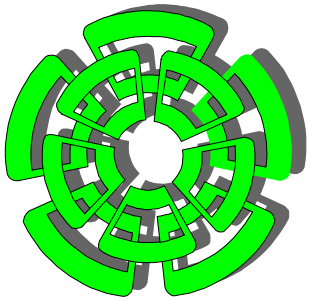

CENTRO DE INVESTIGACIÓN Y DE ESTUDIOS AVANZADOS
DEL
INSTITUTO POLITÉCNICO NACIONAL

DEPARTAMENTO DE CONTROL AUTOMÁTICO

Redes Neuronales Difusas para modelado via agrupanmiento en-línea: Aplicación a un condensador de aspiración

TESIS QUE PRESENTA EL:

M. en C. Andres Ferreyra Ramirez

PARA OBTENER EL GRADO DE
Doctor en Ciencias en la Especialidad de Control Automático

DIRECTOR DE TESIS:
Dr. Wen Yu Liu

México, D.F., 2005

Índice general

Capítulo 1

Introducción

Durante los últimos 30 o 40 años las investigaciones en redes neuronales y en sistemas difusos han sido independientes debido en gran parte a los diferentes orígenes y motivaciones de ambos paradigmas. Hoy es ampliamente conocido que la lógica difusa ofrece un marco muy poderoso para el razonamiento aproximado que intenta modelar el proceso de razonamiento humano en un nivel cognoscitivo. Los sistemas difusos adquieren el conocimiento gracias al dominio del experto el cual es codificado dentro del algoritmo en términos de un conjunto de reglas difusas IF-THEN. Los sistemas difusos emplean este enfoque basado en reglas e interpolan el razonamiento para responder a entradas nuevas. En contraste, las redes neuronales ofrecen una arquitectura altamente estructurada, con capacidades de aprendizaje y generalización, que intenta imitar el mecanismo neurológico del cerebro. Una red neuronal almacena conocimiento de una manera distribuida dentro de sus pesos, los cuales han sido determinados por medio del entrenamiento (aprendizaje) con muestras conocidas. La habilidad de generalización para entradas nuevas está basada en la estructura algebráica inherente de la red neuronal. Ha sido demostrado que ambas tecnologías tienen capacidades de aproximación universal.

Recientemente ha habido un interés considerable en la integración de las redes neuronales y la lógica difusa. Las dos tecnologías pueden ser vistas como complementarias y como consecuencia, el enfoque combinado proporciona un marco de razonamiento aproximado

poderoso que tiene capacidades de aprendizaje y generalización. La tecnología combinada también cubre algunas de las desventajas de los enfoques individuales como por ejemplo la transparencia de la red neuronal y la capacidad de aprendizaje limitada de los sistemas difusos.

En los ultimos años han habido varios sistemas neurodifusos exitosos reportados en la literatura . En general estos son sistemas con una estructura fija que interpretan un sistema difuso en términos de una red neuronal tal que cada paso en el proceso es equivalente a por lo menos una capa de la red. Así, la mayoría de estas arquitecturas tienen al menos una capa correspondiente a fusificación, intersección, implicación y defusificación. Estas arquitecturas difieren de una red neuronal convencional en términos de la uniformidad tanto de los nodos de procesamiento como de la estrategia de la interconexión con el resultado que con frecuencia requieren algoritmos de entrenamiento modificados. Sin embargo, estas arquitecturas no proporcionan un procedimiento sistemático para determinar el número de reglas del sistema de inferencia difuso. Hay arquitecturas que fijan el número de reglas antes de realizar el entrenamiento de la red mientras que otras fijan los conjuntos difusos de la parte IF de las reglas lo que establece un límite para el número de reglas. Por lo tanto, determimar el numero de reglas a utilizar representa un serio problema para los sistemas neurodifusos. Algunos sistemas neurodifusos consideran al número de reglas como un parámetro de diseño más, el cual puede ser determinado utilizando un algoritmo de grupamiento de datos y los datos de entrada-salida del sistema a modelar. Estas técnicas de agrupamiento proporcionan el planteamiento para encontrar prototipos que caracterizan un conjunto de datos. Estos prototipos son entonces utilizados como reglas difusas en un sistema neuro difuso. Los métodos de agrupamiento desasarrollados funcionan bastante bien, sin embargo, procesan los datos fuera de línea por lo que la estructura temporal de los mismos es ignorada, lo que representa un problema cuando se quiere modelar o controlar un sistema en tiempo real. Por lo tanto, resulta interesante el desarrollar técnicas de agrupamiento que procesen los datos en línea para poder modelar sistemas vía redes neuronales difusas en tiempo real.

1.1. Motivación

La selección de reglas difusas encuentra una cantidad considerable de observación heurística para expresar estrategias apropiadas del conocimiento. Es difícil que los expertos humanos examinen todos los datos de entrada-salida de los sistemas complejos para encontrar las reglas apropiadas para un sistema difuso. Se proponen varios acercamientos para generar reglas difusas de datos numéricos [3] [27] [28] [29] [46]. Estos planteamientos consisten de dos fases de aprendizaje: el aprendizaje de la estructura que implica encontrar las variables de entrada principales de todas las posibles, especificando las funciones de membresía, la partición del espacio de entrada y determinando el número de reglas difusas; y el aprendizaje de los parámetros que implica la determinación de los parámetros no conocidos y la optimización de los ya existentes en el modelo, usando algún método de optimización basado en la información lingüística obtenida del humano experto y de los datos numéricos obtenidos del sistema actual que se modelará. Estas dos fases de aprendizaje están interrelacionadas, y ninguna de ellas puede ser realizada independientemente de la otra. Tradicionalmente estas fases se realizan secuencialmente, la actualización de los parámetros se utiliza después de que la estructura se decide. Esto es conveniente solo para la operación off-line. La mayoría de métodos de identificación de la estructura se basan en el agrupamiento de datos como por ejemplo: agrupamiento fuzzy C-means, agrupamiento de montaña, agrupamiento substractivo [28]. Estos planteamientos requieren que todos los datos de entrada-salida estén listos antes de que comenzar a identificar la planta, por lo que son conocidos como off-line. Algunos métodos en línea pueden encontrarse en [33], pero sólo son funcionales para un sistema neuro-difuso especial (self-constructing neural fuzzy inference network).

**Existen dos debilidades principales en estos método que agrupan en línea: (a) En la identificación no lineal del sistema, los datos de la entrada y de la salida están marcados por tiempo, pero sus particiones no consideran el mismo intervalo del tiempo. (b) En los parámetros que actualizan la fase, se utilizan todos los datos. Pero los datos en cada grupo corresponden a diversas reglas. En esta tesis, proponemos un acercamiento que agrupa en línea (de la novela) para superar las dos debilidades antes dichas en la identificación del

sistema. En el proceso de agrupamiento, la relación de tiempo en la entrada y la salida se considera. Porque cada grupo corresponde a una regla, en nuestro algoritmo cada regla es entrenada por sus datos del grupo, esto es más eficaz. También damos unos algoritmos que manejan parametros de ganancia variable en el tiempo, nosotros probamos que los algortimos son estables.

1.2. Objetivo de la tesis

En esta tesis se propone un planteamiento novedoso para el agrupamiento en línea que se pueda aplicar en una clase general de redes neuronales difusas donde tanto la identificación de la estructura como el aprendizaje de parámetros son en línea. El nuevo método de agrupamiento para la identificación de la estructura puede separar datos de entrada-salida en diversos grupos (número de la reglas) de datos de entrada-salida en línea. Para el aprendizaje de parámetros, nuestro algoritmo tiene dos ventajas sobre los otros. Primero, los métodos normales para la identificación de parámetros se basan en una estructura fija (ya que transforman los sistemas difusos en redes neuronales difusas) y en la totalidad de los datos, por ejemplo ANFIS [29], pero después del agrupamiento sabemos que cada grupo corresponde a una regla, así que entrenamos a cada regla por su grupo de datos, lo cual es más efectivo. En segundo lugar, se dá un rango de aprendizaje variante en el tiempo para el algoritmo comúnmente usado (backpropagation), probando que el nuevo algoritmo es estable y más rápido que el algoritmo backpropagation que utiliza un rango de aprendizaje fijo.

1.3. Estructura de la tesis

Esta tesis está estructurada dentro de tres partes lógicas que contienen 6 capitulos.

En la primera parte considerada de *principios básicos y metodologías*, el capitulo 2 presenta una revisión de la literatura que cubre sistemas difusos, redes neuronales, sistemas neuro-difusos e identificación de sistemas dinamicos. Este capítulo presenta una introducción a los tipos principales de sistemas difusos y redes neuronales, mostrando ejemplos de

algunos de estos. Introduce conceptos básicos y metodologías concretas de los sistemas difusos y de las redes neuronales intentado resaltar las ventajas y desventajas de cada una de estas metodologías así como también de resaltar su semajanza. Se discute además el modelado del comportamiento del experto y presenta las posibilidades fundamentales de combinar redes neuronales y sistemas difusos considerando aspectos que juegan un papel importante en la combinación.

En la segunda parte considerada de *identificación de la estructura utilizando agrrupamiento en linea*, el capitulo 3 presenta un procedimiento para modelar sistemas no-lineales utilizando sistemas difusos, el cual consiste de dos etapas: la identificación de la estructura y la estimación de parámetros. En este capítulo se desarrolla la identificación de la estructura del sistema difuso utilizando datos de entrada-salida del sistema a modelar. Se discuten diferentes técnicas de agrupamiento de datos fuera de linea (off line) y se presenta una técnica de agrupamiento substractivo doble en línea. También se presenta una técnica para la extracción tanto de las funciones de membresía como de las reglas difusas en linea.

En la tercera parte considerada de *identificación de la estructura y los parametros*, los capitulos 4 y 5 discuten dos procedimietos diferentes para realizar la identificación de sistemas no-lineales (estimación de parametros de un sistema difuso) con metodos neuro-difusos y en el capitulo 6 se muestra la aplicación exitosa de estos metodos para la identificación de un proceso dinámico no-lineal del mundo real . El capitulo 4 muestra la identificación de sistemas utilizando redes neuronales difusas estáticas, donde se discuten y comparan dos técnicas de entrenamiento: gradiente descendente y aprendizaje estable. En este capitulo, se discute la identificación de parámetro utilizando una o varias reglas difusas para cada grupo obtenido en el proceso de agrupamiento. Por su parte, en el capitulo 5 se discute la identificación de sistemas utilizando redes neuronales difusas dinámicas, donde la identificación se basa en los estados medibles del sistema.

Estas dos técnicas de identificación son aplicadas exitosamente a la *identificación de un Condensador de Aspiración* utilizando una red neuronal difusa en el capitulo 6.

1.4. Publicaciones realizadas

En general, se puede decir que en esta tesis se obtuvo una contribución en el área de **. Las publicaciones realizadas son:

1. Wen Yu and Andrés Ferreyra, System Identification with State-Space Recurrent Fuzzy Neural Networks, *43rd IEEE Conference on Decision and Control*, CDC'04, Paradise Island, Bahamas, 5106-5111, 2004

2. Andrés Ferreyra and Wen Yu, On-line fuzzy neural modeling with structure and parameters updating, *2004 IEEE International Conference on Computational Intelligence for Measurement Systems and Applications*, Boston, USA, 127-130, 2004

3. Andrés Ferreyra and Wen Yu, Identificación de sistemas, SOMI XIX Congreso de Instrumentación, Pachuca Hidalgo, Mex, 123-125, 2004.

4. Andrés Ferreyra, Arquitecturas Neuro-Difusas para Control Inteligente,- - - - - - - - - - - -, - - - - - - - - - -, 2004

5. Andrés Ferreyra and Wen Yu, On-line clustering for nonlinear system identification using fuzzy neural networks with application on EMI sensor, sometido a *Fuzzy Sets and Systems,* 2004

6. Wen Yu and Andrés Ferreyra, Stable fuzzy ientification using recurrent fuzzy neural networks, sometido a *IEEE Trans. Syst., Man, Cybern. B,,* 2004

Capítulo 2

Sistemas Neuro Difusos

2.1. Modelando lógica difusa

2.1.1. Reglas difusas IF-THEN

Las reglas difusas IF-THEN o declaraciones condicionales difusas son expresiones de la forma $IF\ A\ THEN\ B$, donde A y B son etiquetas linguisticas de conjuntos difusos. Debido a su forma consistente, las reglas difusas if-then son confrecuencia empleadas para capturar los modos imprecisos de razonamiento que juegan un papel esencial en la habilidad humana para fabricar decisiones en un ambiente de incertidumbre e impresición. Un ejemplo que describe un hecho simple es:

$$IF \quad \text{x es A}, THEN \quad \text{y es B}$$

donde x y y son variables linguisticas, A y B son valores linguisticos o etiquetas que son caracterizadas por FM's. Con frecuencia la proposición difusa (x es A) es llamada el *antecedente* o la *premisa* mientras que la proposición difusa (y es B) es llamada la *consecuencia* o *conclusión.*

Otra clase de regla difusa if-then, fue propuesta por Takagi and sugeno, tiene conjuntos difusos implicados solo en la parte premisa. Usando las reglas difusas if-then de tipo sugeno,

podemos describir por ejemplo la fuerza de resistencia de un objeto en movimiento de la siguiente manera:

$$IF \text{ velocidad es Alta } THEN \text{ Fuerza} = \text{k*(velocidad)}^2$$

donde, nuevamente, Alta en la parte premisa es una etiqueta linguistica caracterizada por una FM apropiada. Sin embargo, la parte consecuente es descrita por una ecuación no difusa de la variable de entrada, velocidad.

Ambos tipos de reglas difusas if-then han sido usadas extensamente tanto para modelado como para control. A través del uso de etiquetas linguisticas, una regla difusa if-then puede capturar facilmente el alma (espiritu,fantasma) de una "regla general" utilizada por humanos. Desde otro punto de vista, debido a los calificadores en la parte premisa, cada regla difusa if-then puede ser vista como una descripción local del sistema bajo consideración. Las reglas difusas if-then forman la parte esencial del sistema de inferencia difuso, un sistema basado en reglas difusas ampliamente utilizado como un control difuso, que sera discutido en la siguiente sección.

2.1.2. Sistema de Inferencia Difuso

Los sistemas de inferencia difusos son tambien conocidos como sistemas basados en reglas, modelos deifusos, memorias asociativas difusas (FAM), o controles difusos cuando son utilizados como controles. Basicamente un sistema de inferencia difuso esta compuesto de cinco bloques funcionales (figura 2.1):

- Una **base de reglas** que contiene un numero de reglas difusas if-then;
- Una **base de datos** que define las funciones de membresía de los conjuntos difusos utilizados en las reglas difusas;
- Una **unidad de fabricación de decisiones** que realiza las operaciones de inferencia en las reglas;

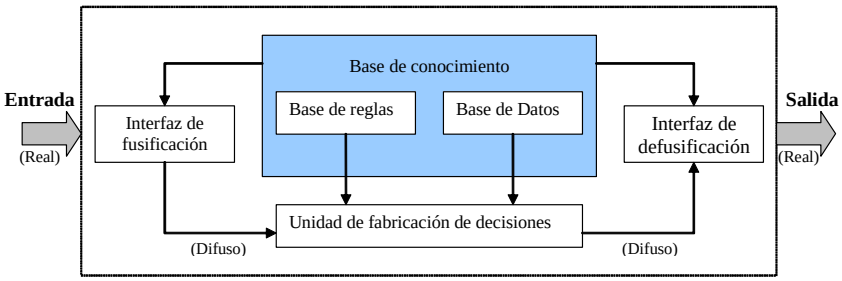

Figura 2.1: Sistema de Inferencia Difuso

- Una **Interfaz de fusificación** que transforma las entradas reales dentro de grados de igualdad con valores linguisticos;
- Una **Interfaz de defusificación** que transforma los resultados difusos de la inferencia dentro de una salida real.

Normalmente, la base de reglas y la base de datos conjuntamente referidos como la **base de conocimiento**.

Los pasos del rasonamiento difuso (operaciones de inferencia sobre las reglas difusas if-then) realizados por el sistema de inferencia difuso son:

1. Comparar las variables de entrada con las FM's en la parte premisa para obtener los valores de membresía (o medidas de compatibilidad) de cada etiqueta linguistica. Este paso es con frecuencia llamado fusificación.
2. Combinar (a través de un operador T-norma especifico, normalmente multiplicación o minimo) los valores de membresía de la parte premisa para obtener la fuerza de disparo (peso) de cada regla.
3. Generar la consecuencia calificada (ya sea difusa o real) de cada regla dependiendo de la fuerza de disparo.
4. Conjuntar la consecuencia calificada para producir una salida real. Este paso es llamado defusificación.

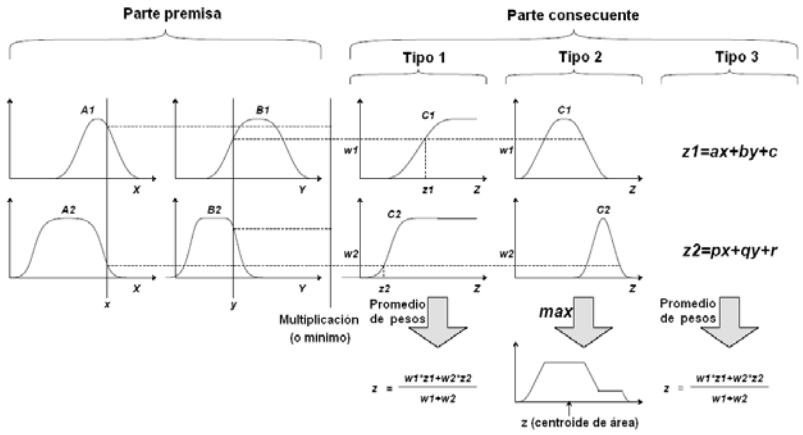

Figura 2.2: Mecanismos de reasonamiento difuso y reglas difusas if-then comúnmente utilizados.

Diferentes tipos de rasonamientos han sido propuestoa en los ultimos años. Dependiendo del tipo de rasonamiento difuso y del tipo de reglas difusas if-then empleadas. la mayoria de sistemas de inferencia difusos puede ser clasificados en tres tipos diferentes (figura 2.2):

Tipo 1: La salida total es el promedio de pesos de cada salida real de cada regla inducida por la fuerza de disparo de la regla (el producto o el minimo del grado de igualdad con la parte premisa) y las funciones de membresía de salida. Las funciones de membresía usadas en este esquema deben de ser monotonicamente no-decrecientes.

Tipo 2: La salida total es derivada por aplicar la operación "max" a la salida difusa calificada (cada una de las cuales es igual a el minimo de la fuerza de disparo y la función de membresía de salida de cada regla). Varios esquemas han sido propuestos para elegir la salida real final basada en la salida difusa total; algunos de ellos son centro de area, bisector de area, media de maximos, criterio maximo, etc.

Topo 3: Las reglas difusas if-then de tipo sugeno son usadas. La salida de cada regla es una combinación lineal de las variables de entrada mas un termino constante, y la salida final es el promedio de pesos de la salida de cada regla.

1. La figura (2.2) utiliza un sistema de inferenci difuso de dos entradas y dos reglas para mostrar diferentes tipos de reglas difusas y rasonamiento difuso mencionados anteriormente. Hay que estar consiente de que la diferencia se encuentra en la especificación de la parte consecuente (funciones de membresía monotonicamente no-decrecientes, campanas o funciones reales) y de esta manera los esquemas de defusificación (promedio de pesos, centroide de area, etc) son tambien diferentes.

Una base de reglas difusas consiste de una colección de reglas difusas de la forma

$$R_u^{(l)} : IF \;\; x_1 \;\text{ is }\; A_1^l \;\text{ and }\; \ldots \;\text{ and }\; x_n \;\text{ is }\; A_n^l, \;\; THEN \;\; y \;\text{ is }\; B^l \tag{2.1}$$

donde A_i^l es un conjunto difuso en $U_i \subset R$ y B^l es un conjunto difuso en $V \subset R$, y $x = (x_1, x_2, \ldots, x_n) \in U$ son las variables de entrada al sistema difuso y $y \in V$ es la salida del sistema difuso.

Las reglas de la forma de (2.1) son conocidas como reglas difusas canonicas ya que incluyen muchos otros tipos de reglas difusas.

La base de reglas difusas es el corazón del sistema difuso en el sentido de que todos los demás componentes son utilizados para implementar estas reglas de una manera razonable y eficiente. La práctica ha demostrado que las reglas difusas en la foma de (2.1) proporcionan un marco muy conveniente para que el humano experto exprese su conocimiento del dominio.

