido externo. Otra consideración, será, la de igualar el valor del umbral a cero, para todas las unidades.

Así, si se considera el efecto del ruido externo, como una contribución a $h_i(t)$ de carácter estocástico y denotando a este *potencial estocástico extra*, como $h_i^{est}(t)$, se tiene, que el *potencial efectivo* que siente la neurona *post-sináptica*, definido como $h_i^{ef}(t)$, puede ser expresado como la suma de $h_i(t)$ y $h_i^{est}(t)$, ésto es:

$$h_i^{ef}(t) = h_i(t) + h_i^{est}(t). \tag{3.2}$$

El modelo consiste en considerar al potencial estocástico, como gausianamente distribuido, de acuerdo a

$$D_{est}(h_i^{est}(t)) = \frac{1}{\sqrt{2\pi\delta^2}} exp\left[-\frac{\left(h_i^{est}(t) - \bar{h}_i^{est}(t)\right)^2}{2\delta^2}\right], \tag{3.3}$$

en donde $\bar{h}_i^{est}(t)$ representa el campo o las contribuciones promedio de la variable aleatoria $h_i^{est}(t)$. Para este caso, se considerará este promedio, como cero (i.e. $\bar{h}_i^{est}(t) = 0$). Lo "ancho" de la distribución, o la desviación estandar representada por δ, es determinada por las fuentes de ruido mencionadas (como podría ser la intensidad de los potenciales estocásticos, a través de las sinapsis).

La probabilidad de que una neurona dispare o no dispare, debido a las contribuciones estocásticas, es igual a la probabilidad de que el potencial estocástico $h_i^{est}(t)$, sea más grande y de signo contrario que el campo $h_i(t)$ (debido a las contribuciones de las demás neuronas). Más explicitamente, y considerando que

$$S_i(t + \Delta t) = signo(h_i^{ef}(t)), \tag{3.4}$$

se puede apreciar que cuando $h_i(t)$ es positivo, entonces, la probabilidad de que $S_i(t+\Delta t) = +1$ puede ser expresada, como

$$Pr(S_i(t + \Delta t) = +1) = Pr(h_i^{est}(t) > -h_i(t)) = Pr(h_i(t) + h_i^{est}(t) > 0). \tag{3.5}$$

De igual forma, cuando $h_i(t)$ es negativo, entonces, la probabilidad de que $S_i(t+\Delta t) = -1$ puede ser expresada, como

$$Pr(S_i(t + \Delta t) = -1) = Pr(h_i^{est}(t) < -h_i(t)) = Pr(h_i(t) + h_i^{est}(t) < 0). \tag{3.6}$$

Utilizando (3.3) conjuntamente con (3.5) se obtiene que

$$Pr(S_i(t+\Delta t)=+1)=Pr(h_i(t)+h_i^{est}(t)>0)=\int_{h_i(t)}^{\infty} D_{est}(h_i^{est}(t))dh_i^{est}(t), \qquad (3.7)$$

$$Pr(h_i(t)+h_i^{est}(t)>0)=\int_{h_i(t)}^{\infty}\frac{1}{\sqrt{2\pi\delta^2}}exp\left[-\frac{(h_i^{est}(t))^2}{2\delta^2}\right]dh_i^{est}(t), \qquad (3.8)$$

en donde, utilizando la función de error

$$erf(x)\equiv\frac{2}{\sqrt{\pi}}\int_0^x dt\, e^{-t^2}, \qquad (3.9)$$

se obtiene

$$Pr(S_i(t+\Delta t)=+1)=Pr(h_i(t)+h_i^{est}(t)>0)=\frac{1}{2}\left[1+erf\left(\frac{h_i^{est}(t)}{\delta\sqrt{2}}\right)\right]. \qquad (3.10)$$

Asimismo, se tiene que la probabilidad de que una neurona no dispare, estará representada por

$$Pr(S_i(t+\Delta t)=-1)=1-Pr(S_i(t+\Delta t)=+1)=\frac{1}{2}\left[1-erf\left(\frac{h_i^{est}(t)}{\delta\sqrt{2}}\right)\right]. \qquad (3.11)$$

El modelo se puede simplificar de forma tal, que las dos posiblidades, (3.10) y (3.11), séan incluídas en una sola expresión. De esta forma se obtiene que la probabilidad de que una neurona se encuentre en el estado $S_i(t+\Delta)=\pm 1$, estará dada por

$$Pr(S_i(t+\Delta t)=\pm 1)=\frac{1}{2}\left[1+erf\left(\frac{h_i(t)S_i(t)}{\delta\sqrt{2}}\right)\right], \qquad (3.12)$$

en donde $h_i(t)$ está dada por la ecuación (3.1). La expresión (3.12), puede ser aproximada con un error del 1% (Amit 1989), por

$$Pr(S_i(t+\Delta t)=\pm 1)=\frac{exp(\beta h_i(t)S_i(t))}{exp(\beta h_i(t)S_i(t))+exp(-\beta h_i(t)S_i(t))}$$

$$Pr(S_i(t+\Delta t)=\pm 1)=\frac{1}{2}\left[1+tanh(\beta h_i(t)S_i(t))\right], \tag{3.13}$$

en donde

$$\beta^{-1}=2\sqrt{2}\delta. \tag{3.14}$$

Una expresión similar a (3.13) con sus correspondientes cambios en los significados de las variables, es la que describe la dinámica de un espín en un sistema de **espines de Ising** (Ising 1925) en contacto con una *fuente térmica* a temperatura $T=\beta^{-1}$ (Binder 1980). Esta dinámica es también conocida como dinámica de Glauber (Glauber 1963); ésta es la razón por la que la forma de la función (3.13) fue escogida para aproximar a (3.12), de forma tal que convenientemente se pudiera establecer una analogía directa, con la mecánica estadística de sistemas de espines de Ising.

La analogía del modelo de memoria asociativa, con el modelo magnético, puede ser mejor entendida si se considera que (Citando a Hopfield (Hopfield 1982)):

"*Cualquier sistema físico cuya dinámica en el espacio fase está dominado por un número substancial de estados locales estables, hacia los cuales el sistema es atraído, puede ser entonces considerado como una memoria general direccionada por contenido. El sistema físico será una memoria útil potencialmente, si, en adición cualquier conjunto prescrito de estados, pueden ser hechos estados estables del sistema*".

Para este caso se tiene, que un sistema físico, que cumple con las condiciones arriba mencionadas, proviene del estudio de sistemas magnéticos desordenados, área mejor conocida como **vidrios de espín**.

3.2 Vidrios de Espín. Los vidrios de espín son sistemas (físicos) magnéticos, que tienen interacciones ferromagnéticas y antiferromagnéticas aleatoriamente distribuidas. La fase de "baja temperatura" -fase de condensación-, de estos sistemas (la fase de vidrio de espín), es en muchas formas un prototipo de estudio de sistemas magnéticos desordenados con interacciones en conflicto. Estudios teóricos han demostrado, que en vidrios de espín con interacciones de largo-alcance donde cada espín interaxiona con el resto de los espines, la energía libre (energía como función del estado de los espines, o de su configuración) tiene una topología muy rica, con muchos mínimos locales, muy cercanos en energía al definido por el estado base del sistema (Morgenstern 1979, Stein 1989, Viana 1985, Viana 1989).

Las propiedades de los vidrios de espín pueden encontrarse en materiales de diferentes tipos, los cuales van desde aleaciones compuestas de metales con impurezas magnéticas

tales como AuFe, CuMn, aislantes como $Eu_xSr_{1-x}S$ o $Cd_{1-x}Mn_xTe$, hasta materiales amorfos tales como $Gd_{.37}Al_{.63}$ (Viana 1985). La universalidad en las propiedades de los vidrios de espín para diferentes materiales indica que sólo algunos ingredientes son responsables de sus propiedades, de hecho se puede decir que el desorden y la competencia de las interacciones magnéticas son comunes a todos los vidrios de espín.

Una diferencia importante entre los vidrios de espín y los sistemas magnéticos comunes, es la existencia de un número muy grande de estados estables de energía mínima del sistema. Un ejemplo de sistema magnético lo representan los ferromagnetos. En el modelo de Ising para estos materiales, sólo existen dos estados de energía mínima, éstos son, con los espines alineados hacia arriba, o hacia abajo[1]. En los vidrios de espín se descubre que existe un número infinito de tales estados de energía mínima, estos estados son en general estados **metaestables**, en el sentido de que aunque cada espín independiente "apunta" en la dirección del campo local creado por los vecinos, existen otros estados del sistema, también estables, que poseen energía menor; así, como consecuencia de las fluctuaciones térmicas, cada estado metaestable decae muy lentamente a uno de los *muchos* estados de **equilibrio térmico del sistema**.

El área de vidrios de espín ha sido mejor entendida gracias al estudio sistemático de sus modelos, de donde se han obtenido numerosos resultados. La herramienta principal que ha sido utilizada para entender el comportamiento cooperativo de estos sistemas magnéticos, ha sido la mecánica estadística. El hecho por el cual esta área es importante para el entendimiento del área de redes neuronales, es debido a que se puede establecer una analogía matemática directa (como se mencionó en la sección anterior) con modelos simplificados de redes neuronales. Lo que resta del capítulo, será dedicado a la presentación y discusión de algunos resultados obtenidos por medio de la mecánica estadística que originalmente fueron desarrollados para entender el comportamiento de vidrios de espín en sistemas de espines de Ising, y que ahora han proporcionado resultados a modelos de redes neuronales, específicamente, al problema de memoria asociativa discutido en capítulo II.

3.3 Mecánica Estadística de Redes Neuronales.

3.3.1 La conexión. Esta sección se dedicará a mostrar algunos resultados obtenidos por medio de la mecánica estadística, para vidrios de espín cuya interpretación puede ser

[1]En el resto de este capítulo, se considerarán, solamente espines de Ising, esto es, con valor discreto $S_i = \pm 1$.

aplicada a modelos de memoria asociativa como el descrito en el capítulo anterior y más complicados como los que consideran efectos de ruido externo.

La analogía Red neuronal - Sistemas magnéticos, fue puesta de manifiesto por Little (Little 1974, Little 1978), aunque fue Hopfield (Hopfield 1982), quien, estudió como tal red neuronal o sistema de espines, puede almacenar y recuperar información, encontrando que (Citando a Hopfield (Hopfield 1982)):

"*El flujo en el espacio fase del sistema, es aparentemente dominado por atractores que son las memorias asignadas nominalmente, cada una de las cuales domina un espacio substancial alrededor de ella. El flujo no es enteramente determinista, y el sistema responde a un estado ambiguo inicial, seleccionando estadísticamente entre los estados de memoria que más se le parecen.*"

Los modelos de Little y Hopfield, difieren en la forma en que la dinámica es llevada a cabo; en el modelo de Little se utiliza una dinámica del tipo paralelo, a diferencia del modelo de Hopfield, en donde se utiliza una dinámica del tipo secuencial aleatorio (véase capítulo II, sección 2.2.1).

El trabajo realizado por Hopfield, proporcionó la idea central de **función de energía**, con la cual los métodos de mecánica estadística pueden ser utilizados.

Se puede demostrar que con una actualización del tipo secuencial aleatoria, y suponiendo que las interacciones entre las neuronas son simétricas (i.e. $J_{ij} = J_{ji}$), es posible construir una **función de energía** similar al hamiltoniano utilizado en sistemas de vidrios de espín.

En este caso, la función de energía será función de la configuración $\{S_i(t)\}$ del sistema, más explícitamente

$$E[S(t)] = -\frac{1}{2}\sum_{ij} J_{ij}S_i(t)S_j(t). \tag{3.15}$$

Esta función siempre disminuye, o permanece constante al evolucionar o ser actualizados los estados de la red durante la fase de reconocimiento. Como ya se mencionó, la ecuación (3.15) puede ser identificada como el "Hamiltoniano" para un vidrio de espín, con lo que la analogía **red neuronal-vidrios de espín** a quedado establecida. Así, los *mínimos locales de energía* de la red neuronal corresponden a los estados metaestables del

vidrio de espín.

La definición de la función de energía fue posible gracias a la simetría en los valores de las interacciones ($J_{ij} = J_{ji}$). En sistemas magnéticos esto siempre es cierto, en redes neuronales biológicas, no necesariamente (véase capítulo I). De esta forma el estudio de las propiedades de los vidrios de espín con interacciones de largo alcance a partir de los métodos de mecánica estadística para sistemas en equilibrio, es equivalente al estudio de las propiedades redes neuronales con conectividad de largo alcance (todas las neuronas están conectadas con todas) y con interacciones simétricas, ya que comparten la misma forma de la función de energía. Como concecuencia, los únicos posibles estados asintóticos de la dinámica, son puntos fijos en el espacio de configuraciones, o configuraciones cuya energía es un mínimo local.

Así, los patrones almacenados son, al menos localmente, mínimos del funcional de la energía $E[S(t)]$.

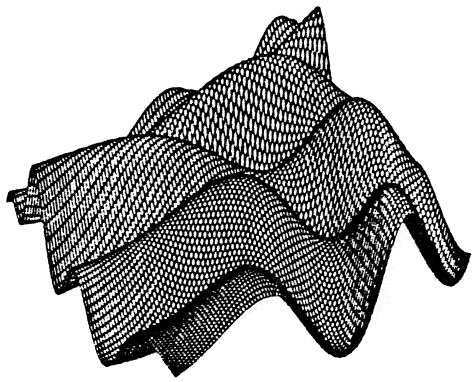


Fig. 3.1 DIAGRAMA MOSTRANDO UNA FUNCIÓN DE ENERGÍA, PICTORICAMENTE REPRESENTADA COMO UNA SUPERFICIE CON MUCHOS MÍNIMOS, AUNQUE EN LA REALIDAD, ESTA FUNCIÓN DE ENERGÍA ES UNA FUNCIÓN QUE DEBE DIBUJARSE EN UN ESPACIO N-DIMENSIONAL.

