UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE INGENIERIA

REDES NEURONALES EN CONTROL Y ROBOTICA

T E S I S

QUE PARA OBTENER EL TITULO DE

INGENIERO EN COMPUTACION

P R E S E N T A

LUIS ANGEL HUIDOBRO GONZALEZ

ASESOR: ING. FRANCISCO RODRIGUEZ RAMIREZ

MEXICO, D. F. 1995

Redes Neuronales en Control y Robótica

A mis padres

...las razones sobran.

ÍNDICE

Agradecimientos

A mis hermanos (Adolfo, Grabiel, Mauricio y Carlos) por permitirme sentir orgulloso de ellos siempre. A Francisco Rodriguez por ser más que un asesor, un amigo. A la Facultad de Ingenieria por darme una preparación profesional. A la Universidad Nacional Autonoma de México por haber sido participe y escenario de mi juventud.

...la ausencia absoluta de carga hace que el hombre se vuelva más ligero
que el aire, vuele hacia lo alto, se distancie de la tierra, de su ser terreno,
que sea real sólo a medias y sus movimientos sean tan libres como insignificantes.
"La Insoportable Levedad del Ser"
M. Kundera

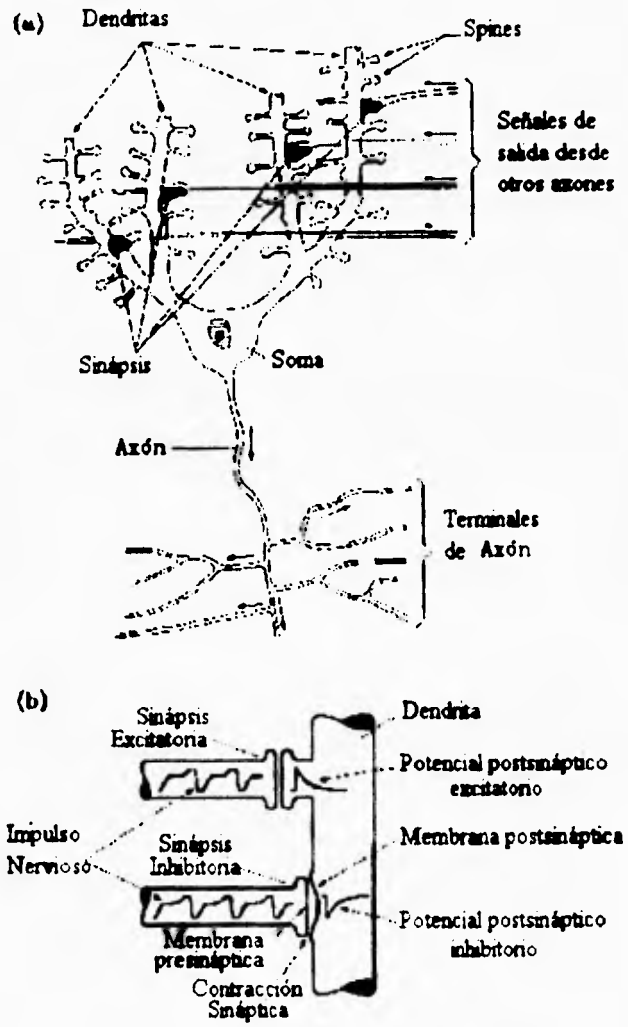
1. INTRODUCCIÓN A LAS REDES NEURONALES ARTIFICIALES

El visionario es el único realista
F. Fellini

Las características principales de las neurocomputadoras son que estas "aprenden" la tarea que van a realizar y no tienen que ser programadas para obtener el resultado deseado; trabajan en forma paralela y no secuencialmente como las máquinas de Von Neumann; están inspiradas en las neuronas biológicas, y están interconectadas entre sí, para formar una red neuronal o neurocomputadora

1.1. Neuronas Biológicas

El sistema nervioso humano, está formado de células llamadas neuronas. Cerca de 10^{11} neuronas participan en unas 10^{15} interconexiones, donde cada neurona comparte algunas características con otras células del cuerpo. La neurona es una célula nerviosa que posee la capacidad de excitarse y propagar el impulso nervioso a otra neurona, siempre en la misma dirección. Tiene regiones especiales de recepción que son las dendritas y de emisión o salida, el axón. Las dendritas se extienden del cuerpo a otras neuronas, donde reciben señales en un punto de conexión llamado sinapsis. El axón, un cilindro estrecho que transporta los impulsos desde el cuerpo a otras células, se ramifica y cada una de estas ramas termina en un pequeño bulbo que casi toca las dendritas de otra neurona. Estos puntos de cuasi-contacto se denominan sinapsis. La transmisión interneuronal es unas veces eléctrica y otras por difusión de productos químicos. Una neurona sólo dispara un impulso eléctrico a lo largo de su axón, si el número de impulsos que llegan a los bulbos terminales de sus dendritas, en un periodo corto de tiempo, es suficiente. Estos impulsos pueden ayudar o impedir el disparo de un impulso y se llaman respectivamente excitadores e inhibidores. La condición para la activación de una neurona, es que la excitación supere la inhibición en una cantidad crítica, llamada umbral, cuando esto sucede, la célula dispara enviando una señal a otras neuronas por el axón.



(a) Estructura esquemática de la neurona biológica
(b) Esquema simplificado de la sinápsis

Figura 1.1 Neurona biológica

1.2. Neuronas Artificiales

Las neuronas artificiales fueron diseñadas para imitar las características principales de una neurona biológica. Es decir, se aplica el conjunto de entradas, cada una representando la salida de otra neurona, cada entrada es multiplicada por un peso correspondiente, y todas las entradas con sus pesos son sumadas para determinar el nivel de activación de la neurona. Existen paradigmas neurocomputacionales que nos permiten representar esto como un conjunto de entradas $\xi_1,\xi_2,....,\xi_n$, las cuales son aplicadas a una neurona artificial, cada señal se multiplica por su peso asociado $\omega_1,\omega_2,.....,\omega_n$; se suman todas las entradas que ya fueron multiplicadas por sus pesos, para producir una salida, esta es procesada por una función de activación "$\varphi(.)$".

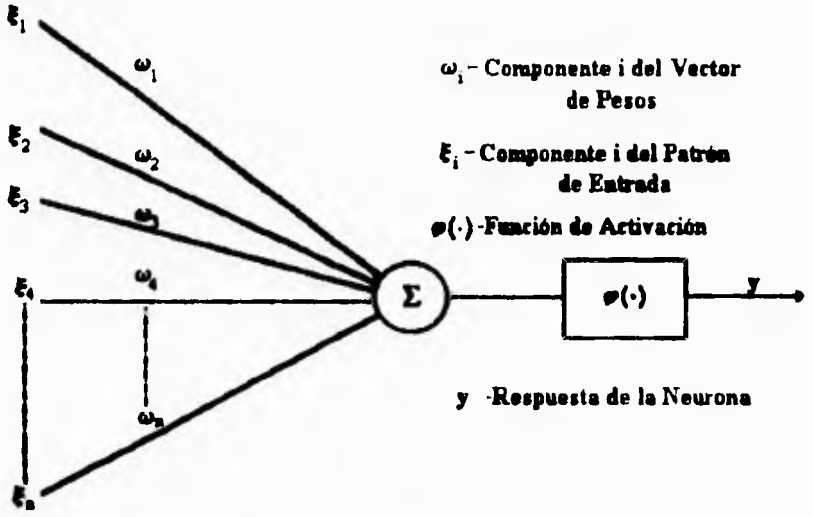


Figura 1.2 Neurona artificial

El poder de las neurocomputadoras, es que nos permite conectar las neuronas en niveles o capas conectadas entre si; lo que llamaremos una red neuronal o neurocomputadora, la red más simple es un grupo de neuronas ordenadas en una capa o

nivel; donde la primera capa o capa de entrada, únicamente distribuye las entradas para ser procesadas por las neuronas. En la siguiente figura se muestra una red neuronal de una capa.

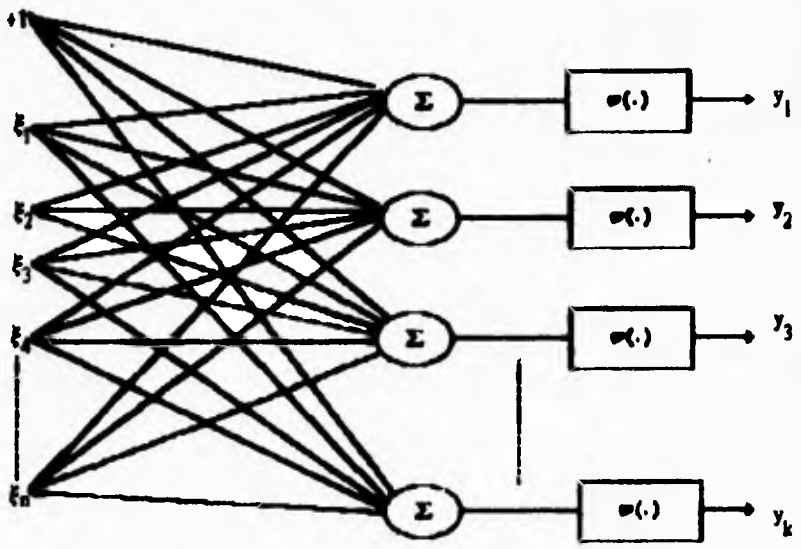


Figura 1.3 Red neuronal de una capa

1.3. Función de Activación

Las señales que llegan a una neurona, son sumadas y multiplicadas por sus respectivos pesos (Net), a esta salida se le aplica la función de activación, de la cual obtenemos una salida no lineal. De estas funciones de activación una de las más conocidas, es la función sigmoide, representada como

$$Out = \frac{1}{1 + e^{-\lambda a}} \qquad (1.1)$$

Existen otras funciones, pero se utilizan las caracteristicas no lineales, ya que la gran mayoria de los problemas presentan un comportamiento no lineal. Una red neuronal con neuronas lineales solo puede resolver problemas que son linealmente separables, y por lo tanto, presentan un comportamiento lineal. Entre las funciones de activación más utilizadas están:

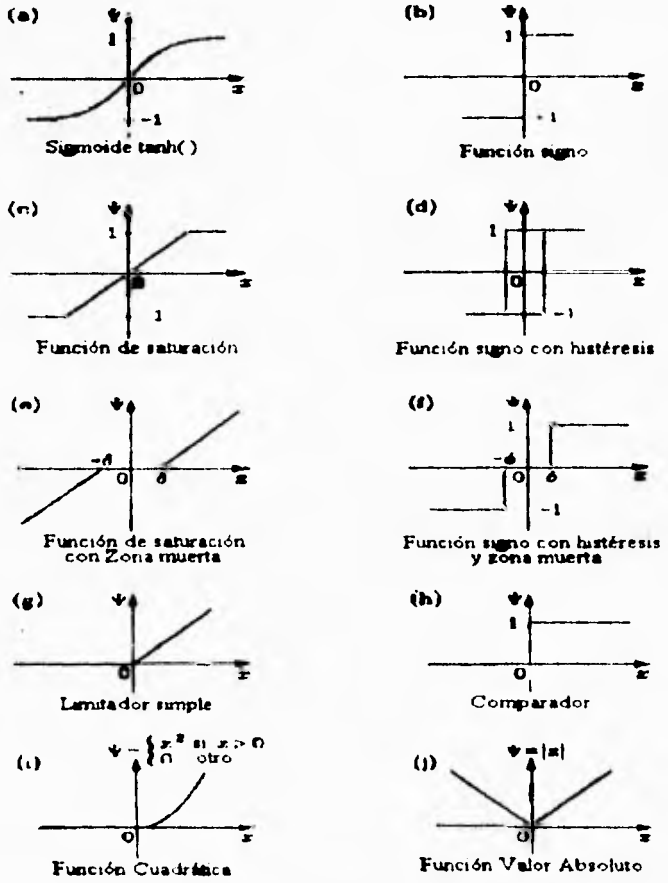


Figura 1.4 Algunas funciones de activación.

Los modelos de redes neuronales son más utilizados en áreas tales como lenguaje y reconocimiento de imágenes, donde se requieren grandes cálculos, y los mejores sistemas

actuales están lejos de igualar el desarrollo humano, las redes neuronales o neurocomputadoras, presentan la ventaja de manejar la información en paralelo, además que las tareas que va a ejecutar, las realiza mediante un "entrenamiento" y "aprendizaje" previo. En lugar de realizar un programa de instrucciones secuencialmente, como en una computadora de Von Neumann, los modelos de redes neuronales usan redes fuertemente paralelas, compuestas de algunos elementos computacionales, conectados por enlaces con pesos variables.

Los elementos computacionales o nodos usados en modelos de redes neuronales son no lineales, son típicamente analógicos, y pueden ser comparados a modernos circuitos digitales. El nodo más simple, suma N entradas con sus respectivos pesos y pasa el resultado a través de una no linealidad (función de activación). El nodo es caracterizado por un umbral interno y por el tipo de no linealidad.

Los modelos de redes neuronales son especificados por la topología de la red, las características del nodo y las reglas de entrenamiento y aprendizaje. Estas reglas especifican un conjunto inicial de pesos, e indican como los pesos deben adaptarse para mejorar su desarrollo. Ambos procedimientos de desarrollo y reglas de entrenamiento, son los tópicos de muchas investigaciones actuales.

Los beneficios potenciales de las redes neuronales, además de las tasas de cálculos provistas por el paralelismo masivo, es que proveen un grado más alto de robustez en comparación con las computadoras secuenciales de Von Neumann. Esto es porque hay más nodos de procesamiento, cada uno con conexiones principalmente locales Algunos algoritmos de redes neuronales también adaptan la conexión de pesos en tiempo, para mejorar el desempeño basado en resultados actuales. La habilidad de adaptar y continuar aprendiendo, es esencial en áreas tal como reconocimiento de voz, donde el entrenamiento

de datos es ilimitado ya que se encuentran nuevas voces, palabras, dialectos, frases y entornos.

1.4. Antecedentes en Neurocomputación

El desarrollo de modelos metemáticos en redes neuronales empezó hace más de 40 años con el trabajo de McCulloch y Pitts, Hebb, Rosenblatt y Widrow; trabajos más recientes por Hopfield, Rumelhart y McClelland, Sejnowski, Feldman, Grossberg y otros, han permitido un resurgimiento en el campo, así como el desarrollo de nuevas redes topológicas y nuevas técnicas analógicas de implantación VLSI. Las redes neuronales proveen una técnica para obtener la capacidad de procesamiento requerido, usando un gran número de elementos de procesamiento operando en paralelo.

A continuación se mencionan seis modelos de redes neuronales que pueden ser usados para clasificación de patrones.

1.5. Redes Neuronales y su Clasificación

La clasificación de redes neuronales son presentadas en el siguiente cuadro:

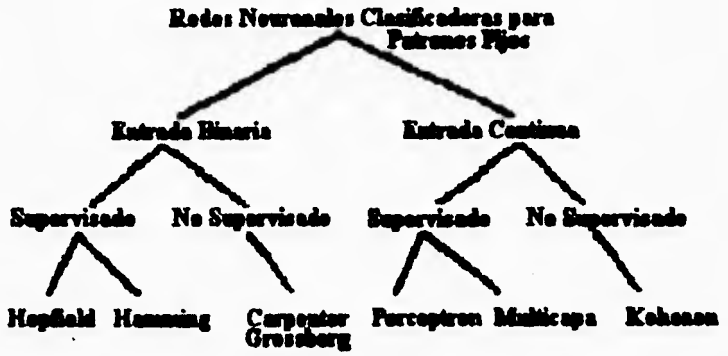


Figura 1.5 Clasificación de redes

Ambos tipos de clasificaciones, determinan cual de las M clases es la más representativa de un modelo de entrada estático desconocido, contando con N elementos de entrada. En un reconocedor de voz, las entradas pueden ser la salida de los valores de un analizador espectral en un instante de tiempo y las clases pueden representar diferentes volúmenes. En un clasificador de imágenes, la entrada puede ser los niveles de escala de gris para cada uno de los pixeles en el dibujo, y las clases pueden representar diferentes objetos.

La clasificación tradicional contiene dos estados. El primero calcula puntos para cada clase y el segundo elige la clase con el puntaje máximo. Las entradas al primer estado, son símbolos que representan valores de N elementos de entrada. Un algoritmo calcula un puntaje para cada una de las M clases, los cuales indican que tan cercana está la entrada para el modelo de esa clase. Este modelo ejemplar es el modelo más representativo de cada clase. Los puntajes son codificados en una representación simbólica y pasan secuencialmente a un segundo estado de la clasificación. Aquí son decodificados, y la clase con el puntaje mayor

es elegida. Una representación simbólica de esa clase se envia para completar la clasificación.

En una red neuronal adaptable que clasifica, los valores de entrada son alimentados en paralelo al primer estado por N conexiones de entrada. Cada conexión tiene un valor analógico el cual puede tomar dos niveles de entrada binaria o puede variar sobre un rango para valores continuos de entrada. El primer estado, calcula el puntaje y envia estos marcadores en paralelo al siguiente estado sobre M lineas analógicas de salida, y se elige el valor máximo de estos valores. El segundo estado tiene una entrada para cada una de las M clases. Una vez que la clasificación ha terminado, sólo la salida correspondiente a la clase más cercana estará en "alto" o encendido; las demás salidas estarán en "bajo" o apagadas. En el diseño, hay salidas para cada clase y esta gran variedad de salidas debe guardarse en estados de procesamiento, cada vez que las clases se consideran diferentes. En el sistema de clasificación más simple, estas lineas de salida pueden ir directamente conectadas a luces con niveles que especifican la identificación de clases, en casos más complicados, pueden pasar por estados de procesamiento, donde las entradas de otras modalidades o dependencias temporales son tomadas en consideración. Si se obtuvo la clase correcta, entonces esta información y las salidas del clasificador, pueden ser retroalimentadas al primer estado del clasificador para adaptar pesos usando un algoritmo de aprendizaje. La adaptación dará una respuesta correcta más cercana, para modelos de entrada exitosas similares a los modelos actuales.

Una red neuronal de clasificación puede realizar tres diferentes tareas: (1) como se describió anteriormente, pueden identificar que clase representa mejor un modelo de entrada, donde se supone que las entradas han sido corruptas por ruido o algún otro proceso. Este es una problema clásico de teoría de decisión; (2) el clasificador puede ser usado como un contenido direccionable o una memoria asociativa, donde se busca un

modelo ejemplar y el modelo de entrada es usado para determinar cual ejemplar producir. Una memoria de contenido direccionable se usa cuando sólo se cuenta con una parte del modelo de entrada y se desea obtener el modelo completo, como por ejemplo, en recuperación bibliográfica de referencias de revistas de información parcial; lo cual normalmente requiere de un tercer estado para regenerar el ejemplar para la clase más probable. Algunas redes neuronales no requieren este estado adicional, tal como la red de Hopfield, la cual es diseñada específicamente como memoria de contenido direccionable y (3) este clasificador puede agrupar las N entradas en M grupos. Este tipo de clasificadores son usados en imágenes y sistemas de transmisión de voz, para reducir el número de bits necesarios para transmitir datos analógicos. En aplicaciones de reconocimiento de voz e imágenes, son usados para comprimir la cantidad de datos que deben ser procesados sin perder información importante. En ambas aplicaciones el número de grupos puede ser especificado previamente, o puede crecer a un límite determinado por el número de nodos disponibles en el primer estado. Esta clasificación se divide en redes con entradas de (a) valores continuos y (b) binarios; y estas a su vez en aquellas que entrenan con o sin supervisión.

1.6. Paradigmas de las Redes Neuronales

Las redes que entrenan sin supervisión, tal como la red de Hopfield o perceptrón, son usados como memorias asociativas o como clasificadores. Estas redes están provistas de niveles que especifican la clase correcta para nuevos modelos de entrada durante el entrenamiento. Algunos clasificadores estadísticos tradicionales, tal como los clasificadores Gaussianos, son entrenados con supervisión usando datos de entrenamiento clasificados. Las redes entrenadas sin supervisión, tal como las características de mapeo de Kohonen formando redes, son usados como cuantificadores de vectores o para formar grupos.

La red de Hamming es una realización de la red neuronal de la clasificación óptima para modelos binarios corruptos por ruido aleatorio. En otros casos los algoritmos de redes neuronales son diferentes de los algoritmos clásicos, por ejemplo, el entrenamiento del perceptrón con el procedimiento de convergencia del perceptrón, se comporta de forma diferente a los clasificadores gaussianos. La red de Kohonen presenta cada modelo nuevo sólo una vez y los pesos son modificados después de cada presentación, esta red forma un número preespecificado de grupos

1.6.1. Red de Hopfield

La red de Hopfield normalmente utiliza entradas binarias; este tipo de redes son más apropiadas cuando la representación binaria es con imágenes en blanco y negro, es decir, los elementos de entrada son valores de pixeles, o código ASCII para representar caracteres de 8-bits. Estas redes son menos apropiadas cuando los valores de entrada son continuos, porque el problema de representación debe ser enfocado para convertir las cantidades analógicas a valores binarios

Esta red puede ser usada como una memoria asociativa o para resolver problemas de optimización, tiene N nodos conteniendo límites de no linealidad, entradas y salidas binarias tomando los valores de +1 y -1. La salida de cada nodo es retroalimentada a todos los nodos por pesos denotados por t_{ij}. La operación de esta red se describe en el cuadro 1. Primero, los pesos se inicializan para todas las clases; un modelo desconocido se presenta a la red en un tiempo cero, forzando que la salida de la red iguale el modelo desconocido. Siguiendo esta inicialización, la red itera en pasos de tiempo discreto usando una formula dada, se considera que la red ha convergido cuando no cambian las salidas en iteraciones sucesivas. El modelo especificado por las salidas del nodo después de la convergencia, es la red de salida

Hopfield y otros han probado que esta red converge cuando los pesos son simétricos, es decir, $t_{ij}=t_{ji}$, y los nodos de salida son actualizados asíncronamente usando las ecuaciones del cuadro 1. Hopfield también demostró que la red converge cuando son utilizadas funciones de no linealidad, similares a la sigmoide. Cuando la red de Hopfield es usada como una memoria asociativa, la red de salida después de la convergencia, es usada como la memoria. Cuando la red de Hopfield es usada como un clasificador, la salida después de la convergencia ,debe ser comparada con los M ejemplares, para determinar si es exactamente igual a un ejemplar, si es igual a un ejemplar, la salida es la clase cuyo ejemplar igualó el modelo de salida, de otra manera el resultado es un "no match"; es decir, no pertenece a ninguna clase.