2.2. Redes Neuronales

En esta sección se proporciona una introducción comprensible de algunos modelos importantes de redes neuronales que son necesarios como antecedentes para los capítulos posteriores. Para aquellos lectores interesados en un tratamiento minucioso de los modelos de redes neuronales que se comentan en esta sección, se hace la referencia a literatura especializada en esta area de investigación.

La sección inicia con una clasificación de las redes neuronales considerando la estructura o el algoritmo de aprendizaje, continua con una discusión de uno de los modelos más impor-

tantes utilizado en un 90 % de las aplicaciones de redes neuronales: el perceptrón multicapa entrenado por el algoritmo de aprendizaje backpropagation. Finalmente se incluyen algunos modelos de redes neuronales recurrentes.

2.2.1. Definición básica

Las redes neuronales (Neural Networks, NNs) consisten normalmente de un número de elementos de procesamiento (PEs) o neuronas interconectadas. Las conexiones son arreglos entre neuronas y la naturaleza de éstas determina la estructura de la red. Como la fortaleza de las conexiones es ajustada o entrenada para alcanzar un comportamiento deseado, la red es gobernada por sus algoritmos de aprendizaje. Las redes neuronales pueden ser clasificadas de acuerdo a sus estructuras o algoritmos de aprendizaje.

Clasificación según su estructura.

En términos de su estructura, las NNs pueden ser divididas en dos tipos: redes neuronales unidireccionales (Feedfordward neural networks) y redes neuronales recurrentes (Recurrent Neural Networks).

Definición 2.1 Feedfordward neural network*s* (FFNNs). *En una FFNN, las neuronas son normalmente agrupadas en capas. Las señales fluyen desde la capa de entrada hasta la capa de salida vía conexiones unidireccionales, las neuronas son conectadas de una capa a la siguiente, pero no dentro de la misma capa. Ejemplos de FFNNs incluyen la perceptrón multicapa (MLP) [49], la red de vector de cuantización de aprendizaje (LVQ) [49] [40], red de control de articulación del modelo cerebral (CMAC) [2] y la red de manejo de datos con método de grupos (GMDH) [25] [68]. La mayoría de FFNNs normalmente realizan un mapeo estático entre un espacio de entrada y un espacio de salida: la salida en un instante dado es una función sólo de la entrada en ese instante.*

Definición 2.2 Recurrent neural networks (RNNs). *En una RNN, las salidas de algunas neuronas son retroalimentadas a las mismas neuronas o a neuronas en capas precedentes.*

Así, las señales pueden fluir tanto en direcciones hacia adelante como hacia atrás. Ejemplos de RNNs incluyen a la Red de Hopfield [49] [21], La red Elman [65] y la red Jordan [65]. Las RNNs tienen una memoria dinámica: sus salidas en un instante dado reflejan las entradas actuales, así como las entradas y salidas previas.

Clasificación según el algoritmo de aprendizaje.

Las RNs son entrenadas por dos tipos principales de algoritmos de aprendizaje: algoritmos de aprendizaje supervisado y no supervisado. Además, existe un tercer tipo, aprendizaje reforzado, que puede ser considerado como una forma especial del aprendizaje supervisado.

Definición 2.3 Aprendizaje supervisado*: Un algoritmo de aprendizaje supervisado ajusta las fortalezas o pesos de las conexiones entre neuronas de acuerdo a la diferencia entre las salidas deseada y actual de la red correspondientes a una entrada dada. De esta manera, el aprendizaje supervisado requiere un* maestro *o* supervisor *para proporcionar las señales de salida deseadas u objetivo. Ejemplos de algoritmos de aprendizaje supervisado incluyen la regla delta, la regla delta generalizada o algoritmo de retropropagación [15] [85] y el algoritmo LVQ [40].*

Definición 2.4 Aprendizaje no supervisado: *Los algoritmos de aprendizaje supervisado no requieren que la salida deseada se conozca. Durante el entrenamiento, sólo los patrones de entrada son presentados a la NN que automáticamente adapta los pesos de sus conexiones para agrupar los patrones de entrada dentro de grupos con características similares. Ejemplos de algoritmos de aprendizaje no supervisado incluyen: el algoritmo de Kohonen [40], la teoría de resonancia adaptiva de Carpenter y Grossberg [49] [8], y los algoritmos de aprendizaje competitivo.*

Definición 2.5 Aprendizaje reforzado: *Como se mencionó anteriormente, el aprendizaje reforzado es un caso especial del aprendizaje supervisado. En lugar de utilizar un maestro para proporcionar la salidas deseadas, un algoritmo de aprendizaje reforzado emplea un crítico sólo para evaluar la calidad de la salida de la NN correspondiente a una entrada dada. Un ejemplo de un algoritmo de aprendizaje reforzado es el algoritmo genético (GA).*

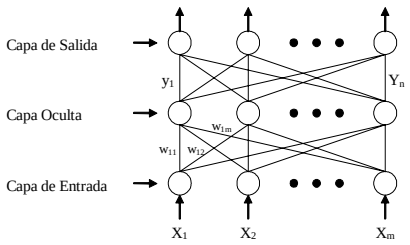

Figura 2.3: Red neuronal de tipo Perceptrón Multicapa (MLP).

2.2.2. Tipos de redes neuronales

Esta sección describe brevemente los ejemplos de las redes neuronales y los algoritmos de aprendizaje asociados citados previamente.

Perceptrón multicapa (MLP)

Las MLPs son quizás el tipo más conocido de FFNN. La figura (2.3) muestra una MLP con tres capas: una capa de entrada, una capa de salida y una capa intermedia o capa oculta. Las neuronas en la capa de entrada sólo actúan como buffers para distribuir las señales de entrada x_i a las neuronas en la capa oculta. Cada neurona j (figura 2.4) en la capa oculta suma todas sus señales de entrada x_i después de sopesarlas con el peso de sus respectivas conexiones w_{ji} desde la capa de entrada y calcula su salida y_j como una función f de la suma,

$$y_j = f\left(\sum w_{ji}x_i\right)$$

f puede ser una simple función de umbral o una sigmoidal, tangente hiperbolica o función de base radial (ver tabla 1.1).

La salida de las neuronas en la capa de salida es calculada de la misma manera.

El algoritmo de backpropagation (BP), un algoritmo de gradiente descendente, son los algoritmos de entranamiento mas comunmente adoptados por MLP. Los cambios Δw_{ji} en los pesos de una conección entre las neuronas i y j son calculadas como:

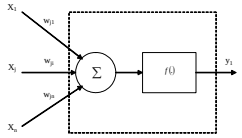

Figura 2.4: Detalles de una neurona en una red neuronal MLP.

$$\delta_j = \left(\frac{\partial f}{\partial net_j}\right)\left(y_j^{(t)} - y_j\right)$$

y para las neuronas ocultas

$$\delta_j = \left(\frac{\partial f}{\partial net_j}\right)\sum_q w_{qj}\delta_q$$

En la ecuación (superior), net_j es la suma total de pesos de las señales de entrada a la neurona y y $y_j\,(t)$ es la salidad objetivo de la neurona j.

Como no hay salidas deseadas para las neuronas ocultas, en la ecuación (inferior), la diferencia entre la salida deseada y la salida actual de la neurona oculta j es reemplazada por la suma de pesos de los términos δ_q ya obtenidos para las q neuronas conectadas a la salida de j. De esta manera, iterativamente, iniciando con la capa de salida, el término δ es calculado para las neuronas en todas las capas y la actualización de pesos determinada por todas las conexiones. El proceso de actualización de peses puede tomar lugar después de la presentación de cada uno de los patrones de entrenamiento (entrenamiento basado en patrones) o después de la presentación del conjunto completo de patrones de entrenamiento (entrenamiento batch). En cualquier caso, una época (epoch) de entrenamiento se dice que ha sido completada cuando todos los patrones de entrenamiento han sido presentados una vez a la MLP.

Para casi toda la mayoría de problemas triviales, varias épocas son requeridas para que la MLP sea entrenada apropiadamente. Un método adoptado comúnmente para aumentar la velocidad de entrenamiento es agregar un término de "momentum.[a] la ecuación (1.2) que efectivamente permite cambiar los pesos previos influenciando el cambio nuevo de pesos, con

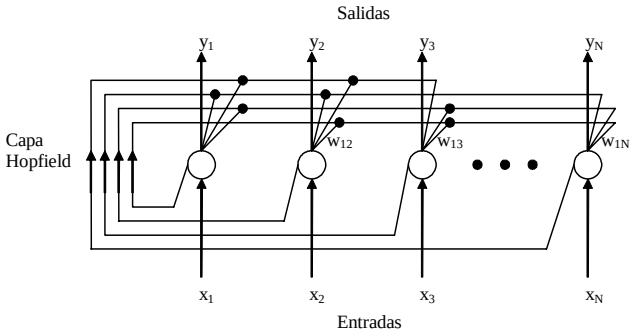

Figura 2.5: Red Hopfield.

$$\Delta w_{ji}(k+1) = \eta \delta_j x_i + \mu \Delta w_{ji}(k)$$

Donde $\Delta w_{ji}(k+1)$ y $\Delta w_{ji}(k)$ son cambios de pesos en las épocas $(k+1)$ y (k) respectivamente y μ es el coeficiente "momentum"

Red Hopfield

La figura (2.5) muestra una versión de la red Hopfield. Esta red acepta normalmente entradas binarias y bipolares (+1 o -1). Tiene una sola capa de neuronas, cada una conectada con todas las otras, dándole una estructura recurrente, según lo mencionado anteriormente.

El entrenamiento de la red Hopfield toma solamente un paso, los pesos de la red w_{ij} son asignados directamente como sigue:

$$w_{ij} = \begin{cases} \frac{1}{N} \sum_{c=1}^{P} x_i^c x_j^c & , \quad i \neq j \\ 0 & , \quad i = j \end{cases}$$

donde w_{ij} es el peso de la conección desde la neurona i hasta la neurona j, y x_i^c (que es o +1 o -1) es el ith componente del patron de entrenamiento de entrada para la clase c, P el numero de clases y N el numero de neuronas (o el numero de componentes en el patron de entrada). Note de la ecuación (1.15) que $w_{ij} = w_{ji}$ y $w_{ii} = 0$, un conjunto de condiciones que garantizan la estabilidad de la red. Cuando se introduce un patron no conocido a la red,

las salidas son inicialmente colacadas (hechas, asignadas) iguales a los componentes de los patrones no conocidos, viz.

$y_i(0) = x_i \qquad 1 \leq i \leq N$

Comenzando con estos valores iniciales, la red itera según la siguiente ecuación hasta alcanzar un estado de energía minimo, es decir, sus salidas se estabilizan a valores constantes:

$$y_i(k+1) = f\left[\sum_{j=1}^{N} w_{ij} y_i(k)\right] \qquad 1 < i \leq N$$

donde f es una función restrictiva fuerte (de limite duro) como

$$f(x) = \begin{cases} -1 & x < 0 \\ +1 & x > 0 \end{cases}$$

Las redes de Jordan y Elman

Las figuras (2.6) y (2.7) muestran una red de Elman y una red de Jordan, respectivamente. Estas redes tienen una estructura multicapa similar a la estructura de la MLPs. En ambas redes, además de una capa oculta ordinaria, hay otras capas ocultas especiales algunas veces llamadas capas de contexto o capas de estado. Estas capas reciven señales retroalimentadas desde la capa oculta normal (en el caso de la red de Elman) o desde la capa de salida (en el caso de la red de Jordan). La red de Jordan tambien tiene conecciones desde cada neurona en la capa contexto de nuevo a sí misma. Con ambas redes, las salidas de las neuronas en la capa contexto, son alimentadas hacia delante a la capa oculta. Si solo las conecciones hacia delante van a ser adaptadas y las conexiones de retroalimentación son preestablecidas (preprogramadas) a valores constantes, estas redes pueden ser consideradas FFNN ordinarias y el algoritmo BP puede ser utilizado para entrenarlas.

2.3. Sistemas Neuro Difusos

Con anterioridad se han discutido tanto las NNs y FSs como sistemas $stand - alone$. Comenzando esta sección, se exploran las características de las combinaciones de ambos

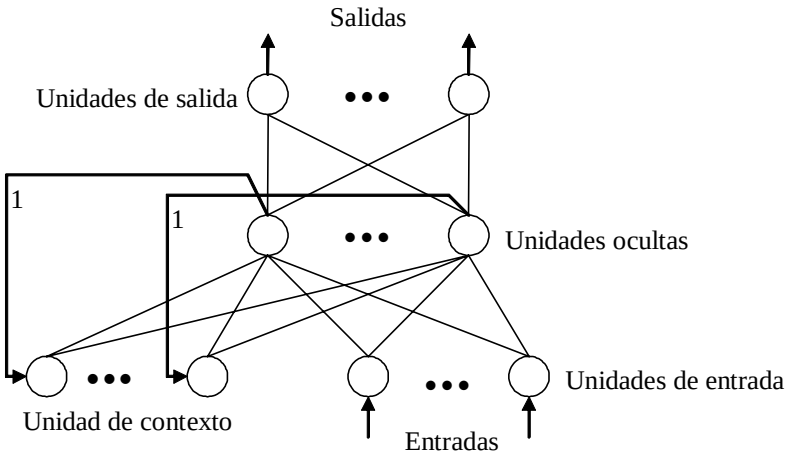

Figura 2.6: Red Elman.

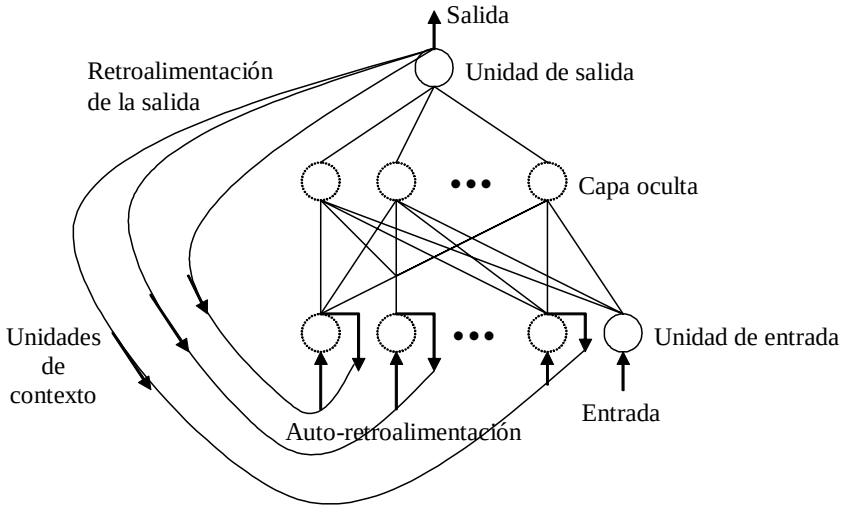

Figura 2.7: Red Jordan.

planteamientos. En esta sección se discuten algunos aspectos fundamentales de las combinaciones neuro-difusas, y en las siguientes secciones se presentan varios planteamientos.

2.3.1. Modelando el comportamiento del experto

Los FSs y las NNs son herramientas computacionales, adecuadas para modelar comportamientos de expertos. Si tenemos conocimiento expresado en reglas lingüísticas, podemos construir un FS, y si tenemos datos o podemos aprender de una simulación, podemos usar NNs. Sin embargo, no es tan fácil como suena. Construir un FS significa especificar ciertos parámetros como conjuntos difusos, t-normas, s-normas, implicaciones difusas, máquina de inferencia, fusificador y defusificador, entre otros. Aunque tengamos reglas lingüísticas, no es tarea fácil. Si usamos NNs, debemos tomar decisiones sobre la arquitectura y los parámetros de aprendizaje.

Podemos también considerar a las NNs y a los FSs como herramientas convenientes para resolver un problema sin tener que analizar el problema en sí a detalle. Al reunir los datos o las reglas lingüísticas de los expertos, podemos concentrarnos en la solución. Nos satisface llegar a un bosquejo de la solución, si la podemos obtener rápido, fácil y a un bajo costo. Sin embargo, si estamos en la necesidad de una solución exacta, sin importar el costo, entonces los planteamientos tradicionales del modelo de la naturaleza del problema, y la derivación de una solución son prioritarios. No tiene sentido sustituir una solución difusa o neuronal para una solución que trabaje sólo por el gusto de usar tal planteamiento.

En la siguiente sección se discutirán las ventajas y desventajas del modelado con NNs y con FSs para modelar el comportamiento experto. Debe puntualizarse que no se desea modelar exactamente al experto. Estamos interesados sólo en un sistema intuitivo y de fácil implementación, que produzca resultados similares a los que pueda producir un experto; por lo que, no podemos presumir que una NN sea un modelo aceptable o admisible de las capacidades de aprendizaje humanas, ni que una regla (difusa) lingüística establezca la base del modelo en la que un humano llega a sus conclusiones. Veremos entonces que una combinación de ambos métodos tiene sentido y se presentarán entonces algunas ideas básicas

de cómo combinarlas.

2.3.2. Combinando Redes Neuronales y Sistemas Difusos

Las NNs y los FSs son utilizados para resolver problemas de aproximación de funciones, pueden ser descritos encontrando valores de salida adecuados para los valores de entrada. Esta es la clase de problemas es en la que nos concentraremos, no se considerarán problemas donde se deban encontrar relaciones (mapeo de una entrada a varias salidas) o procedimientos (mapeo de una entrada a una secuencia de acciones) como soluciones.

Si queremos usar una NN para resolver un problema, éste debe describirse lo suficiente por medio de muestras de datos. Si conocemos datos de entradas/salidas, usamos aprendizaje supervisado (como con retropropagación en perceptrones multicapa), si no conocemos nada acerca de estos datos pero tenemos cuantificados los efectos del error causados por las salidas de la NN, podemos usar el aprendizaje reforzado. Podemos usar las NNs si tenemos datos de entrenamiento pues no necesitamos un modelo matemático del problema ni tampoco ningún conocimiento previo. Sin embargo, no podemos interpretar la solución obtenida del proceso de aprendizaje. La NN es una caja negra, y no podemos usarla para ver si una solución es admisible, es decir, su estado final no puede interpretarse en términos de reglas, lo que significa que no podemos inicializar una NN con conocimientos previos si ya tenemos algunos, pues la red por lo general debe aprender desde el principio. El procedimiento de aprendizaje por si mismo puede tomar mucho tiempo y no tenemos garantía de éxito. Un atributo bien conocido de las NNs es la tolerancia al fallo debido a sus entradas y cambios en su estructura. Sin embargo, si el problema de interés cambia demasiado comparado con los datos de entrenamiento anteriores, se debe re-entrenar pues la red no será capaz de hacer frente al problema. Esto no es garantía de que la reanudación del proceso de entrenamiento la conducirá a una rápida adaptación del problema modificado y puede ser necesario repetir el aprendizaje desde el principio otra vez.

Un FS puede usarse para resolver un problema si tenemos conocimiento sobre la solución con reglas lingüísticas IF-THEN. Definiendo conjuntos difusos adecuados para representar

términos lingüísticos usados en nuestras reglas, podemos crear el FS de esas reglas; no necesitamos un modelo formal para el problema de interés, ni necesitamos datos de entrenamiento, que pudieran tener una obtención costosa. Por otro lado, estamos perdidos sin reglas IF-THEN, y para hecer funcionar al FS podemos necesitar un largo proceso. Para esta etapa del procesos de implementación, no hay método formal, de modo que sólo se puede aplicar la heurística. Además de añadir, quitar o cambiar las reglas difusas de los conjuntos difusos también se usa el peso de las reglas, pero éstos destruyen la semántica de un FS y hacen la interpretación del sistema difícil o imposible. Los FSs también son considerados tolerantes al fallo respecto a pequeños cambios en sus entradas o parámetros del sistema, sin embargo es posible reducir la representación de un FS modificando ligeramente sólo una de sus funciones de membresía. Adaptar manualmente un FS a un problema no es nada fácil debido a que pequeñas modificaciones pueden causar cambios drásticos en la representación. Además, la optimización del sistema puede guiarse por más de un criterio (por ejemplo la velocidad o el consumo de mínima energía), por lo que es deseable tener un proceso automático de adaptación similar al entrenamiento de una NN y soporte de optimización de un FS.