Una investigación más detallada de la funcional de energía de una red neuronal, del tipo Hopfield, muestra que además ésta tiene un número muy grande de otros mínimos locales; sin embargo todos estos **mínimos espurios** son menos pronunciados que aquellos mínimos que coinciden con los patrones almacenados $\{\xi^\mu\}$ $\mu = 1 \ldots P$ (Viana 1988, Viana 1989). Así, éstos últimos corresponden a mínimos *globales* de la superficie de energía $E[S(t)]$, al menos para valores moderados del parámetro de carga $\alpha = \frac{P}{N}$, el cual

indica el grado de utilización de la capacidad de almacenamiento. Un ejemplo de mínimos locales (mínimos espurios), no globales, está dado por las configuraciónes de la red que son combinaciones de tres patrones almacenados, de manera que algunos bits coinciden con los de cada uno de los patrones involucrados. Esquemáticamente:

$$\pm\{\xi^{\eta}\} \pm \{\xi^{\nu}\} \pm \{\xi^{\lambda}\} \quad \text{donde} \quad \{\xi^{\eta}, \xi^{\nu}, \xi^{\lambda}\} \in \{\xi^{\mu}\} \ \mu = 1 \ldots P. \tag{3.16}$$

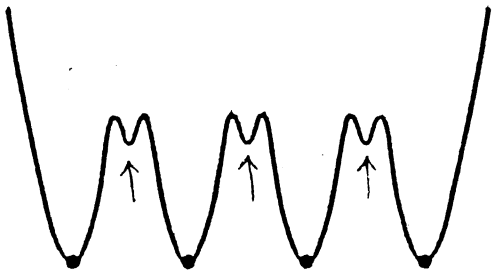


Fig. 3.2 DIAGRAMA ESQUEMÁTICO MOSTRANDO LOS ESTADOS ESPURIOS COMO FUNCIÓN DE LA ENERGÍA LIBRE DEL SISTEMA CUANDO $P << N$ ($i.e.\ \alpha \to 0$), ESTOS ESTADOS ESPURIOS PUEDEN SER VISUALIZADOS COMO MEZCLAS DE LOS PATRONES ALMACENADOS, REPRESENTADOS EN ESTA FIGURA POR LOS MÍNIMOS LOCALES QUE SE ENCUENTRAN ENTRE LOS PATRONES ALMACENADOS Y QUE ESTAN SEÑALADOS POR FLECHAS.

3.3.2 Resultados de la Mecánica Estadística del Modelo de Hopfield.

Las propiedades del modelo de Hopfield utilizando la regla de aprendizaje de Hebb simétrica, fueron estudiadas sistemáticamente mediante los métodos de mecánica estadística para sistemas en equilibrio, considerando el límite termodinámico (número infinito de neuronas) y un número finito de patrones aprendidos (Amit *et. al.* 1985a), i.e $\alpha = 0$. Posteriormente, fue hecho el estudio considerando un número infinito de neuronas y un número infinito de patrones aprendidos (Amit *et. al.* 1985b) como función del **parámetro de carga** $\alpha \neq 0$. A su vez el estudio fue hecho para diferentes niveles ruido T, que equivale a la temperatura en la función de distribución de probabilidades de Boltzman. Los resultados obtenidos, se pueden resumir de la siguiente forma (Ver por ejemplo (Sompolinsky 1988)):

Resultados para $N \to \infty$, P *finita*, con $(0 < T < 1)$. Todos los patrones almacenados, están relacionados con estados base de energía del sistema (En donde $T = 1$ es la temperatura crítica o nivel de ruido crítico, abajo del cual estados ordenados aparecen). Para $0.46 < T < 1$, los patrones almacenados, son los únicos estados estables del sistema. Las fluctuaciones térmicas reducen la probabilidad de que la red quede atrapada en un estado espurio o configuración local metaestable no deseada; conforme se "baja" el nivel de ruido ($T < 0.46$), comienzan a aparecer estados **espurios**. Los estados espurios consisten en "mezclas simétricas" de $2P+1$ patrones almacenados. Conforme el nivel de ruido tiende a cero ($T \to 0$), el número de los estados espurios crece exponencialmente con el número de patrones P.

Resultados para $N \to \infty$, $P \to \infty$, con $(0 < T < 1)$. En este caso, el modelo de Hopfield para una Red Neuronal es estudiado en el límite en que el número de P patrones almacenados se incrementa con el número de neuronas N en la red, como $P = \alpha N$. Para valores finitos de α, se encuentra que al crecer α, la suma de los traslapes de los patrones generados al azar comienza a contribuir importantemente a la función de energía ver ecuación (2.21), hasta que eventualmente desestabiliza a los patrones almacenados. Para $0 < \alpha, \alpha_c \simeq 0.138$ tenemos que un pequeño porcentaje de los bits no puede almacenarce correctamente, este porcentaje aumenta lentamente conforme α crece hasta que para un valor de α cercano a 0.138 existe un punto crítico después del cuál no es posible la recuperación de ninguno de los patrones almacenados.

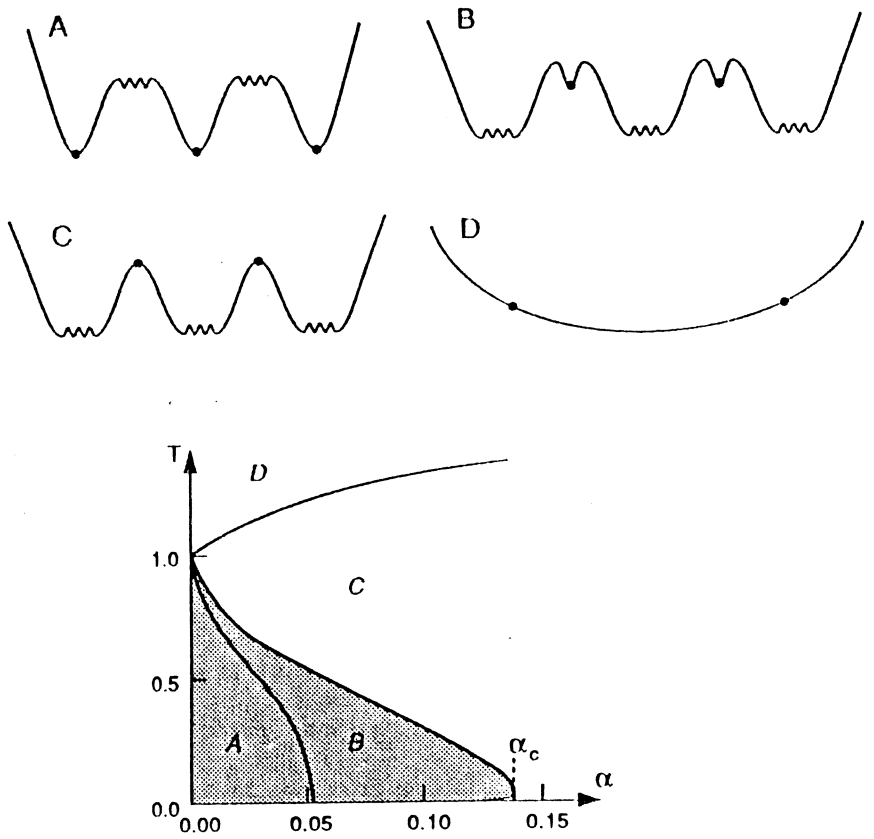


Fig. 3.3 DIAGRAMA DE FASES, OBTENIDO POR AMIT *et.al.* (AMIT *et. al.* 1985*b*), LOS PATRONES ALMACENADOS, SON ESTABLES ÚNICAMENTE EN LA REGIÓN SOMBREADA. EN LA FIGURA SUPERIOR SE MUESTRA UNA SERIE DE DIAGRAMAS, DESCRIBIENDO EL FUNCIONAL DE ENERGÍA CORRESPONDIENTE A DIFERENTES PARTES DEL DIAGRAMA DE FASES.

Otro resultado importante, obtenido por Amit *et.al.* (Amit *et.al.* 1985a), fue demostrar que cuando la dinámica de la red se aproxima asintóticamente a un estado estacionario, la forma como se lleve a cabo la dinámica ó la actualización de los estados de las unidades (i.e. actualización secuencial o paralela), se va haciendo cada vez más irrelevante, estableciendo así, una equivalencia entre el modelo de Little y el modelo de Hopfield.

IV. Simulación Numérica del Modelo de Hopfield

En este capítulo, se presentan simulaciones computacionales del modelo de Hopfield (Hopfield 1982), utilizando un sistema de codificación múltiple de espines, introducido por Penna y Oliveiria (Penna 1989). Este capítulo, puede considerarse como una revisión de dicho algoritmo y de su implementación.

El capítulo inicia, adaptando el modelo de Hopfield para hacer cálculos en un sistema de tamaño finito. Esto se hará con el fin de que el método numérico utilizado pueda entenderse mejor, ya que es técnicamente complicado. Una vez entendida la estrategia a seguir, definida por la nueva versión del modelo, se proseguirá a implementar el algoritmo computacional. Este algoritmo es mejor conocido con el nombre de *Codificación Múltiple de Espines*. El resto del capítulo es dedicado a explicar el algoritmo computacional con más detalle, así como a discutir los posibles cambios en el algoritmo, de forma tal que pueda ser implementado en una computadora digital, de la forma más eficiente.

4.1 El modelo de Hopfield para un sistema de tamaño finito. Como se ha mostrado durante los primeros capítulos de la tesis, la evolución dinámica de los estados de cada una de las unidades en el modelo de Hopfield, se puede representar, fundamentalmente, por la ecuación:

$$S_i(t + \Delta t) = signo\big(h_i(t)\big), \tag{4.1}$$

en donde

$$h_i(t) = \sum_{j=1}^{N} J_{ij} S_j(t), \tag{4.2}$$

representa el campo que siente la unidad i al tiempo t debido a su interacción con el resto de las neuronas. En esta última ecuación el término J_{ij} esta dado por la regla de aprendizaje de Hebb

$$J_{ij} = \frac{1}{N} \sum_{\mu=1}^{P} \xi_i^\mu \xi_j^\mu, \tag{4.3}$$

que representa la matriz de conocimiento (matriz de interacciones). Regresando a la ecuación (4.1) se puede notar que, para efectos de simulación computacional, la

normalización de los valores que $h_i(t)$ puede adquirir no es importante; lo único importante es saber si la cantidad es positiva o negativa. En esta sección, por el contrario, se impondrán límites superiores e inferiores a los posibles valores que la cantidad $h_i(t)$ puede adquirir (i.e. se normalizará). De esta forma, se tiene que si se quiere que $h_i(t)$ este restringido a adquirir **valores entre** -1 **y** $+1$, lo que se tiene que hacer es normalizar (4.2) y (4.3) por los factores correctos. Para este caso y empezando con (4.3), si se desea que J_{ij} tome valores entre -1 y $+1$, se tiene que normalizar por $1/P$. Sabiendo que la cantidad en (4.3) está ya normalizada, el factor de normalización en (4.2) quedaría como $1/N$ en el límite termodinámico. En este punto debe notarse que si en la ecuación (4.3) se define $J_{ij} = 0$ cuando $i = j$, entonces la suma sobre j en (4.2) tendrá un cero, por lo que el factor de normalización en (4.2) será $1/(N-1)$, de forma tal que las auto-interacciones igualadas a cero sean contempladas y no afecten el nuevo intervalo. Así, los posibles valores de $h_i(t)$, con los nuevos factores de normalización, tomarán valores entre -1 y $+1$. Expresando con $h_i'(t)$ el nuevo valor del campo, se obtiene

$$h_i'(t) = \frac{1}{(N-1)} \sum_{j=1}^{N} \frac{1}{P} \sum_{\mu=1}^{P} \xi_i^\mu \xi_j^\mu S_j(t), \tag{4.4}$$

donde $-1 \leq h_i'(t) \leq +1$ y $J_{ij} = 0$ *para* $i = j$, nótese que $S_i(t+1) = signo(h_i(t)) = signo(h_i'(t))$, ya que $S_i(t+1)$ solamente depende del signo de los campos y es independiente de su normalización.

Es importante considerar el factor de normalización $\frac{1}{N-1}$, ya que trabajaremos en muestras de tamaño finito.

4.2 Codificación múltiple de espines. La codificación múltiple de espines, es un algoritmo computacional que se utiliza para optimizar recursos computacionales, más específicamente memoria dinámica y tiempo de "CPU". En el caso del modelo de Hopfield que fue explicado en la sección 4.1, la codificación consistirá en cambiar el estado de una neurona $+1$ y -1 en el modelo tradicional de Hopfield, por 0 y 1 respectivamente. De esta forma se puede utilizar un bit de la memoria de la computadora, para almacenar el estado de la neurona en lugar de los 8 bit's que generalmente se requieren. Así, al requerirse un menor número de bits para almacenar los estados de las unidades y los patrones almacenados, se pueden hacer simulaciones de sistemas mucho más grandes. Otro cambio que se debe implementar es el de substituir la multiplicación aritmética en el modelo de Hopfield por la operación lógica XOR (ó exclusivo), esta última operación estará representada por

$\otimes$. En la siguiente sección, se explica la forma como la codificación múltiple de espines fue llevada a cabo en el modelo de Hopfield.

MULTIPLICACION ARITMETICA	OPERACION XOR
$(+1)\cdot(+1) \rightarrow +1$	$(0) \circledast (0) \rightarrow 0$
$(+1)\cdot(-1) \rightarrow -1$	$(1) \circledast (0) \rightarrow 1$
$(-1)\cdot(+1) \rightarrow -1$	$(0) \circledast (1) \rightarrow 1$
$(-1)\cdot(-1) \rightarrow +1$	$(1) \circledast (1) \rightarrow 0$

Tabla 4.1 TABLA MOSTRANDO LA EQUIVALENCIA ENTRE LA OPERACIÓN DE MULTIPLICACIÓN ARITMÉTICA LOS VALORES $+1$ Y -1, Y LA OPERACIÓN BINARIA LÓGICA XOR CON LOS NUEVOS VALORES 0 Y 1.