La red de Hopfield tiene dos limitaciones cuando se usa como contenido de memoria direccionable. Primero: el número de modelos que pueden ser almacenados y ser llamados es limitado. Si son almacenados muchos modelos, la red puede converger a un modelo falso nuevo, diferente de todos los modelos ejemplares. Un modelo falso producirá una salida que no corresponde cuando la red es usada como un clasificador. Hopfield mostró que esto ocurre de manera poco frecuente con modelos ejemplares que son generados en forma aleatoria y el número de clases (M) es menor a 0.15 veces el número de elementos de entrada o nodos en la red (N). El número de clases a ser guardadas es 0.15N; por ejemplo, una red de Hopfield de sólo 10 clases, puede requerir más de 70 nodos y más de 5.000 conexiones de pesos. Una segunda limitación de la red de Hopfield, es que un modelo ejemplar será inestable, si comparte algunos bits en común con otro modelo ejemplar. Aquí un ejemplar es considerado inestable si es aplicado en un tiempo cero y la red converge a otro ejemplar

Cuadro 1.1 Algoritmo de Hopfield

Paso 1 : Asignar pesos (en esta formula t_{ij} es el peso de conexión desde el nodo i hasta el nodo j y X_i^s el cual puede ser +1 o -1 es el elemento i del ejemplar para la clase s):

$$t_{ij} = \sum_{s=0}^{M-1} X_i^s X_j^s \qquad i \neq j$$

$$t_{ij} = 0 \quad i = j,\ 0 \leq i,j \leq M-1$$

Paso 2 : Inicializa con modelos desconocidos de entrada

$\mu_j(0) = x_j, \quad 0 \leq j \leq N-1$

donde $\mu_j(t)$ es la salida del nodo i en tiempo.

Paso 3 : Itera hasta que converja

$$\mu(t+1) = f_h\left[\sum_{j=0}^{N-1} t_{ij}\mu(t)\right], \quad 0 \leq j \leq M-1 \quad \text{y} \quad f_h \text{ es la función de activación.}$$

Paso 4 : Ir a paso 2.

El proceso es repetido hasta que los nodos de salida no cambien

1.6.2. Red de Hamming

La red de Hopfield generalmente se utiliza para aquellos problemas donde las entradas son generadas eligiendo un ejemplar e invirtiendo valores de bits en forma aleatoria e independientemente con una probabilidad dada. Este es un problema clásico en teoría de comunicaciones. El clasificador óptimo de error mínimo en este caso calcula la distancia de Hamming al ejemplar para cada clase y elige la clase con la distancia mínima de Hamming. La distancia de Hamming, es el número de bits diferentes a los bits del ejemplar correspondiente. Una red de Hamming será aquella que implemente este algoritmo usando componentes de redes neuronales.

La operación de la red de Hamming es mostrada en el cuadro 2. Los pesos y umbrales son inicializados primero en la subred más baja (o primera capa), de tal forma que los puntos de comparación generados por las salidas de la mitad de los nodos sea igual a N menos las distancias de Hamming al modelo ejemplar. Estos puntos van en un rango de 0 a un número de elementos en la entrada (N) y son más altos para aquellos nodos correspondientes a clases con ejemplares que se acercan más a la entrada. Los umbrales y pesos en la subred MAXNET (o segunda capa) son arreglados, esto es, todos los umbrales son puestos en cero y los pesos de cada nodo conectado a si mismo con un valor de 1. Los pesos entre los nodos son inhibitorios con un valor de $-\varepsilon$ donde $\varepsilon < 1/M$.

Una vez que los pesos y umbrales han sido inicializados, se presenta un modelo binario con N elementos al inicio de la red de Hamming. La entrada es removida y MAXNET itera hasta que la salida de sólo un nodo es positiva. La clasificación se completa y la clase elegida es aquella que corresponde al nodo con la salida positiva.

La red de Hamming tiene varias ventajas sobre la red de Hopfield. Implementa el error mínimo óptimo clasificador, cuando los bits de error son aleatorios y de este modo el desempeño de la red de Hopfield será peor que, o equivalente a la red de Hamming en tales

situaciones. Comparaciones realizadas entre las dos redes, en problemas tales como reconocimiento de caracteres, reconocimiento de modelos aleatorios, y referencias bibliográficas han demostrado esta diferencia en desempeño. La red de Hamming también requiere menos conexiones que la red de Hopfield; por ejemplo, con 100 entradas y 10 clases, la red de Hamming requiere sólo 1100 conexiones, mientras que la red de Hopfield requiere casi 10,000. Aunque la diferencia en número de conexiones se incremente cuando el número de entradas se incrementa, porque el número de conexiones en la red de Hopfield crece como el cuadrado del número de entradas, mientras el número de conexiones en la red de Hamming crece linealmente. La red de Hamming también puede ser modificada para tener un clasificador de error mínimo, cuando los errores son generados por inversión de elementos de entrada de +1 a -1 y de -1 a +1 asimétricamente con probabilidades diferentes y cuando los valores de entrada específicos son desconocidos. Finalmente, la red de Hamming no obtiene modelos falsos de salida, los cuales puede producir un resultado que no corresponde.

Esto puede hacerse usando un arreglo de nodos fuertemente limitados con umbrales internos a los valores de umbral deseados. Las salidas de estos nodos serán -1 a menos que las entradas excedan los valores de umbral. Alternativamente, los umbrales pueden tener una entrada común inhibitoria, alimentando a todos los nodos. Este umbral puede variar hasta que la salida de un sólo nodo sea positiva.

Cuadro 1.2 Algoritmo de Hamming

Paso 1 : Asignar Pesos de Conexión

capa l:

$$w_{ij} = \frac{x_i^l}{2}, \theta_j = \frac{N}{2},$$

$$0 \le i \le N-1, 0 \le j \le M-1$$

capa w:

$$t_{kl} = \begin{cases} 1, k = l \\ -\varepsilon, k \ne l, \varepsilon \prec \frac{1}{M}, \end{cases}$$

$$0 \le i \le N-1, 0 \le j \le M-1$$

Paso 2: Inicializa con un modelo desconocido

$$\mu_j(0) = f_t\left(\sum_{i=0}^{N-1} w_{ij} x_i - \theta_j\right)$$

$$0 \le i, j \le M-1$$

$\mu_j(t)$ = salida del nodo j; x_i = entrada; f_t = función de activación no lineal

Paso 3 : Itere hasta que converja

$$\mu_j(t+1) = f_t\left(\mu_j(t) - \varepsilon \sum_{k \ne j} \mu_k(t)\right)$$

$$0 \le j,k \le M\text{-}1$$

Este proceso se repite hasta que converge cuando sólo un nodo permanece positivo

Paso 4. Ir a paso 2.

1.6.3. El Clasificador de Carpenter/Grossberg

Carpenter y Grossberg, en el desarrollo de su teoria de resonancia adaptiva, diseñaron una red la cual forma grupos y entrena sin supervisión. Esta red implanta un algoritmo de agrupación, muy similar al algoritmo de agrupación secuencial por lider. El algoritmo del lider elige la primera entrada como el ejemplar para el primer grupo. La siguiente entrada es comparada al primer grupo ejemplar; sigue al lider y es agrupado con el primero si la distancia con el primero es menor que un umbral; de otro modo es el ejemplar para un nuevo grupo. Este proceso es repetido para las demás entradas. El número de grupos crece con el tiempo y depende del umbral y la distancia usada para comparar las entradas con grupos ejemplares.

La estructura de esta red es similar a la red de Hamming, los puntos se calculan usando conexiones de avance y el valor máximo aumenta usando inhibición lateral entre los nodos de salida. Esta red difiere de la de Hamming, en que las conexiones de retroalimentación son provistas desde los nodos de salida a los nodos de entrada.

La red inicializa todos los ejemplares representados por conexión de pesos a cero. Un umbral llamado vigilante con rangos entre 0.0 y 1.0, determina que tan cerca está el modelo nuevo de entrada a los ejemplares almacenados para considerarlo similar a ese modelo. Un valor cercano a uno requiere una semejanza muy cercana y valores más pequeños aceptan una semejanza más pobre. Las entradas son presentadas secuencialmente al inicio de la red, la entrada se compara en forma paralela con todos los ejemplares almacenados, de la misma forma que en la red de Hamming, para producir puntajes semejantes. El ejemplar con el puntaje más alto es elegido usando inhibición lateral. Este es comparado a la entrada, calculando el cociente del producto punto de la entrada y el ejemplar más semejante (número de bits "1" en común) dividido entre el número de bits "1" en la entrada. Si este cociente es más grande que el umbral de vigilancia, entonces la entrada

se considera similar al ejemplar más semejante y ese ejemplar es actualizado realizando una operación lógica AND entre sus bits y los de entrada. Si el cociente es menor que el umbral de vigilancia, entonces la entrada se considera diferente de todos los ejemplares y se agrega como un ejemplar nuevo. Cada ejemplar adicional requiere un nodo y 2N conexiones para calcular el puntaje.

El algoritmo de Carpenter/Grossberg para entradas con una pequeña cantidad de ruido pueden causar problemas, para un modelo sin ruido, el umbral de vigilancia puede ser puesto de tal forma que dos modelos muy similares sean considerados diferentes para modelos con ruido. El número de ejemplares almacenados puede crecer rápidamente hasta que todos los nodos disponibles sean usados, por lo que se pueden hacer algunas modificaciones para incrementar el desempeño de este algoritmo cuando se introduce ruido, el cual puede incluir una adaptación de los pesos más lentamente y cambio del umbral de vigilancia durante el entrenamiento.

Cuadro 1.3 Algoritmo Carpenter/Grossberg

Paso 1: Inicialización:

$$t_{ij}(0) = 1, \qquad b_{ij}(0) = \frac{1}{1+N}$$

$0 \le j \le N-1, \quad 0 \le j \le M-1$

Sea ρ el umbral de vigilancia con $0 \le \rho \le 1$

y bij, tij las cotas superior e inferior de los pesos para cada instante.

b_{ij} , t_{ij} = pesos, p = umbral de vigilancia

Paso 2: Aplicar entrada nueva

Paso 3: Calcular puntajes

$$\mu_j = \sum_{j=0}^{N-1} b_{ij}(t)\, x_i \qquad 0 \le j \le M\text{-}1$$

μ_j = salida del nodo j; x_i = elemento i de la entrada

Paso 4: Elegir el mejor ejemplar

$$\mu_j^* = \max_j \{\mu_j\}$$

Paso 5: Prueba de vigilancia

(1) $\|X\| = \sum_{i=0}^{N-1} X_i$ (2) $\|T \cdot X\| = \sum_{i=0}^{N-1} t_{ij}^* X_i$

¿ $\dfrac{\|T \cdot X\|}{\|X\|} > \rho$?

SI : Ir a paso 6
NO : Ir a paso 7

Paso 6 : Deshabilita el ejemplar más parecido

La salida del nodo elegido como más parecido en el paso 4 es temporalmente puesto a cero y no toma parte en la maximización del paso 4. Ir a 3.

Paso 7: Adapta el ejemplar más parecido

$$t_{i^*}(t+1) = t_{i^*}(t) X_i$$

$$b_{i^*}(t+1) = \frac{t^*_{ij}(t) X_i}{0.5 + \sum_{i=0}^{N-1} t_{i^*}(t) X_i}$$

Paso 8: Ir a paso 2 (habilitar nodos deshabilitados en 6).

1.6.4. Perceptrón de un Nivel

El perceptrón de un nivel puede usar entradas con valores continuos y binarios. Esta red generó mucho interés cuando se desarrollo inicialmente, debido a su habilidad para reconocimiento de patrones sencillos. El nodo calcula la suma de los pesos de los elementos de entrada, resta un umbral (θ) y pasa el resultado a través de un límite de no linealidad para que la salida "y" sea +1 o -1. La regla de decisión responderá clase A, si la salida es +1 y clase B, si la salida es -1. Una técnica útil para analizar el comportamiento de redes, tal como el perceptrón, es dibujar un mapa de regiones de decisión creado en un espacio multidimensional, medido por las variables de entrada. Estas regiones de decisión especifican que valores de entrada pertenecen a la clase A y cuales a la clase B. El perceptrón forma dos regiones de decisión separadas por un hiperplano. Cuando estas son sólo dos entradas y el hiperplano es una línea, las entradas arriba del límite de la línea pertenecen a la clase A y debajo de la línea a la clase B. Como puede verse, la ecuación del límite de la línea, depende en la conexión de los pesos y el umbral.

Las conexiones de los pesos y el umbral en un perceptrón pueden ser arreglados o adaptados usando una gran variedad de algoritmos. El procedimiento original de convergencia del perceptrón para ajustar pesos, fue desarrollado por Rosenblatt. Los primeros pesos de las conexiones y el valor de umbral, son inicializados a pequeños valores aleatorios diferentes de cero; una entrada con N elementos de valores continuos es aplicada a la entrada y se calcula la salida. La conexión entre los pesos se cambia sólo cuando ocurre un error (usando la formula en el paso 4 del cuadro 4). Esta formula incluye un término de ganancia en un rango de 0.0 a 1.0 y controla la tasa de adaptación; esta ganancia debe ajustarse para satisfacer los requerimientos de adaptación para cambios reales en las distribuciones de entrada y promedio de las entradas pasadas, para proveer estimación de pesos estables

La estructura del perceptrón puede ser usada para realizar ya sea un clasificador Gausiano de máxima verosimilitud o varios, los cuales usan el algoritmo de entrenamiento del perceptrón o uno de sus variantes. La elección depende de su aplicación. El algoritmo de entrenamiento del perceptrón no hace suposiciones concernientes a la configuración de distribuciones fundamentales, pero se centra en errores que ocurren donde las distribuciones se traslapan. De este modo, puede ser mejor que técnicas clásicas y trabajar bien cuando las entradas son generadas por procesos no lineales. El clasificador Gausiano hace fuertes suposiciones acerca de distribuciones fundamentales y es más apropiado cuando las distribuciones son conocidas e igualan la posición gaussiana. El algoritmo de adaptación, definido por el procedimiento de convergencia del perceptrón, es sencillo de implementar y no requiere almacenar más información de la que se presenta en los pesos y los umbrales. El clasificador Gaussiano puede ser adaptable, pero debe almacenar la información extra y los cálculos son más complejos. El procedimiento de convergencia del perceptrón no es apropiado cuando las clases no pueden ser separados por un hiperplano.

Cuadro 1.4 Perceptrón

Si M_{Ai} y σ^2_{Ai} son la media y varianza de la entrada x_i, cuando la entrada es de la clase A y M_{Bi} y σ^2_{Bi} de la entrada x_i de la clase B; y $\sigma_i^2 = \sigma^2_{Ai} = \sigma^2_{Bi}$, entonces los valores de probabilidad son:

$$L_A = -\sum_{i=0}^{N-1} \frac{(x_i - M_{Ai})^2}{\sigma_i^2} = -\sum_{i=0}^{N-1} \frac{x_i^2}{\sigma_i^2} + 2\sum_{i=0}^{N-1} \frac{M_{Ai} x_i}{\sigma_i^2} - \sum_{i=0}^{N-1} \frac{M_{Ai}^2}{\sigma_i^2}$$

y

$$L_B = -\sum_{i=0}^{N-1} \frac{(x_i - M_{Bi})^2}{\sigma_i^2} = -\sum_{i=0}^{N-1} \frac{x_i^2}{\sigma_i^2} + 2\sum_{i=0}^{N-1} \frac{M_{Bi} x_i}{\sigma_i^2} - \sum_{i=0}^{N-1} \frac{M_{Bi}^2}{\sigma_i^2}$$

↑ Término I ↑ Término II ↑ Término III

Un clasificador de máxima probabilidad debe calcular L_A y L_B, y elegir la clase con más probabilidad, el segundo término es el producto de la entrada por los pesos y el tercer término es una constante obtenida del umbral en un nodo tipo perceptrón. Para dos clases se tiene:

$$W_i = \frac{2(M_{Ai} - M_{Bi})}{\sigma_i^2}, \theta = \sum_{i=0}^{N-1} \frac{M_{Ai}^2 - M_{Bi}^2}{\sigma_i^2}$$

1.6.5. Perceptrón Multinivel

Los perceptrones multinivel son redes de avance con uno o más capas de nodos entre la capa de entrada y la de salida, que no están directamente conectados a los nodos de entrada y salida. Los perceptrones multinivel superan algunas de las limitaciones de los perceptrónes de un sólo nivel, pero no fueron usados antes, porque no habia algoritmos de entrenamiento efectivos. Esto ha cambiado recientemente con el desarrollo de nuevos

algoritmos de entrenamiento. Aunque no puede ser probado que estos algoritmos converjan como con perceptrónes de un nivel, se ha mostrado que tienen éxito para algunos problemas de interés.

La capacidad de los perceptrones multinivel proviene de la no linealidad usada con nodos. Si los nodos fueran elementos lineales, entonces una red de un nivel con pesos elegidos apropiadamente pueden duplicar los cálculos realizados por una red multinivel.

Un perceptrón de un sólo nivel forma regiones de decisión de 2 planos. Un perceptrón de dos niveles puede formar cualquier posibilidad ilimitada abarcado por las entradas, tales regiones incluyen polígonos convexos, algunas veces llamados cascos convexos. Aquí el término convexo significa que cualquier línea uniendo puntos en el borde de una región, va sólo a través de puntos dentro de esa región. Cada nodo en el primer nivel se comporta como un perceptrón de un sólo nivel y tiene una salida alta sólo para puntos en un lado del hiperplano formado por sus pesos y offset. Si los pesos de un nodo de salida de N_1 nodos del primer nivel son todos 1.0, y el umbral en la salida del nodo es $N_1-\varepsilon$ donde $0 < \varepsilon < 1$, entonces el nodo de salida será alto, sólo si las salidas de todos los nodos del primer nivel son altas. Esto corresponde a realizar una operación lógica AND en la salida del nodo y resulta en una región de decisión final que es la intersección de todas las regiones del plano medio formadas en el primer nivel. Las intersecciones de esos planos medios, forman regiones convexas como se describió anteriormente. Estas regiones convexas tienen tantos lados como nodos del primer nivel.

Este análisis provee alguna introducción al problema de elegir el número de nodos a usar en un perceptrón de dos niveles. El número de nodos debe ser lo suficientemente grande para formar una región de decisión tan compleja como se requiera por el problema dado. No debe ser muy grande ya que algunos pesos no pueden ser estimados de datos de

entrenamiento disponibles Por ejemplo, dos nodos son suficientes para resolver el problema de OR exclusivo.

Un perceptrón de tres niveles puede formar regiones de decisión arbitrariamente complejas. La prueba depende en hacer una partición la región de decisión deseada en pequeños hipercubos (cuadros donde hay dos entradas),cada hipercubo necesita 2N nodos en el primer nivel (4 nodos cuando hay dos entradas), uno para cada lado del hipercubo, y un nodo en el segundo nivel que toma el AND lógico de las salidas de los nodos del primer nivel. Las salidas de los nodos del segundo nivel serán alto, sólo para entradas dentro de cada hipercubo Los hipercubos son asignados a las regiones de decisión apropiadas, conectando la salida de cada nodo del segundo nivel sólo al nodo de salida correspondiente a la región de decisión; esos hipercubos de los nodos es una operación lógica OR en cada nodo de salida. Una operación lógica OR se llevará a cabo si estas conexiones de los pesos del segundo nivel oculto, al nivel de salida son uno y los umbrales en la salida de los nodos es 0.5. Este procedimiento de construcción puede generalizarse para usar regiones convexas en lugar de pequeños hipercubos y es capaz de generar regiones desconectadas y no convexas.

El análisis anterior, demuestra que no más de tres niveles son usados en redes de perceptrones retroalimentados, porque una red de tres niveles puede generar arbitrariamente regiones de decisión complejas También provee alguna introducción al problema de elegir el número de nodos a usar en un perceptrón de tres niveles El número de nodos en el segundo nivel, debe ser mayor que uno, cuando las regiones de decisión son desconectadas y no pueden ser formadas de un área convexa El número de nodos requeridos para el segundo nivel, en el peor de los casos, es igual al número de regiones desconectadas en distribuciones de entrada El número de nodos en el primer nivel debe ser suficiente para proveer tres o más margenes para cada area convexa generada por cada nodo del segundo

nivel. De este modo debe haber más de tres veces de nodos en el segundo como en el primer nivel.

El algoritmo de retropropagación descrito en el cuadro 5 es una generalización de algoritmos de mínimos cuadrados (LMS least mean square). Usa una técnica de búsqueda gradiente para minimizar una función de costo, igual a la diferencia del cuadrado de la media entre la salida de la red deseada y la actual. La salida deseada de todos los nodos es generalmente baja (0 ó < 0.1), a menos que ese nodo corresponda a la clase de entrada actual, es en tal caso alto (1 ó > 0.9). La red es entrenada inicialmente, eligiendo pesos aleatorios pequeños y umbrales internos; se presentan todos los datos de entrenamiento repetidamente. Los pesos son ajustados después de cada prueba, especificando la clase correcta, hasta que los pesos converjan y la función de costo sea reducida a un valor aceptable. Un componente esencial del algoritmo, es el método iterativo descrito en el cuadro, que propaga términos de error requeridos para adaptar pesos de atrás, desde los nodos en el nivel de salida, a nodos en niveles más bajos.