La razón más importante para combinar FSs con NNs es la capacidad de aprendizaje pues tales combinaciones son capaces de aprender reglas lingüísticas y/o funciones de membresía, o de optimizar las ya existentes. Aprender, en este caso significa crear una base de reglas o funciones de membresía de improvisaciones basadas en el entrenamiento de datos presentados como un problema de aprendizaje fijo o libre [59] [60]. Para crear una base de reglas por lo menos deben tenerse las funciones de membresía iniciales. Considérense las siguientes tres posibilidades:

1. El sistema comienza sin reglas, y crea nuevas reglas hasta que el problema de aprendizaje se resuelve. La creación de una nueva regla se obtiene por un patrón de entrenamiento que no es cubierto lo suficiente por la base de reglas actual [4], [58]. Esta aproximación puede generar grandes bases de reglas si las funciones de membresía no se escogen adecuadamente, por ejemplo si no se translapan lo suficiente. Esto es comparable a una mala generalización de una NN. Dependiendo del problema de aprendizaje es posible obtener una base de reglas inconsistente. El sistema posiblemente deba borrar

algunas reglas después de la generación de la base de reglas.

2. El sistema comienza con todas las reglas que pueden ser creadas debido a la partición de las variables y borra las reglas insuficientes de la base de reglas [56]. Para este procedimiento se necesita una evaluación del funcionamiento de las reglas individuales. Estos planteamientos encuentran problemas complejos cuando se aplican a problemas con gran número de variables. No se permiten bases de reglas inconsitentes porque parte de la evaluación es una prueba de consistencia. Así, mediante este procedimientoes posible obtener bases de reglas con muy pocas reglas.

3. El sistema comienza con una base de reglas (posiblemente escogida al azar) con un cierto número de reglas que son reemplazadas durante el aprendizaje[69] mientras se revisa la consistencia de la base de reglas a cada paso. La desventaja de este planteamiento es el número fijo de reglas y el hecho de que debe implementarse un esquema de evaluación para el borrado de reglas y un procedimiento de análisis de datos para la adquisición de nuevas reglas; de no hacerse, el aprendizaje equivale a una investigación estocástica. En el acontecimiento de degradación del funcionamiento, los reemplazos pueden ser posiblemente cancelados.

Optimizar una base de reglas corresponde parcialmente a aprenderla. Para algunas situaciones existen ya reglas, algunas faltantes deben ser todavía descubiertas y algunas reglas superfluas deben borrarse. En este sentido el tercer planteamiento mencionado antes debe verse como un procesamiento de optimización.

Aprender u optimizar funciones de membresía es menos complejo que adaptar una base de reglas. Las funciones de membresía pueden describirse facilmente por parámetros, que pueden ser optimizados con respecto a la medida del error global. Por medio de restricciones adecuadas, ciertas condiciones pueden ser resueltas, por ejemplo las funciones de membresía deben translaparse. La adaptación de parámetros es una tarea estándard para las NN, y hay muchas combinaciones con FSs para aprender funciones de membresía [55], [56], [59].

Podemos distinguir dos aproximaciones para aprender funciones de membresía:

1. Las funciones de membresía se describen por parámetros que se optimizan en el proceso de aprendizaje.

2. Una NN aprende a producir valores de membresía para valores de entrada dadas usando una muestra de datos.

La segunda aproximación tiene el inconveniente que las funciones de membresía no son explícitamente conocidas y entonces, por lo general se escoge la primera aproximación.

Mientras que la capacidad de aprendizaje es una ventaja desde el punto de vista de un FS, desde el punto de vista de una NN hay ventajas adicionales para sistemas combinados. Debido a que los sistemas neuro difusos se basan en reglas ligüísticas podemos integrar facilmente al sistema conocimiento previo. Las reglas establecidas y las funciones de membresía pueden inicializar al sistema, y esto sustancialmente acorta el proceso de aprendizaje. La adaptación da lugar a una modificación conveniente de la base de reglas y/o de las funciones de membresía, y esto significa que el resultado de aprendizaje se interpreta como un sistema convencional difuso. De este modo, el aspecto de caja negra de una NN se evita , y es posible obtener nuevos conocimientos del sistema.

La arquitectura de un modelo neurodifuso es normalmente determinada por las reglas y los conjuntos difusos usados por el problema en cuestión [55] [56] así que no es necesario especificar ningún parámetro de la red (por ejemplo, el número de neuronas ocultas). Las desventajas que no pueden evitarse, aún por una combinación, son problemas con el proceso de aprendizaje pues no se garantiza un resultado exitoso del entrenamiento o bien, el ajuste puede ser envano.

2.3.3. Que son los Sistemas Neuro-Difusos?

Como puede observarse de la sección anterior, la idea de un sistema neurodifuso es encontrar los parámetros de un FS por medio del aprendizaje de métodos obtenidos de una NN. En esta sección listaremos algunas propiedades básicas de los sistemas neurodifusos antes de considerar ciertos tipos de ellos en la siguiente sección.

Un camino común para aplicar el algoritmo de aprendizaje de un FS es representarlo en una arquitectura especial tipo NN, lo cual es bastante fácil. Entonces un algoritmo de aprendizaje -como el de retropropagación- es usado para entrenar al sistema. Sin embargo, hay algunos problemas: los algoritmos de aprendizaje de una NN son normalmente métodos de gradiente descendente y no pueden aplicarse directamente a un FS porque las funciones usadas para el proceso de inferencia por lo general no son diferenciables. Hay dos soluciones a este problema:

(a) Remplazar las funciones usadas en el FS (como min y max) por funciones diferenciables.

(b) No usar el algoritmo de aprendizaje neuronal a base de gradientes pero sí un procedimiento más apropiado.

Los sistemas neurodifusos modernos por lo general se representan como FFNNs multicapas.

El modelo ANFIS [29] por ejemplo, implementa un FS basado en reglas difusas de tipo Sugeno, el cual está representado por una arquitectura de una FFNN de cinco capas. El ANFIS usa una mezcla de retropropagación (para aprender los parámetros antecedentes) y la estimación de mínimos cuadrados (para determinar los parámetros consecuentes). Un paso en el procedimiento de aprendizaje tiene dos partes: en la primera, los patrones de entrada se propagan, y los parámetros consecuentes óptimos se estiman por un procedimiento iterativo de mínimos cuadrados, mientras que se asume que los parámetros antecedentes son fijados por el ciclo actual a través del conjunto de entrenamiento. En la segunda parte, los patrones se propagan de nuevo, y en esta época, la retropropagación se usa para modificar los parámetros antecedentes, mientras que los parámetros consecuentes permanecen fijos. Este procedimiento es iterativo. Ya que un sistema de inferencia difuso de tipo Sugeno utiliza solo funciones diferenciables, ANFIS va directo a la solución (a).

El modelo GARIC (Generalized Aproximate Reasoning-based Intelligent Control) [5] es una modelo neuro-difuso que implementa un control difuso utilizando varias FFNNs. Consiste de una red de evaluación (AEN) y una red de acción (ASN). La red de acción es una FFNN de

cinco capas la cual utiliza funciones especiales "suave mínimo" las cuales son diferenciables por lo que también hace de (a) su solución.

El problema con la solución (a) es que a veces los modelos no son de fácil interpretación como por ejemplo FS tipo Mamdani. Otros modelos como el NEFCON, EfuNN, SONFIN, FINEST, NEFCLASS y NEFPROX. usan la solución (b) -estos son sistemas difusos tipo Mamdani y usan algoritmos de aprendizaje especiales.

El modelo neuro-difuso NEFCON [57] [62] es diseñado para implementar un control difuso utilizando reglas difusas de tipo Mamdani. NEFCON hace uso de un algoritmo de aprendizaje de tipo de reforzamiento para aprender la base de reglas (aprendizaje de la estructura) y utiliza un algoritmo de retropropagación difuso para el aprendizaje de los conjuntos difusos. NEFCON es capaz de incorporar conocimiento previo a demás de aprender de improvisaciones. Sin embargo el rendimiento del sistema puede ser muy dependiente de factores heurísticos como la razón de aprendizaje, el error medido, etc.

El modelo neuro-difuso EfuNN [35] [36] implementa una base de reglas de tipo Mamdani, basado en una estructura dinámica (creando y borrando estrategias); una sola regla de inferencia, establecida dentro de todas las reglas ganadoras para las reglas de activación del nodo, con un entrenamiento de un paso, como en los casos de entrenamiento y razonamiento.

SONFIN [33] implementa una base de reglas difusas modificadas de tipo Takagi-Sugeno-Kang, que fueron creadas y adaptadas como procedimientos de aprendizaje en línea vía estructura simultánea e identificación de parámetros. SONFIN es quizás el modelo neuro-difuso más costoso computacionalmente hablando. La red es adaptable a las especificaciones y requerimientos de exactitud del usuario. Los parámetros de precondición son ajustados por el algoritmo de retropropagación y los parámetros consecuentes por los algoritmos de mínimos cuadrados o mínimos cuadrados recursivos. Para resaltar la habilidad de representación del conocimiento del SONFIN, una transformación lineal para cada variable de entrada puede incorporarse a la red así que se necesitarán mucho menos reglas, o se alcanzará una mejor aproximación. Transformaciones lineales apropiadas también se aprenden dinámicamente en la fase de identificación de parámetros del SONFIN. SONFIN puede usarse para operaciones normales en cualquier tiempo durante el proceso de aprendizaje sin

la repetición de entrenamientos en los patrones de entrada-salida cuando las operaciones en línea se requieran.

Además, para las FFNNs multicapas hay combinaciones de técnicas difusas con otras arquitecturas de NNs, por ejemplo, los mapas característicos de auto-organización o las memorias asociativas difusas. Algunos planteamientos se abstienen de representar a un FS en una arquitectura neuronal. Estos sólo aplican el procedimiento de aprendizaje a los parámetros del FS, o explícitamente usan una NN para determinarlos.

Hay diferentes planteamientos que tienen mucho en común, pero difieren en aspectos de implementación. Para enfatizar con las características en común de todos estos planteamientos, y darles un significado adecuado al término de *sistemas neuro-difusos,* debemos restringirlos a sistemas que tengan las siguientes propiedades:

(i) Un sistema neurodifuso es aquél que es entrenado por un algorirtmo de aprendizaje derivado (generalmente) de de la teoría de NNs. El procedimiento de aprendizaje (heurístico) opera con información local y sólo causa modificaciones locales en el FS en cuestión. El proceso de aprendizaje no se basa en el conocimiento, sino en el manejo de datos.

(ii) Un FS puede siempre (es decir, antes, durante y después del aprendizaje) ser interpretado como un sistema de reglas difusas. Es posible crear el sistema sin datos de entrenamiento desde el principio e inicializarlo con conocimientos previos de las reglas difusas.

(iii) El procedimiento de aprendizaje de un sistema neurodifuso adquiere sus propiedades semánticas del FS en cuestión. Esto resulta en limitaciones de las posibles modificaciones de los parámetros del sistema.

(iv) Un sistema neurodifuso aproxima una función n-dimensional (desconocida) que es parcialmente dada por los datos de entrenamiento. Las reglas difusas codificadas dentro del sistema representan muestras vagas y pueden verse como prototipos vagos de datos

de entrenamiento. Un sistema neurodifuso no debería verse como una clase de sistema experto(difuso) y no tiene nada que ver con lógica difusa en sentido estrecho.

(v) Un sistema neurodifuso puede ser visto como una especie de NN unidireccional (feedforward) de tres capas. Las unidades en esta red usan t-normas o t-conormas en vez de las funciones de activación normalmente usadas en NNs. La primera capa representa las variables de entrada, la capa intermedia (capa oculta) representa las reglas difusas y la tercera capa representa las variables de salida. Los conjuntos difusos son codificados como pesos difusos de conexión.

Algunos modelos neuro-difusos usan más de tres capas, y codifican los conjuntos difusos como funciones de activación. En este caso, es posible transformarlas en arquitecturs de tres capas.

Esta forma de ver a los FSs ilustra el flujo de datos dentro del sistema y su naturaleza paralela. Sin embargo esta forma de ver a una NN no es un prerrequisito para aplicar un procedimiento de aprendizaje, es sólo por conveniencia y algunos planteamientos neurodifusos no lo utilizan.

Las técnicas neuro-difusas, entoces, se usan para derivar FSs de datos, o para aumentarlos con los ejemplos de aprendizaje. La implementación exacta de los modelos neurodifusos no importa. Es posible usar una NN para aprender ciertos parámetros de un FS, como usando un mapa característico auto-organizado para encontrar reglas difusas, o para ver a los FSs como una NN especial y para aplicar un algoritmo de aprendizaje directamente. En la siguiente sección se discuten algunos posibles tipos de sistemas neurodifusos.

2.3.4. Tipos de Sistemas Neuro-Difusos

En general, se distinguen dos clases de combinaciones entre NNs y FSs que tienen como meta optimizar un FS. En el primer planteamiento, la NN y el FS trabajan por separado. La combinación está en la determinación de ciertos parámetros de un FS por una NN, o un algoritmo de aprendizaje de una NN. Esto puede hacerse fuera de línea (offline) o en

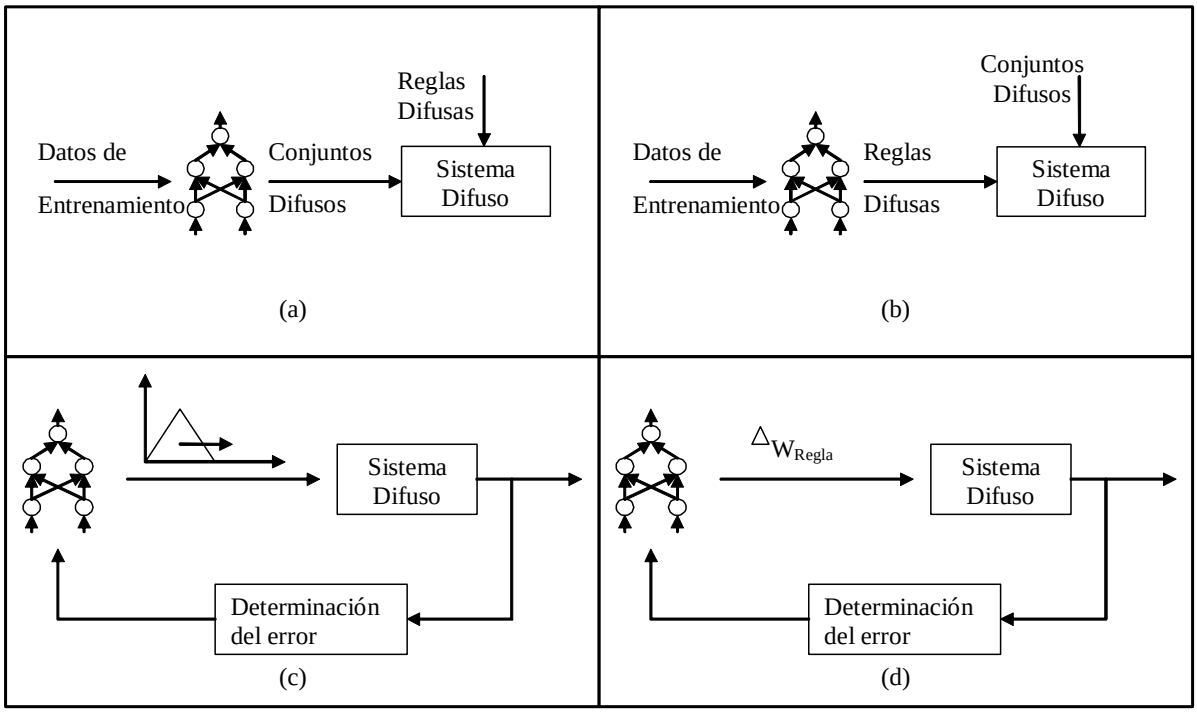

Figura 2.8: Diferentes Sistemas Neuro-Difusos Cooperativos

línea durante el uso de un FS. A esta clase de combinación le llamamos *sistema neurodifuso cooperativo*. Usamos este término porque la NN coopera con el FS ayudándole a encontrar parámetros adecuados.

La segunda clase de combinación define una arquitectura homogénea, generalmente similar a la estructura de una NN. Esto puede hacerse interpretando un FS como una clase especial de NN, o implementando un FS usando una NN. A esta clase de combinación se le llama *sistema neurodifuso híbrido*. Usamos el término "híbrido"porque el sistema resultante puede verse como una NN o como un FS, pero tenemos un solo sistema que no podemos dividir en dos distintos subsistemas.

La figura (2.8) muestra cuatro clases diferentes de sistemas neuro-difusos cooperativos.

(a) Una NN determina las funciones de membresía de los datos de entrenamiento. Esto

puede hacerse determinando parámetros adecuados o por aproximaciones de funciones de membresía con una NN. Los conjuntos difusos se especifican (offline) de este modo se usan junto con reglas difusas dadas separadamente para implementar un FS. Los datos de entrenamiento deben especificar grados para los valores de entrada dados.

(b) Una NN determina las reglas difusas de los datos de entrenamiento. Esto se hace por un planteamiento de agrupamiento normalmente implementada por mapas característicos auto-organizados o arquitecturas neuronales similares. La NN es usada después de implementar el FS y aprende las reglas de forma offline. Las funciones de membresía necesarias deben especificarse por separado. En lugar de NNs pueden usarse algoritmos de agrupamiento difusos.

(c) El sistema aprende los parámetros en línea, es decir durante el uso del FS para adaptar las funciones de membresía. Para este modelo, deben darse las reglas difusas y las funciones de membresía iniciales. Además debe especificarse una medida del error que guíe al proceso de aprendizaje de la NN. Si hay un problema de aprendizaje fijo, este modelo también puede aprender de modo offline. Por lo general no hay una NN real presente en este planteamiento, pero sí un algoritmo de aprendizaje neuronal, por ejemplo una forma de retropropagación.

(d) Una NN (o algoritmo de aprendizaje para NN) determina pesos de reglas para reglas difusas ya sean online u offline; tales factores a menudo se interpretan como la "influencia"de una regla y se multiplican con las salidas de la red. Las precondiciones son como en (c). Sin embargo, la semántica de tales factores de ponderación no es clara. Los pesos de las reglas pueden reemplazarse modificando las funciones de membresía, si esto se hace, los pesos de las reglas pueden destruir la interpretación de los conjuntos difusos. Además, los pesos de las reglas pueden causar valores lingüísticos idénticos para ser representados de forma distinta en diferentes reglas.

Junto con estos modelos cooperativos hay enfoques en los que una NN se usa como preprocesador o postprocesador de un FS (ver figura 2.9). Tales combinaciones no optimizan un

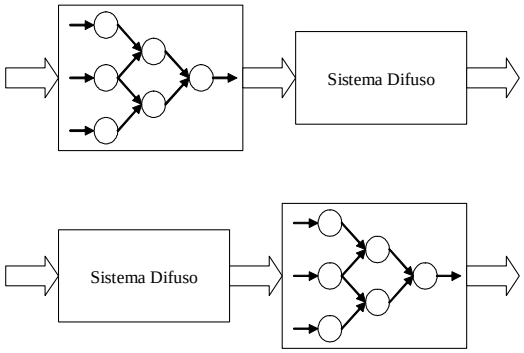

Figura 2.9: Redes Neuronales para procesar Entradas/Salidas de un Sistema Difuso.

FS sólo logran mejorar la representación del sistema combinado. El aprendizaje toma lugar sólo en la NN; el FS queda sin cambios. Las NNs usadas despliegan el comportamiento usual de la caja negra, lo cual hace al enfoque adecuado para situaciones donde la interpretación de los resultados de aprendizaje es de menor importancia, y donde un sistema simple es necesario y pude ser entrenado. En contraste a un enfoque puro de NN, el uso de un FS específico provee la ventaja de una rápida implementación usando conocimientos previos.

Estos modelos *neuro-difusos concurrentes* son usados especialmente si las variables de entrada del controlador no pueden medirse directamente, y deben ser creados por una combinación de valores numéricos. De este modo, la NN puede funcionar como un çompresor de información.ªdaptivo. Por otro lado, la salida de un controlador difuso puede no ser aplicable directamente a un proceso, en cuyo caso puede necesitar combinarse con otros parámetros. Esto puede hacerse mediante una NN. No se habla en este caso de sistemas neuro-difusos pues este término se reserva para enfoques donde los métodos de NNs se usan para determinar los parámetros de un FS.