4.3 Modelo de Hopfield codificado. En esta sección se incorporarán las ideas de codificación múltiple de espines descrita en la sección 4.2, al modelo de Hopfield descrito en la sección 4.1. Para tal efecto, se puede apreciar que los posibles valores de los patrones a aprender $\xi_i^\mu = +1$ y $\xi_i^\mu = -1$ han sido cambiados por $\zeta_i^\mu = 0$ ó $\zeta_i^\mu = 1$, respectivamente. De igual forma se obtiene que las nuevas condiciones iniciales $S_i(t) = +1$ y $S_i(t) = -1$ serán representadas, por $S_i^{\otimes}(t) = 0$ ó $S_i^{\otimes}(t) = 1$ respectivamente[1].

Como ya mencionamos en la sección anterior, la operación aritmética de multiplicación ha sido substituída por la operación lógica XOR (ó exclusivo) representada por $\otimes$. Efectuados estos cambios en el modelo de Hopfield, se puede apreciar que básicamente lo que se hizo fue mapear el dominio de los posibles valores de los estados y de los patrones almacenados. Siguiendo en esta misma dirección, la estrategia que se seguirá será la de restringir al intervalo [0,1] los posibles valores que el campo puede adquirir. Para este efecto, se empezará por encontrar la nueva regla de aprendizaje de Hebb, que se adecúe a este nuevo contexto, para lo cual, como ya se mencionó, la variable tradicional $\xi_i^\mu = \pm 1$ ha sido cambiada, por los ahora posibles valores $\zeta_i^\mu = 0$ ó $\zeta_i^\mu = 1$. Más explicitamente, se obtiene que la nueva representación de la regla de aprendizaje, puede ser expresada como

[1] En este caso se representará con el super-índice $\otimes$ a las nuevas variables correspondientes al modelo de Hopfield.

$$J_{ij}^{\otimes} \propto \sum_{\mu=1}^{P} \zeta_i^{\mu} \otimes \zeta_j^{\mu}. \tag{4.5}$$

En esta misma ecuación se puede apreciar que el factor de normalización apropiado, de forma tal que los valores de $J_{ij}^{\otimes}$ estén restringidos a valores entre $0\,y\,1$, será $1/P$.

Una vez definida la nueva regla de aprendizaje, se procederá a utilizar ésta, en la nueva expresión del campo que siente la neurona i debido a las demás neuronas, así, se tiene que el campo, estará expresado, por

$$h_i^{\otimes}(t) = \sum_{j=1}^{N} J_{ij}^{\otimes} S_j^{\otimes}(t). \tag{4.6}$$

Substituyendo (4.5) en (4.6) y añadiendo el factor de normalización apropiado, se obtiene

$$h_i^{\otimes}(t) \propto \sum_{j=1}^{N} \frac{1}{P} \sum_{\mu=1}^{P} \zeta_i^{\mu} \otimes \zeta_j^{\mu} \otimes S_j^{\otimes}(t). \tag{4.7}$$

De esta ecuación se puede apreciar que el factor de normalización necesario, de forma tal que el resultado $h_i^{\otimes}(t)$ esté entre 0 y 1, es N, ésto es, si se consideran las auto-interacciones (i.e. $J_{ij} \neq 0$ si $(i = j)$ en el modelo tradicional de Hopfield). Sin embargo, si se considera el caso en que $J_{ii} = 0$, de la ecuación (4.7), se puede apreciar que en el producto lógico dentro de las sumas $\zeta_i^{\mu} \otimes \zeta_j^{\mu} \otimes S_j^{\otimes}(t)$ existirá un 1 de más si $S_j^{\otimes}(t) = 1$, por lo que se deberá normalizar por $1/(N+1)$. De igual forma se puede apreciar que si $S_j^{\otimes}(t) = 0$ entonces en la suma existirá un 1 de menos, por lo que en este caso se debe de normalizar por $1/(N-1)$. Utilizando los nuevos factores de normalización en (4.7), se tiene que la nueva ecuación que describe el campo quedará exprezada como

$$h_i^{\otimes}(t) = \frac{1}{N^{\otimes}} \sum_{j=1}^{N} \frac{1}{P} \sum_{\mu=1}^{P} \zeta_i^{\mu} \otimes \zeta_j^{\mu} \otimes S_j^{\otimes}(t), \tag{4.8}$$

en donde $N^{\otimes} = (N+1)$ o $N^{\otimes} = (N-1)$ dependiendo de si $S_j^{\otimes}(t) = 1$ o $S_j^{\otimes}(t) = 0$ respectivamente. Estos factores de normalización asegurarán que el valor del campo tome un valor entre 0 y 1 con lo que solamente restaría hacer la comparación de $h_i^{\otimes}(t)$ con el

valor de 1/2, si el campo es mayor que este valor entonces $S_i^{\otimes}(t+\Delta t)=1$ y si el valor del campo es menor que 1/2, entonces $S_i^{\otimes}(t+\Delta t)=0$, si el valor del campo es igual a 1/2, se optará por $S_i^{\otimes}(t+\Delta t)=S_i^{\otimes}(t)$ (dejar el valor del estado del espín, sin cambio).

4.4 Implementación computacional. En las secciones anteriores, solamente se han renombrado las variables, se han especificado los posibles valores que éstas pueden tomar y se han introducido nuevas operaciones entre ellas. En esta sección se mostrará como estos nuevos cambios, especificamente representados en la ecuación (4.8), pueden ser utilizados para una simulación eficiente del modelo de Hopfield en términos de recursos de computadora. Para tal efecto, se partirá de la Ecuación (4.8) , multiplicandola por un factor de 2, de tal forma, que ahora el intervalo del campo $h_i^{\otimes}(t)$ será $[0,2]$, de esta forma, la comparación del valor del campo, se hará con el valor de 1, en lugar de 1/2 como en la ecuación (4.8). Así el nuevo valor del campo, estará representado por la siguiente ecuación

$$h_i^{\otimes}(t)=\frac{2\cdot\sum_{j=1}^{N}\sum_{\mu=1}^{P}\zeta_i^{\mu}\otimes\zeta_j^{\mu}\otimes S_j^{\otimes}(t)}{PN^{\otimes}}. \tag{4.9}$$

Una vez encontrado el valor del campo, se procede actualizando el estado de la neurona i, de la siguiente forma, si $h_i^{\otimes}(t)>1$ entonces $S_i^{\otimes}(t+\Delta t)=1$ y si $h_i^{\otimes}(t)<1$ entonces $S_i^{\otimes}(t+\Delta t)=0$); por último, si $h_i^{\otimes}(t)=1$ entonces se optará por dejar el estado de la unidad con el mismo valor.

La ecuación (4.9), permitirá optimizar tiempo de computadora, en el sentido de que permitirá hacer una división entera, esto es, una división en la que es posible hacer una comparación del numerador con el denominador; más explicitamente, permitirá saber el resultado 1 ó 0, si el valor del campo es mayor o menor que 1 respectivamente, o para este caso, sabiendo si el numerador es mayor o menor que el denominador. Esta última operación, consume menos tiempo de "CPU" que lo que consumiría la operación de división aritmética.

De la explicación anterior podría parecer que el implementar la división entera no optimiza mucho los recursos computacionales, sin embargo la operación entera, conjuntamente con algunas otras mejoras que se explicarán a continuación, efectivamente optimizan tiempo de computo (tiempo de "CPU") y memoria dinámica, permitiendo así, efectuar simulaciones del modelo de Hopfield con un número grande de unidades.

Siguiendo con la indicaciones señaladas en el artículo de Penna y Oliveiria (Penna

1989), se puede apreciar, en la ecuación (4.9), que el orden de las sumas puede ser cambiado sin alterar el resultado para el campo. Este cambio convierte (4.9) en

$$h_i^{\otimes}(t) = \frac{2 \cdot \sum_{\mu=1}^{P} \sum_{j=1}^{N} \zeta_i^{\mu} \otimes \zeta_j^{\mu} \otimes S_j^{\otimes}(t)}{PN^{\otimes}}, \tag{4.10}$$

de donde se puede apreciar que una forma alternativa, de efectuar la operación

$$\sum_{j=1}^{N} \zeta_i^{\mu} \otimes \zeta_j^{\mu} \otimes S_j^{\otimes}(t). \tag{4.11}$$

Esta forma alternativa, consiste en efectuar primero la operación $\zeta_j^{\mu} \otimes S_j^{\otimes}(t)$ para las N unidades j y contar la cantidad de 1's que han resultado de esta última operación, ya que éste sería el resultado de la suma sobre j. Se continúa comparando el valor de ζ_i^{μ}: si $\zeta_i^{\mu} = 0$, entonces el resultado de la suma (4.11) sería el número de 1's encontrados en la suma sobre j. Por otro lado, si $\zeta_i^{\mu} \neq 0$ entonces el resultado de (4.11) sería el número de unidades N menos el número de 1's encontrados en la suma sobre j. La forma alternativa de realizar la suma sobre j en (4.11) permite efectuar, **simultaneamente**, la operación XOR para 32 unidades (utilizando un procesador de 32-bit's de palabra entera sin signo), lo que reditúa en una reducción del tiempo de computo y de memoria dinámica (RAM) utilizada. Finalmente, esta misma operación es efectuada para las restantes $\frac{N}{32}$ palabras enteras sin signo, de forma tal, que todas las neuronas, sean tomadas en cuenta. Una vez terminada esta operación, se prosigue procesando la información referente a el siguiente patrón; la operación se repite, un número de veces igual al número de patrones que se desea almacenar, más específicamente, con el resto de los P patrones, de forma tal, que todas las unidades, y todos los patrones, son tomados en cuenta. El resultado de cada una de las P operaciones es finalmente sumado, obteniéndose el equivalente a las sumas en el numerador de (4.10). Se prosigue multiplicando por 2, y posteriormente se lleva a cabo la comparación numerador-denominador de forma tal que se actualice el estado del espin $S_i^{\otimes}(t + \Delta t)$. En este punto, se puede decir que el tiempo de cómputo crece proporcional al producto del número de patrones por el número de unidades. Una observación importante es el notar que con este nuevo algoritmo no se tiene una matriz de interacciones con N^2 elementos por lo que la única memoria utilizada es practicamente para almacenar los valores de los patrones a aprender y las condiciones iniciales. Una última mejora que se puede hacer, como lo discute en su artículo G.A. Kohring (Kohring 1990), es calcular los productos $\zeta_j^{\mu} \otimes S_i^{\otimes}(t)$ (**antes** de empezar la actualización) de cada unidad y guardar

estos productos en memoria dinámica (RAM). Después de iniciada la actualización, estos productos deberán ser modificados, si y sólo si, el estado de una neurona necesita ser actualizado. Esta última mejora hará que el programa sea del orden $O(N)$ más rápido, sin embargo hay que hacer notar que los requerimentos de memoria dinámica son del orden $O(\alpha N^2)$ bits.

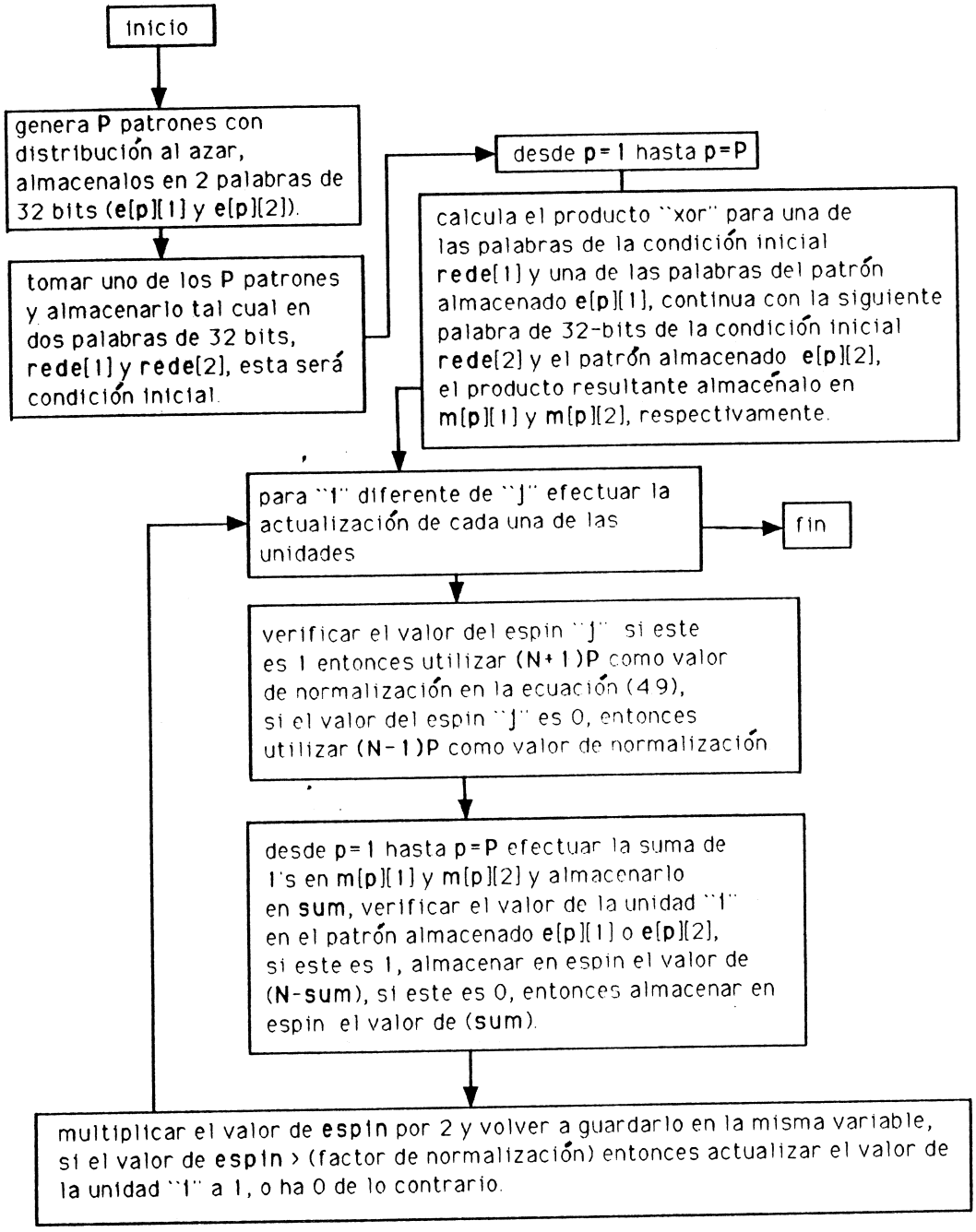


Fig. 4.1 DIAGRAMA DE FLUJO CORRESPONDIENTE A LA SIMULACIÓN DE UNA RED NEURONAL, CON N=64 UNIDADES Y UN NÚMERO VARIABLE DE P PATRONES, DE ACUERDO AL ALGORITMO DE PENNA Y OLIVEIRA.