El algoritmo de retropropagación ha sido probado con un gran número de problemas determinísticos, tal como el problema de OR exclusivo, así como en problemas relacionados con síntesis de voz y reconocimiento, y en problemas relacionados con reconocimiento visual de patrones. Se ha encontrado que es efectivo en algunos casos, para encontrar buenas soluciones a los problemas planteados. Una demostración del poder de este algoritmo, fue probado por Sejnowski, el cual entrenó un perceptrón de dos niveles con 120 unidades escondidas y más de 20,000 pesos para formar letras a reglas de transcripción de fonémas. La entrada a esta red fue un código binario, indicando aquellas letras en una ventana corrediza de siete letras de largo, que fueron movidas sobre una transcripción escrita del texto hablado. La salida deseada era un código binario indicando la transcripción fonética de la letra al centro de la ventana. Después de 50 veces a través de un dialogo

conteniendo 1024 palabras, la tasa de error de transcripción fue sólo 5 % Esta incremento a 22% para una continuación de un dialogo que no fue usado durante entrenamiento.

El mejor desempeño encontrado para el algoritmo de retropropagación es algo sorprendente, considerando que es una técnica de búsqueda que puede encontrar un mínimo local en la función de costo mediante LMS, en lugar del mínimo global deseado. Algunas sugerencias para mejorar el desempeño y reducir la aparición del mínimo local incluido, permitiendo más unidades ocultas, disminuyendo el límite de ganancia usado para adaptar los pesos, y haciendo algunas corridas de entrenamiento empezando con diferentes conjuntos de pesos aleatorios. Cuando es usado con problemas de clasificación, el número de nodos puede ser elegido utilizando las consideraciones descritas arriba. El problema del mínimo local en este caso corresponde a agrupar dos o más regiones de diferente clase en una. Esto puede minimizarse usando múltiples salidas con diferentes pesos aleatorios y una baja ganancia para adaptar los pesos. Una dificultad presentada con el algoritmo de retropropagación, es que en algunos casos, el número de presentaciones de los datos de entrenamiento, requeridos para la convergencia, han sido muy largos (mas de 100 pasos a través de todos los datos de entrenamiento). Aunque un gran número de algoritmos de adaptación más complejos, han sido propuestos para una convergencia más rápida, parece poco probable que las regiones de decisión complejas, formadas por perceptrónes multinivel, puedan ser generadas en algunos entrenamientos, cuando las regiones de clases son desconectadas.

Cuadro 1.5 Algoritmo de Retropropagación

Paso 1: Inicializa pesos con pequeños valores aleatorios.

Paso 2: Presenta entrada y salida deseada.

Paso 3: Calcular la salida actual, usando la función no lineal.

Paso 4: Adapta los pesos

$$w_{ij}(t+1) = w_{ij}(t) + \eta \delta_j x'_j$$

aquí $w_{ij}(t)$ es el peso; x'_i es la salida del nodo i
n es la ganancia y δ_j es el error del nodo j.
Si j es un nodo de salida, entonces:

$$\delta_j = y_j(1 - y_j)(d_j - y_j)$$

d_j es la salida deseada; y_j es la salida actual Si j es un nodo oculto, entonces:

$\delta_j = x'_j(1 - x'_j)\sum_k \delta_k w_{jk}$

la convergencia es mas rápida si un termino temporal es agregado y el cambio de pesos es:

$$w_{ij}(t+1) = w_{ij}(t) + \eta \delta_j x'_j + \alpha\left(w_{ij}(t) - w_{ij}(t-1)\right)$$

donde $0 < \alpha < 1$

Paso 5: Ir a paso 2.

Un teorema interesante que da alguna luz en las capacidades de los perceptrones de multinivel, fué probado por Kolmogorov. Este teorema establece que cualquier función continua de N variables, puede ser calculada usando sólo sumatorias lineales y no lineales, pero funciones de incremento continuo de sólo una variable. Declara que un perceptrón de

tres niveles con N(2N+1) nodos, usando no linealidades incrementando continuamente, pueden calcular cualquier función continua de N variables. Un perceptrón de tres niveles, puede de este modo ser usado para crear cualquier función de probabilidad continua requerida en una clasificación. Desafortunadamente, el teorema no indica como deberán ser elegidos los pesos o no linealidades en la red, o que tan sensitiva es la función de salida, a variaciones en los pesos y funciones internas.

2. REDES NEURONALES ARTIFICIALES EN CONTROL

Si oigo, olvido
si veo, recuerdo
si hago, aprendo
-Confucio

Cuando una nueva tecnología surge y se aplica, de inmediato aparecen dos posturas extremas entre sí -ambas negativas desde el punto de vista científico-, la primera defiende a esta nueva rama y es caracterizada por quienes piensan "*X* es el tema del momento, por lo tanto hay que desarrollar el controlador *X*, reportar los resultados como la panacea del área y minimizar sus defectos" (en este caso particular se puede cambiar a *X* por Adaptable, Neuronal, Difuso o Caótico y cubriríamos así los últimos 15 años del área de Control); la segunda postura simplemente considera que no hay nada nuevo bajo el sol y que los nuevos temas de desarrollo e investigación sólo sirven para distraer a los neófitos del área, quienes con el tiempo verán que las líneas clásicas de investigación son el único camino hacia la verdadera Ciencia. Es evidente, desde el punto de vista de la Ciencia que ambas posturas

dañan el avance del conocimiento ya que, como se verá más adelante, una nueva línea de investigación debe estar fundamentada en los conocimientos y avances obtenidos de una o varias de las ya bien desarrolladas lineas clásicas, completando así una estructura del conocimiento en cierta área de investigación.

2.1. Identificación de Sistemas Lineales y Regresión Lineal

En muchas aplicaciones científicas y de ingeniería un sistema con estructura desconocida (planta) tiene señales de entrada y salida observables o medibles. Una forma de obtener información sobre la dinámica de la planta es simularla con un modelo de estructura flexible que imite a la dinámica del sistema desconocido. Aproximando la respuesta del modelo a aquellas mediciones de la planta mediante cambios adaptables en los parámetros del modelo, de tal forma que una función de error (costo) es minimizada.

Identificación del Sistema es el proceso de construir o seleccionar un modelo para un sistema dinámico desconocido estimando los parámetros del modelo a partir de datos experimentales. El modelo paramétrico más sencillo de un sistema dinámico lineal esta dado por la Regresión Lineal del sistema, que puede formularse como

$$y(t) = \omega^T \phi(t) + e(t) \qquad (2.1)$$

donde y(t) es una señal de salida medible, $\phi(t)=[\phi_1(t),\phi_2(t),...,\phi_n(t)]^T$ es el vector de cantidades conocidas llamado **regresor**, $\omega=[\omega_1,\omega_2,...,\omega_n]^T \in \Re^n$ es el vector de parámetros desconocidos a ser determinados o **vector de estimados** y e(t) es una variable que representa ruido o perturbaciones desconocido(as), dependiendo del contexto.

El problema de regresión lineal puede ilustrarse por ejemplos sencillos.

Ejemplo 2.1

(a) Supóngase que el modelo de la señal es

$$y(t) = a_1 e^{-\alpha_1 t} + a_2 e^{-\alpha_2 t} + \ldots + a_n e^{-\alpha_n t} \qquad (2.2)$$

donde los parámetros α_i i = 1,2,..n (los inversos de las constantes de tiempo) son conocidos mientras que los coeficientes a_i i = 1,2,..n son desconocidos. El problema puede formularse como de regresión lineal con

$$\phi(t) = [e^{-\alpha_1 t}, e^{-\alpha_2 t}, \ldots, e^{-\alpha_n t}]^T, \quad \theta = [a_1, a_2, \ldots, a_n]^T \text{ y } \quad e(t) = 0.$$

(b) Considérese el modelo lineal discreto representado por la ecuación en diferencias siguiente:

$$y(n) = -a_1 y(n-1) - a_2 y(n-2) - \ldots - a_N y(n-N) + b_1 x(n-1) + b_2 x(n-2) + \ldots + b_M x(n-M) \qquad (2.3)$$

El modelo puede ser descrito mediante regresión lineal haciendo

$$\phi(n) = [-y(n-1), -y(n-2), \ldots, -y(n-N), x(n-1), x(n-2), \ldots, x(n-M)]^T,$$

$$\theta = [a_1, a_2, \ldots, a_N, b_1, b_2, \ldots, b_M]^T.$$

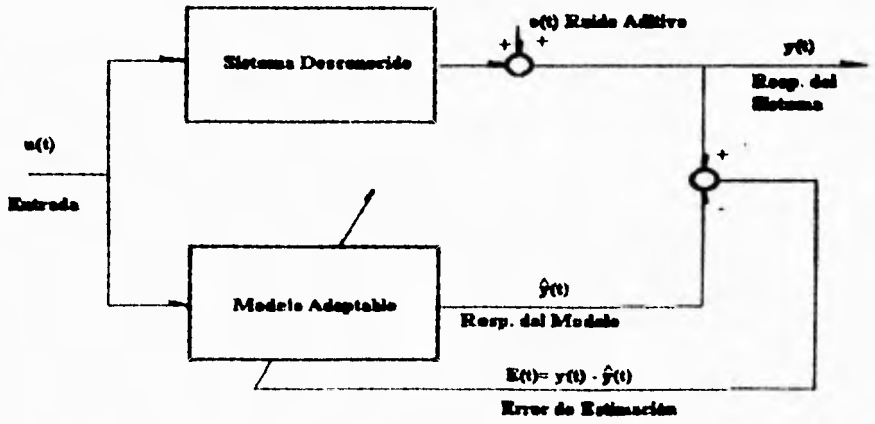


Figura 2.1 Diagrama de Bloques del Problema de Identificación

La regresión lineal puede considerarse como una generalización del problema de decomposición de señal. Sin embargo, en regresión lineal el tiempo t toma valores discretos $t_i = i\tau$, donde τ es el intervalo de muestreo e i es el índice de tiempo o número de ciclo (entero positivo). Las variables continuas en tiempo y(t) y $\phi(t)$ son muestreadas para obtener las variables discretas $y(i\tau)$ y $\phi(i\tau)$, respectivamente. A partir de aquí, se abreviaran $y(i\tau)$ por $y(i) = y^{(i)}$ y $\phi(i\tau)$ por $\phi(i) = \phi^{(i)}$ sin pérdida de generalidad. Como la regresión lineal es determinada en instantes discretos de tiempo se formula como un sistema de ecuaciones lineales

$$y^{(i)} = \omega^T \phi^{(i)} + e^{(i)} \quad (i = 1,2,\ldots,m). \tag{2.4}$$

donde $\omega \in \Re^n$ es el vector de parámetros desconocidos a ser estimados. Usualmente este sistema de ecuaciones lineales es sobre determinado con m >> n dado que el error promedio puede ser reducido a partir de ventanas de medición más amplias. El sistema de ecuaciones lineales anterior puede ser resuelto por una Red Neuronal Artificial (RNA). Especialmente, si el error e(i) es compuesto por ruido Gaussiano de media cero y dispersión conocida (ruido blanco), el problema de regresión lineal puede mapearse a uno de minimización de la función de error (costo) definido por

$$E(\omega) = \sum_{i=1}^{m} \sigma[r_i(\omega)] \tag{2.5}$$

donde

$$r_i(\omega) = y^{(i)} - \omega^T \phi^{(i)} = y(i) - \sum_{j=1}^{n} w_j \phi_j(i) \quad \text{y} \quad \sigma(r_i) = \beta^2 \ln\cosh(r_i/\beta), \text{ con } \beta > 0.$$

La minimización de esta función de error mediante una técnica gradiente descendiente lleva al sistema de ecuaciones diferenciales no lineales

$$\frac{dw_j(t)}{dt} = \mu_j \sum_{i=1}^{m} \psi(r_i)\phi_j(i) \quad j = 1,2,\ldots,n. \tag{2.6}$$

donde $\mu_j > 0$, $\psi(r_i) = \tanh(r_i/\beta)$

Claramente, aplicando la fórmula de Euler podemos transformar el sistema de ecuaciones diferenciales anterior en uno de ecuaciones en diferencias

$$w_j^{k+1} = w_j^k + \alpha^k \sum_{i=1}^{m} \psi(r_i^k)\phi_j(i) \quad k = 0,1,2,... \qquad (2.7)$$

donde $r_i^k = y(i) - \sum_{j=1}^{n} w_j^k \phi_j(i)$ $w_j^k := w_j(t)\Big|_{t=kT}$ y $\alpha^k > 0$ (típicamente $\alpha^k = 1/k^\gamma, \frac{1}{2} < \gamma < 1$).

Con el propósito de mejorar la precisión final en los parámetros estimados se puede agregar una capa auxiliar de filtros de promediación conforme al siguiente algoritmo iterativo

$$\hat{w}_j^{k+1} = \hat{w}_j^k + \frac{1}{k+1}\left(w_j^k - \hat{w}_j^k\right), k = 0,1,2,... \qquad (2.8)$$

Un diagrama de bloques de esta RNA se muestra en la figura 2.2.

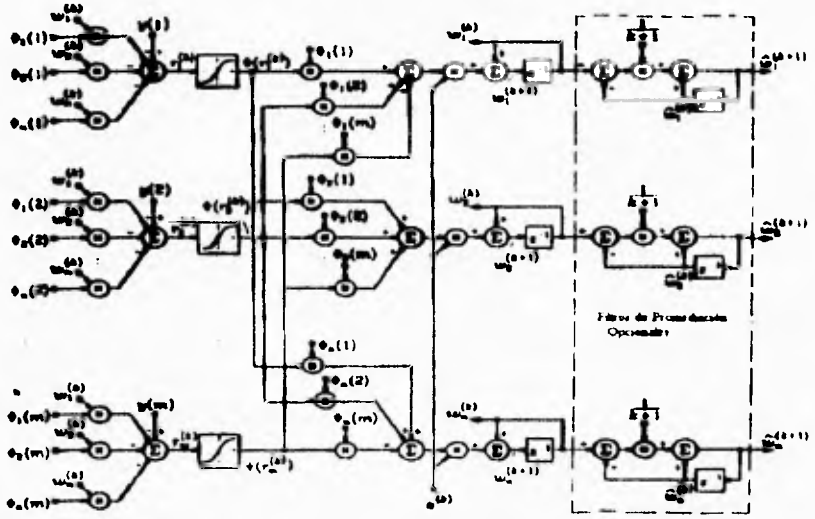


Figura 2.2 RNA para el Problema de Regresión Lineal

El ejemplo anterior es igualmente válido para sistemas estocásticos lineales discretos como para algunos sistemas no lineales que presenten algún comportamiento bilineal, en estos casos el banco de filtros de promediación será necesario pues presentará la estimación óptima por ejemplo para sistemas estocásticos con distribución Gaussiana.

2.2. Detección de Fallas

Cuando se requiere que las variables de estado de un sistema dinámico estén disponibles para realizar en un proceso determinado la detección y aislamiento de fallas, es necesario "observar" los estados a partir de información contenida en la entrada y la respuesta del sistema. El subsistema que realiza la observación de las variables de estado basado en la información recibida de las mediciones de las entrada y las salidas del sistema es conocido como *observador del estado*. Cuando se pretende realizar la detección de fallas en sensores (instrumentos) basta con desarrollar observadores de estado para cada sensor y evaluar los residuos (el error medio entre la respuesta del observador y la del proceso) generados por cada uno de estos observadores.

Dado que una RNA es un buen aproximador a cualquier mapeo acotado, se puede considerar que dicho mapeo es una función vectorial no lineal en general, que representa la relación entrada-salida (dinámica directa) ó salida-entrada (dinámica inversa) de un sistema. Como en el sistema dinámico, la RNA puede establecer internamente la relación entre los estados observables y la entrada al sistema (mediciones que entrenan a la RNA) con los estados no observables (establecidos por la estructura del sistema). Este hecho, permite convertir a la RNA en un observador del estado.

A partir de lo anterior, se procede a sustituir la RNA en un esquema de detección de fallas, si se desea, es también factible elaborar una RNA que permita clasificar los residuos generados por el banco de observadores o bien, conservar la lógica de clasificación clásica. Los resultados de esta aplicación se verán en el siguiente ejemplo.

Ejemplo 2.2

Se tiene un carro que se desplaza a lo largo de una barra guía como se muestra en la siguiente figura.

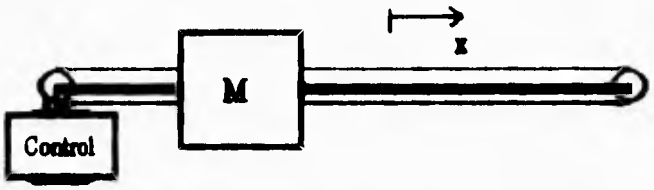


Figura 2.3 Carro con Fricción Seca

Se diseñaron dos RNA de tres capas (5, 15 y 2 neuronas en cada capa, respectivamente) con una función de activación sigmoide (ecuación 1.1) y se entrenaron hasta 35000 iteraciones, la RNA 1 se entrenó para detectar fallas en el sensor de posición y la RNA 2 para el sensor de velocidad, el sistema de carro con fricción es controlado con una retroalimentación del estado suponiendo a la fricción seca como un estado adicional. El patrón de entrenamiento consistio de un desplazamiento de 0 a 30 cm. aún cuando para obtener los resultados siguientes se muestren en un rango de 0.5 a 0.6 por efectos de normalización

- **Para la RNA 1 Detección de Fallas en Posición**

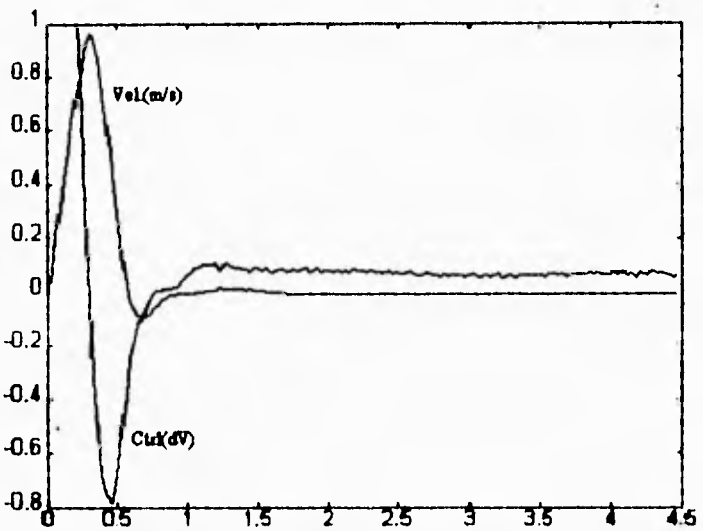


Figura 2.4 Patrón de Entrenamiento (entradas)