La figura (2.10) muestra una representación esquemática de un modelo neurodifuso híbrido en forma de controlador neuro-difuso. La ventaja de tal enfoque híbrido es una arquitectura unificada, que realiza la comunicación entre diferentes sistemas superfluos. Tal sistema es normalmente capaz de realizar aprendizaje online y offline. El módulo neurodifuso en la

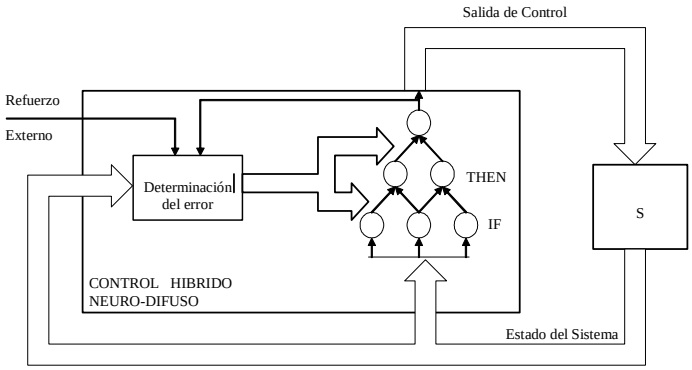

Figura 2.10: Controlador Neuro-Difuso Hibrido

figura (2.10) muestra un componente para determinar la señal de error. Esta componente puede verse como una característica importante del algoritmo de aprendizaje y sólo se muestra para su comprensión.

La idea de un enfoque híbrido es interpretar la base de reglas de un FS en términos de una NN. Los conjuntos difusos pueden verse como pesos mientras que las variables de entrada/salida y las reglas pueden interpretarse como neuronas. De este modo un FS puede verse como un caso especial de NN o puede emularse por una red feedfordward. El algoritmo de aprendizaje trabaja -como es usual para sistemas de conexión- modificando su estructura y/o parámetros, es decir por inclusión o exclusión de neuronas y por adaptación de pesos. Estos cambios causados por el proceso de aprendizaje pueden interpretarse desde el punto de vista tanto de NNs como de FSs. Este último aspecto es importante, porque el comportamiento usual de una caja negra de una NN se elimina. Un proceso de aprendizaje exitoso es idéntico al incremento en conocimiento explícito, representado en la forma de una base de reglas difusas.

Un sistema neurodifuso híbrido aprende en modo supervisado. Hay un problema de aprendizaje fijo o un problema de aprendizaje libre junto con el reforzamiento de la señal. Un problema de aprendizaje fijo puede ser construido al observar a un experto y se usa para el aprendizaje offline. También es posible crear un aprendizaje fijo en el vuelo,es decir, en

línea. Si hay ya una solución para el problema de interés, por ejemplo un controlador de una planta, el sistema neurodifuso puede aprender a reconstruir esta solución; este sería un caso raro porque un nuevo sistema neuro difuso es requerido por lo general si no existen ya soluciones. Sin embargo, si hay una solución en la forma de una NN, es interesante usarla como entrenador de un sistema neuro difuso. Si el aprendizaje es exitoso, obtendríamos una solución interpretable que pueda reemplazar al formador de la NN, que contenga el conocimiento sobre la solución sólo en forma distribuída. El sistema neurodifuso provee este conocimiento explícitamente en forma de reglas lingüísticas.

Es también posible que sólo haya una una solución parcial (convencional) para el problema, un controlador que sólo haga frente a ciertos estados de la planta. En este caso el controlador puede usarse como entrenador de un sistema neuro-difuso, y el número de estados controlados pueden después incrementarse por reforzamiento del aprendizaje.

El reforzamiento del aprendizaje puede llevarse a cabo en línea en el problema real o en una simulación del problema, y es principalmente usado en aplicaciones de control. Los datos de entrenamiento están dados por los estados del sistema y las señales de reforzamiento guían al proceso de aprendizaje; por esto, es necesario saber los estados deseados y conocimiento formulado explicitamente acerca de estados "buenos.º "malos". Por lo general esto es más fácil que proveer un problema de aprendizaje fijo.

Antes de emplear una combinación de NNs y FSs, es necesario revisar si las precondiciones se conocen, es decir si hay suficientes datos de entrenamiento o un procedimiento de evaluación para construir un problema de aprendizaje. Además, debe revisarse si ha de modificarse un FS ya existente o si los conjuntos y reglas difusas han de determinarse por completo.

Si un FS existente es optimizado, las funciones de membresía por lo general son la única parte que cambia. Una modificación de la base de reglas puede significar que una regla deba añadirse o borrarse, pero esto no siempre es deseable, porque la base de reglas refleja el conocimiento considerado como válido. Una mala interpretación de un FS es culpable en este caso de insuficientes funciones de membresía. Si la base de reglas también es modificada durante el aprendizaje, podría significar que las reglas disponibles hasta ese momento sean

usadas como aprendizaje previo. Durante el proceso de aprendizaje puede desarrollarse una base de reglas completamente diferente, removiendo al conocimiento previo completamente del FS.

Para optimizar conjuntos difusos podemos -dependiendo del problema de aprendizaje- usar una de las aproximaciones cooperativas. El enfoque (a) puede usarse sólo si está disponible un problema de aprendizaje fijo que especifique los grados de membresía. Tales datos de entrenamiento por lo general no están disponibles, pero si lo estuvieran, el aprendizaje se realizaría de modo offline y el aprendizaje habría de ser evaluado usando el FS modificado; si su representación no es suficiente, el proceso de aprendizaje tendría que ser resumido con más datos de entrenamiento. El enfoque (c) debe ser usado si es posible el reforzamiento del aprendizaje o si se dispone de un problema de aprendizaje fijo que describa las salidas correctas del FS. Este enfoque es el más usado y puede encontrarse en productos japoneses de consumo.

En principio, es también posible optimizar un FS usando el método mostrado en la figura (2.8 d), porque la multiplicación de la salida de la regla por un factor es equivalente a modificar las funciones de membresía de salida. Sin embargo encontramos problemas semánticos si usamos este método. Similar a este enfoque son las combinaciones neuro/difusas concurrentes (figura 2.9). Tales sistemas son usados también por algunos productos japoneses de consumo que se supone se adaptan constantemente a las necesidades del consumidor.

Si debe diseñarse completamente un FS nuevo, entonces se prefiere un enfoque neurodifuso. Usando un enfoque cooperativo podría ser necesario usar una sucesión de aproximaciones. Después de especificar los conjuntos difusos iniciales, las reglas difusas pueden aprenderse con el modelo de la figura (2.8b) y las funciones de membresía pueden optimizarse con un enfoque (a) o (c).

Sin embargo, un apropiado sistema híbrido tiene la ventaja de llevar todas las tareas en una arquietectura unificada, también es posible usar aprendizaje previo que pueda acelerar al proceso de aprendizaje. El enfoque (b) construye reglas a partir de los datos de entrenamiento, y no permite la inicialización con reglas.

La tabla 10 resume las consideraciones anteriores y muestra bajo qué criterio nuestros

modelos generales de combinaciones neuro-difusas son aplicables.

2.4. Identificación de sistemas dinámicos via neuro difusos

Un sistema dinámico puede ser descrito por dos tipos de modelos: modelo de entrada-salida y modelo estado-espacio. En este capítulo describimos el uso de NNs difusas para aprender a actuar como ambos tipos de modelos.

Modelo de Entrada-Salida

Un modelo de entrada-salida describe un sistema dinámico basado en datos de entrada salida. En el dominio discreto, un modelo de entrada-salida puede ser del tipo NARMAX [9]. Un modelo de entrada-salida asume que la salida nueva del sistema puede ser predecida por las entradas y salidas pasadas del sistema. Si un sistema es además supuesto a ser determinístico, invariante en el tiempo, solo-entrada-solo-salida (SISO), el modelo se convierte en

$$y_p(k) = f\left(y_p(k+1), y_p(k-2), \ldots, y_p(k-n), u\left((k-1), u(k-2), \ldots, u(k-m)\right)\right) \quad (2.2)$$

donde $[u(k), y_p(k)]$ representa la pareja de entrada-salida del sistema en el tiempo k. Los números enteros positivos n y m son respectivamente el número de salidas pasadas (también llamada el orden del sistema) y el número de entradas pasadas. En la práctica m es normalmente más pequeño o igual a n. La función f puede ser una función estática que mapea las entradas y salidas pasadas a una nueva salida.

Si el sistema es lineal, f es una función lineal y la ecuación (2.2) puede ser reescrita como

$$y_p(k) = a_1 y_p(k+1) + a_2 y_p(k-2), \ldots, + a_n y_p(k-n) + b_1 u(k-1) + b_2 u(k-2), \ldots, + b_m u(k-m) \quad (2.3)$$

donde $a_i\,(i = 1, 2, \ldots, n)$ y $b_i\,(i = 1, 1, \ldots, m)$ son constantes reales.

Modelo de Estado-Espacio

Un sistema dinámico puede ser descrito también por un modelo de estado-espacio. El modelo de estado-espacio de un sistema dinámico de orden n, no-lineal, invariante en el tiempo, multi-entrada-multi-salida (MIMO) es como sigue

$$\begin{aligned} x(k+1) &= \Phi[x(k), u(k)] \\ y(k) &= \Psi[x(k)] \end{aligned} \tag{2.4}$$

donde $x(k) = [x_1(k), x_2(k), \ldots, x_n(k)]^T$ es el vector de estados de n componentes del sistema, $u(k) = [u_1(k), u_2(k), \ldots, u_r(k)]^T$ es el vector de entrada del sistema, y $y(k) = [y_1(k), y_2(k), \ldots, y_m(k)]^T$ es el vector de salida del sistema. Φ y Ψ son mapeos no-lineales. Si el sistema es lineal, la ecuación (2.4) se convierte en

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) \\ y(k) &= Cx(k) \end{aligned} \tag{2.5}$$

donde A,B y C son matrices de (n x n),(n x r) y (m x n).

2.4.1. Identificación basada en entradas y salidas del sistema

La tarea de la identificación de sistemas es esencialmente encontrar mapeos convenientes que puedan aproximar los mapeos implicados en un sistema dinámico.

La ecuación (2.2) puede ser representada por el diagrama a bloques mostrado en la figura (2.11). Puede verse de la figura (2.11) que cuando se utilizan datos de entrada-salida, el sistema dinámico está definido por la función f y los enteros m y n. Si m y n son dados, la única tarea es encontrar f. Cabe mencionar que f no cambia con el tiempo para sistemas invariantes en el tiempo.

De acuerdo a la configuración de un sistema de identificación basado en redes neuronales, pueden haber dos estructuras de identificación: la estructura en paralelo (ver figura 2.12), y la estructura serie-paralelo (ver figuara 2.13). Estas pueden ser extendidas para generar un sistema de identificación basado en redes neuronales difusas.

En la estructura en paralelo, la red y el sistema reciben las mismas entradas externas; las salidas del sistema no son utilizadas como entradas a la red. El sistema y la red son

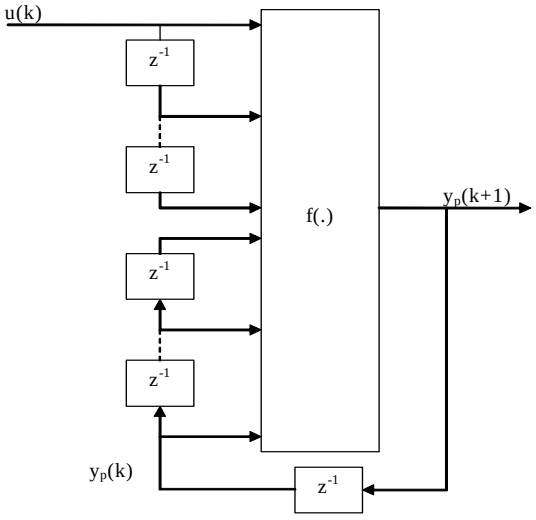

Figura 2.11: Modelo de Entrada-Salida

dos procesos independientes que comparten las mismas entradas externas. Sus salidas no interfieren la una con la otra.

En la estructura serie-paralelo, nuevamente la red y el sistema reciben las mismas entradas externas, pero la salida del sistema es parte de las entradas a la red. El sistema y la red ya no son dos procesos independientes. El comportamiento dinámico de la red es afectado por el sistema.

Cuando las estructuras de las figuras (2.12) y (2.13) son empleadas para identificación, el sistema es considerado a ser de entrada-limitada-salida-limitada (BIBO) y estable en la presencia de una entrada. Sin embargo, si la estructura paralela es empleada, no puede ser garantizado que el proceso de aprendizaje de los pesos convergerá o el error entre la salida del sistema y la de la red tenderá a cero.

La red tanto en la estructura en paralelo como en la estructura serie-paralelo requiere (n+m) unidades de entrada y una unidad de salida. Ambas estructuras son analizadas en [54] utilizando redes neuronales y se hace una diferencia entre estas estructuras:

1. El modelo Serie-Paralelo: En este caso las salidas medidas son utilizadas como los argu-

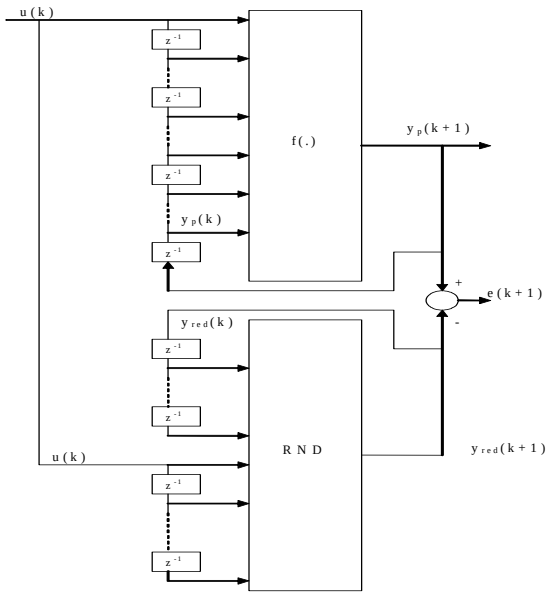

Figura 2.12: Estructura de Identificación Paralela. RND= Red Neuronal Difusa.

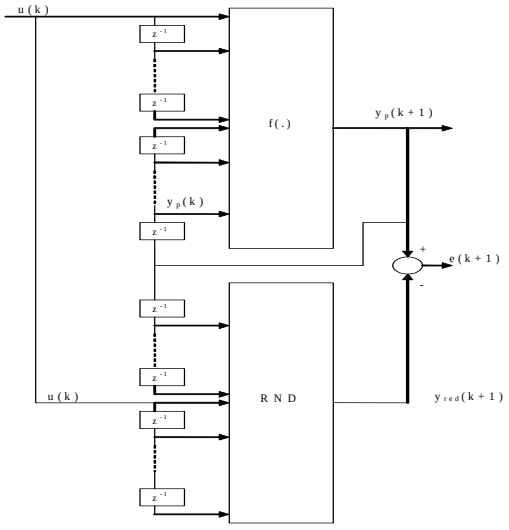

Figura 2.13: Estructura de Identificación Serie-Paralelo. RND= Red Neuronal Difusa.

mentos de entrada al modelo de la red neuronal, por ejemplo en el caso deterministico:

$$y_{net}(k) = f\left(y_p(k+1), y_p(k-2), \ldots, y_p(k-n), u\left((k-1), u(k-2), \ldots, u(k-n)\right)\right) \tag{2.6}$$

donde $y_{net}(k)$ es la salida estimada. Debido a que no hay repetición (recurrence) en la ecuación (2.6), esto corresponde a una *red neuronal feedforward*. Uno tiene un mapeo no-lineal estático del espacio de entrada al espacio de salida de la red neuronal

2. Modelo Paralelo: Aquí las salidas estimadas y_{net} son utilizadas como argumento de entrada al modelo de la red neuronal.

$$y_{net}(k) = f\left(y_{net}(k+1), y_{net}(k-2), \ldots, y_{net}(k-n), u\left((k-1), u(k-2), \ldots, u(k-n)\right)\right) \tag{2.7}$$

Debido a la repetición (recurrence) de y_{net} en la ecuación (2.5), esto corresponde a una *red neuronal recurrente*.

2.4.2. Identificación basada en estados medibles del sistema

En la ecuación (2.4), $\Phi(.)$ y $\Psi(.)$ son funciones arbitrarias que pueden o no ser lineales. Estos son los mapeos para aproximar la red neuronal difusa. La función $\Phi(.)$ es un mapeo que mapea el estado $x(k)$ y la entrada $u(k)$ dentro del nuevo estado $x(k+1)$. El otro es un mapeo $\Psi(.)$ que transforma el estado $x(k)$ dentro de la salida $y(k)$. Cuando la red neuronal difusa es utilizada para identificar al sistema descrito en la ecuación (2.4), se hacen dos suposiciones: (i) todos los estados del sistema son medibles; (ii) el sistema es estable.

Basado en las suposiciones anteriores, la arquitectura dibujada en la figura (2.14) puede ser utilizada para la identificación del sistema. La figura (2.14) muestra que son necesarias dos redes. La red 1 tiene a las entradas del sistema y a los estados como sus señales de entrada. Así, la capa de almacenamiento de la red 1 (input buffer layer) tiene (n+r) unidades. Sin embargo, ya que la red 1 tiene los estados nuevos predecidos del sistema como sus señales

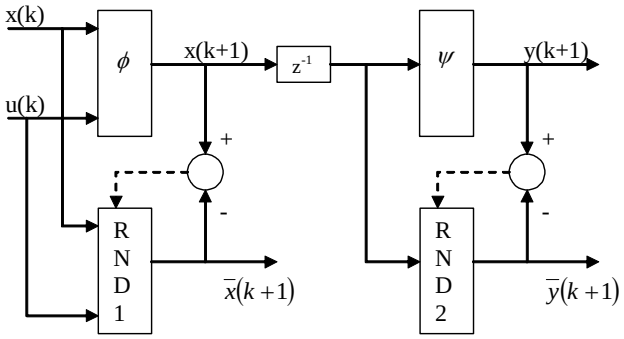

Figura 2.14: Identificación basada en estados medibles del sistema.

de salida, su capa de salida necesita n unidades. El número de capas ocultas y unidades ocultas puede decidirse de acuerdo a la exactitud requerida. Similarmente, la red 2 necesita n unidades de entrada y m unidades de salida.

Capítulo 3

Identificación de la estructura con Agrupamiento en línea

Las técnicas cualitativas de modelado de sistemas tradicionales tienen limitaciones significantes. En la mayoría de los casos, es bastante dificil describir adecuadamente el comportamiento de un sistema no-lineal por modelos matemáticos, especialmente cuando la estructura del sistema no es conocida. Incluso si uno conoce la estructura, la representación del modelo numérico normalmente llega a ser irrelevante y computacionalmente ineficiente cuando la complejidad crece. Después de todo, hay mucha incertidumbre,dinámica imprevisible, interacciones mutuas, y otros fenómenos no conocidos que no pueden ser matemáticamente modelados en todo. Los sistemas de difusos se desarrollaron en un intento por obtener más flexibilidad y más capacidad efectiva de manejo y procesamiento de incertidumbres en sistemas complicados y mal definidos. Los FSs establecen un planteamiento lingüístico para modelar sistemas, el cual puede ser formulado entres características distinguidas:

1. El uso de variables lingüísticas en lugar de variables numéricas.

2. La caracterización de relaciones simples entre variables con reglas difusas IF-THEN.

3. La formulación de relaciones complejas por algoritmos de razonamiento difuso.

Por lo tanto, la característica central del FS es que está basado en conceptos de particionamiento difuso de la información. La habilidad de fabricación de decisiones del modelo difuso depende de la existencia de una base de reglas y un mecanismo de inferencia difusa.