El figura (4.1), se muestra el diagrama de flujo correspondiente al algoritmo com-

putacional de Penna y Oliveiria (Penna 1989), diseñado para aprender un número variable de patrones, en una red neuronal con un número constante de 64 neuronas. Los resultados de la simulación, para esta red, se muestran en las gráficas (5.5a) y (5.6) (véase siguiente capítulo). Para detalles técnicos, refierase al apendice A, en donde se ha incluído el código del programa utilizado, específicamente para esta red.

4.5 Especificaciones del programa utilizado. Para simular el modelo de Hopfield, se implementó el algoritmo computacional de Penna y Oliveira (esquematicamente descrito en la sección anterior). Las simulaciones computacionales, se hicieron en una computadora "*Sun Sparc Station*", que tiene tamaño de palabra entera sin signo de 32-bit's, y sistema operativo *UNIX* (Kerningam 1984), todo el código del programa fue desarrollado en lenguaje de programación "C", para el cual existen los compiladores (Kerningam 1978) más eficientes.

V. Implementación del Algoritmo de Penna y Oliveira

5.1 Objetivo de las simulaciones. El objetivo de este capítulo, será el de implementar el algoritmo computacional de Penna y Oliveira para el modelo de Hopfield explicado en el capítulo IV, así como el de utilizar el método de análisis señal a ruido desarrollado en el capítulo II, para interpretar los resultados obtenidos. Se programaran una serie de simulaciones computacionales, con este fin.

El plan de trabajo para desarrollar las simulaciones, será de la siguiente forma:

1) Implementar el algoritmo computacional de Penna y Oliveira, para simular una red de 32 unidades, la cual ha sido entrenada para aprender un número variable de patrones y la cual ha utilizado uno de los patrones aprendidos como condición inicial (sin bits equivocados), esto con el objeto de comprobar si los patrones se almacenaron correctamente. El tamaño de la red se escogió de 32 unidades, ya que este es el tamaño de palabra entera sin signo en el procesador de la computadora en donde se llevaron a cabo las simulaciones. La simulación arriba mencionada, se efectuará para diferentes tiempos t de actualización de la red. Se utilizará el análisis señal a ruido para interpretar los resultados.

2) Comprobar que la red de 32 unidades, además de almacenar los patrones correctamente, es capaz de corregir errores en la condición inicial, más específicamente, verificar que los patrones almacenados tienen una cuenca de atracción alrededor del estado correspondiente al patrón almacenado (véase capítulo II), de tal forma que para valores de P y N adecuados, la red es capaz de corregir errores en las condición inicial. Para tal efecto, se efectuarán simulaciones con un número constante de patrones aprendidos, en las que se considerará a la condición inicial, como uno de los patrones aprendidos, pero con un número determinado de bits equivocados. La simulación arriba mencionada, se efectuará para diferentes tiempos t de actualización de la red.

3) Observar el desempeño de la red, considerando un número creciente de unidades N y de patrones almacenados P, utilizando como condición inicial uno de los patrones aprendidos sin bits equivocados, para diferentes tiempos t de actualización de la red.

5.2 Simulaciones para una red de 32 unidades con un número variable de patrones a aprender, sin bits equivocados en la condición inicial. La forma como se procederá a verificar el funcionamiento correcto del método numérico, será partiendo del hecho de que las simulaciones fueron efectuadas en una computadora con palabra entera

sin signo de 32-bits, por lo que se procedió a efectuar la simulación para una red de 32 unidades. La forma como se llevó a cabo esta simulación, fue utilizando como condición inicial de la dinámica de la red, uno de los patrones almacenados (sin errores); se dejó que los estados de cada uno de los espines de la red fueran actualizados de acuerdo al algoritmo explicado en el capítulo IV, utilizando como variable independiente, el número de patrones P a almacenar. El resultado de la evolución dinámica de la red neuronal, se puede resumir en la figura (5.1), en donde se muestra el número promedio[1] de bits equivocados después de una actualización ($t = 1$) como función del número de patrones almacenados (eje horizontal inferior) y del parámetro de carga de información α (eje horizontal superior).

Así, la figura (5.1), muestra una serie de puntos, cada uno de los cuales representa, el número promedio de bits equivocados durante la fase de recuperación después de una actualización ($t = 1$). El promedio se efectuó para un total de 1024 simulaciones por punto. La figura (5.1) también muestra a la variable independiente como el número de patrones a almacenar, y la variable dependiente el promedio de bits recuperados incorrectamente para $t = 1$; en el eje superior se ha graficado el valor de α, que para este caso, es proporcional al número de patrones almacenados, tomando en cuenta que el número de unidades permaneció constante durante toda la simulación. En el eje lateral derecho, se ha graficado el error obtenido, expresado en porcentaje de bits incorrectos; este error es proporcional al número de bits recuperados incorrectamente y al tamaño de la red.

[1] Estos promedios consideraron un número de condiciones iniciales diferentes pero equivalentes, para este caso en especial, las barras de error fueron calculadas tomando en cuenta, 1024 estados de la red al tiempo $t = 1$, y son una medida de que tan dispersos quedaron éstos, de su valor promedio, en unidades de bits incorrectos (desviación estándard).

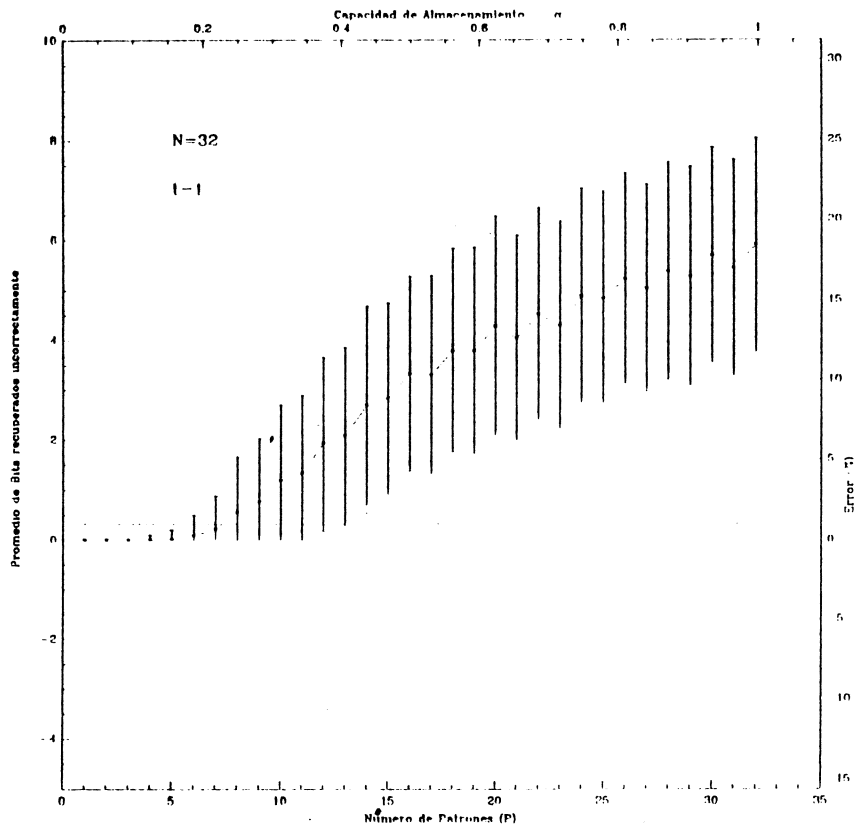


Figura 5.1 GRÁFICA MOSTRANDO EL NÚMERO PROMEDIO DE BITS EQUIVOCADOS EN EL ESTADO DE UNA RED NEURONAL DE 32 UNIDADES, AL TIEMPO $t = 1$, LA CUAL A SIDO ENTRENADA CON UN NÚMERO VARIABLE DE PATRONES APRENDIDOS.

En la figura (5.1), la linea horizontal punteada corresponde a un error del 1% en la recuperación; así, para el caso de 32 unidades, este error corresponderá a 0.32 bits equivocados. De esta forma, se puede apreciar que para valores de P igual ó menor que 4 ($\alpha < 0.125$), la mayoría de las veces que se llevó a cabo la dinámica de la red, al tiempo $t = 1$ no obtuvo errores que excedieran el 1%.

Así, se puede apreciar que cuando la red fue entrenada con 3 patrones ($\alpha \simeq 0.93$), ésta no obtuvo ningún error en la recuperación; ésto es, en promedio la red reconoció y mantuvo estables, todos los bits del patrón, la mayoría de las 1024 veces. Esto se puede apreciar de la gráfica ya que el punto correspondiente, que representa el promedio, esta ubicado en cero errores y no tiene barras de error. De la misma figura , se puede observar, que

cuando la red fue entrenada con 32 patrones, la red incurrió en un error que en promedio tuvo 6 bits equivocados; de aquí se puede observar además, que el punto correspondiente tiene barras de error asociadas (en unidades de bits equivocados), indicando, que tan distribuídos estuvieron los bits equivocados al tiempo $t = 1$, con respecto al promedio.

Los resultados arriba mencionados, también ponen de manifiesto la efectividad del análisis señal a ruido al predecir muy acertadamente, los valores límite para P_{max} obtenidos en esta simulación (véase capítulo II).

La figura (5.2), muestra una serie de 4 gráficas que serán interpretadas de la misma forma que en la figura (5.1), con la única diferencia de que ahora, las gráficas corresponden a tiempos diferentes, ésto es, una vez que todas las unidades de la red han sido actualizadas, se procede a utilizar el estado al tiempo $t = 1$ una vez más, pero ahora, como condición inicial. Este proceso es repetido 4 veces más, generando así, 4 gráficas correspondientes a actualizaciones sucesivas (i.e. $t = 2, 3, 4$ y $t = 5$).

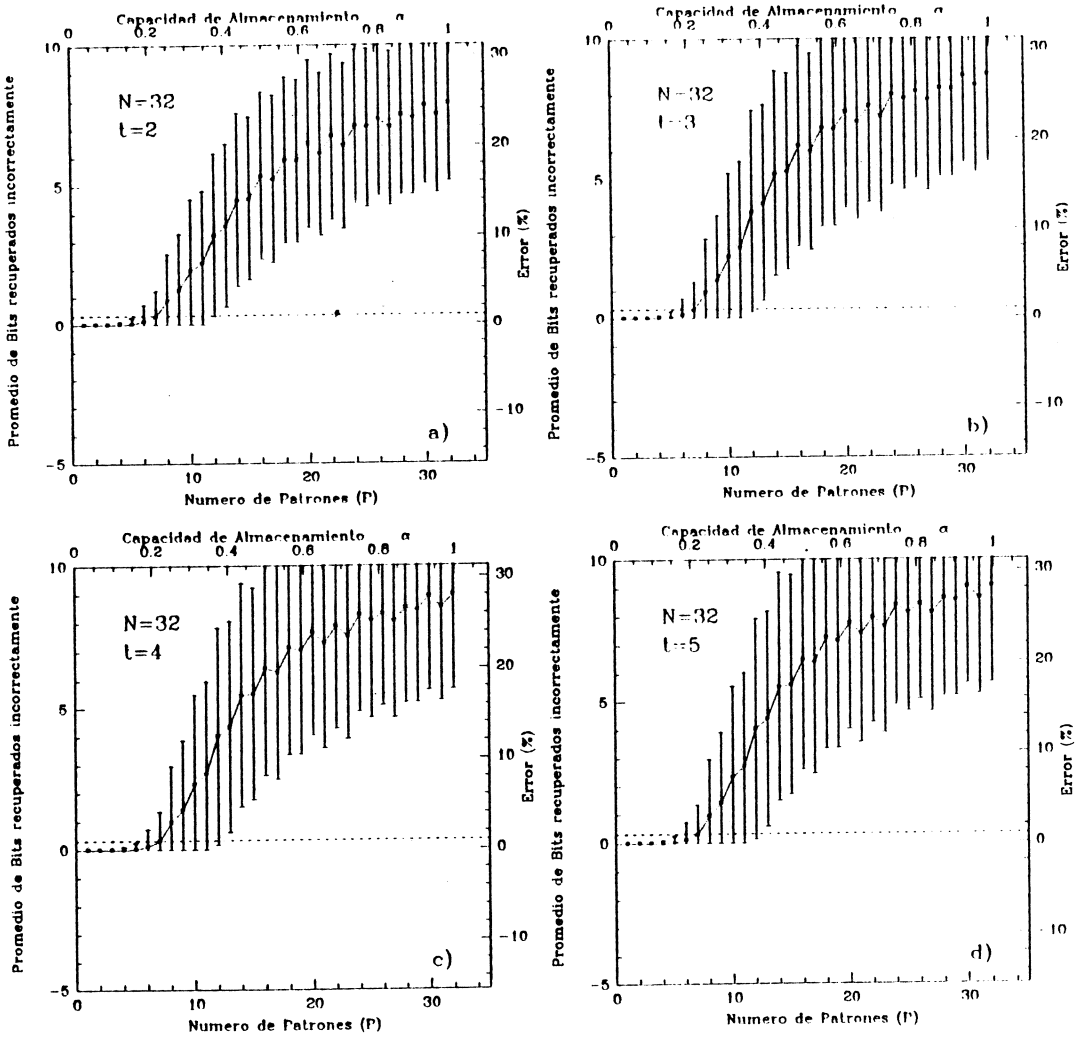


Figura 5.2 SERIE DE GRÁFICAS MOSTRANDO EL NÚMERO PROMEDIO DE BITS EQUIVOCADOS AL TIEMPO t, PARA. *a*) $t = 2$, *b*) $t = 3$, *c*) $t = 4$, *d*) $t = 5$, PARA UNA RED NEURONAL DE 32 UNIDADES, LA CUAL A SIDO ENTRENADA, CON UN NÚMERO VARIABLE P DE PATRONES APRENDIDOS.