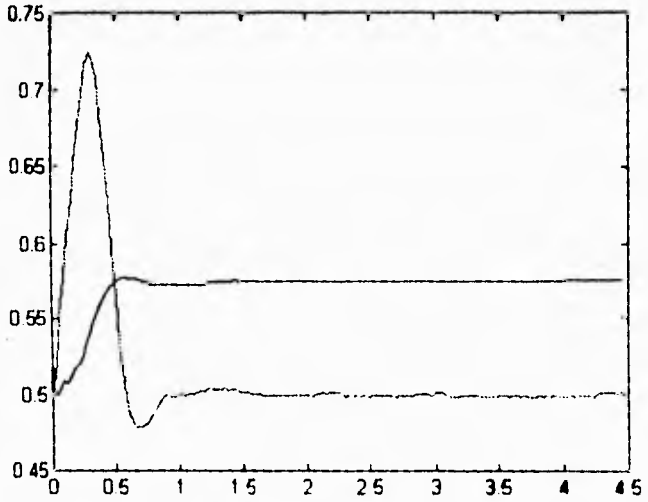


Figura 2.5 Sistema sin Falla

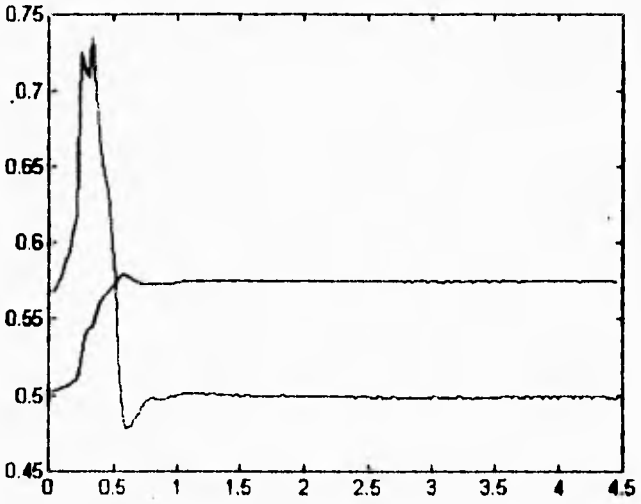


Figura 2.6 Respuesta de RNA 1 a Entrenamiento

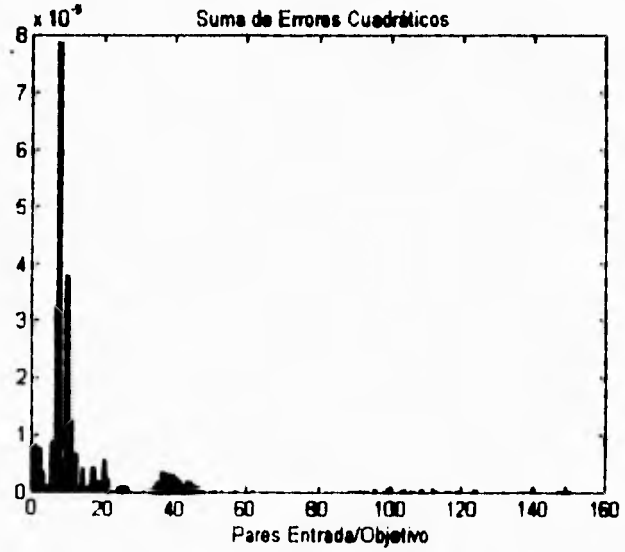


Figura 2.7 Residuo obtenido de la posición RNA 1

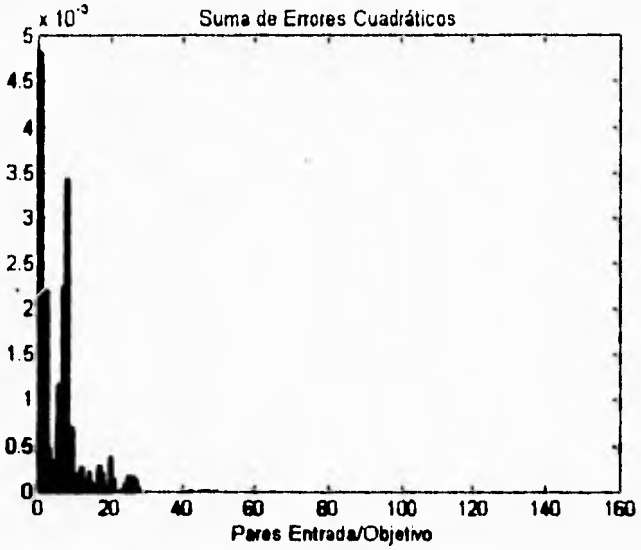


Figura 2.8 Residuo obtenido de la velocidad RNA 1

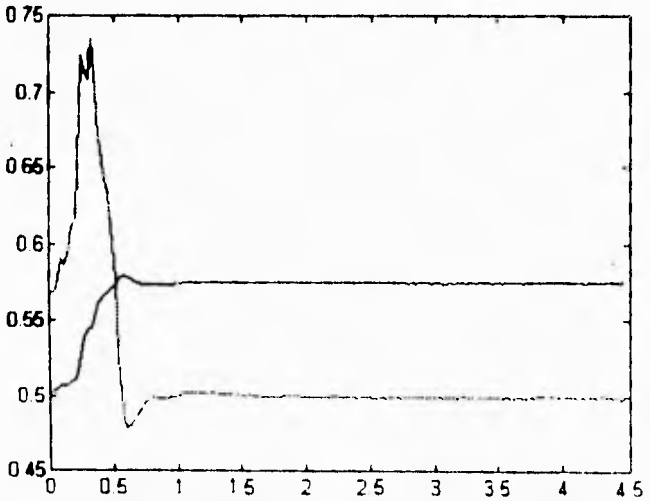


Figura 2.9 Respuesta de RNA 1 a Sistema sin Falla

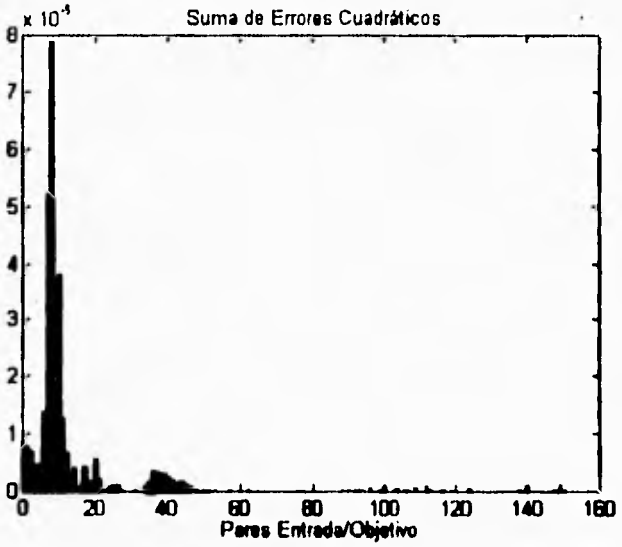


Figura 2.10 Residuo de Posición sin Falla RNA 1

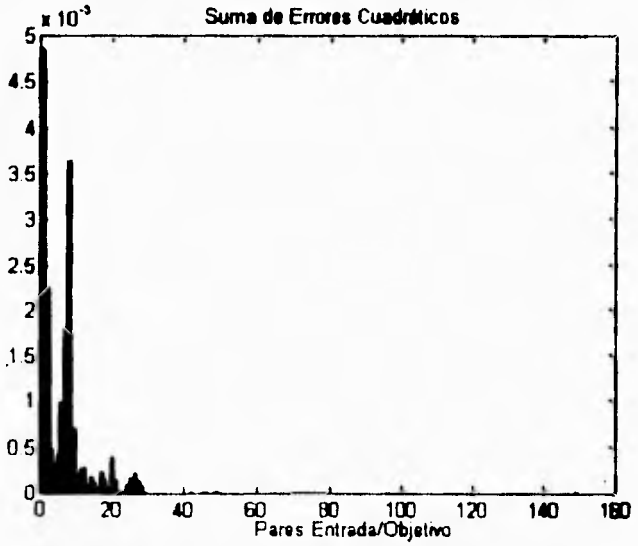


Figura 2.11 Residuo de Velocidad sin Falla RNA 1

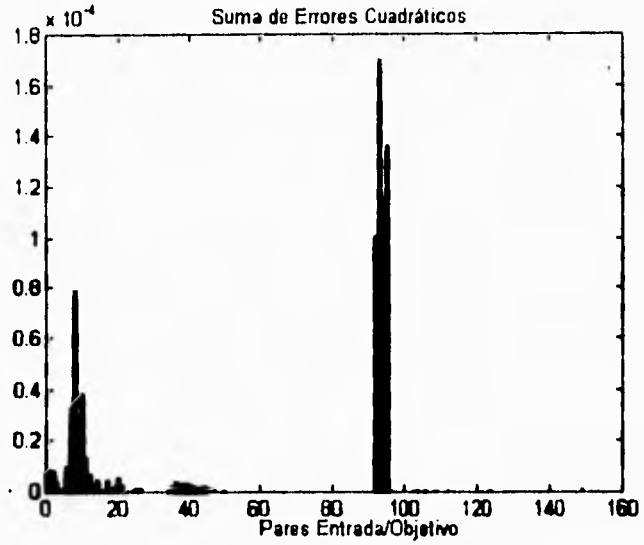


Figura 2.12 Residuo de Posición con Falla RNA 1

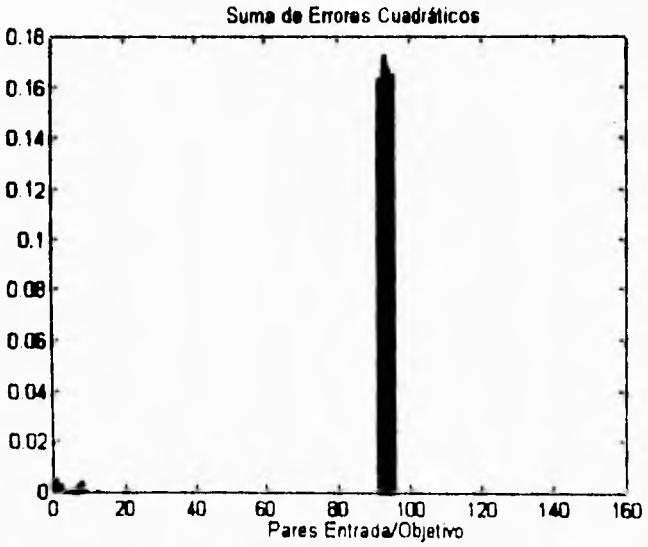


Figura 2.13 Residuo de Velocidad con Falla en Posición RNA 1

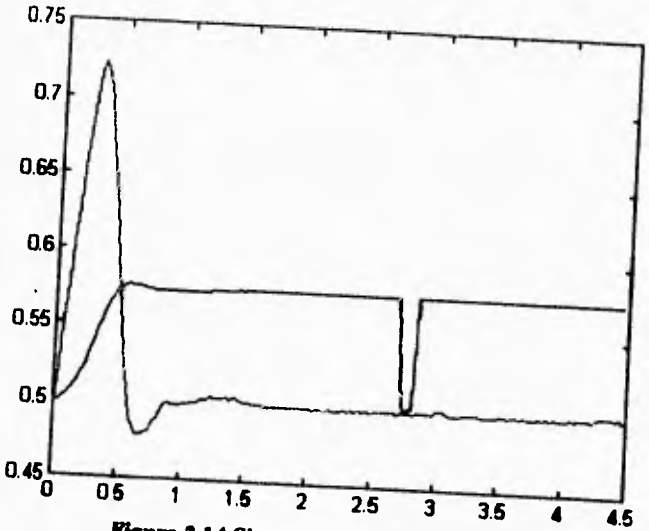


Figura 2.14 Sistema con Falla en Posición

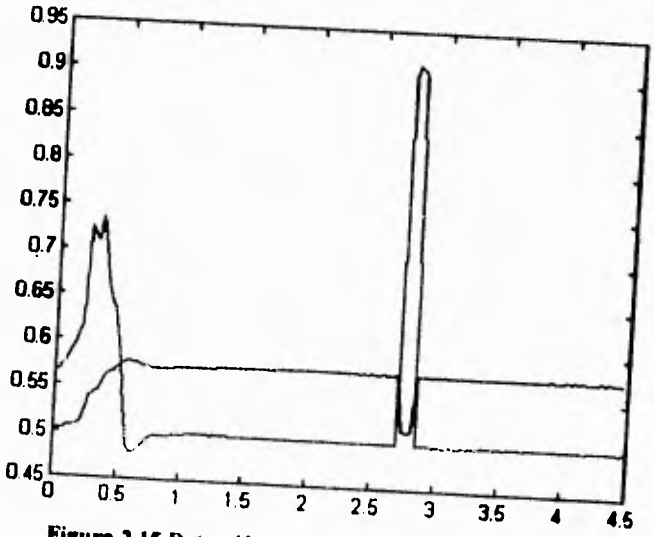


Figura 2.15 Detección de Falla en posición por RNA 1

- **Para la RNA 2 Detección de Fallas en Velocidad**

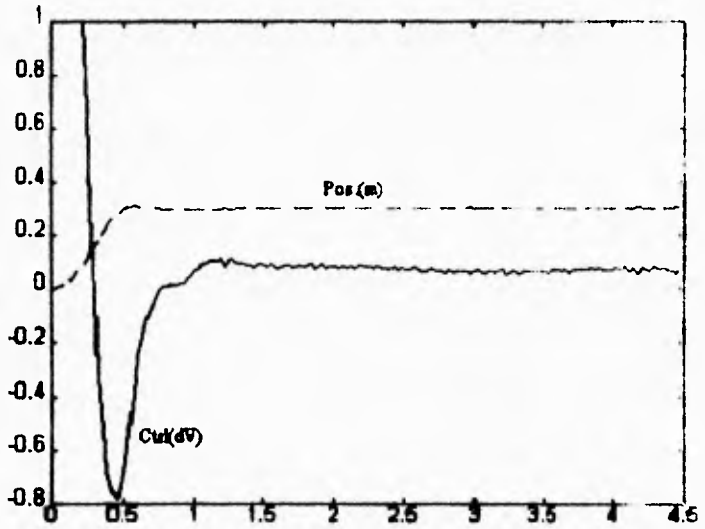


Figura 2.16 Patrón de Entrenamiento (entradas)

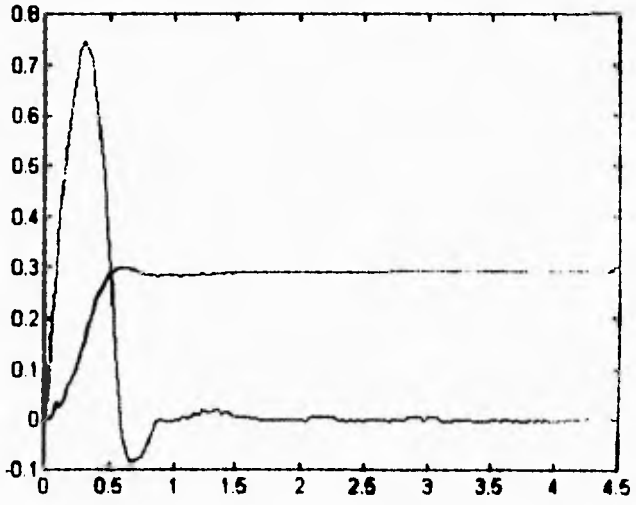


Figura 2.17 Sistema sin Falla

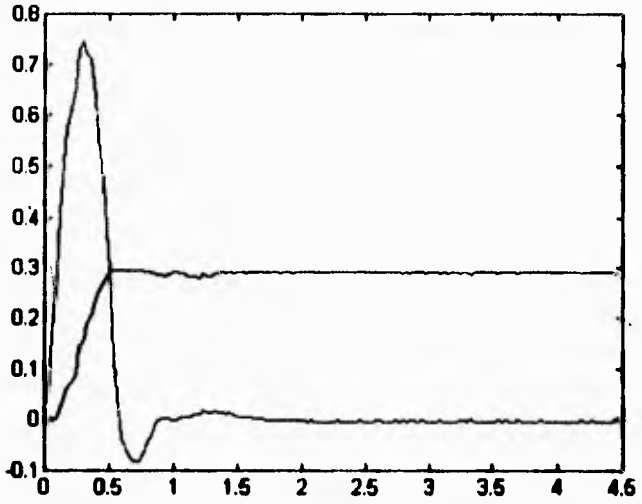


Figura 2.18 Respuesta RNA 2 a Entrenamiento

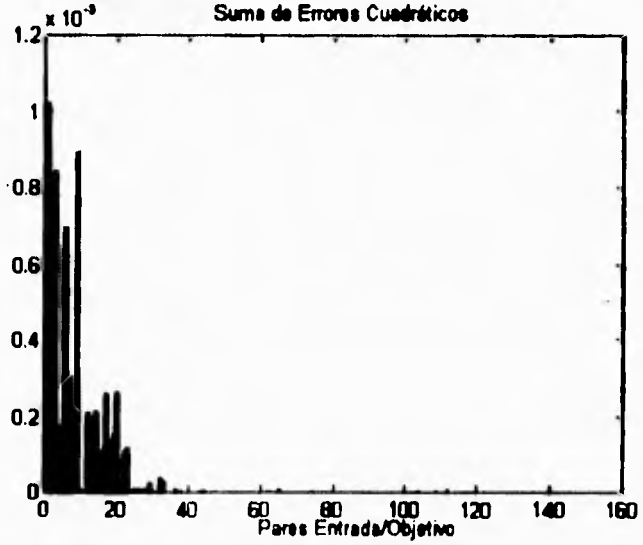


Figura 2.19 Residuo obtenido de la posición RNA 2

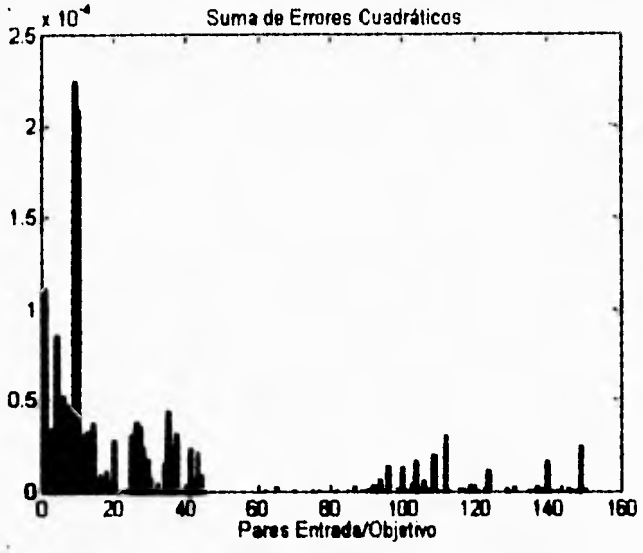


Figura 2.20 Residuo obtenido de la velocidad RNA 2

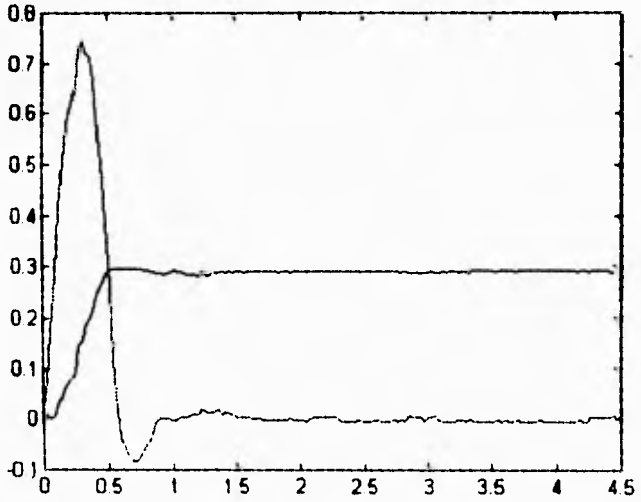


Figura 2.21 Respuesta RNA 2 a Sistema sin Falla

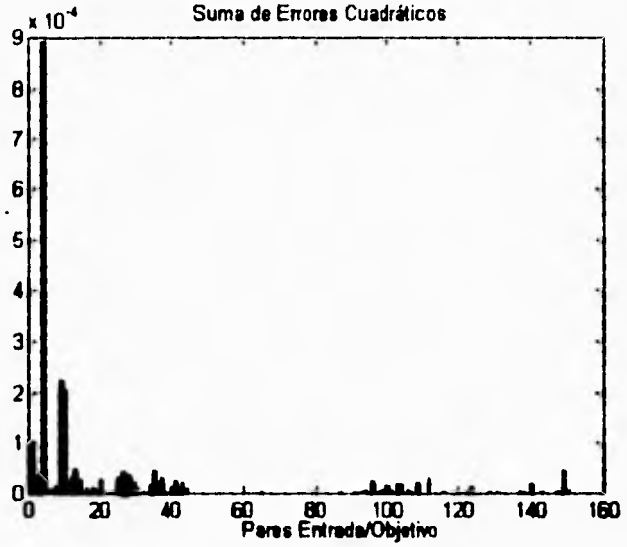


Figura 2.22 Residuo de Posición sin Falla RNA 2

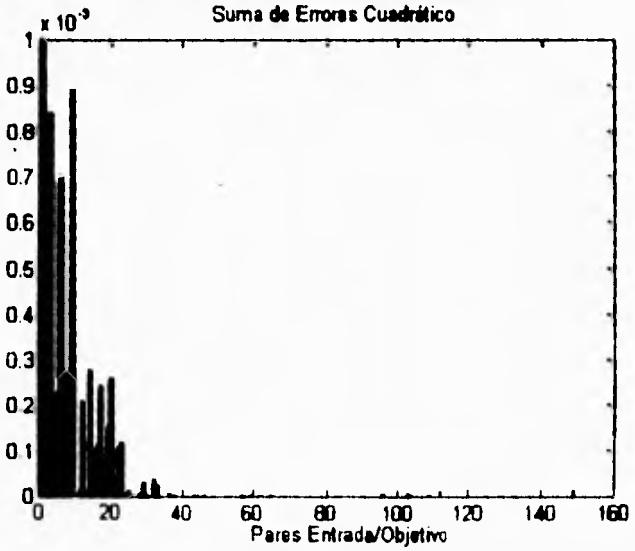


Figura 2.23 Residuo de Velocidad sin Falla RNA 2

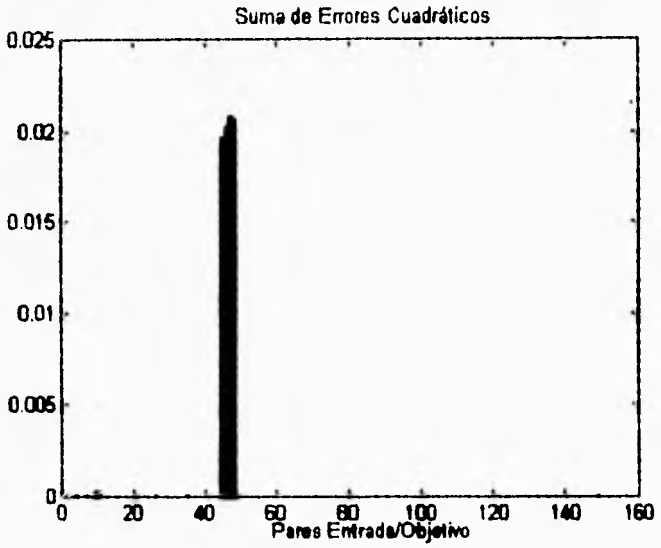


Figura 2.24 Residuo de Posición con Falla en Velocidad RNA 2

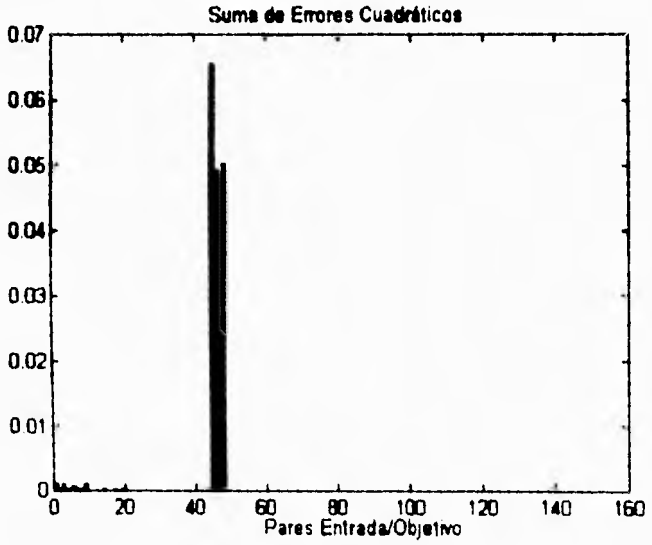


Figura 2.25 Residuo de Velocidad con Falla RNA 2

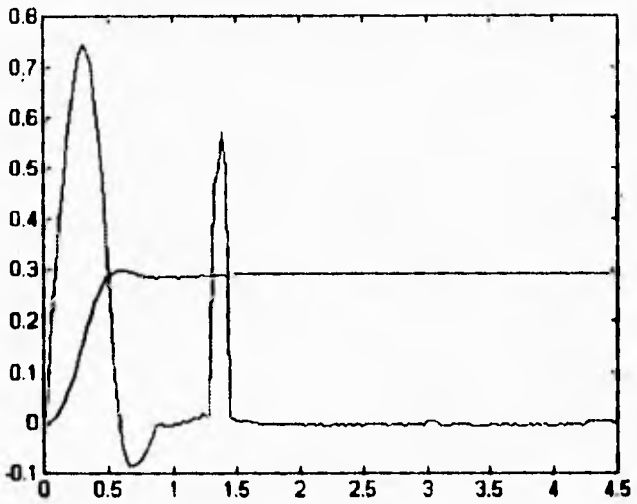


Figura 2.26 Sistema con Falla en Velocidad

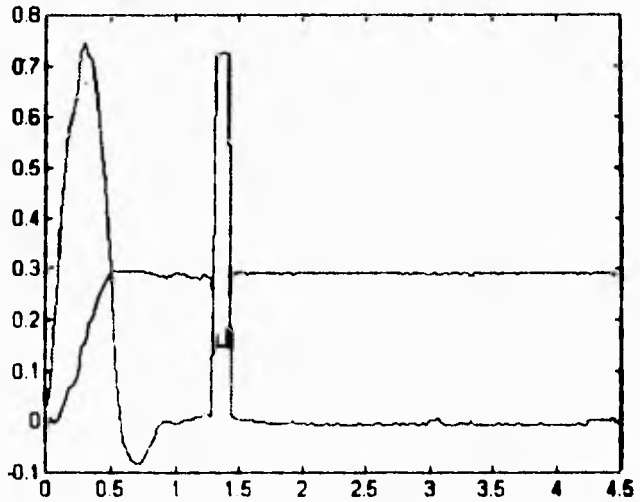


Figura 2.27 Detección de Falla en Velocidad

Los resultados anteriores muestran cómo la estimación del estado es factible a través de RNA, sin la necesidad de asignar una estructura (lineal en este caso), que implique un conocimiento detallado de la dinámica del proceso Sin embargo, es importante hacer notar

que se requiere conocer el grado de correlación entre las señales para poder detectar la falla en uno de los sensores a partir de la señal obtenida del otro sensor. A partir de esto, es evidente que bajo el efecto de falla simultanea en los sensores este esquema de detección de fallas resulta inoperante.