Dependiendo de cómo es representada la información del sistema a modelar dentro de un sistema de inferencia difuso (SIF), podemos distinguir dos técnicas de modelado:

1. En la primer técnica, las reglas difusas son expresiones de la forma

$$\mathrm{R}^l\text{: IF } x_1 \text{ is } A_1^l \text{and } x_2 \text{ is } A_2^l \text{ and } \cdots \ x_n \text{ is } A_n^l \text{ THEN } y_1 \text{ is } B_1^l \text{ and} \cdots y_m \text{ is } B_m^l \quad (3.1)$$

   en donde tanto la parte antecedente como la parte consecuente están formadas por etiquetas lingüísticas (A y B) de conjuntos difusos, las cuales son caracterizadas por funciones de membresía apropiadas. En esta técnica, el modelo difuso es esencialmente una expresión cualitativa del sistema que utiliza expresiones de lenguaje natural. Este tipo de modelo difuso ha sido investigado por muchos investigadores [73], en [80] se discuten dos de los principales representantes de esta técnica de modelado difuso: el modelo tipo Mandani y el modelo tipo Tsukamoto

2. En la segunda técnica, las reglas difusas son expresiones de la forma

$$IF\ x \text{ es } A_1 \text{ and } y \text{ es } B_1 \text{ THEN } f_1 = p_1 x + q_1 y + r_1 \quad (3.2)$$

   que tienen una parte antecedente difusa y una parte consecuente funcional. Donde f normalmente es una función polinomial que permite que la salida de la regla sea una combinación lineal de las variables de entrada más un termino constante. En esencia estos modelos son una combinación de modelos difusos y no-difusos. Esta técnica fue inicialmente propuesta por Takagi-Sugeno-Kang y es referida como modelo difuso TSK. El modelo difuso TSK tiene efectivamente el potencial para expresar información cualitativa y es computacionalmente eficiente. Sin embargo, estamos involucrados con un marco más general de modelado cualitativo, por lo tanto, es preferible adoptar el tipo más general de reglas difusas IF-THEN, que consiste de variables difusas tanto en su antecedente como en su consecuencia.

En la literatura actual podemos encontrar dos planteamientos básicos bien definidos para la metodología del modelado difuso. En el primer planteamiento se asume que se cuenta con la ayuda de un humano experto, el cual posee todo el conocimiento necesario del sistema a modelar. La información a priori sobre el sistema es utilizada entonces para construir de manera subjetiva la descripción linguistica del sistema. Sin embargo este planteamiento directo para construir modelos difusos tiene algunas limitaciones y su principal desventaja es la subjetividad y la falta de generalización y dependencia del conocimiento del experto que algunas veces puede faltar. El segundo planteamiento, fue inspirado por la teoría clásica de sistemas y está basado en el uso de datos de entrada-salida para construir el modelo difuso. Este es un planteamiento más objetivo y formal ya que utiliza datos disponibles sobre el sistema que pueden aumentar el conocimiento del humano o incluso generar nuevo conocimiento. En el lenguaje de la teoría de sistemas, este planteamiento puede ser considerado como un proceso de identificación de sistemas.

En el procedimiento para modelar sistemas utilizando FSs (ver figura 3.1), existen diferentes temas principales a tratar sobre el mecanismo de razonamiento, por ejemplo, cómo la información está implícita en cada regla, cómo se infiere nueva información del conjunto de reglas y cómo los valores inferidos son transformados en datos numéricos. Otro tema importante a tratar relacionado con el planteamiento objetivo del modelado difuso es el problema de identificación del FS. De manera similar a la identificación de sistemas convencional, el problema puede ser dividido en dos fases como se muestra en la figura (3.1): La *identificación de la estructura* implica encontrar las variables de entrada importantes de todas las posibles variables de entrada, especificando las funciones de membresía, la partición del espacio de entrada, y determinando el numero de reglas difusas. La *estimación de parámetros* implica la determinación de parámetros no conocidos en el modelo utilizando algún método de optimización basado tanto en la información lingüística obtenida del humano experto como de los datos numéricos obtenidos del sistema actual a ser modelado. La especificación de la estructura y la estimación de parametros están interelacionados, y ninguno de ellos puede ser independientemente identificado sin considerar el otro. En las secciones siguientes se discuten las dos fases para la identificación de un modelo difuso.

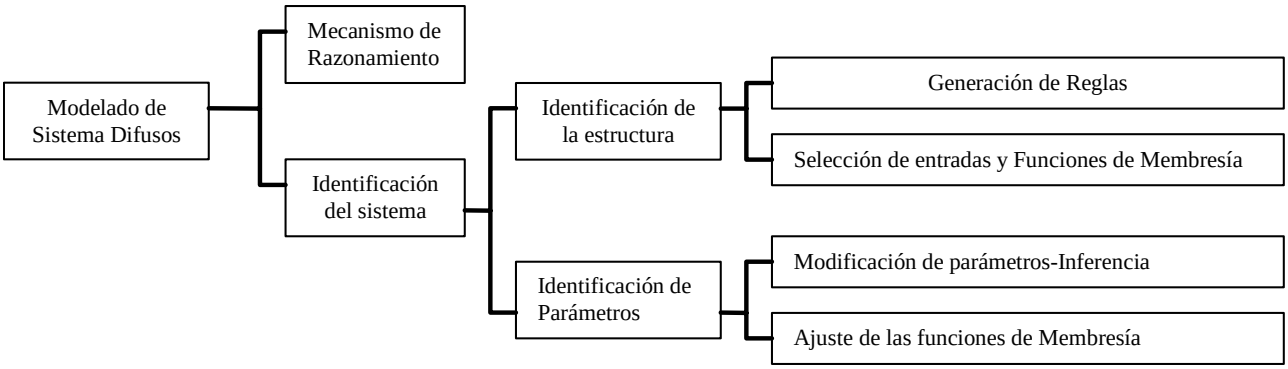

Figura 3.1: Diagrama de flujo del modelado de Sistemas Difusos.

La identificación de la estructura de un FS es posible por construir bastantes reglas con las funciones de membresía de entrada y salida apropiadas. Esta meta puede ser dividida dentro de dos fases separadas.

1. Generación de reglas

2. Formación de las funciones de membresía.

Cuando los datos de entrada-salida están disponibles a priori, el agrupamiento difuso es una técnica que puede ser utilizada para la identificación de la estructura. En [73] se propone un planteamiento intuitivo para la generación de reglas difusas IF-THEN,el cual está basado en el agrupamiento de datos de entrada salida; ésta es una idea simple y aplicable especialmente para sistemas con un gran número de variables de entrada. En [18] se considera que la esencia del método para la identificación de la estructura está en el agrupamiento y la proyección. En ambos planteamientos, el espacio de salida es particionado utilizando un algoritmo de agrupamiento difuso (FCM), partición que puede ser siempre considerada como un espacio de una dimensión en el modelo difuso. En segundo lugar, las particiones (grupos) son proyectados dentro del espacio de cada una de las variables de entrada. La partición de salida y su correspondiente partición de entrada son las consecuencias y antecedentes de las regla IF-THEN respectivamente. Utilizando este método, el paso de generación de reglas puede ser separado del paso de selección de entradas, como se discute a continuación.

3.1. Agrupamiento off-line

Para obtener un conjunto de N reglas difusas condicionales capaces de representar el sistema bajo estudio, los algoritmos de agrupamiento son especialmente buenos, ya que permiten un particionamiento dispersado del espacio de datos de entrada-salida, cuyo resultado es encontrar sólo las reglas relevantes. Comparando con los métodos de particionamiento basados en rejilla, los algoritmos de agrupamiento tienen la ventaja de evitar la explosión de la base de reglas, un problema conocido como la "maldición de dimensionalidad". Algunos investigadores utilizan métodos de particionamiento basados en rejilla sin embargo en se concluye que los resultados nos son tan buenos como los obtenidos de técnicas de agrupamiento, por diferentes razones.

Desde un punto de vista general y conceptual, agrupar significa particionar una colección de datos en grupos o subconjuntos disjuntos, con el dato en una agrupación teniendo varias propiedades que los distingan de los datos de otros grupos. El propósito del agrupamiento es extraer agrupamientos naturales de un conjunto de datos, produciendo una representación concisa del comportamiento de un sistema. Los algoritmos de agrupamiento han sido utilizados extensamente para organizar, clasificar y comprimir datos y para construir modelos. Los algoritmos de agrupamiento pueden ser divididos en dos clases: batch y en línea. Los algoritmos batch procesan los datos fuera de linea, por lo tanto, la estructura temporal de los mismos es ignorada. Los algoritmos en línea por su parte asumen que los datos son producidos por un proceso estacionario por lo que estos no se mueven ni varían. En esta situación los datos pueden ser muestreados y agrupados con un algoritmo batch. Jang [28] describe cuatro de las técnicas de agrupamiento off-line más representativas utilizadas frecuentemente en el modelado difuso: Agrupamiento C-means (o K-means), agrupamiento difuso C-means (FCM), el método de agrupamiento de montaña (mountain clustering method), y agrupamiento substractivo (subtractive clustering).

La idea del agrupamiento difuso (FCM) es dividir los datos de salida dentro de particiones difusas que se traslapan unas con otras. Por lo tanto, la contención de cada dato para cada grupo es definida por un grado de membresía entre $[0, 1]$.

El FCM particiona una colección de n vectores x_i, $i = 1, \ldots, n$ dentro de c grupos difusos, y encuentra un centro de grupo para cada grupo de manera tal que una función de costo es minimizada. El FCM emplea particiones difusas de manera tal que un punto dado puede pertenecer a varios grupos con un grado de pertenencia especificado por grados de membresía entre 0 y 1.En otras palabras, la matriz de membresía U esta permitida para tener elementos con valores entre 0 y 1. Sin embargo, imponente normalización estipula que la sumatoria de grados de pertenencia para un conjunto de datos siempre es igual a la unidad:

$$\sum_{i=1}^{c} u_{ij} = 1, \ \forall \ \ j = 1, \ldots, n. \tag{3.3}$$

La función de costo para FCM es:

$$J\left(U, c_1, \ldots, c_c\right) = \sum_{i=1}^{c} J_i = \sum_{i=1}^{c} \sum_{j}^{n} u_{ij}^m d_{ij}^2 \tag{3.4}$$

donde u_{ij} esta entre 0 y 1; c_i es el centro de grupo del grupo difuso i; $d_{ij} = \|c_i - x_j\|$ es la distancia euclidiana entre el $i - ésimo$ centro de grupo y el $j - ésimo$ punto de dato; y $m \in [1, \infty)$ es un exponente de peso.

La condición necesaria para que la ecuación (3.4) alcance un mínimo puede ser encontrada por realizar una nueva función de coste $\overline{J}$ como sigue:

$$\begin{aligned} \overline{J}\left(U, c_1, \ldots, c_n, \lambda_1, \ldots, \lambda_n\right) &= J\left(U, c_1, \ldots, c_c\right) + \sum_{j=1}^{n} \lambda_j \left(\sum_{i=1}^{c} u_{ij} - 1 \right) \\ &= \sum_{i=1}^{c} \sum_{j}^{n} u_{ij}^m d_{ij}^2 + \sum_{j=1}^{n} \lambda_j \left(\sum_{i=1}^{c} u_{ij} - 1 \right) \end{aligned} \tag{3.5}$$

donde λ_j, $j = 1$ a n, son los multiplicadores de Lagrange para las n restricciones en la ecuación (3.3).Para diferenciar $\overline{J}\left(U, c_1, \ldots, c_n, \lambda_1, \ldots, \lambda_n\right)$ con respecto a todos sus argumentos de entrada, la condición necesaria para que la ecuación (3.4) alcance su mínimo es:

$$c_i = \frac{\sum_{j=1}^{n} u_{ij}^m x_j}{\sum_{j=1}^{n} u_{ij}^m} \tag{3.6}$$

y

$$u_{ij} = \frac{1}{\sum_{k=1}^{c} \left(\frac{d_{ij}}{d_{kj}}\right)^{\frac{2}{(m-1)}}} \tag{3.7}$$

El algoritmo FCM es simplemente un procedimiento interactivo a través de las dos condiciones necesarias anteriores.

El algoritmo FCM determina los centros de grupo c_i y la matriz de membresía utilizando los siguientes pasos:

Paso 1: Inicializar la matriz de membresía U con valores aleatorios entre 0 y 1 de manera tal que la restricción de la ecuación (3.3) sea satisfecha.

Paso 2: Calcular c centros de grupos difusos c_i, $i = 1, \ldots, c$, utilizando la ecuación (3.6).

Paso 3: Calcular la función de costo con la ecuación (3.4). Parando si:

- La función de costo está por debajo de un cierto valor de tolerancia o si la mejora sobre iteraciones previas está por debajo de un cierto umbral.
- Si un número máximo de iteraciones es alcanzado.

Paso 4: Calcular una nueva U utilizando la ecuación (3.7).

Los centros de grupo pueden también ser primero inicializados y entonces llevar acabo el procedimiento iterativo. No se asegura que el algoritmo FCM converja a una solución óptima. El rendimiento depende de los centros de grupo iniciales, por lo tanto podríamos utilizar otros algoritmos más rápidos para determinar los centros de grupo iniciales, tales como: El algoritmo de Agrupamiento de Montaña propuesto por Yager y Filev [86] o El algoritmo de Agrupamiento Substractivo propuesto por Chiu [12]; o correr el algoritmo FCM varias veces, y en cada vez iniciando con un conjunto diferente de centros de grupo iniciales.

El algoritmo de agrupamiento difuso es:

1. Un algoritmo de tipo supervisado ya que se necesita indicarle cuántos grupos tiene que formar, lo que representa su principal desventaja.

2. Un algoritmo que hasta la fecha no cuenta con una base teórica para elegir de manera óptima el exponente de pesos (m).

3. Un algoritmo que requiere la localización inicial de los centros de grupo y éste conocimiento no es necesariamente disponible a priori.

La identificación de estos problemas así como el planteamiento de las posibles soluciones son tratados en [18].

3.2. Agrupamiento en línea: substractivo doble en línea

Los algoritmos de agrupamiento en línea como se mencionó anteriormente, asumen que los datos son producidos por un proceso estacionario, por lo que no se mueven ni varían. En esta situación los datos pueden ser muestreados y agrupados con un algoritmo batch. Sin embargo, existen muchos problemas del mundo real en los cuales los datos son producidos por un cierto tipo de proceso no-estacionario, en donde el problema se convierte en cómo reconocer que un nuevo proceso comenzó (llegada de un dato de un grupo nuevo) y asignar un centro de grupo para representarlo. En las secciones siguientes se presentan dos planteamiento diferente para tratar el agrupamiento en línea de datos producidos por procesos no-estacionarios.

En el contexto de técnicas de identificación neuro difusas, el esquema tanto del agrupamiento difuso como de otros algoritmos de agrupamiento basados en optimización, pueden conducir a trabajo de cómputo excesivo ya que realizan una fase de optimización innecesaria previa al entrenamiento de la red. También, el agrupamiento progresivo [14] y los algoritmos de mezcla de agrupamientos [42] compatibles son computacionalmente costosos y necesitan medidas para validación de grupos individuales. Por lo tanto, a pesar de su potencial, todo estos algoritmos son demasiado complejos para una simple inicialización de una red neuronal difusa. En el mismo contexto el algoritmo de agrupamiento substractivo (subtractive clustering) posee algunas ventajas interesantes ya que es un algoritmo eficiente que no requiere

optimización, por esta razón es una buena elección para la inicialización de las redes neuro difusas.

El agrupamiento substractivo es esencialmente una forma modificada del método de montaña y está basado en una medida de densidad de cada dato en el espacio característico. La idea es encontrar regiones en el espacio característico con altas densidades de datos. El punto con el mayor número de vecinos es seleccionado como centro para un grupo. Los datos dentro de un radio difuso preespecificado son entonces removidos (substraidos), y el algoritmo busca un nuevo dato con el mayor número de vecinos. Esto continua hasta que todos los datos son examinados.

Considere una colección de Z datos$\{x_1, \ldots, x_z\}$ en un espacio m-dimensional. Suponiendo que todos los puntos están normalizados en cada dimensión, el conjunto de datos Z está limitado por un hypercubo. En el algoritmo, cada punto es visto como un centro de grupo potencial y una medida de densidad en el punto x_i es definida como

$$D_i = \sum_{j=1}^{N} \exp^{-\alpha \|z_i - z_j\|^2} \tag{3.8}$$

donde $\alpha = 4/r_a^2$ y $r_a > 0$ define el radio de vecindad para cada centro de grupo. Por lo tanto, la densidad asociada con cada centro de grupo depende de su distancia con respecto a todos los demás puntos, destacando a grupos con densidades altas donde la vecindad es densa.

Después de calcular la densidad para cada punto, el dato con la densidad más alta es seleccionado como el primer centro de grupo. Sea z_1^* el centro del primer grupo y D_1^* su densidad. Entonces, la densidad para cada punto z_i^* es reducido, especialmente para los puntos cercanos a el centro de grupo, por:

$$D_i = D_i - D_1^* \exp^{-\beta \left\| z_i - z_1^* \right\|^2} \tag{3.9}$$

donde $\beta = 4/r_b^2$ y $r_b > 0$ representa el radio de la vecindad para la cual la reducción de densidad puede ocurrir. El radio para la reducción de densidad puede ser algo más extendido que el radio de vecindad para evitar grupos estrechamente espaciados. Típicamente, $r_b =$

1,5r_a. Ya que los puntos que están cerca al centro de grupo pueden tener su densidad reducida de manera fuerte, la probabilidad de que uno de estos puntos sea elegido como el siguiente grupo es baja. Este procedimiento (selección de centros y reducción de densidad) es realizado iterativamente, hasta que el criterio de paro es alcanzado. Este procedimiento es ilustrado en (3.2).

Donde, ε^{up} especifica un umbral sobre el cual el punto es seleccionado como un centro sin ninguna duda, y ε^{down} especifica el umbral abajo el cual el punto es definitivamente rechazado. El tercer caso es cuando el punto es caracterizado por un buen equilibrio entre tener una densidad alta suficiente y estar bastante lejos de los grupos determinados antes. Típicamente, $\varepsilon^{up} = 0{,}5$ y 0,15.

Una propiedad del algoritmo es que el número de grupos a obtener no es pre-especificado. Sin embargo, es importante notar que el parámetro $radii = r_a$ está directamente relacionado con el número de grupos (centros) encontrados. De esta manera, un radio pequeño puede conducir a un número alto de reglas que, si es excesivo puede resultar un probrema de overfitting . Por otro lado, un radio grande puede conducir a un número pequeño de grupos, que puede originar un problema de underfitting, y también, modelos con reducida representación de exactitud. Por lo tanto, en la práctica es necesario probar varios valores para $radii$ y seleccionar el más adecuado de acuerdo a los resultados obtenidos. Sin embargo, a pesar del hecho de que algunos $radii$ pueden ser probados, este parametro da una pista inicial en el número de grupos necesarios [64].Esto constituye una importante ventaja sobre los algoritmos de agrupamiento basados en optimización, cuando poca información es conocida respecto al mejor numero de grupos. Otra ventaja del agrupamiento substractivo es que el algoritmo es robusto al ruido, ya que la salida (outliers) no están influenciando significativamente la elección de los centros, debido a su baja densidad.

Al final del agrupamiento, un conjunto de g reglas del tipo de (3.1) o (3.2) puede ser obtenido. Cada grupo representa una regla. Sin embargo ya que el procedimiento del agrupamiento está conducido en un espacio multidimencional, los conjuntos difusos deben ser obtenidos antes de establecer las reglas, es decir, a estas alturas el algoritmo de agrupamiento sólo nos proporciona el número de reglas (numero de grupos) que necesitamos para poder

caracterizar al sistema a modelar.

El algoritmo de agrupamiento substractivo doble puede resumirse en los siguientes pasos:

1. Establecer el periodo de tiempo t.
2. Realizar el muestreo y almacenamiento de los datos en cada tiempo k, $[x(k), y(k)]$. Se recomienda que $N \geq 10t$, donde N es igual al número de datos almacenados durante el periodo t.
3. Realizar una primer etapa de agrupamiento utilizando el algoritmo de agrupamiento substractivo con $radii \leq 0{,}2$.
4. Establecer los centros de grupos obtenidos en el paso 3 como los nuevos datos a agrupar.
5. Realizar una segunda etapa de agrupamiento utilizando el algoritmo de agrupamiento substractivo con $radii \geq 0{,}4$.
6. Registrar los centros de grupo finales y resetear el valor de k. Ir a 2.

3.3. Extracción de funciones de membresía en línea

La técnica utilizada para extraer las funciones de membresía depende grandemente del tipo de algoritmo de agrupamiento utilizado. Por ejemplo, cuando se aplica el algoritmo de agrupamiento difuso FCM a un conjunto de datos, el resultado final es la así llamadas matriz de membresía difusa. En la figura (3.3) se muestra un conjunto de datos de seis puntos y los valores de membresía para el primer renglón de las particiones FCM.

Los grupos generados, son conjuntos difusos multidimencionales (discretos) que se traslapan. Cada agrupamiento difuso puede ser transformado en una regla difusa, por proyectar los grados de membresía de los datos de entrenamiento a su dimensión única correspondiente.