En la figura (5.2), también se puede apreciar la avalancha de errores, que ocurre, cuando el valor del número de patrones es mayor que un cierto valor crítico de $P_{max} = 5$ (para $N = 32$), una vez que la red ha sido actualizada sucesivamente, para tiempos $t = 2, 3, 4, 5$. De ésta figura, se puede apreciar que para valores de $P > P_{max} = 5$, algunos patrones empiezan a ser recuperados con algunos errores que exceden el 1%, al menos, una vez que todas las unidades han sido actualizadas 5 veces, de tal forma, que la red, no retiene la información correctamente. De esta última observación, se puede inferir, que para valores de P mayores o iguales a 5, se obtendrán errores en la recuperación, mayores al 1% y eventualmente, si se sigue actualizando la red neuronal, un fenómeno de avalancha, se presentará, en donde los estados al tiempo t no guardarán, nada, o casi nada de parecido, con la condición inicial. Así, al ser actualizada la red, una y otra vez, los únicos valores que permanecerán invariantes, serán los que corresponden a puntos fijos definidos para cuando el número de patrones es igual o menor que 4, ésto garantiza que los errores obtenidos, serán menores que el 1% como se puede apreciar en la gráfica.

Con esta simulación quedó de manifiesto, el funcionamiento del algoritmo para la red descrita con los parámetros arriba mencionados, así como la habilidad de ésta para almacenar, reconocer y retener un número variable de patrones sin ruido.

Observaciones. En este punto, es importante remarcar el hecho de que las simulaciones, fueron hechas para tiempos finitos de actualización, en las que el tiempo más grande de actualización de la red fue $t = 5$. De forma tal que en estas simulaciones, no fueron contemplados los efectos de seguir actualizando la red para tiempos mayores. Para este ejemplo en particular, en el que se inicializa al estado de la red en un punto fijo de la dinámica, y en el que se estudia su evolución, se puede inferir, que una vez que la red empieza a producir errores en el estado de la red al tiempo $t = 1$, estos seguirán produciendose en actualizaciones sucesivas.

5.3 Simulaciones para una red de 32 unidades con un número constante de patrones por aprender, considerando como condición inicial, uno de los patrones almacenados con un número de bits incorrectos. El objetivo de estas simulaciones, será el de verificar que los patrones almacenados tienen una cuenca de atracción alrededor del estado del patrón almacenado (véase capítulo II), de tal forma que para valores de P y N adecuados, la red es capaz de corregir errores en las condición inicial. Para tal efecto, se efectuaron simulaciones con un número constante de patrones aprendidos, en las que se consideró como condición inicial a uno de los patrones aprendidos pero con un número

determinado, de bits equivocados. El estado de la red se verificó para diferentes tiempos t de actualización de ésta.

A continuación, se presenta una serie de simulaciones realizadas con un número variable de errores en la condición inicial y cuyos resultados se presentan en la figura (5.3). En la figura, se puede apreciar una serie de 4 gráficas en las cuales se contempla la evolución temporal del estado de la red para 10 condiciones iniciales distintas cada una de las 40 evoluciones temporales que se presentan, fueron efectuadas, desde el tiempo $t = 0$ hasta el tiempo $t = 20$, y representan en su conjunto, un total de 800 simulaciones, representadas por un punto cada una de ellas.

En la gráfica (5.3a), se muestran 10 evoluciones temporales, llevadas a cabo con un número variable de errores en la condición inicial cada una de ellas, en donde se puede apreciar que para $\alpha = 0.03$ como es el caso de (5.3a), el estado de la red para t suficientemente grande será independiente del número de errores en la condición inicial, reteniendo, reconociendo y corrigiendo errores en la condición inicial y recuperando así el patrón almacenado sin errores en el primer paso de actualización.

Para el caso en que $\alpha = 0.12$ como es el caso de (5.3b), se puede apreciar que la mayoría de las 10 evoluciones temporales, terminaron en el patrón almacenado, sin ningún error, mostrandose así, la capacidad de la red para corregir errores en la condición inicial. Debe notarse sin embargo, que 2 de las evoluciones, no terminaron en el patrón almacenado, quedando "estancadas" en un estado estable de la dinámica, pero que no corresponde al patrón almacenado, a este patrón se le conoce como memoria "espuria" (véase capítulo III).

Para el caso en que $\alpha = 0.25$ como es el caso de (5.3c), se puede apreciar que 6 de las evoluciones temporales, no terminaron en ninguno de los patrones almacenados, quedando así atrapadas en estados estables de la dinámica ó memorias espurias.

Para el caso en que $\alpha = 0.37$ como es el caso de (5.3d), se puede apreciar que la mayoría de las 10 evoluciones temporales, además de que ya no pueden retener la información almacenada, degradan esta de una actualización a otra, terminando así, para tiempos suficientemente grandes, en estados estables que no representan al patrón a recuperar ó memorias espurias.

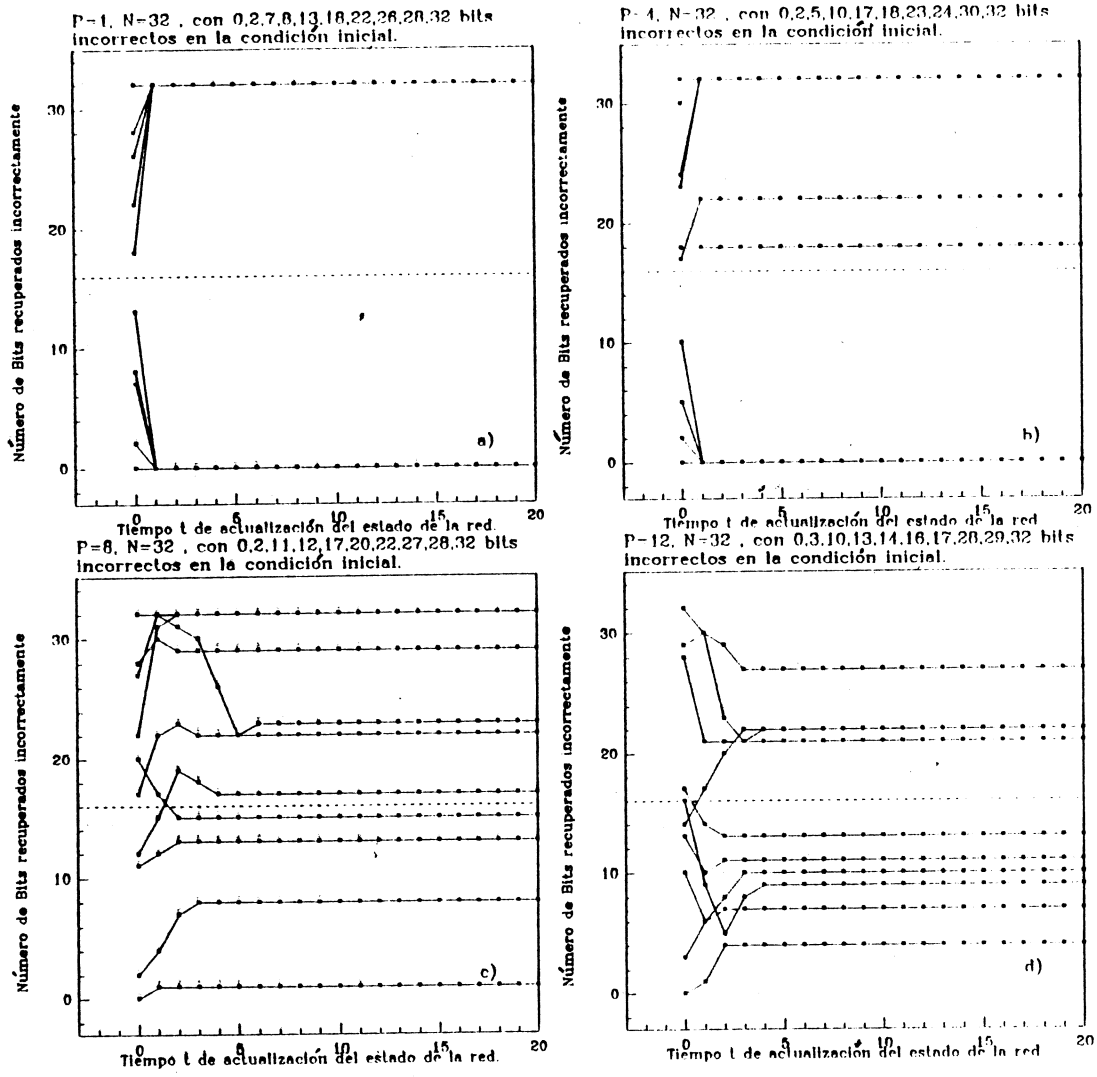


Figura 5.3 SERIE DE GRÁFICAS MOSTRANDO EL NÚMERO DE BITS EQUIVOCADOS PARA UN TIEMPO VARIABLE t DE ACTUALIZACIÓN DEL ESTADO DE LA RED, PARA UNA RED DE 32 UNIDADES. CON *a*) $P = 1$, *b*) $P = 4$, *c*) $P = 8$, *d*) $P = 12$ Y UN NÚMERO VARIABLE DE ERRORES EN LA CONDICIÓN INICAL.

Debe notarse que en el conjunto de simulaciones, cuyos resultados se presentaron en la figura (5.3), no se efectuaron promedios, debido a la naturaleza de la simulación, a continuación, se presentarán los resultados de simulaciones, llevadas a cabo para diferentes tiempos de actualización, con un número constante de patrones y neuronas, en el cual se efectuarán promedios sobre los estados de la red para diferentes tiempos, asimismo, se considerará como variable independiente, al número de errores en la condición inicial.

La serie de gráficas (5.4a), (5.4b) y (5.4c) muestran los resultados de los valores promedio de los estados al tiempo t, para tres actualizaciones sucesivas $t = 1, 2$ y 3 respectivamente, en una red de 32 unidades, cuando ésta ha sido entrenada con un número constante de patrones $P = 3$. Las simulaciones fueron efectuadas, para un número variable de bits equivocados, con respecto a uno de los patrones aprendidos, en la condición inicial. Así, se puede apreciar que para cuando el número de patrones y el número de neuronas permanece constante (i.e. $\alpha = 0.09$, constante), entonces, se obtiene que la red funciona perfectamente. Inclusive, además de reconocer y retener los patrones almacenados, rectifica los errores introducidos, verificándose así la capacidad y funcionameinto de la red como una memoria asociativa direccionada por contenido. Los promedios, fueron realizados, sobre un ensemble de 512 condiciones al tiempo $t = 1, 2, 3$.

La serie de gráficas (5.4d), (5.4e) y (5.4f) muestran los resultados de los valores promedio de los estados al tiempo t de la red, para tres actualizaciones sucesivas $t = 1, 2$ y 3 respectivamente, en una red de 32 unidades, cuando ésta ha sido entrenada con un número constante de patrones $P = 10$. Las simulaciones fueron efectuadas para un número variable de errores, en la condición inicial. Así, se puede apreciar, que para cuando el número de patrones y el número de neuronas permanece constante (i.e. $\alpha = 0.31$, constante), la red ya no es capaz de retener la información correctamente, obteniéndose errores en los estados de la red al tiempo $t = 1, 2, 3$. Así, se presenta un fenómeno de avalancha de errores, a través de actualizaciones sucesivas, como se puede apreciar en la gráfica (5.4f), la cual muestra que los errores en los estados promedio de la red para $t = 3$, se han incrementado y se han esparcido de su valor promedio, como lo muestran las barras de error. Los promedios, fueron realizados, sobre un ensemble de 512 condiciones al tiempo $t = 1, 2, 3$.

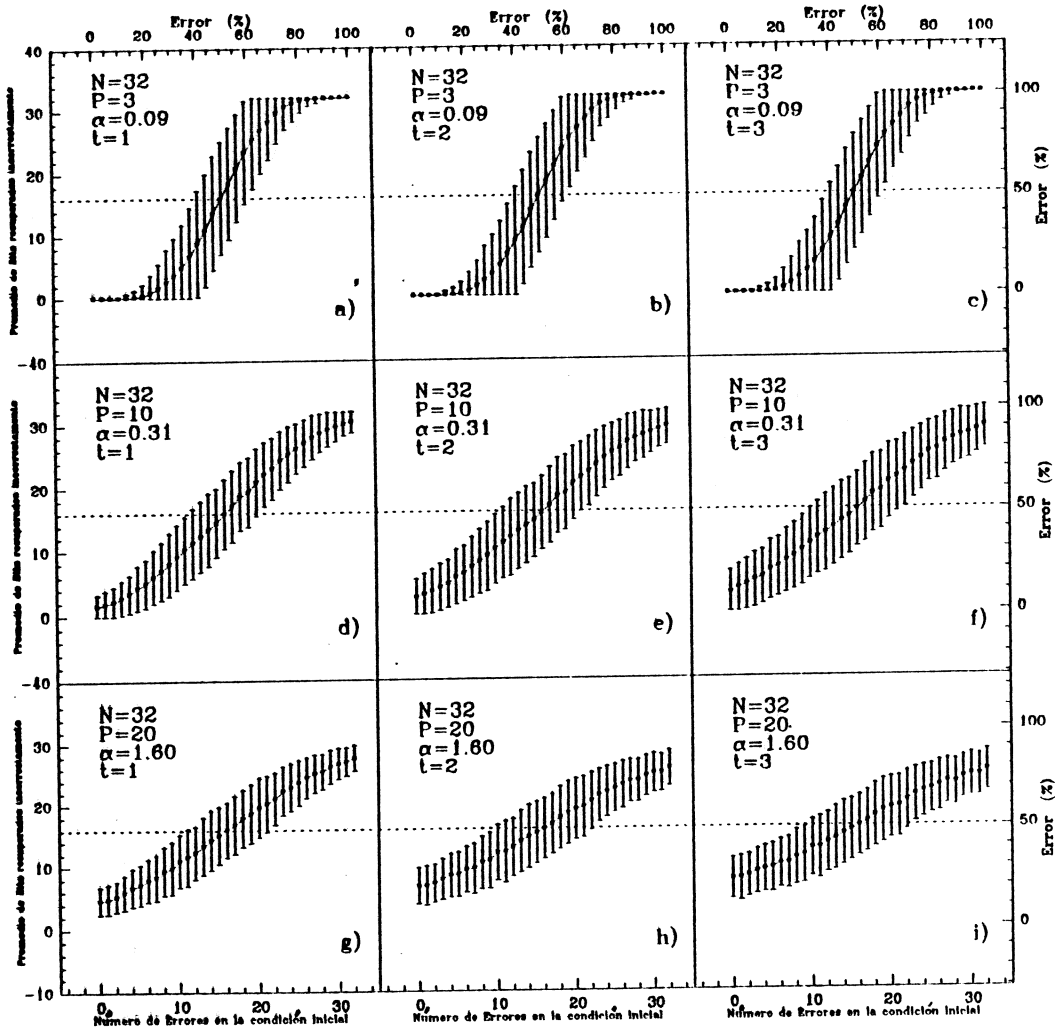


Figura 5.4 SERIE DE GRÁFICAS MOSTRANDO EL NÚMERO PROMEDIO DE BITS EQUIVOCADOS AL TIEMPO *a*) $t = 1$, *b*) $t = 2$, *c*) $t = 3$, CON $N = 32$ Y $P = 3$, AL TIEMPO *d*) $t = 1$, *e*) $t = 2$, *f*) $t = 3$, CON $N = 32$ Y $P = 10$, AL TIEMPO *g*) $t = 1$, *h*) $t = 2$, *i*) $t = 3$, CON $N = 32$ Y $P = 20$, LA CUAL A SIDO ENTRENADA, CON UN NÚMERO VARIABLE, DE ERRORES EN LA CONDICIÓN INICIAL.