2.3. Aplicaciones

Actualmente, existe una gran cantidad de aplicaciones de RNA en el área de Control, a continuación se presenta una breve descripción de algunos de los trabajos que se consideraron como ejemplos clásicos en el área o bien, por ser una aportación novedosa.

Junhong Nie y D.A. Linkens **A hybrid neural network based self organizing controller** Int. J. Control,vol. 60, 2, Febrero 1994, p.p.197-222	En este estudio se presenta una novedosa construcción de un controlador multivariable basado en una red neuronal híbrida que consiste de una red competitiva de estructura variable y de una red neuronal retropropagada. Se desarrollaron tanto los algoritmos de aprendizaje como las estructuras de auto organización; sin embargo, la representación del conocimiento y el razonamiento son hechos por la estructura de red y no por la interface lógica. Por otro lado, comparada con el paradigma de red neuronal retropropagada, el controlador posee características distintivas, así se tienen las ventajas de la eficiencia computacional y la capacidad de aprendizaje que mantienen la claridad de los paradigmas a través de la red neuronal retropropagada. Los estudios extensivos en problemas multivariables de la presión arterial demuestran no solo la confiabilidad y la eficiencia del sistema, sino que ofrece aspectos que permiten el mejor entendimiento del sistema tales como su adaptabilidad y habilidad de aprendizaje en una amplia gama de situaciones.

Reynold Chu, Rahmat Shoureshi y Manoel Tenorio **Neural Networks for System Identification** IEEE Control Systems Magazine American Control Conference, 1989, p.p.31-35	En esta publicación se tratan dos formas de uso de las redes neuronales para la identificación de sistemas dinámicos. La primera forma es una red neuronal Hopfield que es usada para implementar una estimación mediante mínimos cuadrados para sistemas variantes en el tiempo e invariables en el tiempo. La segunda forma, la cual se usa en el dominio de la frecuencia utiliza un conjunto de funciones de bases ortogonales y análisis de Fourier para construir un sistema dinámico en términos de coeficientes de Fourier.
Jiabin Hao, Shaohua Tan y Joos Vandewalle **A Rule-Based Neural Controller for Inverted Pendulum** IEEE International Joint Conference on N.N., 1993, p.p. 534-539	Este documento demuestra como un control neuronal heuristico puede ser usado para resolver un problema de control complejo. Para poder balancear el péndulo, el controlador requiere mover el carro hacia atrás de las vías. A través de la solución de este problema, se trata de ilustrar el control heuristico neural con descomposición o separación de tareas, teniendo como elementos principales la regla de extracción e implantación. Especializando en el problema del péndulo el control global es descompuesto en subtareas: posicionamiento del péndulo y posicionamiento del carro. Así tres subcontroladores neuronales son diseñados para controlar y coordinar cada subtarea.

Kumpati Narendra y Sruehasis Mukhopadhyay **Intelligent control using Neural Network** IEEE Control Systems Magazine on Neural Networks in Control Systems, Abril 1992, p.p.11-18	El control inteligente usando redes neuronales puede ser aplicado a sistemas dinámicos complejos en presencia de fallas estructurales. Después de ocurrir una falla estructural se asume que el sistema dinámico puede encontrarse en un número finito de configuraciones, correspondiendo cada una de estas a cada controlador de estabilización. Un controlador de 2 niveles es usado en el cual el nivel más alto detecta la falla usando un reconocimiento de patrones activando así un controlador de estabilización. En el segundo nivel, un controlador adaptable se usa para optimizar la estabilidad del sistema . Se utiliza una red neuronal en el nivel inferior para identificar y controlar la planta y como reconocedor de patrones en el nivel superior para detectar la falla. Los resultados presentados de la simulación muestran que dada la eficiencia de los controladores multinivel basados en redes neuronales pueden usarse para diseñar controladores para sistemas dinámicos no lineales.

Richard S. Sutton, Andrew G. Barto y Ronald J. Williams **Reinforcement learning is Direct Adaptive Optimal Control** IEEE Control Systems Magazine on Neural Networks in Control Systems, Abril 1992, p.p.19-22	Los problemas de control pueden dividirse en dos grupos. El primer grupo es el problema de regulación y seguimiento donde el objetivo es darle seguimiento a la trayectoria de referencia. El segundo grupo es el problema del control óptimo, donde el objetivo es extremar la función del controlador, la cual no siempre está definida en términos de la trayectoria de referencia. Los métodos adaptables para problemas del 1er tipo son bien conocidos, sin embargo poco útilizados para control óptimo, es más, hay estudios que revelan que comparado con métodos indirectos en los cuales se reprocesa la información en cada estado de un modelo de un sistema de control, los sistemas adaptables de control se vuelven muy atractivos como controladores óptimos. En este artículo se ve el aprendizaje reforzado como una computación simple y directa hacia el control adaptable para sistemas no lineales. También se enfoca al método aprendizaje-Q debido a sus capacidades analíticas para un caso de control adaptable.

B. Wu **An introduction to neural networks and their applications in manufacturing** J.Intell. Manuf., Diciembre 1993, p.p.391-403	Cada día es más frecuente mencionar que las funciones de manufactura están siendo con cada día más interdisciplinarias con las nuevas técnicas que continuamente son introducidas y adoptadas. Las recientes aplicaciones de las redes neuronales en la manufactura proporcionan un ejemplo de esta tendencia. este documento examina las estructuras y funciones de las redes neuronales así como provee ejemplos de sus aplicaciones.
A. Freitas da Rocha **Evolutive fuzzy neural networks** IEEE International Conference on Fuzzy Systems, Marzo 1992, p.p. 493-500	Los autores describen la combinación de redes neuronales difusas con algoritmos genéticos produciendo un paradigma de aprendizaje flexible y poderoso. El aprendizaje evolutivo combina herramientas complementarias a través del ajuste sináptico de parámetros y deduce el aprendizaje a través de la modificación de la topología de la red para así adaptar el sistema de conocimiento al ambiente predominante. Los algoritmos para el desarrollo de la máquina de aprendizaje son presentados. El criterio difuso se basa en la entropía propuesta para seleccionar la mejor arquitectura para una red neuronal difusa.

E. Wong **Neural computing and stochastic optimization** Future Tendencies in Computer Science, Control and Applied Mathematics International Conference, Diciembre 1993, p.p. 339-342	Una de las promesas de las redes neuronales es que han tenido la capacidad de aprendizaje y las capacidades de cómputo dados por un sistema nervioso central. A la fecha el poder de las redes neuronales de aprender ha sido extensamente explotado y los resultados han sido motivantes. Existe un algoritmo que puede usarse sistemáticamente para ajustar el parámetro de la red neuronal tomando como base un muestreo de las entradas para la cual la respuesta deseada es conocida por la red. A la fecha, la principal aplicación ha sido la clasificación del los patrones reconocidos. Para estos problemas, la falta de un modelo analítico hace de la programación realmente sencilla. En este documento se analizan un tipo de redes neuronales dinámicas que pueden ser usadas para encontrar la solución óptima a una gran cantidad de problemas.

Kumpati S. Narendra y Kannan Parthasarathy **Gradient methods for the optimization of dynamical systems containing neural networks** IEEE Trans. on Neural Networks, Marzo 1991, p.p.252-262	Las redes neuronales "multinivel" han sido usadas satisfactoriamente en el reconocimiento de patrones. La retropropagación es uno de los métodos comúnmente usados para ajustar los parámetros de las redes neuronales. En estudios recientes los autores han sugerido el uso de redes neuronales multinivel para la identificación y control de sistemas dinámicos no lineales y propuesto una extensión del método de retropropagación a dichos problemas. Es más, los autores consideran que estas redes neuronales encuentran un incremento en su uso en los subsistemas.
Esther Levin, Raanan Geuirtzmann y Gideon F. Inbar **Neural network architecture for adaptive system modeling and control** Neural Networks, vol 4, 1991, p.p. 185-191	En este artículo se propone una arquitectura computacional basada en redes neuronales discretas no lineales para el modelado y control de sistemas adaptables. Estas novedades son estructuras paralelas y distribuidas para procesamiento de señales con el potencial de ir mejorando su desenvolvimiento a través de un aprendizaje dinámico. En este artículo la arquitectura de red de entrada retardada - estado retardado y un esquema general de capacitación son descritos. Se siguiere así mismo, utilizar una red neuronal para la solución de la representación de señales analógicas. Se incluyen resultados ilustrativos de las simulaciones que muestran un buen y robusto desenvolvimiento en varios de los casos

Naveen V. Bhat, Peter A. Minderman, Thomas McAvoy y NAm San Wang **Modelling chemical process systems via neural computation** IEEE Control Systems Magazine, 1990, p.p. 24-30	Esta publicación trata el uso de redes neuronales para el modelado de sistemas en procesos químicos no lineales. Tres casos son considerados : un reactor de estado estático, un Ph-imetro de tanque y la interpretación de datos obtenidos por un biosensor. En todos los casos se utiliza con éxito una red neuronal retropropagada. Una ventaja de las redes neuronales es que son paralelas y como resultado pueden resolver problemas mucho más rápido que una computadora serial, aún más, las redes neuronales tienen la habilidad de aprender. Las redes neuronales así aprenden a gobernar las relaciones involucradas en el entrenamiento de la base de datos.
Sinnasamy R. Naidu, Evanghelos Zafiriou y Thomas J. McAvoy **Use of neural networks for sensor failure detection in a control system** IEEE control Systems Magazine (American Control Conference), 1990, p p. 49-55	Este documento analiza el uso de una red neuronal retropropagada como detector de fallas en sistemas de control. Se discuten entre otros tópicos, el paradigma de la retropropagación junto con algoritmos de detección de fallas. Este algoritmo es aplicado a una estructura de control interno para una planta lineal de primer orden. Comparando con otros métodos tradicionales la técnica de retropropagación muestra la habilidad de discernir con precisión las fallas supercríticas de las subcríticas

Kumpati S. Narendra y Kannan Parthasarathy **Identification and control of dynamical systems using neural networks** IEEE Transactions on Neural Networks, Marzo 1990, p.p. 4-27	A través de este estudio se demuestra que las redes neuronales pueden ser usadas para la identificación y control de sistemas dinámicos no lineales. Este articulo hace énfasis tanto en la identificación como en el control de dichos sistemas. Los modelos tratados son el multicapa y las redes recurrentes todas conectadas en configuraciones especiales por lo que hay una necesidad de estudiarlas en forma conjunta. Los resultados de la simulación muestran que la identificación y los esquemas de control adaptable propuestos son costeables. Los conceptos y definiciones usadas así como las interrogantes teóricas son descritas también a lo largo del estudio.
Anthony N. Michel y Jay A. Farrell **Associative memories via artificila neural networks** IEEE Control Systems Magazine, 1990, p.p. 6-17	En este documento se presentan varias técnicas de diseño como el Ouder Product Method (OPM) y el Projection Learning Rule (PLR), que pueden usarse para redes neuronales continuas y discretas para formar memorias asociativas. La aplicación de estas técnicas se demuestra a través de ejemplos en los cuales se pueden apreciar sus ventajas y desventajas.

Derrich H. Nguyen y B. Widrow **Neural networks for self-learning control systems** IEEE Control Systems Magazine, 1990, p.p. 18-23	Las redes neuronales pueden usarse para resolver problemas de control no lineales. Este artículo muestra como una red neuronal puede aprender de su propio desarrollo para controlar un sistema dinámico no lineal. Un emulador aprende a identificar las características dinámicas del sistema, a su vez, un controlador aprende a controlar el sistema dinámico real. El proceso de aprendizaje continúa cuando el controlador y emulador mejoren y den seguimiento al proceso físico. Un ejemplo usado fue el de un trailer de una caja donde la red neuronal será quien controle el volante mientras el trailer avanza en reversa. El controlador es capaz de dirigir el trailer a partir de casi cualquier posición inicial, demostrándose así que esta técnica puede emplearse para una gran variedad de problemas no lineales.

Stephen H. Lane, David A. Handelman y Jack J. Gelfand **Theory and development of higher-order CMAC neural networks** IEEE Control Systems Magazine, Abril 1992, p.p.23-30	El Modelo Controlador de Articulación Cerebral (CMAC) es capaz de aprender funciones no lineales extremadamente rápido dada su naturaleza de actualización de los pesos. El perfil rectangular de los campos vectoriales del CMAC produce aproximaciones de funciones discontinuas sin las derivadas analíticas inherentes. La habilidad de aprender las funciones y sus derivadas es de suma importancia para el desarrollo de muchos filtros adaptables, estimadores y algoritmos de control. Se demuestra que usando campos de funciones B-spline junto con esquemas de direcciones generales del CMAC conducen al desarrollo de redes neuronales CMAC de alto nivel las cuales pueden aprender las funciones y sus derivadas permitiéndose el desarrollo de arquitecturas que pueden entrenarse utilizando técnicas de retropropagación de errores.

Gordon Kratt y David Campagna **A comparison between CMAC neural network control and two traditional adaptive control systems** IEEE Control Systems Magazine, 1990, p.p. 36-43	Este articulo compara un controlador basado en una red neuronal similar al modelo de controlador cerebral articulado (CMAC) con 2 controladores adaptables tradicionales, un STR (Regulador Autoajustable) y un MRAC (Modelo de Referencia). Los tres sistemas son comparados conceptualmente y a través de pruebas de simulación con el mismo problema de control de bajo orden. Los resultados se obtienen para un sistema lineal y sin ruido, para el caso en que se añade ruido y para cuando se controla un sistema no lineal. Se realizan comparaciones respecto a la estabilidad del sistema de lazo cerrado, velocidad de adaptabilidad, rechazo al ruido, número de cálculos requeridos, desenvolvimiento del sistema de rastreo y el grado de desarrollo teórico. Los resultados indican que l red neuronal aproxima bien las funciones en ruido, funciona para sistemas lineales y no lineales y puede ser implementada eficientemente en sistemas de gran escala.

Fu Chuang Chen **Backpropagation neural netwoks for nonlinear self-tuning adaptive control** IEEE Control Systems Magazine, 1990, p.p. 44-48	Una red neuronal retropropagada es aplicada a un problema de rastreo no lineal autoajustable. Las técnicas tradicionales de control adaptable autoajustable pueden tratar solo con sistemas lineales y solo algunos no lineales. Las redes neuronales retropropagadas tienen la capacidad de aprender de la no linealidad arbitraria y muestran gran potencial de aplicación al control adaptable. Se propone y analiza un esquema que combina a la red neuronal retropropagada con la técnica de control adaptable autoajustable. Los resultados de la simulación muestran que el esquema de auto sintonización puede tratar a un gran número de linealidades desconocidas.
Panos J. Antsaklis **Neural networks in control systems** IEEE Control Systems Magazine, 1990, p.p. 3-5	Las crecientes necesidades tecnológicas de la sociedad moderna requieren del desarrollo de nuevas técnicas que son necesarias para resolver los problemas de control. las redes neuronales con su paralelismo masivo y capacidad de aprendizaje ofrecen mejores soluciones al menos para algunos de estos problemas. Por ahora, la comunidad del control ha escuchado de las redes neuronales y se pregunta si estas pueden proveer de mejores soluciones en el control de problemas ya viejos o al menos una solución a los problemas que han permanecido a los mejores esfuerzos.

Charles W. Anderson **Learning to control an inverted pendulum using neural networks** IEEE Control Systems Magazine, Abril 1989, p.p. 31-37	El péndulo invertido es simulado como una tarea de control con el objetivo de aprender el balanceo del péndulo sin conocimiento previo de la dinámica del sistema. En contraste con otras aplicaciones de las redes neuronales, el comportamiento de retroalimentación se asume como no disponible en cada paso apareciendo solo una falla en la señal cuando el péndulo cae o alcanza la horizontal. Para resolver estas tareas, el controlador debe tratar con aspectos de retraso de la evaluación, aprendizaje de la incertidumbre y de las funciones no lineales. Los métodos de reforzamiento y diferencia temporal son usados para el aprendizaje para evitar las condiciones de inestabilidad y balanceo del péndulo.

Jiabin Hao, Shaohua Tan y Joos Vandewalle **A rule-based neural controller for inverted pendulum system** IEEE International Joint Conference on N.N., 1993, p.p. 534-539	Este documento trata de demostrar como un control neuronal heuristico puede ser usado para resolver un complejo problema de control. Para poder balancear el péndulo, el controlador requiere mover el carro hacia atrás de las vías. A través de la solución de este problema, se trata de ilustrar el control heuristico neural con descomposición o separación de tareas, teniendo como elementos principales la regla de extracción e implantación. Especializando en el problema del péndulo el control global es descompuesto en subtareas: posicionamiento del péndulo y posicionamiento del carro. Así tres subcontroladores neuronales son diseñados para controlar y coordinar cada subtarea.

Saifu Akhyar y Sigeru Omatu **Neuromorphic self-tuning PID Controller** IEEE /INNS Int. Joint Conf. on Neural Networks1993, p.p. 552-557	Se ha probado que las redes neuronales pueden aproximar cualquier función continua con un error mínimo. Así las redes neuronales permiten representar cualquier sistema no lineal, aún más, la teoria del control adaptable está limitado en sistemas lineales. aquí usando un mapeo de redes neuronales, se desarrolla un método adaptable de PID basado en la retropropagación de las redes neuronales. Los resultados de la simulación muestran que el método propuesto puede identificar los parámetros del controlador PID cuando se implementan tanto la planta lineal como la no lineal.
Can Isik y A. Mete Cakmakci **Identification of a nonlinear multivariable dynamic process using feedforward networks** IEEE /INNS Int. Joint Conf. on Neural Networks 1993, pp. 564-567	Los aspectos prácticos de identificación de un sistema dinámico no lineal de múltiples entradas, múltiples salidas usando redes neuronales prealimentadas(feed forward) son mencionados en este documento. Usando mediciones de 25 entradas y sus variables internas en el proceso, el proceso primario de salida es estimado por una red neuronal que ha sido parcialmente conectada. Dos problemas de patrón de conectividad se comparan y los problemas encontrados durante su desarrollo son sumarizados. La eficacia del estimador es demostrada comparando la salida del proceso en el dominio de la frecuencia.

Asriel U. Levin y Kumpati S. Narendra **Stabilization of nonlinear dynamical systems using neural networks** IEEE International Joint Conference on N.N., 1992, p.p.275-280	Aquí se trata el problema de la estabilización de sistemas dinámicos alrededor del punto de equilibrio. El principal objetivo es indicar como basándose en resultados de la teoria de control no lineal, la mayoria de los controles viables puede diseñarse usando redes neuronales.
Panos J. Antsaklis **Neural networks in control systems** IEEE Control Systems Magazine on Neural Networks in Control Systems, Abril 1992, p.p.8-10	Lo más avanzado en el área de redes neuronales en los sistemas de control se plasma en esta edición. La necesidad de tener sistemas de control dinámicos que estén en posibilidad de cumplir con las condiciones de incertidumbre provoca que las redes neuronales dada su habilidad de aprender, aproximar funciones, reconocer patrones y su basta implantación en hardware sean la opción más acertada. Las redes neuronales pueden realizarse en muchas funciones esenciales de los sistemas de control con un alto grado de autonomia, este es el segundo articulo especial de la IEEE dedicada a las redes neuronales en los sistemas de Control.

Yashundo Takhashi **Control Adaptable Predictivo de Sistemas Dinámicos No Lineales Usando Redes Neuronales** **Paper Code** : NN WE P2 OR-2-3 Mayo 1992 II-UNAM	En este documento, una red neuronal se entrena para simular el comportamiento dinámico de entrada-salida de un objeto de control. Luego usado como estimador de respuesta de una planta a partir de la cual el valor optimo de control de entrada se determina en cada punto de muestreo dando a la planta el óptimo control predictivo si esta es invariable en el tiempo. Las simulaciones realizadas en varios sistemas no lineales y variantes en el tiempo han mostrado en muchos casos el nivel más próximo al valor óptimo de control.

3. APLICACIONES EN ROBÓTICA

Cuanto más pesada sea la carga, más a ras de tierra
estará nuestra vida, más real y verdadera será.
"La Insoportable Levedad del Ser"
M. Kundera.

Hasta este punto, se puede distinguir la cercana conexión entre el control con RNA y la Teoria de Control Adaptable; de hecho, se puede inferir que las RNA comprenden una clase especial de controladores no lineales adaptables con propiedades muy importantes.