Ejemplo 3.1 *En [6] consideran el conjunto de datos* $Z = \{(x_1, y_1), \ldots (x_{13}, y_{13})\}$ *(puntos pequeños en la figura 3.4) que representa un conjunto de entradas escalar* $(p = 1)$ *y salida*

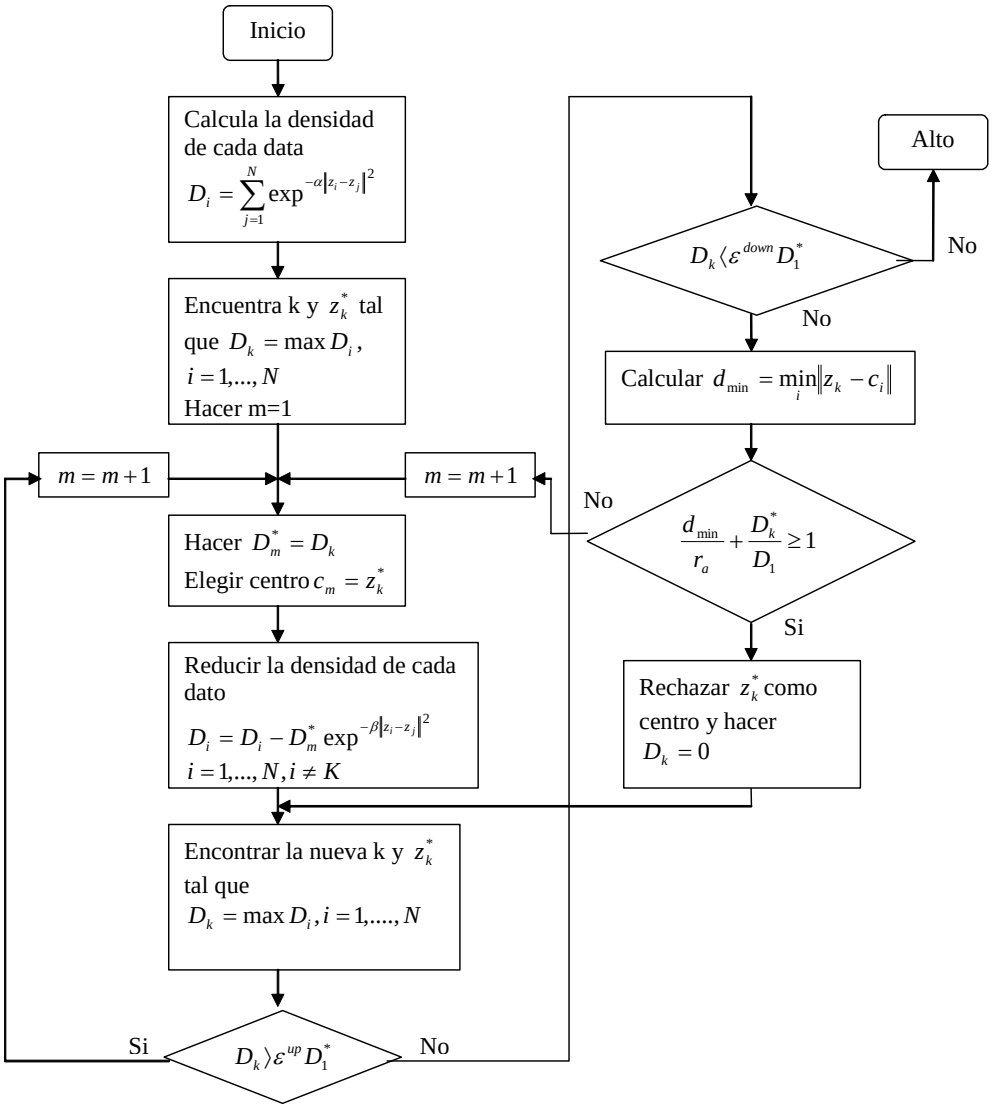

Figura 3.2: Carta de flujo del algoritmo de agrupamiento substractivo.

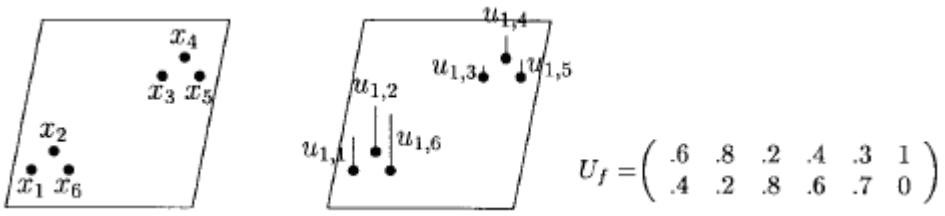

Figura 3.3: Un conjunto de datos y un ejemplo de la partición FCM para los conjuntos

escalar $(q = 1)$ *.Construyen las funciones de membresía en la figura (3.4) con el algoritmo FCM en el cual* $m = 2$ *y* $c = 2$*. La figura (3.4) muestra los centros de grupo (puntos grandes) y la membresía* $U_1 = \{u_{11}, \ldots, u_{113}\}$ *(líneas verticales) obtenidas después de 100 iteraciones. Los puntos* (x_k, y_k, u_{1k}) *están en la superficie mostrados como la rejilla punteada.*

La proyección de estos puntos a los ejes x *e* y *son mostrados al frente y a la izquierda. Estos puntos (muestras) son interpretados como funciones de membresía no conocidas* $\mu_{11}, \mu_{12} : \mathbb{R} \to [0, 1]$ *(curvas punteadas)*

Es decir, las muestras para estas funciones de membresía pueden ser obtenidas por proyección de $\{(x_{11}, \cdots, x_{p1}, y_{11}, \cdots, y_{q1}, u_{i1}), \cdots, (x_{1n}, \cdots, x_{pn}, y_{1n}, \cdots, y_{qn}, u_{in})\}$ *para cada una de las dimensiones de las* p *entradas y las* q *salidas.*

Así para cada grupo y cada variable se obtiene un histograma que puede ser aproximado ya sea por conectar los grados de membresía por una línea, o -más preferentemente- por una función de membresía parametrizada que debe ser tanto normal como convexa y colocar los grados de membresía proyectados tan bien como sea posible [37]. Este planteamiento puede resultar en formas de funciones de membresía difíciles de interpretar (ver figuras 3.5 y 3.6).

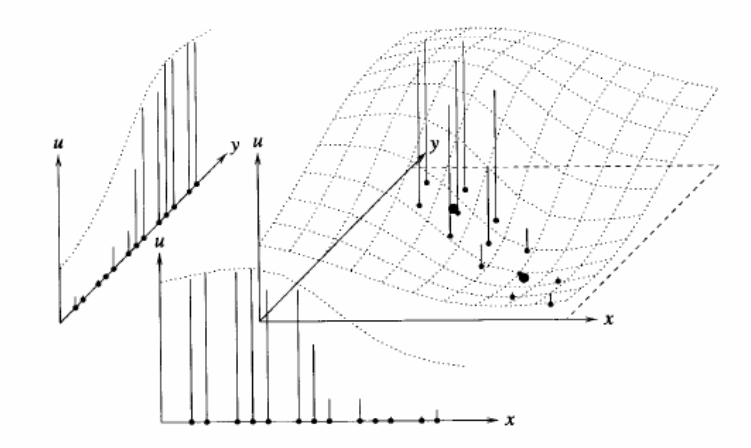

Figura 3.4: Cluster 1 para Z y sus proyecciones.

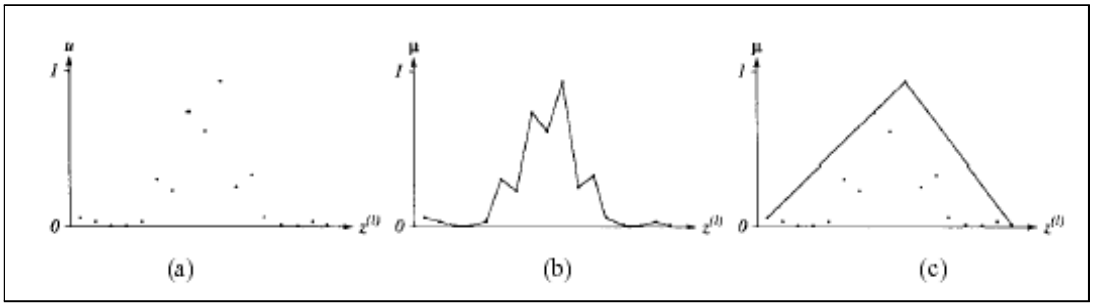

Figura 3.5: Muestras de membresía, Interpolación y Finalización convexa.

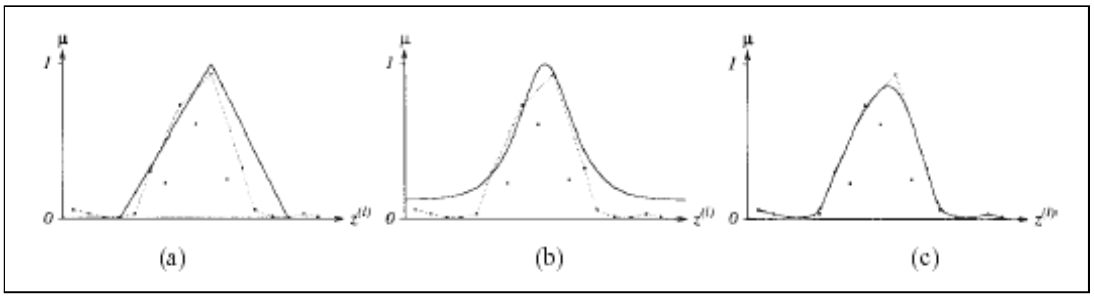

Figura 3.6: Aproximación con triangulos, funciones de Cauchy, y B-splines

Este procedimiento también causa una pérdida de información debido a que el producto cartesiano de las funciones de membresía inducidas no produce un grupo difuso exactamente (ver Figura 3.7). Esta pérdida de información es más fuerte en el caso de hyperelipsoides orientadas arbitrariamente. Para hacer este problema fácil de manejar, es posible buscar solo hyperelipsoides de ejes paralelos [38].

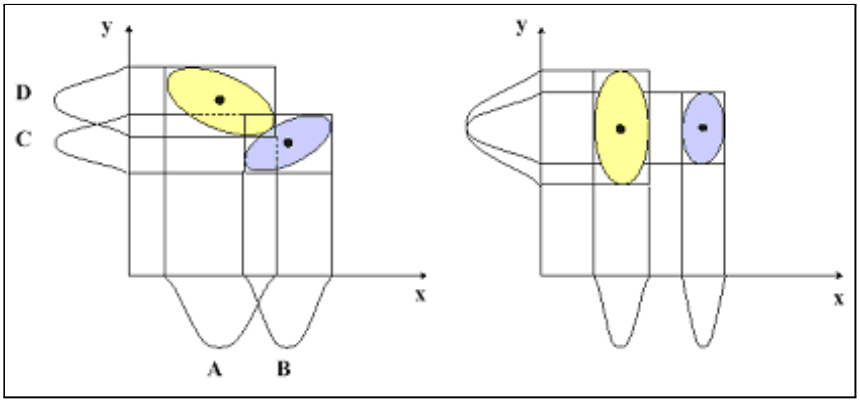

Figura 3.7: Si los grupos en forma de hyperelipsoides son proyectados para obtener reglas difusas, ocurre una perdida de información y pueden obtenerse particiones difusas inusuales.

Normalmente, las reglas difusas obtenidas por proyectar los grupos no son fáciles de interpretar, debido a que los conjuntos difusos son inducidos individualmente para cada regla. Para cada característica podría haber tantos conjuntos difusos diferentes como grupos. Algunos de estos conjuntos difusos pueden ser similares, aunque ellos normalmente no son idénticos. Para una buena interpretación es necesario tener una partición difusa de algunos conjuntos difusos donde cada uno representa claramente un concepto lingüístico.

En [67] se tratan detalladamente las diferentes formas de aproximar los histogramas así como también se trata el problema de la interpretabilidad de las funciones de membresía y la pérdida de información que se genera en la proyección de los grados de membresía.

Por otra parte, cuando se aplica el algoritmo de agrupamiento substractivo a un conjunto de datos, cada uno de los grupos obtenidos puede constituir un prototipo para un comportamiento particular del sistema bajo análisis. Así, cada grupo puede ser utilizado para definir

una regla difusa, capaz de describir el comportamiento del sistema en alguna región del espacio de entrada-salida. Cada uno de los términos lingüísticos de la parte antecedente, es asociado con una función de membresía definida como sigue

$$\mu_A(z_i) = \exp^{-\alpha\left(z_i - z^*_{ri}\right)^2} \tag{3.10}$$

Donde, z_i denota un valor numérico relacionado a la i-ésimma dimensión de entrada y z^*_{ri} es la i-ésima coordenada en el vector m-dimensional z^*_r. La ecuación (3.10) resulta del cómputo de la densidad asociada a cada dato en el espacio de datos. *En otras palabras, como cada eje del espacio multidimensional se refiere a una variable, los centros de las funciones de membresía para esa variable son obtenidos por proyectar los centros de cada grupo en los ejes correspondientes. Mientras que los anchos, son obtenidos en base al radio de vecindad r_a, definido mientras se realiza el agrupamiento substractivo.* Por ejemplo, en la figura 3.9 se muestra el trazo de las FMs correspondientes a los cuatro centros de grupo obtenidos en la figura 3.8; como se puede observar, el centro de cada una de las FMs corresponde a cada uno de los centros de grupo.

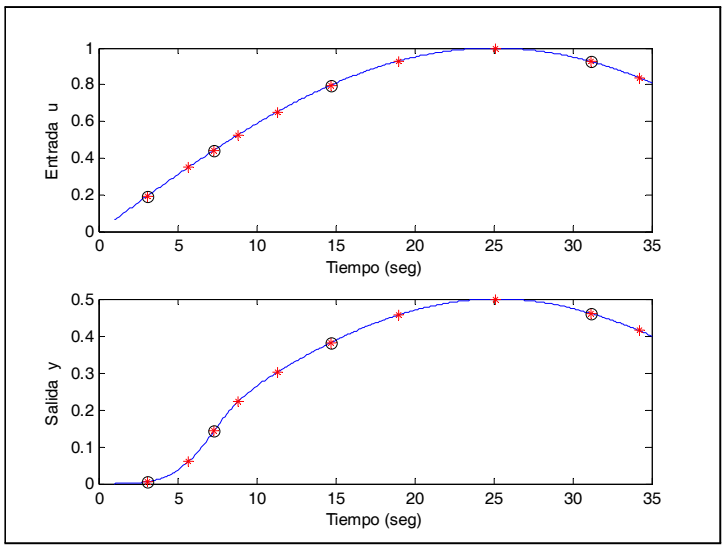

Figura 3.8:

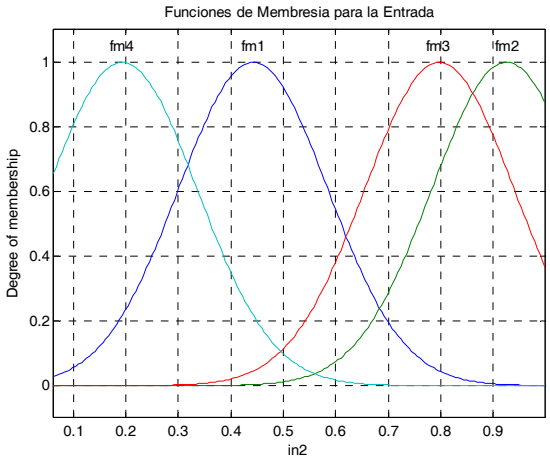

Figura 3.9: Funciones de Membresía para los centros de grupo de la figura 3.8.

Ya que las funciones de membresía tipo Gaussiana son utilizadas, su desviación estándar es calculada por

$$\sigma_{ij} = r_a \frac{\text{máx}\left(z_{kj}\right) - \text{mín}\left(z_{kj}\right)}{\sqrt{8}}, \ k = 1, ..., N. \tag{3.11}$$

Como se puede percibir, la extracción de las funciones de membresía es mucho más sencilla utilizando el agrupamiento substractivo que utilizando el agrupamiento difuso ya que la FM tiene una forma parametrizada de tipo Gaussiana bien definida lo que elimina la pérdida de información que se genera con el agrupamiento difuso cuando se hace la proyección de los valores de membresía de cada uno de los datos. Además, los centros calculados por el algoritmo substractivo son tomados directamente como centros de las FMs antecedentes y la amplitud de dichas FMs es automáticamente generada al elegir el radio de vecindad para los grupos (r_a) . Es por esto que el algoritmo de agrupamiento substractivo se ha convertido en el algoritmo ideal para obtener un modelo difuso inicial del sistema a modelar, el cual puede ser optimizado a través de técnicas neuro difusas.

Por otra parte, una vez obtenidas las FMs para cada uno de los grupos establecido por el método de agrupamiento, se procede a generar las reglas difusas IF-THEN. En la figura (3.10) se muestran tres grupos y sus correspondientes FMs, los cuales pueden ser utilizados

para estimar las reglas siguientes

$$R_1 : IF \;\; \mu_{11}(x) \;\; THEN \;\; v_1(y) = \mu_{12}(y)$$
$$R_2 : IF \;\; \mu_{21}(x) \;\; THEN \;\; v_1(y) = \mu_{22}(y)$$
$$R_3 : IF \;\; \mu_{31}(x) \;\; THEN \;\; v_1(y) = \mu_{32}(y)$$

Podemos asignar las etiquetas lingüísticas (S,M,L) a las FMs como se muestra en la figura y obtener

$$R_1 : IF \;\; x = S \;\; THEN \;\; y = S$$
$$R_2 : IF \;\; x = M \;\; THEN \;\; y = M$$
$$R_3 : IF \; x = L \;\; THEN \;\; y = L$$

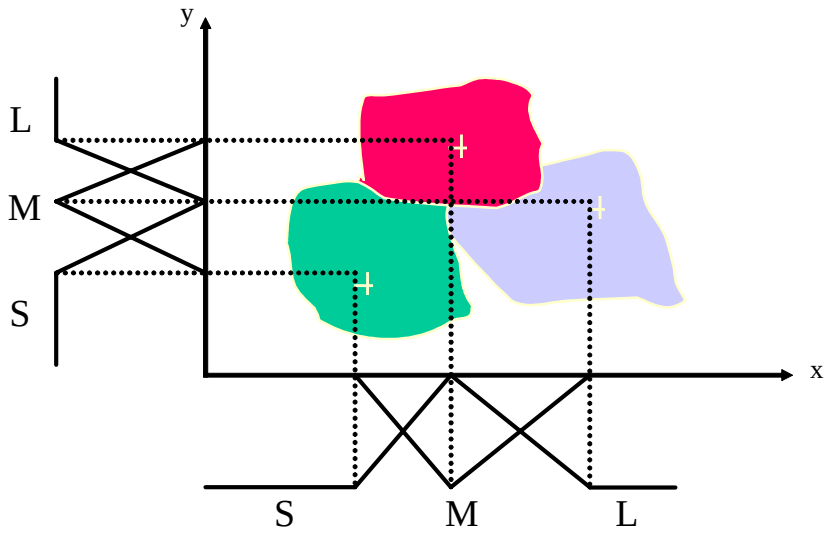

Figura 3.10: Agrupamiento y su relación con las reglas difusas

3.4. Simulación

En este planteamiento tratamos un conjunto de problemas que comparten las siguientes caracteristicas: una escala de tiempo corta es seudo escionaria, mientras que en una escala

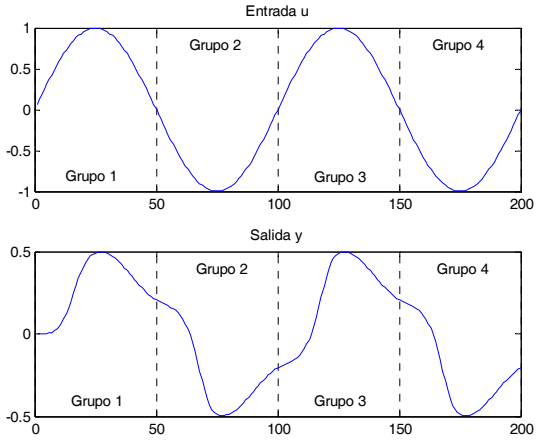

Figura 3.11: Presentación secuencial de los datos de la figura

de tiempo larga el proceso tiene una propiedad secuencial. Consideremos por ejemplo los datos mostrados en las figuras (3.11) y (3.12), en donde los grupos de datos fueron producidos secuencialmente. Los puntos en cada grupo fueron generados en un proceso estacionario. Primero, todos los puntos del grupo 1 (primeros 50 pts) llegaron aleatoriamente. Estos fueron seguidos por la llegada aleatoria de todos los puntos del grupo 2 (siguientes 50 pts). Posteriormente, los datos del grupo 2 fueron seguidos por todos los puntos del grupo 3 (siguientes 50 pts) y así sucesivamente.