La serie de gráficas (5.4g), (5.4h) y (5.4i) muestra los resultados de los valores promedio de los estados al tiempo t, para tres actualizaciones sucesivas $t = 1, 2$ y 3 respectivamente, en una red de 32 unidades, cuando ésta ha sido entrenada con un número constante de patrones $P = 20$. Las simulaciones fueron efectuadas, para un número variable de errores, en la condición inicial. Así, se puede apreciar, que para cuando el número de patrones y el número de neuronas permanece constante (i.e. $\alpha = 1.60$, constante), se puede apreciar que los efectos observados en la serie de gráficas (5.4d), (5.4e) y (5.4f), se vuelven más evidentes mostrando la inutilidad de la red a retener la información para actualizaciones sucesivas, inclusive, cuando se ha utilizado uno de los patrones como condición inicial, sin ruido agregado. El fenómeno de avalancha, se puede apreciar más claramente, en el incremento de las barras de error, a través de actualizaciones sucesivas, como lo muestra la gráfica (5.4i), en la cual los errores en los estados al tiempo $t = 3$, se han incrementado y se han esparcido de su valor promedio, como lo muestran las barras de error, los promedios, fueron realizados, sobre un ensemble de 512 condiciones al tiempo $t = 1, 2, 3$.

Observaciones. En un punto fijo se cumple la condición

$$\sum_{i=1}^{N}\Big(S_i(t+1) - S_i(t)\Big) = 0, \tag{5.1}$$

en nuestras simulaciones no comprobamos si se llegó o no al punto fijo, lo cual hace poco útiles nuestros resultados. Sin embargo, el código del algoritmo que se incluye en el apéndice fué modificado de manera que comprueba, después de cada iteración si se llegó ó no a un mínimo. Como se puede apreciar, para valores de P y de N apropiados, efectivamente no existieron errores en la recuperación del patrón almacenado, almenos, para un valor propicio de bits equivocados, en la condición inicial.

5.4 Simulaciones para redes de 64, 128, 256 y 512 unidades con un número variable de patrones por aprender, sin bits equivocados en la condición inicial. A continucación se presentan los resultados algunas simulaciones, el objetivo de estas simulaciones, fue principalmente el de explorar las propiedades computacionales del algoritmo discutido en el capítulo IV, para cuando se utiliza un número creciente de unidades y de patrones almacenados. Los resultados de las simulaciones, se presentan a continuación.

La figura (5.5) muestra los valores promedio de los estados al tiempo t de la dinámica de la red, cuando ésta ha sido entrenada con un número variable de patrones. Esta figura muestra las gráficas corespondientes a redes de 64, 128, 256 y 512 unidades, una vez que se

ha llevado a cabo el primer paso de actualización ($t = 1$); ésto es, cuando todas la unidades de la red han sido actualizadas una sóla vez. En estos casos los promedios fueron efectuados sobre ensembles de estados de la red al tiempo $t = 1$, ensembles compuestos por 512, 400, 300 y 100 elementos, para las redes de 64, 128, 256 y 512 unidades, respectivamente.

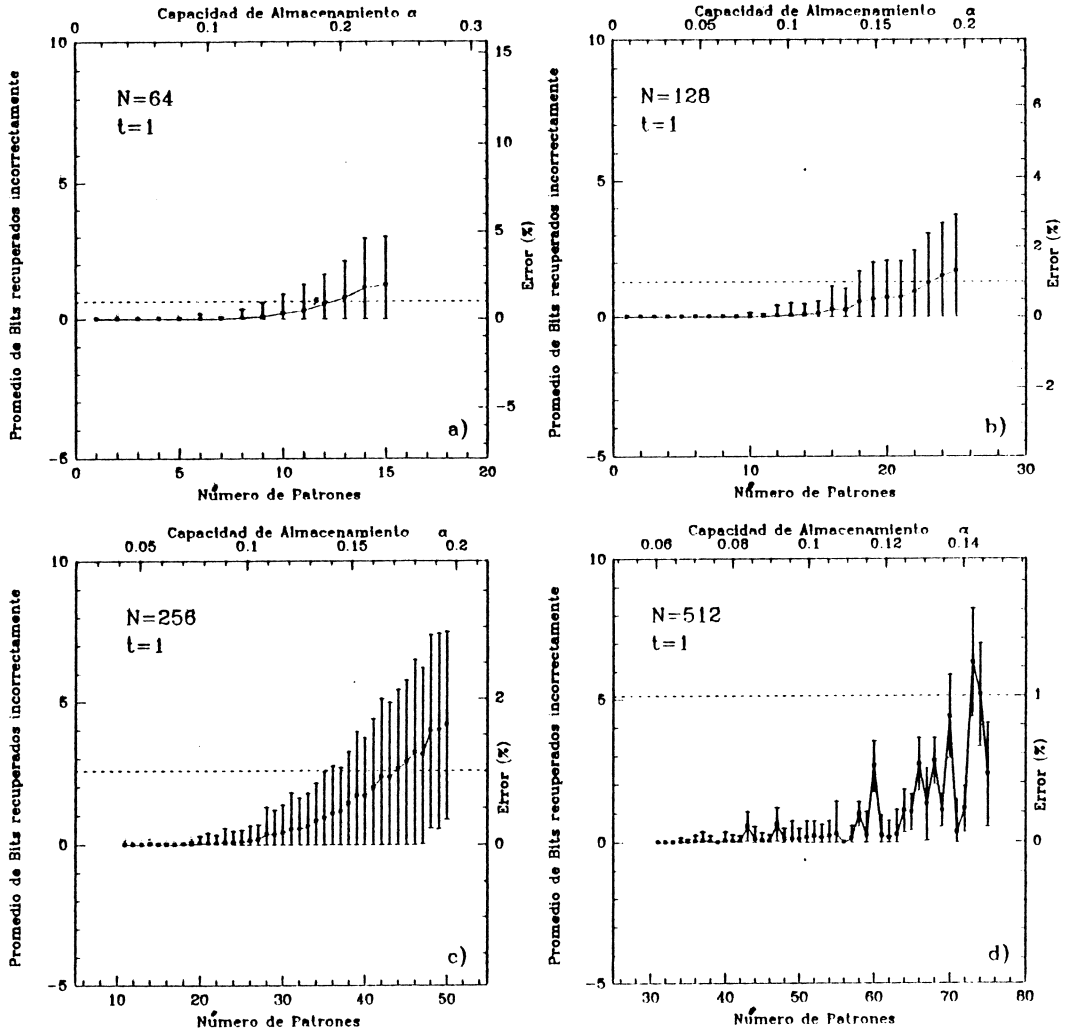


Figura 5.5 SERIE DE GRÁFICAS MOSTRANDO EL NÚMERO PROMEDIO DE BITS EQUIVOCADOS AL TIEMPO $t = 1$ PARA UNA RED CON N ELEMENTOS DONDE *a*) $N = 64$, *b*) $N = 128$, *c*) $N = 256$, *d*) $N = 512$. ESTAS REDES HAN SIDO ENTRENADAS, CON UN NÚMERO VARIABLE DE PATRONES.

La figura (5.6) muestra los resultados de los valores promedio de las actualizaciones para los 4 tiempos posteriores, en la red de 64 unidades. Estos promedios fueron tomados sobre ensembles de 512 condiciones al tiempo $t = 2, 3, 4, 5$.

La figura (5.7) muestra los resultados de los valores promedio de las actualizaciones para los 4 tiempos posteriores, en la red de 128 unidades. Estos promedios fueron tomados sobre ensembles de 400 condiciones al tiempo $t = 2, 3, 4, 5$.

La figura (5.8) muestra los resultados de los valores promedio de las actualizaciones para los 4 tiempos posteriores, en la red de 256 unidades. Estos promedios fueron tomados sobre ensembles de 300 condiciones al tiempo $t = 2, 3, 4, 5$.

La figura (5.9) muestra los resultados de los valores promedio de las actualizaciones para los 4 tiempos posteriores, en la red de 512 unidades. Estos promedios fueron tomados sobre ensembles de 100 condiciones al tiempo $t = 2, 3, 4, 5$.

Observaciones. Es importante remarcar el hecho de que las simulaciones, fueron hechas para tiempos finitos de actualización, en las que el tiempo más grande de actualización de la red fue $t = 5$. De forma tal que en estas simulaciones, no fueron contemplados los efectos de seguir actualizando la red para tiempos mayores. Para este ejemplo en particular, en el que se inicializa al estado de la red en un punto fijo de la dinámica, y en el que se estudia su evolución, se puede inferir, que una vez que la red empieza a producir errores en el estado de la red al tiempo $t = 1$, estos seguirán produciendose en actualizaciones sucesivas.

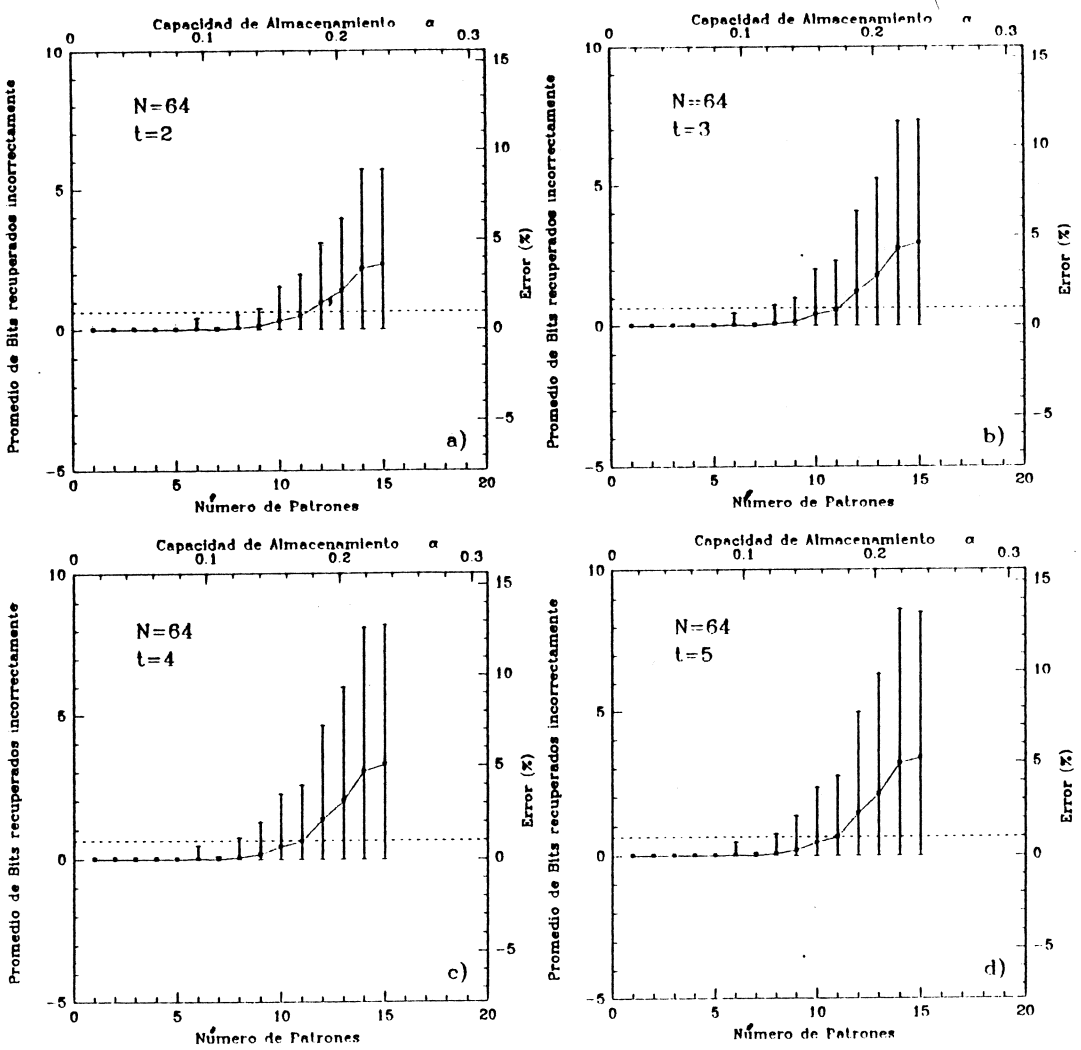


Figura 5.6 SERIE DE GRÁFICAS MOSTRANDO EL NÚMERO PROMEDIO DE BITS EQUIVOCADOS AL TIEMPO t DONDE *a*) $t = 2$, *b*) $t = 3$, *c*) $t = 4$, *d*) $t = 5$, PARA UNA RED NEURONAL DE 64 UNIDADES, LA CUAL A SIDO ENTRENADA, CON UN NÚMERO VARIABLE DE PATRONES APRENDIDOS.