En control adaptable es necesario hacer ciertas suposiciones tales como las condiciones para desacoplamiento del controlador en el control adaptable indirecto, o sobre la capacidad de aproximación al sistema, las cuales pueden no cumplirse En contraste, un controlador basado en RNA no requieren este tipo de condiciones

3.1. Dinámica del Robot de Estructura Rígida

En cierto sentido la aplicación de controladores con RNA a robots de estructura rígida parece ser muy natural. La razón principal de ello es que la dinámica del robot satisface algunas propiedades importantes, incluyendo la de pasividad, que son fáciles de preservar en lazo cerrado haciendo uso de RNA. Esto motiva la aplicación de esta técnica a robots de este tipo. La dinámica de estos manipuladores se discute a continuación.

3.1.1. Dinámica del Robot y sus Propiedades

La dinámica de un manipulador de n eslabones rígidos se expresa mediante las ecuaciones de Euler-Lagrange por:

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{V}\left(\mathbf{q},\dot{\mathbf{q}}\right)\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}) + \mathbf{F}\left(\dot{\mathbf{q}}\right) + \tau_d = \tau(t) \qquad (3.1)$$

con $q(t) \in \Re^n$ el vector de uniones, $M(q)$ la matriz de inercia, $V(q,\dot{q})$ la matriz de fuerzas de coriolis, $G(q)$ el vector de gravedad, y $F(\dot{q})$ la fricción. Las perturbaciones desconocidas acotadas son denotadas por τ_d, y $\tau(t)$ es el torque de control.

Se requieren las siguientes propiedades en la dinámica del robot:

1. La matriz $M(q)$ es simétrica positiva definida acotada por $m_1 \mathbf{I} \le \mathbf{M}(\mathbf{q}) \le m_2 \mathbf{I}$ con m_1, m_2 constantes positivas conocidas.
2. La matriz $V(q,\dot{q})$ es acotada por $V_b(\mathbf{q})\|\dot{\mathbf{q}}\|$ con $V_b(\mathbf{q}) \in C^1(s)$.
3. La matriz $\dot{\mathbf{M}} - 2\mathbf{V}$ es antisimétrica.
4. Las perturbaciones desconocidas son acotadas, es decir, $\|\tau_d\| < b_d$ con b_d constante positiva conocida.

3.1.2. Seguimiento de Trayectoria

Una importante aplicación dentro del control de robots es que el manipulador siga una trayectoria prescrita.

Error en Dinámica. Dada una trayectoria deseada $q_d(t) \in \Re^n$ el *error de seguimiento* está definido por:

$$e(t) = q_d(t) - q(t) \tag{3.2}$$

Es típico en robótica el definir un filtro de error de seguimiento por:

$$r(t) = \dot{e} + \Lambda e \tag{3.3}$$

donde $\Lambda = \Lambda^T > 0$ es una matriz de parámetros de diseño, usualmente diagonal. Diferenciando r(t) y usando (3.1), la dinámica del robot puede expresarse en términos del error de seguimiento filtrado como:

$$M\dot{r} = -Vr - \tau + f + \tau_d \tag{3.4}$$

donde la función no lineal importante es:

$$f(x) = M(q)(\ddot{q}_d + \Lambda\dot{e}) + V(q,\dot{q})(\dot{q}_d + \Lambda e) + G(e) + F(\dot{q}) \tag{3.5}$$

y por instancia se define el vector

$$x = [e^T \dot{e}^T q_d^T \dot{q}_d^T \ddot{q}_d^T]^T. \tag{3.6}$$

Una posible entrada de control para el seguimiento de trayectoria está dada por

$$\tau_d = \hat{f} + K_v r \tag{3.7}$$

con $K_v = K_v^T > 0$ una matriz de ganancia y $\hat{f}$ un estimado de f obtenido por un medio todavía no especificado. Usando este control, el sistema en lazo cerrado es:

$$M\dot{r} = -(K_v + V_m)r + \tilde{f} + \tau_d \equiv -(K_v + V_m)r + \zeta_m \tag{3.8}$$

donde el error de estimación funcional está dado por

$$\tilde{f} = f - \hat{f} \tag{3.9}$$

Este es un error del sistema en donde el error de seguimiento filtrado es manejado por el error de estimación funcional. El control τ_0 incorpora un término proporcional-derivativo (PD) en $\mathbf{K}_v \mathbf{r}(t) = \mathbf{K}_v(\dot{\mathbf{e}} + \Lambda \mathbf{e})$.

El Problema de Control. En lo subsecuente, se empleará (3.8) para enfocar la selección de algoritmos de entrenamiento de RNA que aseguren la estabilidad del error de seguimiento filtrado r(t). Entonces, dado que la ecuación (3.3), con la entrada considerada como r(t) y la salida como e(t) describe un sistema estable.

Además, el diseño del controlador debe garantizar que tanto el error r(t) como la señal de control sean acotados. Es importante señalar que la conclusión anterior se sustenta en el hecho que el estimado $\hat{\mathbf{f}}$ es acotado. Más aún, para un buen desempeño, la cota sobre r(t) debe ser en un sentido "suficientemente pequeña".

Propiedad de Pasividad. La dinámica (3.8) desde $\zeta_*(t)$ hasta r(t) es un sistema estrictamente pasivo, es decir, es un sistema que disipa energía.

3.2. Controlador Neuronal para el Robot de Estructura Rígida

Este controlador consiste de la estrategia de control definida por (3.7) donde la función estimada $\hat{\mathbf{f}}$ es provista por una RNA. Dado que se debe garantizar acotamiento en los pesos y en el error de seguimiento la regla delta -estándar en el entrenamiento de RNA- no es suficiente para ello y por lo tanto se deben agregar términos extra a los requerimientos de aprendizaje

3.2.1. Algunas Suposiciones y Hechos

Las suposiciones aquí establecidas pueden ser ciertas en toda situación practica, y son un estándar en la literatura existente.

Suposición 1. La función no lineal (3.5) es aproximada por cualquier RNA de 2 ó 3 capas (una intermedia oculta) con un error de aproximación acotado $\varepsilon < \varepsilon_M$.

A menos que la RNA sea minimal existen varias combinaciones de pesos tales que satisfagan la suposición anterior. Los "mejores" pesos son aquellos que satisfacen la condición de acotamiento bajo alguna norma (p.e. Frobenius, Euclidiana, etc.). Para desarrollos subsecuentes sólo se necesita saber que dichos pesos existen; sus valores iniciales no importan.

Por conveniencia de notación se define la matriz de todos los pesos como

$$\mathbf{Z} = \begin{bmatrix} \mathbf{W} \\ \mathbf{V} \end{bmatrix} \tag{3.10}$$

por consistencia dimensional se rellena W ó V con columnas de ceros.

Suposición 2. La trayectoria deseada es acotada en el sentido de una norma

$$\left\| \begin{matrix} q_{d_1} \\ \vdots \\ q_{d_n} \\ \vdots \\ q_{d_n} \end{matrix} \right\| \leq Q_d \tag{3.11}$$

donde $Q_d \in \Re$ es una constante conocida

El siguiente hecho sigue directamente de las suposiciones y definiciones previas.

Hecho 1. Para cada instante t, x(t) en (3.6) está acotada por

$$\|x\| \le c_1 Q_e + c_2 \|r\| \qquad (3.12)$$

para constantes c_i calculables positivas.

3.2.2. Estructura del Controlador

La estructura de la RNA empleada como controlador se puede definir ahora; esta aparece en la figura siguiente, donde $\underline{q} \equiv \left[q^T \quad \dot{q}^T\right]^T$, $\underline{e} \equiv \left[e^T \quad \dot{e}^T\right]^T$. Es importante señalar que la RNA no es un desarrollo ex profeso para este problema, sin embargo, la estructura de la RNA sigue al tratamiento del error en la dinámica del robot y las propiedades de este.

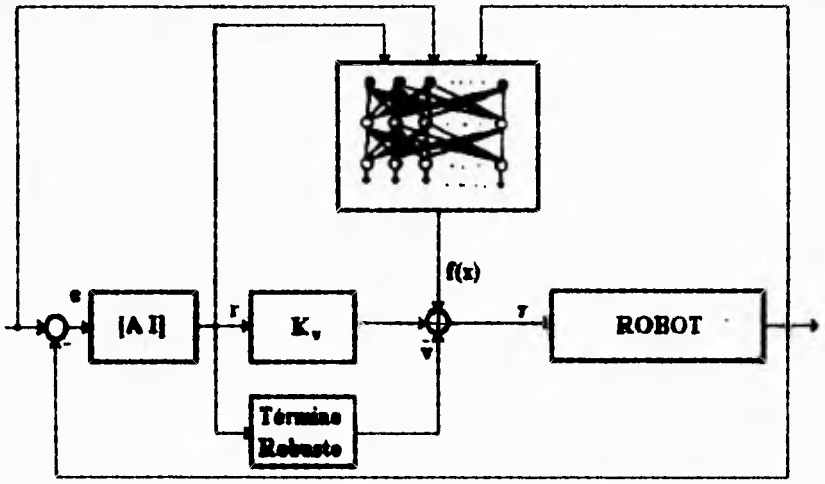


Figura 3.1 Estructura del Controlador basado en una RNA

RNA como Controlador. Se define la RNA que sea la estimada funcional de (3.5) por

$$\hat{f}(x) = \hat{W}^T \sigma\left(\hat{V}^T x\right) \qquad (3.13)$$

donde $\hat{W}$ y $\hat{V}$ son los valores estimados de los pesos de la RNA W y V, respectivamente. Estos estimados son provistos por el algoritmo de aprendizaje. Con $\tau_0(t)$ definido en (3.7), se define la señal de control

$$\tau = \tau_0 - v(t) = \hat{W}^T\sigma\left(\hat{V}^T x\right) + K_v r - v(t) \qquad (3.14)$$

con v(t) como función robustificante como se verá a continuación.

Teorema 3.1

Sea la trayectoria deseada acotada por (3.11). Tómese la señal de control para el robot (3.1) como (3.14) con el término robustificante siguiente

$$v(t) = -K_z\left(\left\|\hat{Z}\right\|_F + Z_M\right)r \qquad (3.15)$$

donde $K_z > \|r\|\left\|\hat{Z}\right\|_F + g$ y g es función de la magnitud media de las perturbaciones; Z_M es una cota máxima para la matriz de todos los pesos. Entonces, para que el error de seguimiento y los pesos estimados de la RNA $\hat{W}$ y $\hat{V}$ sean Globalmente Asintóticamente Acotados (GAA) tenemos:

$$\dot{\hat{W}} = F\hat{\sigma}r^T - F\hat{\sigma}'\hat{V}^T x r^T - \kappa F|r|\hat{W} \qquad (3.16)$$

$$\dot{\hat{V}} = Gx\left(\hat{\sigma}'^T\hat{W}r\right)^T - \kappa G|r|\hat{V} \qquad (3.17)$$

donde $F=F^T>0$, $G=G^T>0$ y $\kappa>0$.

La siguiente tabla muestra en resumen las ecuaciones para los algoritmos de la RNA.

Tabla 3.1 Controlador Basado en RNA

RNA Controlador: $\tau = \hat{W}^T \sigma\left(\hat{V}^T x\right) + K_v r - v(t)$

Término Robustificante: $v(t) = -K_z\left(\left\|\hat{Z}\right\|_F + Z_M\right) r$

Actualización de Pesos:

$$\dot{\hat{W}} = F\hat{\sigma} r^T - F\hat{\sigma}'\hat{V}^T x r^T - \kappa F \|r\| \hat{W}$$

$$\dot{\hat{V}} = G x\left(\hat{\sigma}'^T \hat{W} r\right)^T - \kappa G \|r\| \hat{V}$$

Señales:

$e(t) = q_d(t) - q(t)$ Error de Seguimiento

$r(t) = \dot{e} + \Lambda e$ Error de Seguimiento Filtrado

$x = [\, e^T \; \dot{e}^T \; q_d^T \; \dot{q}_d^T \; \ddot{q}_d^T \,]^T$ Vector de Entradas a la RNA

Parámetros de Diseño:

Ganancias K_v , K_z y matrices de aprendizaje F y G simétricas y positivas definidas.

Z_M una cota sobre la matriz de todos los pesos

A partir de estos elementos se pueden desarrollar diferentes aplicaciones sobre robots de estructura rígida, como el ejemplo siguiente.

Ejemplo 3.1 Control de un robot de 2 eslabones.

El robot planar de 2 eslabones rígidos es usado extensamente dentro de la literatura para propósitos ilustrativos aparece en la figura siguiente. Su dinámica está dada extensamente en la bibliografía por lo que aquí no se presenta al detalle.

$$\begin{bmatrix} m_1 l_{c1}^2 + m_2 l_1^2 + I_1 & m_2 l_1 l_{c2} \cos(q_2 - q_1) \\ m_2 l_1 l_{c2} \cos(q_2 - q_1) & m_2 l_{c2}^2 + I_2 \end{bmatrix} \ddot{q} + \begin{bmatrix} -m_2 l_1 l_{c2} \operatorname{sen}(q_2)\dot{q}_2 & -m_2 l_1 l_{c2} \operatorname{sen}(q_2)(\dot{q}_1 + \dot{q}_2) \\ -m_2 l_1 l_{c2} \operatorname{sen}(q_2)\dot{q}_1 & 0 \end{bmatrix} \dot{q} +$$
$$+ \begin{bmatrix} (m_1 l_{c1} + m_2 l_1) g \cos(q_1) + m_2 l_{c2} g \cos(q_1 + q_2) + \varphi \\ m_2 l_{c2} g \cos(q_1 + q_2) + \varphi \end{bmatrix} = \bar{\tau}$$

(3.18)

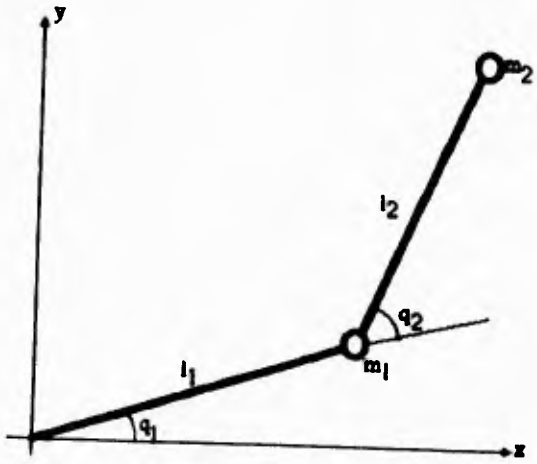


Figura 3.2 Robot Planar de 2 Eslabones

Las variables de unión son $q = [q_1 \quad q_2]^T$ y la fricción es φ. Tomamos los parámetros de la armadura como l_1=1m. l_2=1m. l_{c1}=0.5 m. l_{c2}=0.5 m. m_1=0.8Kg. m_2=2.3 Kg., y seleccionamos $q_{1d}(t)$ =sen(t) q_{2d}= cos(t). A partir de estas variables se desarrolla el controlador con una RNA, es importante hacer notar que el controlador no requiere del conocimiento de la dinámica del robot y tampoco de la asignación de una estructura como en el caso adaptable.

Una selección de señales más propia que $x=[\ e^T\ \dot{e}^T\ q_d^T\ \dot{q}_d^T\ \ddot{q}_d^T]^T$ se requiere para poder despreciar la fricción y alcanzar las ventajas arriba citadas, el vector x seleccionado debe ser válido para n eslabones, hacemos entonces

$$x = \left[\zeta_1^T \quad \zeta_2^T \quad \cos(q_d)^T \quad \text{sen}(q_d)^T \quad \dot{q}_d^T \quad \text{sign}(\dot{q}_d)^T\right]^T \tag{3.19}$$

donde $\zeta_1 = \ddot{q}_d + \Lambda \dot{e}$, $\zeta_2 = \dot{q}_d + \Lambda e$. El controlador es el mismo de la figura 3.1 con 10 neuronas en la capa intermedia. Los resultados mostrados se obtienen aplicando el algoritmo especificado por la Tabla 3.1 con κ=0.1, K_v=diag(20, 20), F=diag(10, 10), Λ=diag(5, 5) y G=diag(10,10).

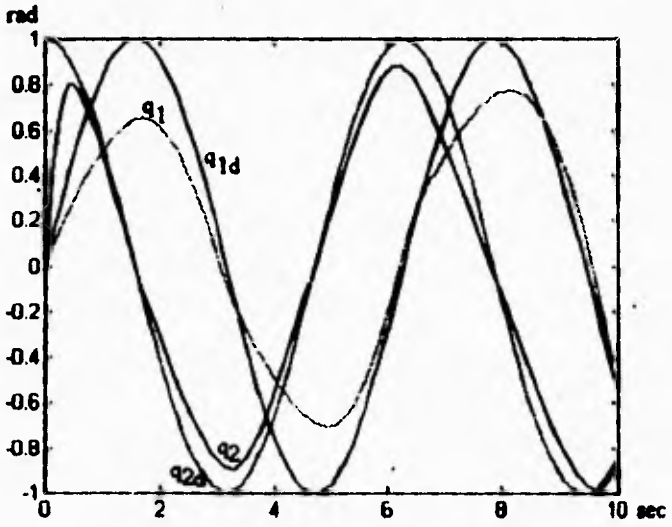


Figura 3.3 Respuesta del Sistema sin RNA Ángulos actuales y deseados

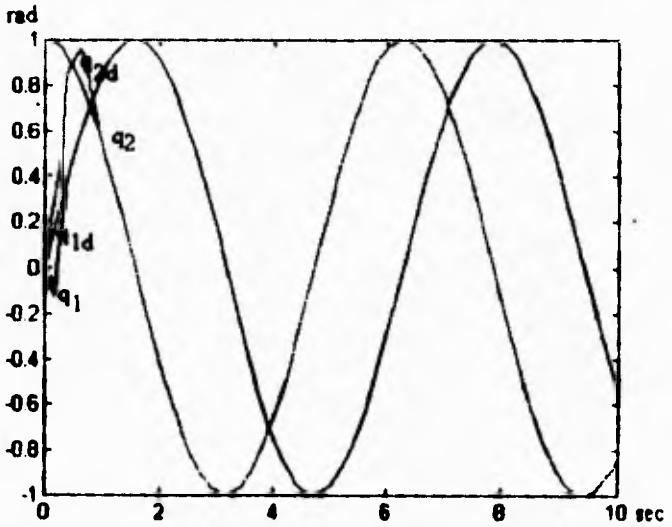


Figura 3.4 Respuesta del Sistema con RNA Ángulos actuales y deseados

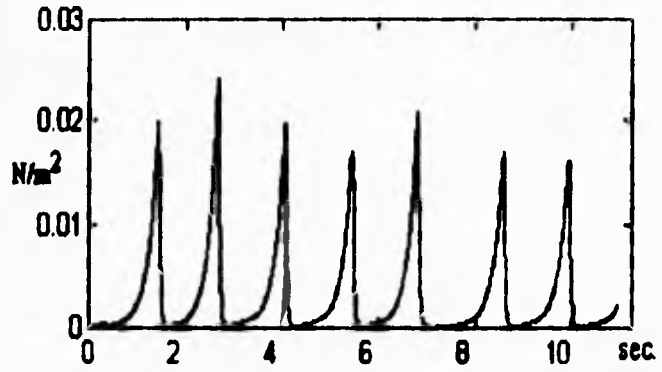


Figura 3.5 Corrección de Fricción Seca

A partir de las gráficas anteriores resulta evidente que el controlador sin la RNA estabiliza en el sentido entrada acotada-salida acotada a la armadura, la compensación de la fricción seca (φ) por parte de la RNA se puede obtener también bajo la Teoría de Control Adaptable, en este caso, un controlador paramétrico es diseñado a partir de la estructura del sistema (dinámica del robot) pero de parámetros desconocidos. La validación de los parámetros implica que la matriz de inercia sea simétrica; además que ésta y la de fuerzas coriolis sean ambas positivas definidas y que la matriz Jacobiana empleada para calcular la energía cinética no se haga singular en ningún punto de la trayectoria. En cambio, el controlador propuesto sólo requiere que las señales correspondan a un sistema pasivo para asegurar que la respuesta de la RNA ejerza un efecto correctivo.

3.3. Aplicaciones

Actualmente, existe una gran cantidad de aplicaciones de RNA en Robótica, la mayor parte de ellas se encuentran en el campo de la Visión Artificial, el Reconocimiento de Patrones, los problemas de Dinámica y Cinemática Directa e Inversa (determinación de trayectorias y de señales de control), a continuación se presenta una breve descripción de algunos de los trabajos que se consideraron como aplicaciones clásicos en el área.

Michael Kuperstein y Jorge Rubinstein **Implementation of an adaptive neural controller for sensory motor coordination** IEEE Control Systems Magazine, Abril 1989, pp. 25-30	Se presenta la teoría y prototipo de un controlador neuronal llamado INFANT, el cual aprende coordinación motor-sensorial. INFANT se adapta a cambios inesperados en la geometría del sistema físico motor y de la localización, figura, y tamaño de los objetos. Puede tomar un objeto sin tener ninguna información respecto a él. Está diseñado de tal forma que puede ser generalizado para la coordinación de cualquier número de miembros de entrada. INFANT está implantado con un procesador de imágenes, cámaras estéreo, y un brazo de robot de cinco grados de libertad.

Huan Liu, Thea Iberall y George A. Bekey **Neural network architecture for robot hand control** IEEE Control Systems Magazine, Abril 1989, pp. 38-43	Un sistema de control de una mano de robot, llamado GeSAM, está en desarrollo en la Universidad del Sur de California. Lo primordial en la arquitectura de GeSAM es proveer un controlador genérico de una mano de robot que esté basado en las funciones prensiles humanas. Se enfoca a las relaciones entre las primitivas geométricas de los objetos y las formas en que la mano puede desarrollar la presión. El artículo presenta cómo las relaciones entre las primitivas de los objetos y las formas de asirlos, pueden ser aprendidos por una red neuronal adaptativa. Además se trata de que el sistema pueda desarrollar improvisación, para evitar el tedioso trabajo de cómputo ocupándose particularmente de cada caso.
Kevin M. Passino, Michael A. Sartori y Panos J. Antsaklis **Neural computing for numeric to symbolic conversion in control systems** IEEE Control Systems Magazine, Abril 1989, pp. 44-52	La computación neuronal ofrece facilidades enormes en cuanto a la clasificación de patrones. Una red neuronal, del tipo llamado *perceptrón multicapa*, es usado para clasificar datos numéricos y asignarles símbolos apropiados. Esta conversión numérica a simbólica es un tipo de extracción de información, similar a la reducción de datos en reconocimiento de patrones. El perceptrón es empleado como un convertidor numérico-simbólico para sistemas con eventos discretos controlando y supervisando variables continuas de sistemas dinámicos.