En este ejemplo, la escala de tiempo corta es el número de puntos en cada grupo, mientras que la escala de tiempo larga es el proceso completo. Es decir, la escala de tiempo tiene una relación directa con los puntos de los grupos.

El algoritmo propuesto considera que si los datos tiene una propiedad secuencial, entonces se puede seleccionar un periodo de tiempo que fije los datos a ser almacenados y agrupados por un algoritmo de tipo batch como es el caso del algoritmo de agrupamiento substractivo discutido en la sección. Por ejemplo, si seleccionamos un periodo de tiempo $t = 35\ seg$ para los datos secuenciales de la figura 3.11 y aplicamos el agrupamiento substractivo los grupos obtenidos son como los que se muestran en la figura 3.13

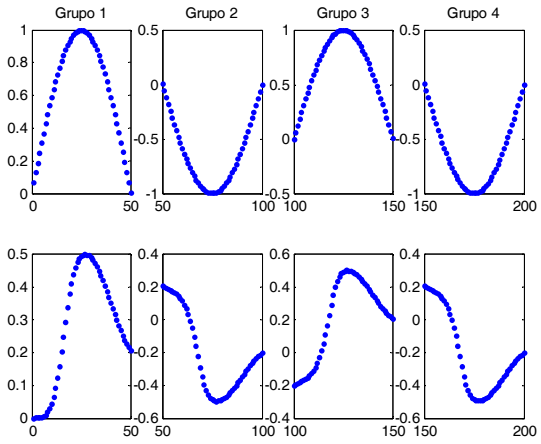

Figura 3.12: Presentacion de los datos

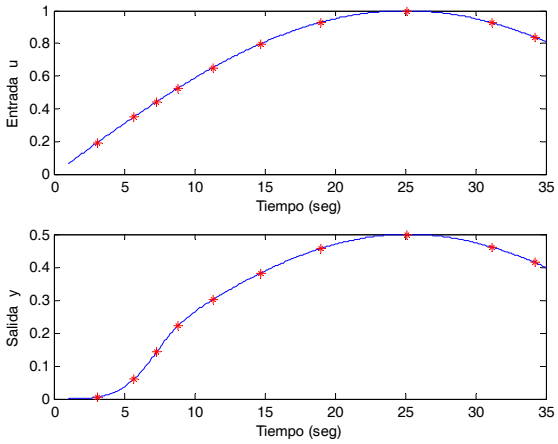

Figura 3.13:

El número de grupos encontrados está directamente relacionado al radio radii. Un radio pequeño conduce a un número grande de grupos pequeños, mientras que un radio grande conduce a un número pequeño de grupos grandes. El algoritmo propone seleccionar una valor del radio lo suficientemente pequeño (radii=0.2) con el fin de obtener la mayor cantidad de

grupos que nos permitan capturar la estructura de los datos.

Las distancias entre los centros de los grupos generados deben ser revisadas, con el fin de combinar los centros mas cercanos. Esto da lugar a la creación de centros redundantes.Para lograr esto,en el algoritmo se propone realizar una segunda etapa de agrupamiento substractivo con un valor de radio mayor al de la etapa anterior (por ejemplo, radii=0.5) ver figura 3.14.

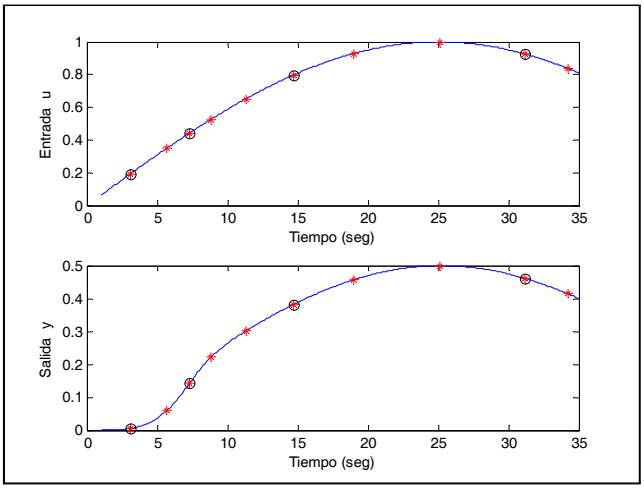

Figura 3.14:

Capítulo 4

Identificación de parametros para redes neuronales difusas dinámica

El aprendizaje de gradiente descendente y el algoritmo de retropropagación son usados siempre para ajustar los parámetros de las funciones de membresía y los pesos de defusificación. La lenta velocidad de aprendizaje y el mínimo local son importantes retrocesos de estos algoritmos de aprendizaje[47]. Algunas modificaciones se derivan. [10] se sugiere un aprendizaje BP robusto para resistir los efectos del ruido y rechazar el amontonamiento de errores durante la aproximación. [83] usando funciones de membresía B-splinese minimiza la función objetivo, la velocidad de convergencia es improvisada. Las redes neuronales RBF se aplicaron en [78] para determinar la estructura y parámetros de los sistemas neuronales difusos.A pesar de sus excelentes aplicaciones, los algoritmos mencionados con anterioridad, tienen un problema durante su entrenamiento, carecen de análisis estables.

Para los ingenieros es muy importante asegurar estabilidad en la teoría antes de aplicar una técnica de modelado neuro-difuso en un sistema real. Es bien conocido que la identificación normal de algoritmos (por ejemplo gradiente descendente o mínimos cuadrados) se estabiliza por condiciones ideales. En presencia de dinámica no modelada, estos procedimientos adaptivos pueden ir facilmente a la estabilidad. La falta de de robustez de la identificación de parámetros se demostró en [16] y se volvió un tema de controversia en los 1980s, cuando

se sugirieron técnicas robustas de modificación para identificacion adaptiva [24].

La actualización de las funciones de membresía del modelado neuro-difuso puede ser considerada como parámetro de identificación; el aprendizaje normal de gradiente descendente y el algoritmo de retropropagación son estables cuando el modelo neuronal difuso puede coincidir exactamente con la planta no lineal. Generalmente tenemos que aplicar algunas modificaciones a estos algoritmos de tal modo que los procesos de aprendizaje sean robustamente estables. El operador proyección es muy efectivo para asegurar los parámetros limitados para el modelado difuso [80] y los sistemas neuro-difusos [44]. Otro método generalizado es usar técnicas de modificación robusta [24] en el modelado neuro difuso. Por ejemplo, [82] aplica un cambio de modificación-σ, para prevenir el aglutinamiento de parámetros. En [81], los autores sugieren una velocidad de aprendizaje estable y óptima sin modificaciones robustas, pero esta velocidad óptima es difícil de encontrar.

En este capítulo se propone una nueva red neuronal difusa recurrente, llamada red neuronal difusa recurrente de estado- espacio (state-space recurrent fuzzy neural networks (SRFNN)), donde la alimentación hacia atrás ocurre a la salida de todo el sistema difuso. Este capítulo proporciona redes neuronales difusas recurrentes de tipo Mamdani [51], la forma final es una ecuación de estado-espacio no lineal, así que es muy conveniente modelar un proceso nolineal que sea expresado en un marco estado-espacio. Cuando las FMs de la parte consecuente (parte THEN) se desconocen, se usa el aprendizaje gradiente descendente. Cuando las FMs de la parte premisa(parte IF) se desconocen se usa el algoritmo de retropropagación. Las velocidades de aprendizaje variantes en el tiempo se obtienen con aproximaciones de estabilidad entrada-a-estado (input-to-state stability (ISS)) para actualizar los parámetros de las funciones de membresía, estas leyes de aprendizaje pueden asegurar estabilidad en el proceso de entrenamiento. Comparado con [77], los algoritmos propuestos en este trabajo son más simples (sin zona muerta) y pueden ser calculados directamente por datos de entrada/salida.

4.1. Redes neuronales difusas recurrentes estado-espacio

Para identificar los siguientes sistemas no lineales de estado-espacio

$$x(k+1) = f\left[x\left(k\right), u\left(k\right)\right] \tag{4.1}$$

donde $u\left(k\right) \in \Re^m$ es el vector de entrada, $x\left(k\right) \in \Re^n$ es un vector de estado, y $y\left(k\right) \in \Re^l$ es el vector de salida, f es generalmente una función suave no lineal $f \in C^\infty$, usamos un modelo difuso que se presenta como una colección de reglas difusas de la siguiente forma:

La regla i es

(sin control de entrada)

$$\begin{array}{c} \text{IF } x_1\left(k\right) \text{ is } A_{1i} \text{ and } x_2\left(k\right) \text{ is } A_{2i} \text{ and } \cdots x_n\left(k\right) \text{ is } A_{ni} \\ \text{THEN } \widehat{x}_1\left(k+1\right) \text{ is } B_{1i} \text{ and} \cdots \widehat{x}_n\left(k+1\right) \text{ is } B_{ni} \end{array}$$

o (con control de entrada)

$$\begin{array}{c} \text{IF } x_1\left(k\right) \text{ is } D_{1i} \text{ and } x_2\left(k\right) \text{ is } D_{2i} \text{ and } \cdots x_n\left(k\right) \text{ is } D_{ni} \\ \text{THEN } \widehat{x}_1\left(k+1\right) \text{ is } u_1 C_{1i} \text{ and} \cdots \widehat{x}_n\left(k+1\right) \text{ is } u_m C_{mi} \end{array}$$

o (recurrente)

$$\widehat{x}\left(k+1\right) = A\widehat{x}\left(k\right)$$

donde, la matriz $A \in \Re^{n\times n}$ es una matriz estable que será especificada después, α es una constante positiva, el siguiente teorema muestra cómo seleccionarla. $\widehat{x}_j\left(k\right)$ es el estado del modelo difuso; A_{ji}, B_{ji}, C_{ji} y D_{ji} son conjuntos difusos estándares[80]. Usamos l reglas difusasIF-THEN $(i = 1, 2 \cdots l)$ para representar el mapeo de un vector lingüístico de entrada $[\widehat{x}_1\left(k\right) \cdots \widehat{x}_n\left(k\right), u_1 \cdots \ u_m] \in \Re^{n+m}$ a un vector lingüístico de salida$[x_1(k+1) \cdots \widehat{x}_n\left(k+1\right)]$. Para cada variable de entrada x_i hay l_i conjuntos difusos. En el caso de una conexión completa, $l = l_1 \times l_2 \times \cdots l_n$. Usando máquina de inferencia difusa, fusificador centro-promedio y defusificador singleton [80], la $p-$ésima salida del sistema lógico difuso puede expresarse

como:

$$\begin{aligned}\widehat{x}_p(k+1) &= \left(\sum_{i=1}^{l} w_{1pi}\left[\prod_{j=1}^{n}\mu_{A_{ji}}\right]\right) / \left(\sum_{i=1}^{l}\left[\prod_{j=1}^{n}\mu_{A_{ji}}\right]\right) = \sum_{i=1}^{l} w_{1pi}\phi_{1i} \\ \widehat{x}_p(k+1) &= \left(\sum_{i=1}^{l} u_i w_{2pi}\left[\prod_{j=1}^{n}\mu_{D_{ji}}\right]\right) / \left(\sum_{i=1}^{l}\left[\prod_{j=1}^{n}\mu_{D_{ji}}\right]\right) = \sum_{i=1}^{l} w_{2pi}\phi_{2i}u_i\end{aligned} \tag{4.2}$$

donde $\mu_{A_{ji}}$ y $\mu_{D_{ji}}$ son las FMs de los conjuntos difusos A_{ji} y D_{ji}; w_{1pi} es el punto al cual $\mu_{B_{pi}} = 1$; w_{2pi} es el punto al cual $\mu_{C_{pi}} = 1$. Si definimos

$$\begin{aligned}\phi_{1i} &= \prod_{j=1}^{n}\mu_{A_{ji}} / \sum_{i=1}^{l}\prod_{j=1}^{n}\mu_{A_{ji}} \\ \phi_{2i} &= \prod_{j=1}^{n}\mu_{D_{ji}} / \sum_{i=1}^{l}\prod_{j=1}^{n}\mu_{D_{ji}}\end{aligned} \tag{4.3}$$

(4.2) puede expresarse en forma de matriz

$$\widehat{x}(k+1) = A\widehat{x}(k) \ + W_1\sigma_1[x(k)] + W_2\sigma_2[x(k)]U(k) \tag{4.4}$$

donde el parámetro $W_1 = \begin{bmatrix} w_{111} & & w_{11l} \\ & \ddots & \\ w_{1n1} & & w_{1nl} \end{bmatrix}$, $W_2 = \begin{bmatrix} w_{211} & & w_{21l} \\ & \ddots & \\ w_{2n1} & & w_{2nl} \end{bmatrix}$, vectores de datos $\sigma_1[x(k)] = [\phi_{11}\cdots\phi_{1l}]^T$, $\sigma_2[x(k)]U(k) = [\phi_{21}u_1\cdots\phi_{2m}u_m 0,\cdots 0]^T$. Este es un modelo serie-paralelo (la entrada es del estado de la planta) [53]. La estructura de las redes neuronales difusas recurrentes de estado-espacio serie-paralelo se muestra en la figura (4.1).

Si las reglas se cambian como

$$\begin{gathered}\text{IF } \widehat{x}_1(k) \text{ is } A_{1i} \text{ and } \widehat{x}_2(k) \text{ is } A_{2i} \text{ and } \cdots \widehat{x}_n(k) \text{ is } A_{ni} \\ \text{THEN } \widehat{x}_1(k+1) \text{ is } B_{1i} \text{ and} \cdots \widehat{x}_n(k+1) \text{ is } B_{ni}\end{gathered}$$

o

$$\begin{gathered}\text{IF } \widehat{x}_1(k) \text{ is } D_{1i} \text{ and } \widehat{x}_2(k) \text{ is } D_{2i} \text{ and } \cdots \widehat{x}_n(k) \text{ is } D_{ni} \\ \text{THEN } \widehat{x}_1(k+1) \text{ is } u_1 C_{1i} \text{ and} \cdots \widehat{x}_n(k+1) \text{ is } u_m C_{mi}\end{gathered}$$

o

$$\widehat{x}(k+1) = A\widehat{x}(k)$$

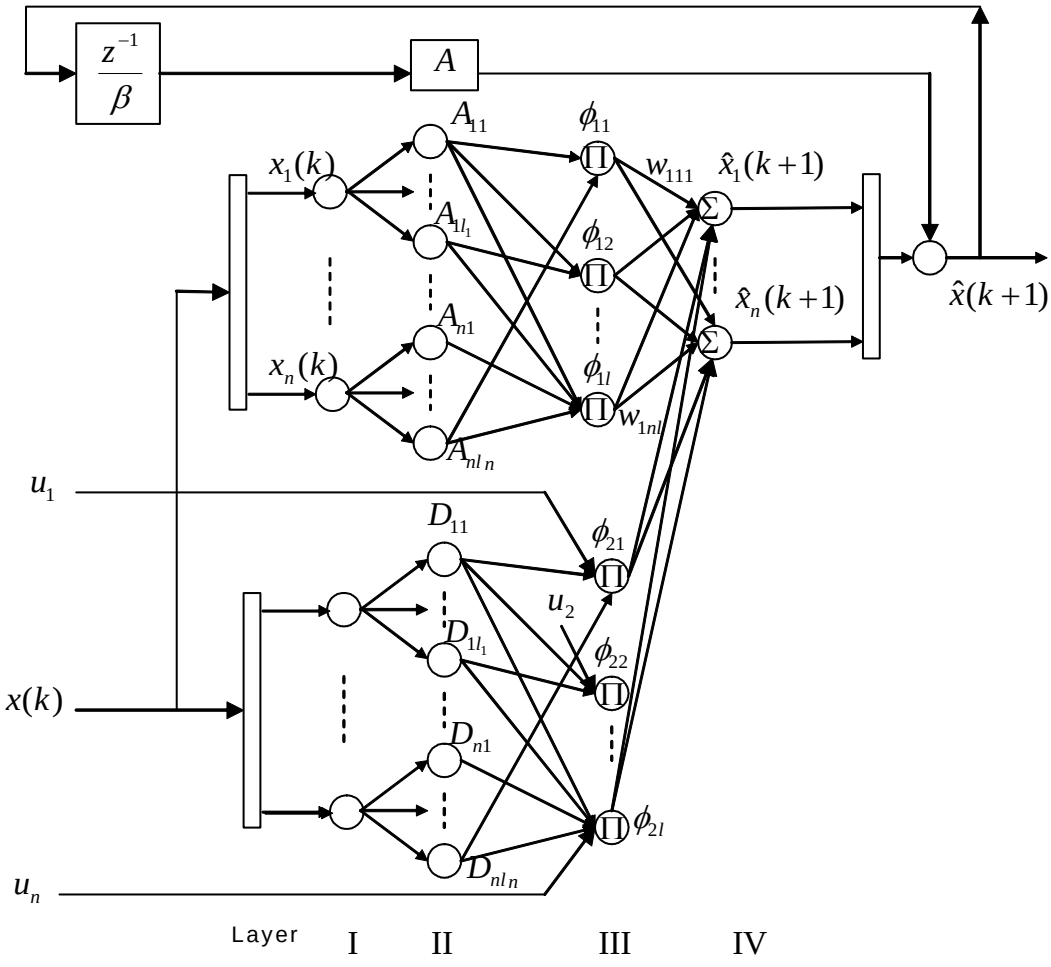

Figura 4.1: Series-parallel state-space recurrent fuzzy neural networks

La red neuronal difusa recurrente es

$$\widehat{x}\left(k+1\right)=A\widehat{x}\left(k\right)\ \ +W_{1}\sigma_{1}\left[\widehat{x}\left(k\right)\right]+W_{2}\sigma_{2}\left[\widehat{x}\left(k\right)\right]U\left(k\right) \tag{4.5}$$

(4.5) es modelo paralelo (la entrada es de estado del modelo) [53]. La estructura de la red neuronal difusa recurrente de estado-espacio paralelo se muestra en figura (4.2). En este trabajo sólo se discute el modelo serie-paralelo, algunos resultados similares pueden obtenerse del modelo paralelo.

La parte de la alimentación hacia adelante es una red neuronal de cuatro capas discutida en varios trabajos [46][80]. La capa I acepta al vector lingüístico de entrada $x\left(k\right)$. Cada nodo de la capa II representa el valor de la función de membresía de la variable lingüística. Los nodos de la capa III representan las reglas difusas. La capa IV es la capa de salida, las ligas entre las capas III y IV están completamente conectadas por la matriz de pesos W. Las capas I y II son la parte premisa. Las capas III y IV son la parte consecuente.