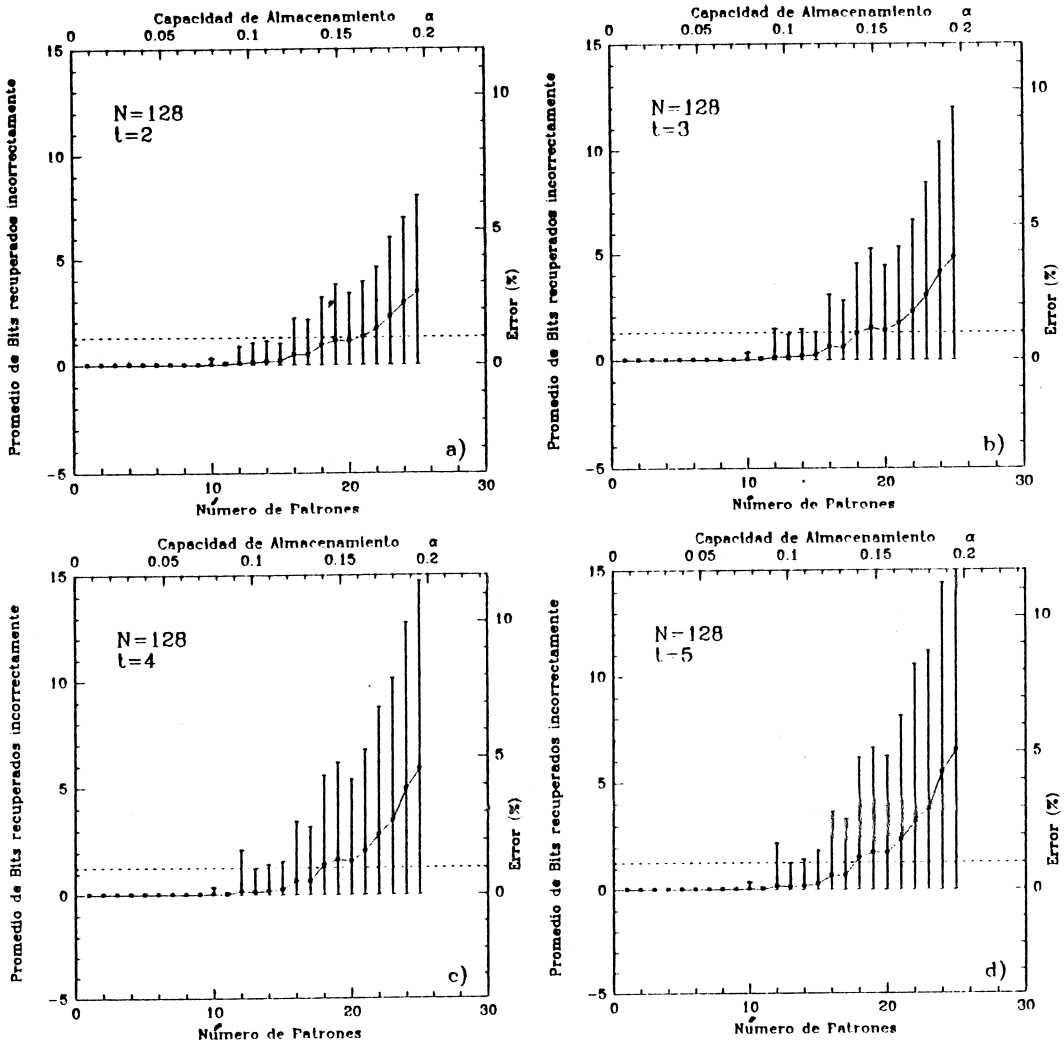


Figura 5.7 SERIE DE GRÁFICAS MOSTRANDO EL NÚMERO PROMEDIO DE BITS EQUIVOCADOS AL TIEMPO t DONDE *a*) $t = 2$, *b*) $t = 3$, *c*) $t = 4$, *d*) $t = 5$, PARA UNA RED NEURONAL DE 128 UNIDADES, LA CUAL A SIDO ENTRENADA, CON UN NÚMERO VARIABLE DE PATRONES APRENDIDOS.

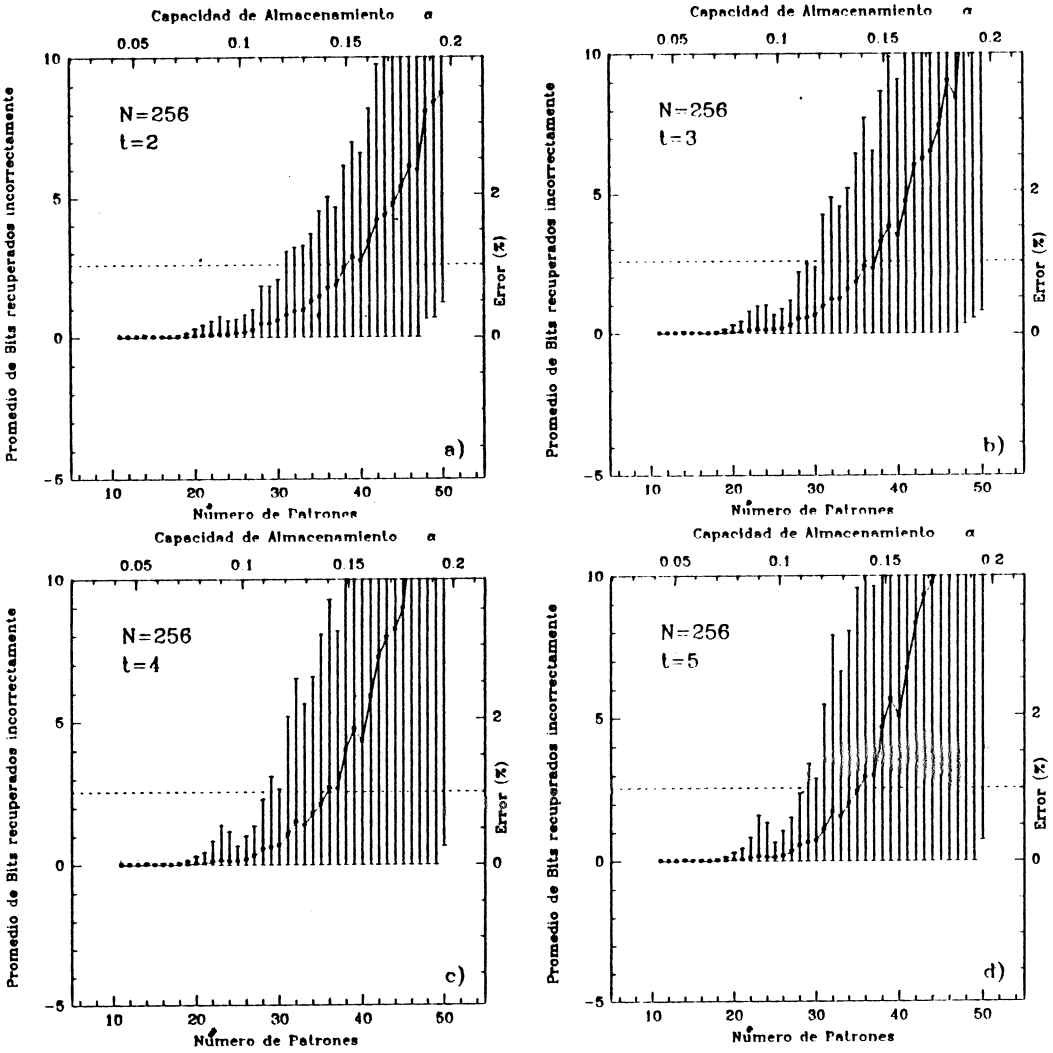


Figura 5.8 SERIE DE GRÁFICAS MOSTRANDO EL NÚMERO PROMEDIO DE BITS EQUIVOCADOS AL TIEMPO t DONDE *a*) $t = 2$, *b*) $t = 3$, *c*) $t = 4$, *d*) $t = 5$, PARA UNA RED NEURONAL DE 256 UNIDADES, LA CUAL A SIDO ENTRENADA, CON UN NÚMERO VARIABLE DE PATRONES APRENDIDOS.

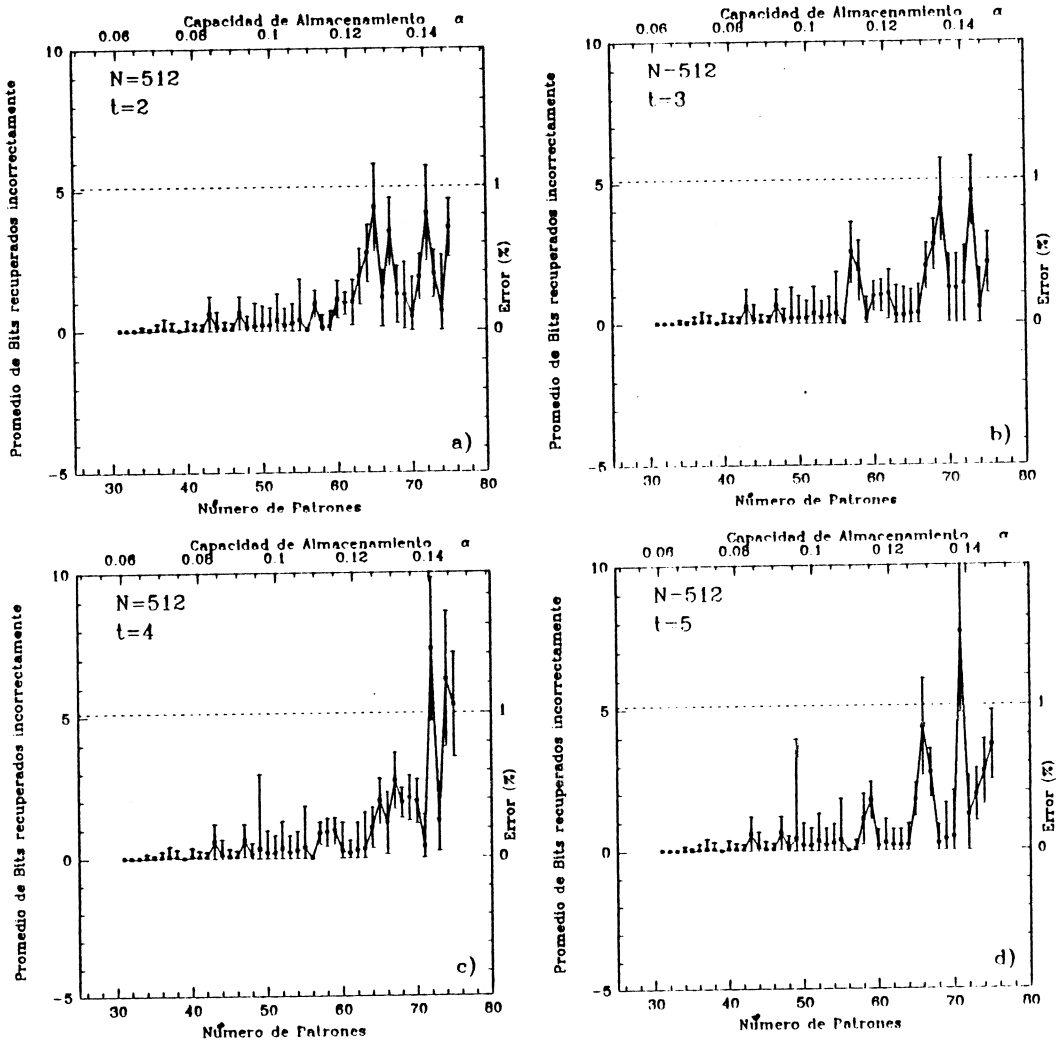


Figura 5.9 SERIE DE GRÁFICAS MOSTRANDO EL NÚMERO PROMEDIO DE BITS EQUIVOCADOS AL TIEMPO t DONDE *a*) $t = 2$, *b*) $t = 3$, *c*) $t = 4$, *d*) $t = 5$, PARA UNA RED NEURONAL DE 512 UNIDADES, LA CUAL A SIDO ENTRENADA, CON UN NÚMERO VARIABLE DE PATRONES APRENDIDOS.

5.5 Especificaciones Computacionales. Los programas, que generaron los resultados de éste capítulo, así como los programas para reducir los datos, fueron escritos en lenguaje de programación C (Kerningam 1978). El código del programa, para el cálculo de los promedios, varianzas y desviaciones estandard, fue tomado en su totalidad de *Numerical Recipes in C* (Press 1988). Las gráficas fueron producidas por el paquete de graficado Super Mongo (Lupton 1989) y todos los cálculos fueron efectuados en una computadora Sun Sparc Station, con sistema operativo Unix (Kerningam 1984).

VI. Conclusiones

Se efectuó una revisión bibliográfica referente al modelo de Hopfield, comenzando por definir algunos conceptos fundamentales con el fin de entablar un marco de referencia en términos gramaticales, para lo que fue escrito el capítulo I, el cual cuenta con las definiciones necesarias, al menos dentro del universo de palabras utilizadas en la realización de esta tesis.

Posteriormente, se definió el problema a resolver (Capítulo II), conocido como problema **memoria asociativa** utilizando el formalismo de redes neuronales. Una vez planteado el problema, se procedió a encontrar una solución analítica a éste, utilizando un análisis estadístico comúnmente conocido como análisis *señal a ruido*, se puede concluir por resultados teóricos más sofisticados como los expuestos en el capítulo III, así como por los encontrados al realizar las simulaciones computacionales, que el análisis señal a ruido, es efectivamente capaz de predecir el comportamiento colectivo de la evolución dinámica de la red. Estando la dinámica de la red, definida por el sistema de ecuaciones descrito en el capítulo II.