Rolf Eckmiller **Neural nets for sensory and motor trajectories** IEEE Control Systems Magazine, Abril 1989, pp. 53-58	Este articulo presenta una introducción al uso de redes neuronales artificiales para el manejo de trayectorias sensoriales y motoras. En esta investigación, las funciones en el dominio del tiempo representan las trayectorias son separadas de los parámetros espaciales los que son almacenadas en un mapa neuronal, y los parámetros de tiempo son guardados en un segundo mapa neuronal. Una investigación llamada lattice triangular neuronal es sugerido para implementar los mapas neuronales.

W. Thomas Miller, Robert P. Hewes, Filson H. Glanz y L. Gordon Kraft **Real time dynamic control of an industrial manipulator using a neural network based learning controller** IEEE Transactions on Robotics and Automation, Vol. 6, No. 1 Febrero 1990, pp. 1-9	La extraordinaria complejidad de la mayoría de los problemas de control de robots y la idea de un sistema robótico general, han dejado muchas discusiones acerca del uso de redes neuronales en control de robots. Se propone una técnica de control para aprendizaje la cual utiliza una extensión de una red CMAC desarrollada por Albus, y se presentan los resultados de los experimentos de control en tiempo real que involucran aprendizaje de la dinámica de un robot industrial de cinco ejes durante movimientos rápidos. Durante cada ciclo de control , se usó un esquema de entrenamiento para ajustar los pesos en la red conforme a la aproximación al modelo dinámico del robot a regiones apropiadas en el espacio de control. Simultáneamente, la red fue utilizada durante cada ciclo de control para predecir los actuadores requeridos para seguir una trayectoria deseada, y estos manejadores fueron usados como términos de avance en paralelo para ajustar el controlador con una ganancia lineal de retroalimentación. Los errores en las trayectorias que fueron encontrados eran convergentes a valores muy pequeños después de unos cuantos intentos de entrenamiento, errores que son relativamente despreciables. Se investigó también los efectos del tamaño de la memoria en la red y las características de las trayectorias en el aprendizaje del sistema.

Owen Holland y Martin Snaith **Q -learning with generalisation: an architecture for real world reinforcement learning in a mobile robot** IEEE/INNS International Joint Conferences on Neural Networks 1992, Vol. I-287-292	El uso de robots reales en vez de simulaciones se ha incrementado severamente, ya que las arquitecturas, al realizar aprendizaje sólo funcionan por un determinado tiempo y un cierto número de pruebas. Estos argumentos son presentados para soportar el hecho de que los perceptrones multicapa son inapropiados porque están exentos de mezclar nuevo aprendizaje con lo adquirido anteriormente. La estructura del Aprendizaje-Q de Watkins, un esquema cerrado de aprendizaje de estados y tiempo discreto relacionado con programación dinámica y capaz de interpretar las conexiones, parece adecuado, y se presentan los detalles para ser generalizado en información posterior. Una simple representación de los componentes no aprendidos de estado interno en términos de la historia reciente de percepciones y acciones se propone para uso en navegación entre acontecimientos y ambientes donde los acontecimientos son raros. Se describe un desarrollo reciente basado en un robot móvil (FRANK) el cual tiene un mecanismo conocido de percepción basado en neuronas para operar en un ambiente humano no estructurado y una computadora a bordo para implementar las modificaciones al algoritmo Q y la codificación PAS.

Hai-Long Pei, T. P. Leung y Qi-Jie Zhou **Backward construction** IEEE/INNS International Joint Conferences on Neural Networks 1992, Vol. I-293-298	Aunque las reglas de aprendizaje en general son innegablemente aplicables a los problemas de aprendizaje de un motor, se sabe de la complejidad de manejar un robot, una estrategia directa de entrenamiento no tiene un comportamiento convergente distintivo, lo cual involucra mucho tiempo consumido. En el artículo se propone una estrategia para descomponer la tarea en varias partes. La investigación emplea la idea de retroconstrucción para construir progresivamente la estrategia del motor deseado. Esta investigación está demostrada en detalle con un ejemplo del aprendizaje de un manipulador en el control de fuerza-posición.

Sukhan Lee **A new neural net approach to robot 3D perception and visuo-motor coordination** IEEE/INNS International Joint Conferences on Neural Networks 1992, Vol. I-299-307	Se presenta una investigación en redes neuronales para la coordinación visual-motor de un robot. Se referencia a esta investigación como "neurobótica", establece redes neuronales basadas en los errores visuales como mecanismo fundamental para la implantación de la coordinación visual-motor. El error visual es producido para protección del espacio de tareas en la percepción en 3D de la red y generar un cambio incremental en la configuración del brazo en el espacio 3D. La percepción en 3D se forma de una colección de neuronas llamadas "celdas indices". Cada celda indice representa una posición 3D asociada con la triangulación de un par de pixeles en las retinas virtuales, donde estas retinas virtuales están definidas con orientación fija de los ojos y densidades uniformes de pixeles, y representan los ojos internos del robot. Las configuraciones del brazo en el espacio 3D son planeadas conforme a la red de percepción en 3D que traduce la información a los ángulos correspondientes que debe tomar. La red para movimiento del brazo está implementada con un mapa bidireccional entre la 3D y el espacio contiguo basado en una jerarquía de auto organización.

L. Monostori y D. Barschdorff **Artificial neural networks in intelligent manufacturing** Robotics & Computer-Integrated Manufacturing Great Britain 1992 Vol. 9, No. 66, pp. 421-437	En sistemas inteligentes de manufactura, se presentan situaciones y problemas sin precedentes e inesperados, que esperan ser resueltos correctamente con base en información incompleta e imprecisa. Esto pareceria realizable sólo gradualmente, a través de soluciones parciales integradas dentro de los sistemas de manufactura actuales. Este articulo pone especial atención a otra aproximación, llamada redes neuronales artificiales o sistemas conectivos, que tienen la habilidad de integrar información de múltiples sensores, funciona en tiempo real, posee conocimientos efectivos de representación y puede aprender o adaptarse. Por esta razón de perfeccionamiento se hacen diversas pruebas de estructuras de redes neuronales artificiales y algoritmos de aprendizaje, y junto con aplicaciones comunes de técnicas de redes en diferentes campos dan la manufactura inteligente. Se resalta el procedimiento de aprendizaje de retropropagación más popular, con mayor importancia en técnicas de aceleración, y de competitividad en aprendizaje, que tiene buenas expectativas en futuras aplicaciones. Este articulo examina las aplicaciones conocidas de redes neuronales y sus perspectivas en manufactura inteligente. Hace un especial énfasis en la inteligencia de la máquina, particularmente en los siguientes aspectos: fusión e integración multisensorial, aprendizaje de modelado de procesos, control adaptable, monitoreo, diagnóstico y control de calidad. También se sugiere un sistema hibrido, que combine las dos aproximaciones simbólica y conectiva para aprovechar los beneficios que ambas brindan

Zheng Geng y Leonard S. Haynes **Neural networks solution for the forward kinematics problem of a Steward platform** Robotics & Computer-Integrated Manufacturing Great Britain 1992 Vol. 9, No. 66, pp. 485-495	Una estructura de red neuronal múltiple llamada cascada CMAC (Cerebellar Model Arithmetic Computer) se propone para resolver el problema del movimiento en adelanto de una clase de manipuladores, llamados plataforma Stewart. Las redes en cascada CMAC proveen una capacidad de aprendizaje rápido y la habilidad de capturar la dirección general y encontrar los detalles de mapas no lineales desconocidos. El desempeño de las redes en cascada CMAC es comparado con las populares redes de retropropagación en el mismo problema, y los resultados muestran que las redes propuestas aprenden mucho más rápido que las de propagación para este problema.

Hideki Hashimoto, Takashi Kubota, Masaaki Kudou y Fumio Harashima **Self organizing visual servo system based on neural networks** IEEE Control Systems Magazine, Abril 1992, pp. 31-36	Se propone el concepto de auto organización para manipuladores basando el control en redes neuronales. La posición del efector final y la orientación dada por el ciclo de control usando datos visuales generan las entradas de control necesarias para un manipulador. El objetivo es mover el efector final a un lugar donde el manipulador pueda tomar fácilmente un objeto dado. El mapeo no lineal entre datos de imágenes y ángulos es aprendido usando dos redes neuronales. El sistema se auto organiza por cualquier configuración del manipulador una vez que aprendió el proceso. La efectividad del sistema propuesto es confirmado por simulaciones de computadora. En los años recientes, los sistemas de control basados en visión han recibido mucha atención, especialmente en el campo de los robots inteligentes. Tales sistemas pueden hacerle frente a la mayoría de los problemas que limitan a los robots actuales. Usando sensores externos, un robot puede tener una conducta adaptativa: el robot es capaz de tener flexibilidad a los cambios en su ambiente y ejecutar tareas inteligentes. Para lograr el control de decisiones en un robot en un espacio referenciado de tareas, es necesario transformar los sensores de salida en espacio de decisiones. Varios investigadores han discutido las posibilidades de la aplicación de redes neuronales en control de robots Se propone el uso de dos redes para lograr el control del manipulador; una red global que controle las señales de objetos a grandes distancias y una red local para las pequeñas

Luis C. Rabelo y Xavier J. R. Avula **Hierarchical neurocontroller architecture for robotic manipulation** IEEE Control Systems Magazine, Abril 1992, pp. 37-41	Se presenta la arquitectura de neurocontrolador jerárquico consistente de dos sistemas de redes neuronales artificiales para la manipulación de un brazo de robot. La red de mayor nivel participa en la delineación del espacio de trabajo del brazo del robot y el proceso de transformación de coordenadas y de decisión de movimiento. La red menor provee la correcta secuencia del control de las acciones. Un ejemplo entero ilustra las capacidades de la arquitectura incluyendo velocidad, adaptabilidad y eficiencia computacional.
D. J. Cooper, L. Megan y R. F. Hinde, Jr. **Disturbance pattern classification and neuro adaptive control** IEEE Control Systems Magazine, Abril 1992, pp. 42-48	Una estrategia de adaptación basada en el análisis de patrones expuestos en el historial reciente de los errores de controles y procesos de manipulación de variables de entrada. El foco de la estrategia está en la adaptación de los requerimientos durante las perturbaciones en la carga. Específicamente, los patrones asociados con la manipulación de entradas con perturbación son analizados para determinar cuales de éstas tienen la potencia para generar un cambio en el proceso. Entonces, el patrón correspondiente es analizado para determinar la adaptación del modelo interno del controlador. Un vector de cuantización de la red es analizado como herramienta para la implantación del método. La estrategia está limitada a la simple adaptación de un parámetro donde la ganancia del modelo interno del controlador es el parámetro a ajustar. Se presentan los detalles y la demostración del método usando un proceso de simulación construido para corregir la estrategia. Un modelo basado en un controlador es empleado en este trabajo.

Y. C. Pati, P. S. Krishnaprasad y Martin C. Peckerar **An analog neural network solution to the inverse problem of "Early Taction"** IEEE Transactions on Robotics and Automation Abril 1992, Vol. 8, No. 2, pp. 196-212	Se examina una aplicación de las redes neuronales analógicas de dos niveles de procesamiento para datos de sensores táctiles. En analogía al término visión primitiva, se tiene un primer nivel de procesamiento requerido de sensado táctil en la tactilidad primaria. Asociado con casi todas las realizaciones de sensores táctiles, el problema fundamental inverso debe ser resuelto. Dichas soluciones son demandadas a la computación. Sin embargo los datos que se tienen de los sensores para la solución del problema se ven interferidos por ruido. La teoría de redes eléctricas no lineales es empleada para describir funciones de energía para una clase de redes no lineales y presentar el equilibrio de los estados, para que las redes propuestas correspondan a las soluciones regularizadas. Un regulador de entropía es incorporado dentro de la función de energía de la red para recobrar la distribución normal de fuerzas. Para realizar la demostración se emplearon simulaciones en computadora y un prototipo de circuito integrado propuesto como red.

Sumio Watanabe y Masahide Yoneyama **An ultrasonic visual sensor for 3D object recognition using neural networks** IEEE Transactions on Robotics and Automation Abril 1992, Vol. 8, No. 2, pp. 240-249	Combinando imágenes ultrasónicas con redes neuronales, se ha desarrollado un nuevo sistema de reconocimiento de objetos en 3D para uso en visión de robots. Las imágenes ultrasónicas son usadas para calcular las imágenes iniciales de objetos 3D. Estas imágenes son pasadas a redes neuronales que 1) identifican la categoría de los objetos, 2) estiman su localización, y 3) improvisan imágenes 3D. En el artículo de explica el método para las imágenes ultrasónicas en 3D, se proponen tres estructuras de red para el análisis de las imágenes, y se demuestra lo práctico del experimento con resultados, con gran exactitud y rapidez en el reconocimiento usando solo un pequeño conjunto de transductores.
Hillel J. Chiel, Randall D. Beer, Roger D. Quinn y Kenneth S Espenschied **Robustness of a distributed neural network controller for locomotion in a hexapod robot** IEEE Transactions on Robotics and Automation Junio 1992, Vol. 8, No. 3, pp. 293-303	Un controlador para locomoción con redes neuronales distribuidas, basado en neurobiología de un insecto, ha sido usado para controlar un robot hexópodo. La robustez del controlador se explica mediante un simple sensor o componente central. Una compleja interrelación entre los elementos centrales de la neurona y las entradas de los sensores dan la respuesta a la robustez del controlador y la habilidad para generar rangos continuos de marcha. Los resultados sugieren que los controladores de redes neuronales inspiradas biológicamente ofrecen un método robusto en el control de robots.

Dan Simon **Neural network based robot trajectory generation** IEEE/INNS International Joint Conferences on NN 1993, pp.540-545	La interpolación del mínimo movimiento que un robot debe realizar en una articulación empleando un número arbitrario de nodos es realizado mediante una red neuronal alambrada. Este mínimo movimiento de la articulación para una trayectoria es deseable por su similitud con los movimientos humanos, aumentando así la exactitud. Las trayectorias resultantes numéricas son bastante mejores que las funciones analíticas en tiempo. Esta aplicación toma la interpolación como una construcción de un simple problema cuadrático del ángulo en el dominio continuo y discreto en el dominio del tiempo. El tiempo es discretizado de acuerdo al controlador del robot. Las salidas de la neurona definen los ángulos de las articulaciones y los Lagrangianos. Este artículo discute la optimización de la red y su aplicación a la planeación de rutas del robot, presentando algunos resultados de la simulación, y las comparaciones entre el método con redes neuronales y otros métodos de planeación de mínimos movimientos en una trayectoria.
Kunio Iwata, Joydeep Gosh y Yoan Shin **Time optimal control using PI-SIGMA network** IEEE/INNS International Joint Conferences on NN 1993, pp. 546-551	Se propone un controlador basado en redes neuronales pi-sigma para obtener trayectorias de tiempo óptimo. Esta red facilita el uso de un conocimiento a priori del problema en la determinación de los pesos adecuados en la inicialización. Un algoritmo de Contracción de Delta Descendente es introducido para el control adaptable evitando la necesidad de un modelo de identificación y la subsecuente retropropagación del error. La simulación se hizo usando un controlador para el actuador de un manejador de disco duro mostrando una trayectoria uniforme sin sobrepaso

D. H. Rao y M. M. Gupta **Dynamic neural controller with somatic adaptation** IEEE/INNS International Joint Conferences on NN 1993, pp. 558-563	En la aplicación de redes neuronales al control de sistemas de funciones no lineales, usualmente sigmoidal, ésta se mantiene constante, y la ganancia de la función sigmoidal es determinada por la técnica de prueba y error. La selección heurística de la ganancia, determina la forma de la función sigmoidal que continúa constante, pudiendo limitar la red neuronal a sistemas complejos que involucran sistemas dinámicos no lineales. Los efectos de una ganancia sigmoidal en sistemas dinámicos no lineales se discuten en el artículo. Los algoritmos de aprendizaje y adaptación para determinar la ganancia sigmoidal óptima, son resultados derivados de la neurona.
Chris M. Jubein, Nikitas J. Dimopoulos **Identificatio of a PUMA 560 two link robot using a stable neural network** IEEE/INNS International Joint Conferences on NN 1993, pp. 568-572	Se presenta un proceso de entrenamiento para una red neuronal que es asintóticamente estable. El proceso de entrenamiento es un método gradiente el cual es adaptado de la interconexión de pesos así como de la relajación constante y la información de las funciones de activación usadas, así como la minimización entre el error las respuestas esperadas y obtenidas. También se da un método para mantener la estabilidad en el proceso de entrenamiento. Se empleó una red para simular un sistema no lineal y un robot de dos eslabones PUMA -560.

Thomas Martinetz y Klaus Schulten **A neural network with Hebbian like adaptation rules learning visuomotor coordination of a PUMA robot** IEEE/INNS International Joint Conferences on NN 1993, pp. 820-822C	Se presenta un algoritmo híbrido de redes neuronales que emplea superposición de mapas lineales, aplicado a las tareas de aprendizaje de las posiciones de un brazo de robot. El aprendizaje y control del posicionamiento está acompañado de una red para entrada visual desde un par de cámaras. Además del aprendizaje de las relaciones desconocidas de entrada-salida a priori, desde las cámaras en ángulos correspondientes, la red provee al robot la capacidad de hacer movimientos de corrección a través de retroalimentación. Esto permite dividir el posicionamiento en un movimiento inicial, un lazo abierto de correcciones guiadas por la retroalimentación, logrando movimientos del brazo similares a los humanos. Fue empleado un robot PUMA 560, el algoritmo de redes neuronales dio un resultado de error de posicionamiento de 1.3 mm., el límite menor dado por la resolución de las cámaras usadas. Dado que en los lazos de retroalimentación, las reglas de corrección de error mediante mínimos cuadrados(LMS) para los pesos en la red siguen las reglas de aprendizaje de la forma Hebbian, excepto que en vez de dar al producto una excitación pre y postsináptica, el producto determina sus ajustes conforme a sus derivadas en el tiempo:

D. Sbarbaro-Hofer, D. Neumerkel y K. Hunt **Neural control of a steel rolling mill** IEEE, Control Systems Magazine Junio 1993, pp. 69-75	En el trabajo descrito en este artículo , se revela la gran importancia que provee una fresadora de acero al mundo de las aplicaciones de control con neuronas no lineales. Diversas estructuras de control se basan en modelos neuronales de plantas de simulación. Los resultados de los controladores neuronales, son una mezcla de modelos de control interno y modelos predictivos de control, son comparados con el desempeño de un controlador PI convencional. Explotando las ventajas de las técnicas del modelado no lineal, todas las aproximaciones neuronales incrementan la precisión del control. En este caso, la combinación de un modelo neuronal como un compensador con un controlador de retroalimentación de tipo integral dan los mejores resultados.

S. T. Venkataraman, S. Gulati, J. Barhen y N. Toomarian **A neural network based identification of enviroments models for compliant control of space robots** IEEE Transactions on Robotics and Automation, Vol. 9, No. 5 Octubre 1993, pp. 685-697	Muchos de los sistemas robóticos en el espacio podrian requerir de operar en ambientes de incertidumbre o desconocidos. El problema es la identificación de cuales ambientes deben ser los considerados. En particular, redes neuronales son empleadas para identificar con que ambientes un robot establece contacto. Los resultados de ambas funciones aproximación e identificación de parámetros son presentadas. La estructura del modelo del ambiente considerado es relevante a dos aplicaciones espaciales: ejecución cooperativa de tareas por robots y astronautas, y adquisición de muestras durante exploración planetaria. Los complementos en los experimentos de los movimientos han sido desarrollados por un brazo de robot, colocado en contacto con un ambiente electromecánico de un grado de libertad. En estos experimentos, las fuerzas de contacto deseadas son computadas usando redes neuronales, dando el movimiento para la trayectoria deseada.

Ted Hesselroth, Kakali Sarkar, P. Patrick van der Smagt y Klaus Schulten **Neural network control of a pneumatic robot arm** IEEE Transactions on Systems, Man, and Cybernetics, Vol. 24, No. 1 Enero 1994, pp. 28-38	Un algoritmo de mapas neuronales ha sido empleado para controlar un brazo de robot neumático de cinco articulaciones, tomando una retroalimentación continua desde dos cámaras de video. Dos controles de posiciones del brazo, están formados por 200 neuronas constituyendo un espacio de trabajo tridimensional dentro de un sistema coordenado de cuatro dimensiones dado por las dos cámara, así como un conjunto de matrices Jacobianas de 3 x 4 para interpolación entre estas posiciones. La orientación de agarre fue concebida mediante la adaptación de una matriz Jacobiana de 1 x 4 para una cuarta articulación. . La aplicación de correcciones repetidas en cada posición dan como resultado un algoritmo de control robusto. La red neuronal empleada en SoftArm encierra grandes analogías con los modelos de red empleados en los mapas de visualización del cerebro. Por conclusión, dada la similitud entre el SoftArm y el sistema muscular natural el problema de control tiene implicaciones en el control visual-motor de tipo biológico.

Terence D. Sanger **Neural network learning control of robot manipulators using gradually increasing task difficulty** IEEE Transactions on Robotics and Automation, Vol. 10, No. 3 Junio 1994, pp. 323-333	La extensión del aprendizaje de trayectorias es un método incremental para el entrenamiento de redes neuronales artificiales en la aproximación dinámica inversa de un manipulador. Los datos de entrenamiento cercanos a una trayectoria deseada se obtienen por pequeñas variaciones del parámetro de trayectoria de una región. El parámetro puede ser promedio de velocidad, forma de la ruta, ganancia de retroalimentación, o cualquier otro tipo de variable controlable. Una solución aproximada a la dinámica local inversa para cada valor del parámetro es usada como guía de aprendizaje para el siguiente valor del parámetro. Las condiciones de convergencia están dadas por dos variaciones en el algoritmo. Los ejemplos de aplicaciones son presentados con un manejador de brazo de robot de dos articulaciones y una simulación de brazo redundante de tres articulaciones, ambos simulando el uso de un punto de control de equilibrio.