Así, el análisis señal a ruido es efectivamente una herramienta útil para el entendimeinto del modelo de Hopfield de Memoria Asociativa utilizado en esta tesis, como lo corroboran los resultados obtenidos en el capítulo V. Es importante, sin embargo, entender que los resultados analíticos obtenidos por medio del análisis señal a ruido utilizado, son válidos únicamente dentro de un espacio de constricciones del modelo de Hopfield y a temperatura cero $T = 0$.

Una vez entendido el problema de memoria asociativa, se procedió simularlo numéricamente, utilizando el esquema de codificación múltiple de espines, cuyos detalles fueron presentados en el capítulo IV. En este punto, sería bueno concluir que el modelo numérico utilizado, implementando las mejoras debidas a la codificación múltiple de espines, así como de las mejoras utilizadas dentro de este enfoque, son efectivamente la mejor forma de implementar actualmente el algoritmo computacional del modelo de Hopfield en una computadora, ya que permite optimizar los recursos computacionales disponible. Asimismo, es importante resaltar que estas mismas simulaciones pueden ser extendidas a un número muy grande de modelos similares al de Hopfield, por ejemplo los modelos en los cuales la temperatura es finita, o modelos para los cuales $T \neq 0$. Así como para modelos a los cuales se les puede incorporar diversas reglas de aprendizaje. Así, el trabajo presentado en esta tesis pone de manifiesto la posibilidad de extender el método numérico a modelos

más elaborados, cumpliéndose así uno de los objetivos principales de esta tesis, que era el de poder explorar las propiedades computacionales del modelo de Hopfield.

Para concluir, debe señalarse que el modelo utilizado en esta tesis, no es el único, ni tampoco es el que describe con mayor precisión el comportamiento de la dinámica de la red del modelo analizado, sin embargo, es justo decir que éste fue escogido debido a su simplicidad matemática y por la facilidad de poner de manifiesto de una manera clara y sencilla la naturaleza inherente en el modelo estudiado. La forma más óptima de resolver el problema en el futuro y que representa hoy un área de intensa investigación, será proporcionado por arquitecturas computacionales que trabajen más acorde con la naturaleza de éste, ya que no tienen las limitantes de los sistemas computacionales que ofrecen los denominados de Von Newman.

APENDICE A

```
/*Programa solo64.c, originalmente desarrollado por Penna T.J.P. y Oliveiria P.M.C., (ver
(1989) Journal Of Physics, A: Mathematical and General, 22, L-719), y adaptado
para realizar las simulaciones de esta tésis, de una red neuronal de 64 unidades, la cual
a realizado 512 simulaciones diferentes por cada patrón que es agregado como variable
independiente. Las figuras 5.5a y 5.6 muestran los resultados de la simulación.
 16 Diciembre de 1992*/


#include <stdio.h>
#include <math.h>
#include <string.h>
#include <time.h>
#define SZ 32
#define SI 31
static char *s6 = "n64p"; /*utilizado para los nombres de los archivos, con 512 resultados
                            por cada archivo, en este caso, se generan 15 archivos,
                            uno por cada patrón que es agregado, por ejemplo :
                           ``n64p8.dat" contiene los resultados de 512 simulaciónes diferentes,
                             para una red de 64 unidades y 8 patrones. */

/*inicio del programa*/
main() {
double floatnneu,floatrui;
float random,rui,promedio;
int ran,i,hi,l,k,j,m,p1,t,iter,nmai,nmeno,cambio;
unsigned int bit[32],
rede[100],      /*guarda el patrón a recuperar o la condición inicial de hasta 32*100 bits*/
e[200][100],    /*guarda hasta 200 patrones a ser almacenados, de hasta 3200 bits cada patrón*/
se[100],        / *guarda la condición inicial para despues compararla con la condición final y
                  saber los bits equivocados*/
ma[200][100],  /*guarda los valores de los productos XOR , para la suma sobre j
                 en la ecuación (4.11), de forma que se optimize la dinámica de la red*/
tempo,tamanho,npadroes,spin,ruido,ps,temp,sum,nneurons,sumu,si,tfinal,tinic,bytrand;
char  we[4];
FILE *yane, *patr, *patro;

/*lee del archivo ``datos.dat" el número de neuronas a ser simulado*/

scanf("%d",&nneurons);

for(i=0;i<SZ;i++) bit[i]=(1<<i);
floatnneu=nneurons;
```

```
/*efectuar la simulación, para un total de 15 patrones (npadroes), (i.e. efectuar
la simulación de la red con el número de patrones como la variable
independiente)*/

for(npadroes=0;npadroes<15;npadroes++){

ruido=0;
floatrui=ruido;
rui=floatrui/floatnneu;

tamanho=nneurons/SZ;

/*factores de normalizacion*/
nmai=nneurons+1;
nmeno=nneurons-1;

time(&tinic);

patr=fopen("nombres.dat","w");
fprintf(patr,"%d",npadroes);
fclose(patr);
strcpy(s6,"n64p");
patro=fopen("nombres.dat","r");
fscanf(patro,"%s",we);
strcat(s6,we);
fclose(patro);
yane=fopen(s6,"w");

/*efectúa la actualización de la red 512 veces diferentes, de forma tal que
se pueda generar un ensemble de soluciones y se pueda efectuar un promedio*/

for(iter=0;iter<512;iter++){
      time(&tempo);

/*genera P patrones, con el valor de cada uno de los bits, generado al azar*/

for(i=0;i<npadroes;i++)
  {
   for(j=0;j<tamanho;j++)
      {
       bytrand=0;
       for(k=0;k<32;k++)
          {
           ran=rand();random=ran/2147483648.0;
           if(random<=0.50) bytrand=bytrand|bit[k];
          }
       e[i][j]=bytrand;
      }
  }
```

```
/*toma uno de los P patrones que fueron generados al azar, y utilízalo como
                condición inicial*/

ran=rand();random=ran/2147483648.0;
p1=floor(random*npadroes);
for(m=0;m<tamanho;m++)
  {
   rede[m]=e[p1][m];
   se[m]=e[p1][m];
  }

/*calcula los valores de los productos XOR, para la sumatoria de j,
como lo describe en su artículo G.A. Khoring, véase capítulo 4, de la tésis*/

 for(j=0;j<tamanho;j++)
   {
    for(l=0;l<npadroes;l++) rha[l][j]=e[l][j]^rede[j];
   }


/*efectuar la actualización para el estado de la red 100 veces
(i.e. solo hasta t=100)*/
cambio=0;
for(hi=0;hl<100;hl++)
  {
   sumu=0;
   for(i=0;i<tamanho;i++)
     {
      temp=rede[i]^se[i];
      while (temp)
            {
             temp&=(temp-1);
             sumu++;
            }
     }
```

```
/*actualizar los valores de las 64 unidades de forma secuencial, de acuerdo con el algoritmo de
Penna y Oliveiria, descrito en la sección 4.4 de la tésis.*/

for(i=0;i<nneurons;i++)
   {
    t=i&SI;
    j=i/SZ;
   spin=0;
   for(ps=0;ps<npadroes;ps++)
      {
       sum=0;
       for(i=0;i<tamanho;i++)
          {
           temp=ma[ps][i];
          while (temp)
                {
                 temp&=(temp-1);
                 sum++;
                }
          }
       spin+=(e[ps][j]&bit[t])?(nneurons-sum):sum;
      }
si=(rede[j]&bit[t])?(nmai*npadroes):(nmeno*npadroes);
spin=spin<<1;
if(spin!=si)
 {
  spin=(spin>si)?bit[t]:0;
  if((rede[j]&bit[t])!=spin)
    {
     rede[j]^=bit[t];cambio++;
     for(i=0;i<npadroes;i++) ma[i][j]=e[i][j]^rede[j];
    }
 }

}/*fin de la actualización de una unidad, continuar con la siguiente unidad i*/

  if(cambio==0) break;
 /* Parar de hacer las simulaciones, ya que se ha llegado a un punto fijo de la configuraci\'on*/
/* continuar con otra simulación (i.e. siguiente iter)*/

     sum=0;
     for(i=0;i<tamanho;i++)
        {
         temp=rede[i]^se[i];
         while (temp)
               {
                temp&=(temp-1);
                sum++;
               }
        }


promedio=32.0;
fprintf(yane,"%u %u %d %f %u\n",sumu,sum,p1,promedio,0);
```

```
time(&tfinal);
tfinal-=tinic;

      }/*fin de las actualizaciones para un tiempo (i.e. t=1)*/

}/*fin de una de las 512 (i.e. iter ) simulaciónes de la red, por patrón*/


fprintf(yane,"\nAcabe en %2ld:%2ld:%2ld:\n",(tfinal/3600),(tfinal/60)%60,tfinal%60);
fclose(yane); /*escribe el resultado en un archivo
                  correspondiente a uno de los patrones*/

};/* fin de la simulación para los P patrones*/
}/*fin de la función main*/
_
```

BIBLIOGRAFIA

Alkon Daniel L., (1989) *Scientific American, July*, 42.

Amit D.J., Gutfreund H., Sompolinsky H., (1985a) *Phys. Rev.* **A32**, 1007.

Amit D.J., Gutfreund H., Sompolinsky H., (1985b) *Phys. Rev. Lett.* **55**, 1530.

Amit, D., (1989) *Modeling Brain Function.* Cambridge: Cambridge University.

Binder K., (1980) *Fundamental Problems in Statistical Mechanics V*, Ed. By Cohen E.G. (North-Holland, Amsterdam)

Bhattacharya R.N., Whymire E.C., (1990), *Stochastic Processes With Applications*, Pag. 17,

Glauber R., (1963) Time-Dependent Statistics of the Ising Model, J. Math. Phys. **4**, 294.

Ising E., (1925), Beitrag zur Theorie des Ferromagnetismus, *Z. Physik* **31**, 253.

Hebb, D.O., (1949) *The Organization of Behavior.* Wiley, New York.

Hertz, J.A., Krogh A., Palmer G.R., (1991) *Introduction To The Theory Of Neural Computation.* Vol. I, (Addison-Wesley).

Hopfield, J.J., (1982) *Proc. Natl. Acad. Sci. USA* **79**, 2554.

Ising E., (1925), Beitrag zur Theorie des Ferromagnetismus, *Z. Physik* **31**, 253.

Kerningam B.W., Pike R., (1984) *The UNIX Programing Environment.* , (Englewood Cliffs, NJ: Prentice-Hall).

Kerningam B.W., Ritchie D.M., (1978) *The C Programing Language.* , (Englewood Cliffs, NJ: Prentice-Hall).

Kohring G.A., (1990) *J. Stat. Phys.* , **59** , 1077.

Little W.A., (1974) The Existence of Persistent States in the Brain. *Math. Biosci.* **19**, 101-120.

Little W. A., Shaw G. L., (1978) Analytic Study of the Memory Capacity of a Neural Network. *Math. Biosci.* **39**, 281.

Lupton R., Monger P., (1989) *Super Mongo for Sun Work Stations*, Preprint.
-Original de Tonry L. J., (1982) *Mongo An interactive Plotting Program.*

McCulloch, W.S., Pitts, W., (1943), A Logical Calculus of Ideas Immanent in Nervous Activity, *Bull. Math. Biophys.* **5**, 115-133.

Minsky M. L., Papert S. A., (1968) *Perceptrons*, The MIT Press.

Morgenstern I., Binder K., (1979) *Phys. Rev. Lett.* , **43** , 1615.

Müller B., Reindhardt J., (1991) *Physics of Neural Networks: Neural Networks*, Pag. 27, Spinger-Verlag.

Neher E., Sakmann B., (1992) *Scientific American, March*, 44.

Penna T.J.P., Oliveira P.M.C., (1989) *J. Phys. A: Math. Gen.* , **22** , L-719.

Press W.H., Flannery B.P., Teukolsky S.A., Vetterling W.T., (1988) *Numerical Recipes in C*, Pag. 473, Cambridge University Press.

Ramón y Cajal, S., (1913) *Estudios sobre la degeneración y regeneración del sistema nervioso.*

Reichl L.E., (1980), *"A Modern Course In Statistical Physics"*, P. 151, University of Texas Press.

Singer, W., (1987) *The Neural and Molecular Bases of Learning* ed. J.P. Changeux and M. Konishi, (New York: Wiley) p. 301.

Sompolinsky H., (1988), "Statistical Mechanics of Neural Networks" *Physics Today*, December, Pag. 70.

Stein D.L., (1989), "Spin Glasses" *Scientific American*, July, Pag. 52.

Sun Spark Station, *References*

Talavera, C., (1992) *Optimización de una Red Neuronal Entrenada por Retropropagación*, Tesis de Licenciatura, *Universidad Autónoma de Baja California*, Pag. 33.

Tolouse G., (1989) *J. Phys. A: Math. Gen.* **22**, 1959.

Viana, L., (1985) *Ph.D. Thesis, Manchester University.*

Viana, L., (1988) *J. Phys. (France)*, **49**, 205.

Viana, L., Cota E., Martínez C., (1990) *Basins of attraction and spurious states in Neural Networks*, Precedings of the XI Sitges Conference, Springer-Verlag.

Viana, L., (1990) *Memoria Natural y Artificial* Fondo de Cultura Económica (México), No. **88**.

LECTURA RECOMENDADA

- **Amit, D.** (1989) *Modelling Brain Function*. Cambridge: Cambridge University.

- **Hertz, J.A., Krogh A., Palmer G.R.** (1991) *Introduction To The Theory Of Neural Computation*. Vol. I, (Addison-Wesley).

- **Hemmen Van, Dommany E. Eds.** (1991) *Physics of Neural Networks: Introduction to Neural Networks*, Spinger-Verlag.

- **Mezard M., Parisi G., Virasoro M. A.** (1986) *Spin Glass Theory and Beyond*, World Scientific Publications, Singapore, 1986.

- **Müller B., Reindhardt J.** (1991) *Physics of Neural Networks: Neural Networks*, Spinger-Verlag.

- **Sompolinsky H.** (1988), "Statistical Mechanics of Neural Networks" *Physics Today*, December, Pag. 70.

- **Viana, Laura** (1990) *Memoria Natural y Artificial* Fondo de Cultura Económica (México), No. 88.