K. H. Kyung, B. H. Lee y M. S. Ko **Acceleration based learning control of robotic manipulators using a multi layered neural network** IEEE Transactions on Systems, Man, and Cybernetics, Vol. 24, No. 8, Agosto 1994, pp. 1265-1272	Se presenta un método de compensación no lineal basado en redes neuronales para el control de trayectorias en manipuladores. Una red neuronal perceptrón multicapa (MLP) es usado para predecir el accionar de los actuadores del robot para seguir una trayectoria, y estos accionares predictivos son aplicados al robot como compensaciones de alimentación en paralelo al controlador lineal de retroalimentación. Un esquema de aprendizaje basado en aceleración es propuesto para ajustar los pesos de conexión en la red para formar un modelo dinámico aproximado del robot. Se presentan los resultados de la simulación que se proponen para hacer que el error converja rápidamente en el sistema de aprendizaje.

4. CONCLUSIONES

..., sólo nos queda una alternativa:
tomar el mundo como un todo y convertirlo
en objeto de nuestro juego; convertirlo en juguete.
"La Inmortalidad"
M. Kundera

En las secciones anteriores se describió el uso de RNA en estructuras de control no lineales, en esta parte se presenta un sumario de las diferentes ventajas de esta tecnología, así como el planteamiento de nuevas direcciones en la aplicación de las mismas Un compendio de las aplicaciones de RNA se encuentra en la última sección de cada capítulo con el propósito de hacer una revisión del estado del arte en las áreas de Robótica y Control En base a esta información se obtienen las conclusiones siguientes

4.1. Sobre las Redes Neuronales Artificiales

A lo largo de la primera parte de este trabajo se presentaron cinco de los paradigmas más usados en las aplicaciones de RNA dentro de la Ingeniería. Los algoritmos descritos aquí, son componentes de una gran variedad de sistemas complejos, algunos aún en desarrollo, en los que las RNA hacen las veces de un cerebro. Sin embargo, no se trata de un cerebro en el sentido humano, sino basado en un planteamiento más formal, ya que el desarrollo de estas aplicaciones está en manos de la Ingeniería Neuronal originada a partir de las técnicas ya establecidas de procesamiento de señales, análisis estadístico y organización de la información. De esta forma, la Ingeniería Neuronal puede distinguirse de las llamadas Neurociencias que tienen como propósito el emular al cerebro humano propiamente.

El propósito de la Ingeniería no está fundamentado en imitar al ser humano en sus características biológicas, pero es necesario tener en cuenta los aspectos de desarrollo que nos han permitido ser la especie más exitosa de la Tierra para poder combinarlos adecuadamente con los conocimientos que ya se han alcanzado, para beneficio del propio ser humano. En este sentido, la Ingeniería Neuronal es un área en desarrollo de la Ingeniería, pues solamente los ingenieros aplican tecnología y conocimiento.

4.2. Redes Neuronales Artificiales en Sistemas Dinámicos

El uso de RNA en Sistemas de Control puede verse como un paso natural en la evolución de la metodología de control para alcanzar nuevos retos. Como características específicas de Redes Neuronales en Control se pueden definir las siguientes propiedades de estas

- *Sistemas No Lineales.* Las Redes Neuronales tienen gran campo de aplicación en control de sistemas no lineales dada su capacidad de aproximación a cualquier mapeo.

- *Procesamiento Paralelo Distribuido.* La estructura fuertemente paralela de las Redes Neuronales las proveen de una capacidad muy grande de tolerancia a fallas.

- *Sistemas Multivariables.* Como extensión del punto anterior, las Redes Neuronales tienen la capacidad de tratar varias señales de entrada y por supuesto, tener varias salidas

- *Generalización de Aprendizaje.* Las Redes Neuronales aprenden haciendo uso de datos pasados, al presentársele nueva información con características semejantes, esta posee la capacidad de generalizar y tratarla eficientemente.

- *Fusión de Datos.* Las Redes Neuronales pueden tratar simultáneamente con información cualitativa y cuantitativa. Es decir, se encuentran en el limite entre la inteligencia artificial tradicional (heuristica) y los sistemas de Ingenieria clásicos.

Por supuesto, aún quedan aspectos inciertos sobre las RNA, muchos de estos aspectos se ven reflejados en la información relevante sobre los ejemplos del presente trabajo:

1 En todos los ejemplos presentados, el tamaño de la RNA (el número de capas, el número de neuronas por capa) se escogió por ensayo y error para aproximar a las respuestas deseadas, es decir, no existe una regla para definir la arquitectura de una RNA

2. La selección de entradas a la RNA no tuvieron que satisfacer condiciones de excitación persistente, como en los casos de Control Adaptable e Identificación, únicamente requieren cierta correlación entre ellas, es decir, pertenecer a un sistema dinámico.

3. Como los algoritmos utilizados son la aplicación del Método de Gradiente Descendiente es necesario que la función de costo asociada tenga un mínimo global para garantizar la convergencia del algoritmo. El poder "brincar" de un mínimo local implica agregar al algoritmo una parte adaptativa sobre el coeficiente de convergencia.

4. Todas las RNA utilizadas sirven como aproximadores no lineales, esto significa que no requieren que les sea asignada una estructura, en lugar de esto, se le asocia una relación entrada-salida que bien puede ser contenida por la dinámica de la RNA si esta es bastante compleja.

4.3. Redes Neuronales Artificiales en el Futuro

Si se revisan cuidadosamente las secciones de aplicaciones que se presentan en este trabajo, se podrá apreciar la diversidad de combinaciones en las que se pueden desarrollar sistemas basados en RNA. Uno de los objetivos del presente es el sensibilizar al lector a realizar trabajos basados en RNA, por ejemplo, se presenta un trabajo sobre el control del péndulo invertido con características operativas diferentes del concepto de retroalimentación ya conocido, en este caso sería interesante validar la efectividad de estos conceptos mediante un trabajo experimental que los compruebe.

Por otra parte, hay trabajos que retoman herramientas ya clásicas de Control como un controlador PID auto ajustado neuromorficamente lo que representa una adaptación sin estructura y que resultaría interesante implantar en sistemas no lineales para comprobar su desempeño.

Dentro del área de robótica se encuentra quizas el mayor número de trabajos que faciliten el uso de RNA como clasificadores, reconocedores de patrones y herramientas de procesamiento de imágenes, además de controladores. Es interesante apreciar que en esta área la integración de sistemas sensorimotores recibe una fuerte atención por parte de los investigadores. Regresando al péndulo invertido, un sistema sensor-motor puede implantarse en él con una cámara CCD calibrada respecto a la vertical y con un motor acoplado que funcione como controlador de tipo "Bang-Bang", puede usarse una RNA que reconozca el patron "péndulo invertido" respecto a la vertical ejerciendo una acción sobre el motor para alcanzar este objetivo de la misma forma en que paramos un palo de escoba en nuestra mano.

Trabajos de esta índole pueden realizarse tanto en seminarios de titulación como en proyectos de materias optativas generando de esta forma antecedentes que fortalezcan el avance académico dentro de una Facultad sin necesidad de comprometer un costo significativo pues sólo se requiere una computadora personal y una tarjeta de adquisición de datos para conectarse a distintos procesos

APÉNDICE

Código de la Red Neuronal empleado para MATLAB .

```
% Neurodin
%
% Red Neuronal de 3 capas (1 intermedia oculta)
% de proposito general
%
help neurodin
clg
hold off
axis('normal')
pausetime=0;
off=0 5;

% Define Vectores de Entrada (pP) y Salida (pT)

pP=[tansig(X1) tansig(X2) tansig(X3) tansig(X4)]';
pT=[tansig(Y1) tansig(Y2) tansig(Y3) tansig(Y4)]';

% Inicializa Arquitectura de RN.
%
% R  -> # Entradas
% S1 -> # Neuronas Capa 1
% S2 -> # Neuronas Capa 2
% S3 -> # Neuronas Capa 3 == # Salidas

[pR, pQ]=size(pP);
[pS3, pQ]=size(pT);
pS1=5; pS2=15;


% Inicializa Pesos y Bias

z=menu('INICIALIZACION DE PESOS Y BIAS ',
     'Inicializa por Nguyen-Widrow',
     'Inicializa RANDOM ',
     'Continua entrenando (DEFAULT) ');
if z==1

        [pW1,pB1]=nwtan(pS1,pR);
```

```
        [pW2,pB2]=nwtan(pS2,pS1);
        pW3=rands(pS3,pS2)*0.5;
        pB3=rands(pS3,1)*0.5;

% Parametros de Entrenamiento

        disp_freq=100;
        freq_change=300;
        inc=100;
        max_epoch=7000;
        err_goal=0.01;
        lr=0.0034432;
        flops(0);
        TP=[disp_freq max_epoch err_goal lr];
        epoch=0;

elseif z==2
      [pW1,pB1]=rands(pS1,pR);
        [pW2,pB2]=rands(pS2,pS1);
        [pW3,pB3]=rands(pS3,pS2);

% Parametros de Entrenamiento

        disp_freq=100;
        freq_change=300;
        inc=100;
        max_epoch=7000;
        err_goal=0.01;
        lr=0.00344342;
        flops(0);
        TP=[disp_freq max_epoch err_goal lr];
        epoch=0;

else
        fprintf('Continua Entrenando...\n');
end
```

```
% Entrenamiento

red1=0;
old_pSSE=0;
pSSE=0;
inc2=300;


while red1==0
        pA1=tansig(pW1*pP,pB1);
        pA2=tansig(pW2*pA1,pB2);
        pA3=tansig(pW3*pA2,pB3);
        pE=pT-pA3;
        pSSE=sumsqr(pE);
        pTR=[pSSE; lr];
        if epoch==freq_change
                if lr>0.01
                        lr=0.01*rand;
                end
                if pSSE > old_pSSE
                        lr=lr*0.7;
                else
                        lr=lr/0.31;
                end
                if lr<1e-6
                        lr=0.001*rand;
                end
                old_pSSE=pSSE;
                freq_change=freq_change+inc2;
        end
        if epoch==disp_freq
                plot(t,[pA3; pT]);pause2(1.5);
                disp_freq=disp_freq+inc;
        end
        if pSSE > err_goal
          pD3=deltatan(pA3,pE);
          pD2=deltatan(pA2,pD3,pW3);
          pD1=deltatan(pA1,pD2,pW2);
          [pdW1, pdB1] = learnbp(pP,pD1,lr);
          [pdW2, pdB2] = learnbp(pA1,pD2,lr);
          [pdW3, pdB3] = learnbp(pA2,pD3,lr);
          pW1=pW1+pdW1; pB1=pB1+pdB1;
          pW2=pW2+pdW2; pB2=pB2+pdB2;
          pW3=pW3+pdW3; pB3=pB3+pdB3;
```

```
        else
          red1=1;
        end
        epoch=epoch+1;
end


% Muestra Progreso

totalflops=flops;
plot(pt,pE);
pause
barerr(pA3,pT);
pause
plot(t,[pA3; pT]);
pause
fprintf('Average de %.0f flops/epoch.\n',...
round(totalflops/epoch));
if z==1 fprintf('Por  Nguyen-Widrow.\n')
end
fprintf('La Suma Final de Error Cuadratico es %g \n',pSSE);
fprintf('El entrenamiento fue ');
if pSSE<err_goal
        disp('ADECUADO.')
else
        disp('NO satisfactorio.')
end
```

BIBLIOGRAFÍA

Frank L. Lewis ***Applied Optimal Control & Estimation: Digital Design and Implementation*** Prentice Hall Englewood Cliffs NJ 1992

Benjamin C. Kuo ***Automatic control systems*** 6 ed. Prentice Hall Englewood Cliffs NJ 1991

Benjamin C. Kuo ***Digital control systems*** 2 ed. Saunders College Pub. Fl 1992

Mark W. Spong and M. Vidyasagar ***Robot dynamics and control*** Jhon Wiley NY 1989

Frank L. Lewis, K. Liu, and A. Yesildirek **Neural net robot controller with guaranteed tracking performance** Proc. IEEE Int. Symp. Intelligent Control, pp. 225-231,Aug. 1993

Junhong Nie and D.A. Linkens **A hybrid neural network based self organizing controller** Int. J. Control, vol. 60, 2, Febr. 1994, p.p.197-222

Reynold Chu, Rahmat Shoureshi and Manoel Tenorio **Neural Networks for System Identification**IEEE Control Systems Magazine American Control Conference, 1989, p.p.31-35

Jiabin Hao, Shaohua Tan and Joos Vandewalle **A Rule-Based Neural Controller for Inverted Pendulum** IEEE International Joint Conference on N.N., 1993, p.p. 534-539

Kumpati Narendra and Sruchasis Mukhopadhyay **Intelligent control using Neural Network** IEEE Control Systems Magazine on Neural Networks in Control Systems, Apr. 1992, p.p.11-18

Richard S. Sutton, Andrew G. Barto and Ronald J. Williams **Reinforcement learning is Direct Adaptive Optimal Control** IEEE Control Systems Magazine on Neural Networks in Control Systems, Apr. 1992, p.p.19-22

B Wu **An introduction to neural networks and their applications in manufacturing** J.Intell. Manuf., Dec. 1993, p.p.391-403

A. Freitas da Rocha **Evolutive fuzzy neural networks** IEEE International Conference on Fuzzy Systems, March 1992, p.p. 493-500

E. Wong **Neural computing and stochastic optimization** Future Tendencies in Computer Science, Control and Applied Mathematics Intrnal. Conf., Dec. 1993, p.p. 339-342

Kumpati S. Narendra and Kannan Parthasarathy **Gradient methods for the optimization of dynamical systems containing neural networks** IEEE Transactions on Neural Networks, March 1991, p.p.252-262

Esther Levin, Raanan Geuirtzmann and Gideon F. Inbar **Neural network architecture for adaptive system modeling and control** Neural Networks, vol 4, 1991, p.p. 185-191

Naveen V. Bhat, Peter A. Minderman, Thomas McAvoy and NAm San Wang **Modelling chemical process systems via neural computation** IEEE Control Systems Magazine, 1990, p.p. 24-30

Sinnasamy R. Naidu, Evanghelos Zafiriou and Thomas J. McAvoy **Use of neural networks for sensor failure detection in a control system** IEEE Control Systems Magazine (American Control Conference), 1990, p.p. 49-55

Kumpati S. Narendra and Kannan Parthasarathy **Identification and control of dynamical systems using neural networks** IEEE Transactions on Neural Networks, March 1990, p.p. 4-27

Anthony N. Michel and Jay A. Farrell **Associative memories via artificila neural networks** IEEE Control Systems Magazine, 1990, p.p. 6-17

Derrich H. Nguyen and B. Widrow **Neural networks forself-learning control systems** IEEE Control Systems Magazine, 1990, p.p. 18-23

Stephen H. Lane, David A. Handelman and Jack J. Gelfand **Theory and development of higher-order CMAC neural networks** IEEE Control Systems Magazine, Apr. 1992, p.p.23-30

Gordon Kratt and David Campagna **A comparison between CMAC neural network control and two traditional adaptive control systems** IEEE Control Systems Magazine, 1990, p.p. 36-43

Fu Chuang Chen **Backpropagation neural netwoks for nonlinear self-tuning adaptive control** IEEE Control Systems Magazine, 1990, p.p. 44-48

Panos J. Antsaklis **Neural networks in control systems** IEEE Control Systems Magazine, 1990, p.p. 3-5

Charles W. Anderson **Learning to control an inverted pendulum using neural networks** IEEE Control Systems Magazine, Apr. 1989, p.p. 31-37

Jiabin Hao, Shaohua Tan and Joos Vandewalle **A rule-based neural controller for inverted pendulum system** IEEE International Joint Conf. on N.N., 1993, p.p. 534-539

Saifu Akhyar and Sigeru Omatu **Neuromorphic self-tuning PID Controller** IEEE /INNS Int. Joint Conf. on Neural Networks1993, p.p. 552-557

Can Isik and A. Mete Cakmakci **Identification of a nonlinear multivariable dynamic process using feedforward networks** IEEE /INNS Int. Joint Conf. on Neural Networks 1993, pp. 564-567

Asriel U. Levin and Kumpati S. Narendra **Stabilization of nonlinear dynamical systems using neural networks** IEEE International Joint Conference on N.N., 1992, p.p.275-280

Panos J. Antsaklis **Neural networks in control systems** IEEE Control Systems Magazine on Neural Networks in Control Systems, Apr. 1992, p.p.8-10

Yashundo Takhashi **Control Adaptable Predictivo de Sistemas Dinámicos No Lineales Usando Redes Neuronales** Paper Code NN WE P2 OR-2-3 Mayo 1992, II-UNAM

Michael Kuperstein and Jorge Rubinstein **Implementation of an adaptive neural controller for sensory motor coordination** IEEE Control Systems Magazine, Apr. 1989, pp. 25-30

Huan Liu, Thea Iberall and George A. Bekey **Neural network architecture for robot hand control** IEEE Control Systems Magazine, Apr. 1989, pp. 38-43

Kevin M. Passino, Michael A. Sartori and Panos J. Antsaklis **Neural computing for numeric to symbolic conversion in control systems** IEEE Control Systems Magazine, Apr. 1989, pp. 44-52

Rolf Eckmiller **Neural nets for sensory and motor trajectories** IEEE Control Systems Magazine, Apr. 1989, pp. 53-58

W. Thomas Miller, Robert P. Hewes, Filson H. Glanz and L. Gordon Kraft **Real time dynamic control of an industrial manipulator using a neural network based learning controller** IEEE Trans. on Robotics and Automation, Vol. 6, No. 1, Febr. 1990, pp. 1-9

Owen Holland and Martin Snaith **Q -learning with generalisation: an architecture for real world reinforcement learning in a mobile robot** IEEE/INNS International Joint Conferences on Neural Networks 1992, Vol. I-287-292

Hai-Long Pei, T. P. Leung and Qi-Jie Zhou **Backward construction** IEEE/INNS International Joint Conferences on Neural Networks 1992, Vol. I-293-298

Sukhan Lee **A new neural net approach to robot 3D perception and visuo-motor coordination** IEEE/INNS International Joint Conferences on Neural Networks 1992, Vol. I pp. 299-307

L. Monostori and D. Barschdorff **Artificial neural networks in intelligent manufacturing** Robotics & Computer-Integrated Manufacturing GB 1992 Vol. 9, No. 66, pp. 21-437

Zheng Geng and Leonard S. Haynes **Neural networks solution for the forward kinematics problem of a Steward platform** Robotics & Computer-Integrated Manufacturing GB 1992 Vol. 9, No. 66, pp. 485-495

Hideki Hashimoto, Takashi Kubota, Masaaki Kudou and Fumio Harashima **Self organizing visual servo system based on neural networks** IEEE Control Systems Magazine, Apr. 1992, pp. 31-36

Luis C. Rabelo and Xavier J. R. Avula **Hierarchical neurocontroller architecture for robotic manipulation** IEEE Control Systems Magazine, Apr. 1992, pp. 37-41

D. J. Cooper, L. Megan and R. F. Hinde, Jr. **Disturbance pattern classification and neuro adaptive control** IEEE Control Systems Magazine, Apr. 1992, pp. 42-48

Y. C. Pati, P. S. Krishnaprasad and Martin C. Peckerar **An analog neural network solution to the inverse problem of "Early Taction"** IEEE Transactions on Robotics and Automation, Apr. 1992, Vol. 8, No. 2, pp. 196-212

Sumio Watanabe and Masahide Yoneyama **An ultrasonic visual sensor for 3D object recognition using neural networks** IEEE Transactions on Robotics and Automation, Apr. 1992, Vol. 8, No. 2, pp. 240-249

Hillel J. Chiel, Randall D. Beer, Roger D. Quinn and Kenneth S. Espenschied **Robustness of a distributed neural network controller for locomotion in a hexaped robot** IEEE Transactions on Robotics and Automation Jun. 1992, Vol. 8, No. 3, pp. 293-303

Dan Simon **Neural network based robot trajectory generation** IEEE/INNS International Joint Conferences on NN 1993, pp.540-545

Kunio Iwata, Joydeep Gosh and Yoan Shin **Time optimal control using PI-SIGMA network** IEEE/INNS International Joint Conferences on NN 1993, pp. 546-551

D. H. Rao and M. M. Gupta **Dynamic neural controller with somatic adaptation** IEEE/INNS International Joint Conferences on NN 1993, pp. 558-563

Chris M. Jubein, Nikitas J. Dimopoulos **Identificatio of a PUMA 560 two link robot using a stable neural network** IEEE/INNS International Joint Conferences on NN 1993, pp. 568-572

Thomas Martinetz and Klaus Schulten **A neural network with Hebbian like adaptation rules learning visuomotor coordinetion of a PUMA robot** IEEE/INNS International Joint Conferences on NN 1993, pp. 820-822C

D. Sbarbaro-Hofer, D. Neumerkel and K. Hunt **Neural control of a steel rolling mill** IEEE, Control Systems Magazine Jun. 1993, pp. 69-75

S. T. Venkataraman, S. Gulati, J. Barhen and N. Toomarian **A neural network based identification of enviroments models for compliant control of space robots** IEEE Transactions on Robotics and Automation, Vol. 9, No. 5 Oct. 1993, pp. 685-697

Ted Hesselroth, Kakali Sarkar, P. Patrick van der Smagt and Klaus Schulten **Neural network control of a pneumatic robot arm** IEEE Transactions on Systems, Man, and Cybernetics, Vol. 24, No. 1 jan. 1994, pp. 28-38

Terence D. Sanger **Neural network learning control of robot manipulators using gradually increasing task difficulty** IEEE Transactions on Robotics and Automation, Vol. 10, No. 3 Jun. 1994, pp. 323-333

K. H. Kyung, B. H. Lee and M. S. Ko **Acceleration based learning control of robotic manipulators using a multilayered neural network** IEEE Transactions on Systems, Man, and Cybernetics, Vol. 24, No. 8 Aug. 1994, pp. 1265-1272