4.2. Identificación de parte ¨IF¨ vía SRFNN

El sistema no lineal identificado se representa como (4.1). Asumimos que la planta es BIBO estable, es decir, $x\left(k\right)$ y $u\left(k\right)$ están limitadas. Según el teorema de Stone-Weierstrass [20], esta suave función generalmente no lineal puede escribirse como

$$x(k+1)=Ax\left(k\right)+W_{1}^{*}\sigma_{1}\left[x\left(k\right)\right]+W_{2}^{*}\sigma_{2}\left[x\left(k\right)\right]U\left(k\right)+\mu\left(k\right) \tag{4.6}$$

donde W_{1}^{*} y W_{2}^{*} son pesos constantes que pueden minimizarse a $\mu\left(k\right)$, $\mu\left(k\right)$ es el error de modelado. Como σ_{1} y σ_{2} son funciones limitadas, $\mu\left(k\right)$ es limitada como $\mu^{2}\left(k\right)\leq\overline{\mu}$, $\overline{\mu}$ es una constante positiva desconocida. El neuro error de identificación se define como:

$$e\left(k\right)=\widehat{x}\left(k\right)-x\left(k\right)$$

De (4.6) y (4.6), entonces

$$e\left(k+1\right)=Ae\left(k\right)+\widetilde{W}_{1}\left(k\right)\sigma_{1}\left[x\left(k\right)\right]+\widetilde{W}_{2}\left(k\right)\sigma_{2}\left[x\left(k\right)\right]U\left(k\right)+\zeta\left(k\right) \tag{4.7}$$

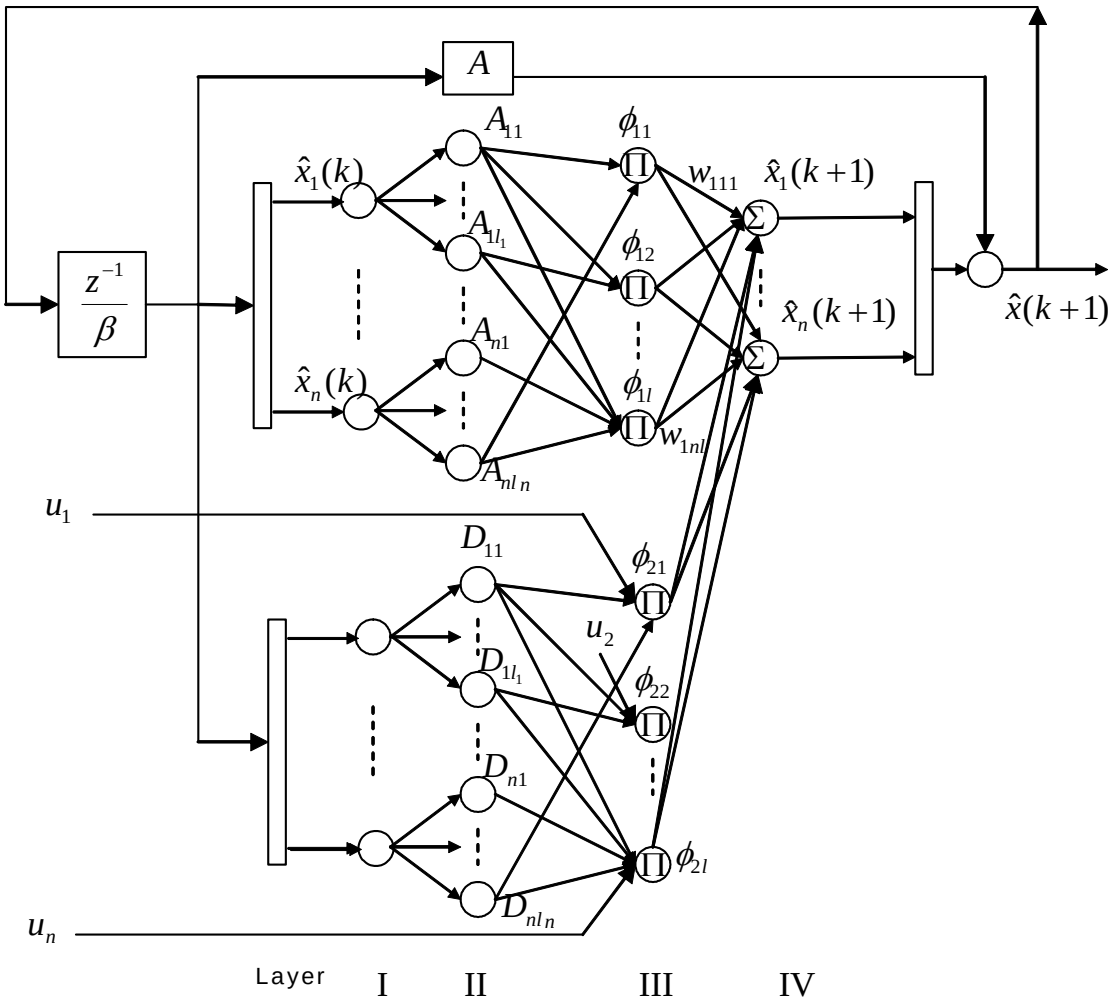

Figura 4.2: Parallel state-space recurrent fuzzy neural networks

donde $\widetilde{W}_1(k) = W_1(k) - W_1^*$, $\widetilde{W}_2(k) = W_2(k) - W_2^*$, $\varepsilon_1(k)$ es un error de aproximación de segundo orden. σ' es una deriativa de la funcion de activación nolineal $\sigma(\cdot)$ para el punto $W_1(k)$. Como ϕ es una función de activación sigmoidal, $\varepsilon_1(k)$ está delimitada por $\|\varepsilon_1(k)\|^2 \leq \overline{\varepsilon}_1$, $\overline{\varepsilon}_1$ es una constante positiva desconocida, donde $\zeta(k) = \varepsilon_1(k) + \varepsilon_2(k) - \mu(k)$. El siguiente teorema da un algoritmo de aprendizaje estable de una red neuronal de una sola capa de tiempo discreto.

Teorema 4.1 *Si usamos una red neuronal difusa recurrente (4.4) para identificar una planta no lineal (4.1) y los eigenvalores de A se seleccionan como* $-1 < \lambda(A) < 0$*, la siguiente ley de actualización sin modificación robusta puede identificar al error* $e(k)$ *limitado(estable en un sentido* L_∞*)*

$$\begin{aligned} W_1(k+1) &= W_1(k) - \eta(k)\,\sigma_1 e(k)^T \\ W_2(k+1) &= W_2(k) - \eta(k)\,u(k)\sigma_2 e^T(k) \end{aligned} \tag{4.8}$$

donde si $\eta(k) = \dfrac{\eta}{1 + \|\sigma\|^2 + \|u\sigma\|^2}; 0 < \eta \leq 1$

Demostración. Seleccionamos la función de Lyapunov como

$$V(k) = \left\|\widetilde{W}_1(k)\right\|^2 + \left\|\widetilde{W}_2(k)\right\|^2$$

donde $\left\|\widetilde{W}_1(k)\right\|^2 = \sum_{i=1}^{n} \widetilde{w}_1(k)^2 = tr\left\{\widetilde{W}_1^T(k)\,\widetilde{W}_1(k)\right\}$. Para la ley de actualización (4.8)

$$\widetilde{W}_1(k+1) = \widetilde{W}_1(k) - \eta(k)\,\sigma_1 e(k)^T$$

Entonces,

$$\begin{aligned} &\Delta V(k) = V(k+1) - V(k) \\ &= \left\|\widetilde{W}_1(k) - \eta(k)\,\sigma_1 e(k)^T\right\|^2 - \left\|\widetilde{W}_1(k)\right\|^2 \\ &+ \left\|\widetilde{W}_2(k) - \eta(k)\,u(k)\sigma_2 e^T(k)\right\|^2 - \left\|\widetilde{W}_2(k)\right\|^2 \\ &= \eta^2(k)\,\|e(k)\|^2\,\|\sigma_1\|^2 - 2\eta(k)\left\|\sigma_1\widetilde{W}_1(k)\,e^T(k)\right\| \\ &+ \eta^2(k)\,\|e(k)\|^2\,\|u(k)\sigma_2\|^2 - 2\eta(k)\left\|u(k)\sigma_2\widetilde{W}_2(k)\,e^T(k)\right\| \end{aligned}$$

Usando (4.7) y $\eta(k) \geq 0$,

$$\begin{aligned}
&-2\eta(k)\left\|\sigma_1\widetilde{W}_1(k)\,e^T(k)\right\| - 2\eta(k)\left\|u(k)\sigma_2\widetilde{W}_2(k)\,e^T(k)\right\| \\
&\leq -2\eta(k)\left\|e^T(k)\right\|\|e(k) - Ae(k) - \zeta(k)\| \\
&= -2\eta(k)\left\|e^T(k)\,e(k) - e^T(k)\,Ae(k) - e^T(k)\,\zeta(k)\right\| \\
&\leq -2\eta(k)\left\|e^T(k)\,e(k)\right\| \\
&+2\eta(k)\,e^T(k)\,Ae(k) + 2\eta(k)\left\|e^T(k)\,\zeta(k)\right\| \\
&\leq -2\eta(k)\|e(k)\|^2 + 2\eta(k)\,\lambda_{\text{máx}}(A)\|e(k)\|^2 \\
&+\eta(k)\|e(k)\| + \eta(k)\|\zeta(k)\|^2
\end{aligned}$$

Desde $0 < \eta \leq 1$,

$$\begin{aligned}
&\Delta V(k) \leq \eta^2(k)\|e(k)\|^2\|\sigma' x(k)\|^2 \\
&+\eta^2(k)\|e(k)\|^2\left\|u(k)\phi' x(k)\right\|^2 + 2\eta(k)\,\lambda_{\text{máx}}(A)\|e(k)\|^2 \\
&-\eta(k)\|e(k)\|^2 + \eta(k)\|\zeta(k)\|^2 \\
&= -\eta(k)\left[1 - 2\lambda_{\text{máx}}(A)\right] \\
&+\eta(k)\,\frac{\|\sigma' x(k)\|^2 + \left\|u(k)\phi' x(k)\right\|^2}{1 + \|\sigma' x(k)\|^2 + \left\|u(k)\phi' x(k)\right\|^2}\eta e^2(k) \\
&+\eta_k\zeta^2(k) \leq -\pi e^2(k) + \eta\zeta^2(k)
\end{aligned} \tag{4.9}$$

donde $\pi = \dfrac{\eta}{1+\kappa}\left[1 - \dfrac{\kappa}{1+\kappa}\right]$, $\kappa = \underset{k}{\text{máx}}\left(\|\sigma' x(k)\|^2 + \left\|u(k)\phi' x(k)\right\|^2\right)$. Entonces $-1 < \lambda(A) < 0$, $\pi > 0$

$$n\,\text{mín}\left(\widetilde{w}_i^2\right) \leq V_k \leq n\,\text{máx}\left(\widetilde{w}_i^2\right)$$

donde $n \times \text{mín}\left(\widetilde{w}_i^2\right)$ y $n \times \text{máx}\left(\widetilde{w}_i^2\right)$ son $\mathcal{K}_\infty$-funciones, y $\pi e^2(k)$ es una $\mathcal{K}_\infty$-función, $\eta\zeta^2(k)$ es una $\mathcal{K}$-función, entonces V_k admite la función suave de ISS-Lyapunov función como en la *Definción 2*. Del *Teorema* 1, la dinámica de la identificación del error es estable entrada-a -salida (input -to-output). La entrada "INPUT" corresponde al segundo término de la la última línea en (4.9), *es decir*,el error modelado $\zeta(k) = \varepsilon_1(k) + \varepsilon_2(k) - \mu(k)$, el estado "STATE" corresponde al primer término de la última línea en (4.9), es decir, el error de identificación(k). Debido a que la entrada "INPUT" $\zeta(k)$ está limitada por y, la dinámica es ISS, el estado"STATE" $e(k)$ está limitado. ■

Comentario 4.1 *(4.8) es el algoritmo de gradiente descendente, con el que la velocidad de aprendizaje normalizada η_k es variante en el tiempo para asegurar que el proceso de identificación es variable. La ley de aprendizaje es simple de usar porque no requiere de cuidar cómo seleccionar una mejor velocidad de aprendizaje para asegurar tanto una rápida convergencia y estabilidad. No requiere información previa.*

Comentario 4.2 *La condición $\beta \|e(k+1)\| \geq \|e(k)\|$ es zona muerta. Si β se selecciona lo suficientemente grande, la zona muerta es lo suficientemente pequeña.*

4.3. Identificación de parte ¨THEN¨ vía SRFNN

Con la planta identificada se ve como una caja-negra, no podemos construir reglas difusas para la parte premisa $\mu_{A_{ji}}$ y $\mu_{D_{ji}}$.(ver [76][83][81]). El objetivo de modelar con redes neuronales es encontrar los valores centrales W_1 de $B_{1i} \cdots B_{mi}$ y W_2 de $C_{1i} \cdots C_{mi}$, como las FM´s $A_{1i} \cdots A_{ni}$ y $D_{1i} \cdots D_{ni}$, tales que las redes neuronales difusas (4.4) puedan seguir a la planta no lineal (4.1). Para simplificar el cálculo, discutimos el caso de $U = 0$, luego D y C desaparecen. En este trabajo las FMs Gaussianas son explotadas para identificar reglas difusas que se definen como $\mu_{A_{ji}} = \exp\left(-\frac{(x_j - c_{ji})^2}{\rho_{ji}^2}\right)$.

El modelo neuro difuso puede expresarse como:

$$\widehat{x}(k+1) = A\widehat{x}(k) + W_1(k)\,\sigma_1[x(k)] \tag{4.10}$$

Se define

$$z_i = \prod_{j=1}^{n} \exp\left(-\frac{(x_j - c_{ji})^2}{\rho_{ji}^2}\right)$$
$$a_q = \alpha \sum_{i=1}^{l} w_{1qi} z_i, \qquad b = \sum_{i=1}^{l} z_i$$

Luego

$$\widehat{x}_q = \frac{a_q}{b}$$

El proceso no lineal identificado (4.1) puede representarse como

$$\widehat{x}_q = \frac{\sum_{i=1}^{l} w_{qi}^* \prod_{j=1}^{n} \exp\left(-\frac{\left(x_j - c_{ji}^*\right)^2}{\rho_{ji}^{*2}}\right)}{\left[\sum_{i=1}^{l} \prod_{j=1}^{n} \exp\left(-\frac{\left(x_j - c_{ji}^*\right)^2}{\rho_{ji}^{*2}}\right)\right]} - \mu_q \tag{4.11}$$

donde w_{qi}^*, c_{ji}^* y ρ_{ji}^{*2} son parámetros desconocidos que pueden minimizar un modelado dinámico μ_q.

En caso de tres variables independientes, la función suave f tiene una fórmula de Taylor dada por

$$f(x_1, x_2, x_3) = \sum_{k=0}^{l-1} \frac{1}{k!} [\left(x_1 - x_1^0\right) \frac{\partial}{\partial x_1} + \left(x_2 - x_2^0\right) \frac{\partial}{\partial x_2} + \left(x_3 - x_3^0\right) \frac{\partial}{\partial x_3}]_0^k f + R_l$$

donde R_l es el residuo de la fórmula de Taylor. Sea x_1, x_2, x_3 sus correspondientes w_{pi}^*, c_{ji}^* y σ_{ji}^{*2}, x_1^0, x_2^0, x_3^0 sus correspondientes w_{pi}, c_{ji} y σ_{ji}^2,

$$\begin{aligned} x_q + \mu_q = \widehat{x}_q + \sum_{i=1}^{l} \left(w_{qi}^* - w_{qi}\right) z_i / b + \sum_{i=1}^{l} \sum_{j=1}^{n} \frac{\partial}{\partial c_{ji}} \left(\frac{a_q}{b}\right) \left(c_{ji}^* - c_{ji}\right) \\ + \sum_{i=1}^{l} \sum_{j=1}^{n} \frac{\partial}{\partial \rho_{ji}} \left(\frac{a_q}{b}\right) \left(\rho_{ji}^* - \rho_{ji}\right) + R_{1q} \end{aligned} \tag{4.12}$$

Usando la regla de la cadena, podemos obtener

$$\begin{aligned} &\frac{\partial}{\partial c_{ji}} \left(\frac{a_q}{b}\right) = \frac{\partial}{\partial z_i} \left(\frac{a_q}{b}\right) \frac{\partial z_i}{\partial c_{ji}} = \left(\frac{1}{b} \frac{\partial a_q}{\partial z_i} + \frac{\partial}{\partial z_i} \left(\frac{1}{b}\right) a_q\right) \left(2 z_i \frac{x_j - c_{ji}}{\sigma_{ji}^2}\right) \\ &= \left(\frac{w_{qi}}{b} - \frac{a_q}{b^2}\right) \left(2 z_i \frac{x_j - c_{ji}}{\sigma_{ji}^2}\right) = 2 z_i \frac{w_{qi} - \widehat{y}_q}{b} \frac{x_j - c_{ji}}{\sigma_{ji}^2} \\ &\frac{\partial}{\partial \rho_{ji}} \left(\frac{a_q}{b}\right) = \frac{\partial}{\partial z_i} \left(\frac{a_q}{b}\right) \frac{\partial z_i}{\partial \rho_{ji}} = 2 z_i \frac{w_{qi} - \widehat{x}_q(k+1)}{b} \frac{\left(x_j - c_{ji}\right)^2}{\rho_{ji}^3} \end{aligned}$$

En forma matricial,

$$x_q(k+1) + \mu_q = \widehat{x}_q(k+1) - \widetilde{W}_q Z(k) - D_{Zq} \overline{C}_k E - D_{Zq} \overline{B}_k E$$

Definimos al error de identificación como:

$$\begin{aligned} e_q &= \widehat{x}_q(k+1) - x_q(k+1) \\ e_q &= Z(k) \widetilde{W}_q + D_{Zq} \overline{C}_k E + D_{Zq} \overline{B}_k E + \mu_q - R_{1q} \end{aligned} \tag{4.13}$$

donde R_{1q} es un error de aproximación de segundo orden de la serie de Taylor, $q = 1 \cdots m$. Definnimos $e(k) = [e_1 \cdots e_m]^T$, entonces

$$e(k) = Ae(k) + \widetilde{W}_k Z(k) + D_z(k) \overline{C}_k E + D_z(k) \overline{B}_k E + \zeta(k) \tag{4.14}$$

Por la limitación de la función Gausiana ϕ podemos asumir que μ en (4.11) está limitada, también R_1 es limitada. Entonces $\zeta(k)$ en (4.14) es limitada. El siguiente teorema proporciona un algoritmo de retropropagación estable para redes neuronales difusas de tiempo discreto.

Teorema 4.2 *Si usamos una red neuronal (4.10) para identificar una planta no lineal (4.1), los eigenvalores de A se seleccionan como $-1 < \lambda(A) < 0$, y el siguiente algoritmo de retropropagación puede hacer que el error de identificación $e(k)$ esté limitado.*

$$\begin{aligned} W_{k+1} &= W_k - \eta_k e(k) Z(k)^T \\ c_{ji}(k+1) &= c_{ji}(k) - 2\eta_k z_i \tfrac{w_{pi} - \widehat{y}_p}{b} \tfrac{x_j - c_{ji}}{\rho_{ji}^2} (\widehat{x}_q - x_q) \\ \rho_{ji}(k+1) &= \rho_{ji}(k) - 2\eta_k z_i \tfrac{w_{pi} - \widehat{y}_p}{b} \tfrac{(x_j - c_{ji})^2}{\rho_{ji}^3} (\widehat{x}_q - x_q) \end{aligned} \tag{4.15}$$

donde si $\eta(k) = \dfrac{\eta}{1 + \|Z\|^2 + 2\|D_z\|^2}$; $0 < \eta \leq 1$. *El promedio del error de identificación satisface*

$$J = \lim_{T \to \infty} \sup \frac{1}{T} \sum_{k=1}^{T} e^2(k) \leq \frac{\eta}{\pi} \overline{\zeta} \tag{4.16}$$

dode $\pi = \dfrac{\eta}{1+\kappa} \left[1 - \dfrac{\kappa}{1+\kappa} \right] > 0$, $\kappa = \max\limits_k \left(\|Z\|^2 + 2\|D_z\|^2 \right)$, $\overline{\zeta} = \max\limits_k \left[\zeta^2(k) \right]$

Demostración. Definamos $\widetilde{c}_{ji}(k) = c_{ji}(k) - c_{ji}^*$, $\widetilde{b}_{ji}(k) = \sigma_{ji}(k) - \sigma^*(k)$, el elemento de $\widetilde{C}_k$ se expresa como $\widetilde{c}_{ji}(k) = \left[\widetilde{C}_k \right]$. Luego

$$\left[\widetilde{C}_{k+1} \right] = \left[\widetilde{C}_k \right] - 2\eta_k z_i \frac{w_{qi} - \widehat{y}_q}{b} \frac{x_j - c_{ji}}{\rho_{ji}^2} (\widehat{x}_q - x_q)$$

Seleccionamos una matriz positiva L_k definida como

$$L_k = \left\| \widetilde{W}_k \right\|^2 + \left\| \widetilde{C}_k \right\|^2 + \left\| \widetilde{B}_k \right\|^2 \tag{4.17}$$

Similar a la demostración del teorema 1, usando (4.14) tenemos

$$\begin{aligned}
\Delta L_k &= \left\|\widetilde{W}_k - \eta_k e(k) Z(k)^T\right\|^2 + \left\|\widetilde{C}_k - \left[2\eta_k z_i \tfrac{w_{qi}-\widehat{y}_q}{b} \tfrac{x_j - c_{ji}}{\rho_{ji}^2} (\widehat{x}_q - x_q)\right]\right\|^2 \\
&+ \left\|\widetilde{B}_k - \left[2\eta_k z_i \tfrac{w_{qi}-\widehat{y}_q}{b} \tfrac{(x_j-c_{ji})^2}{\rho_{ji}^3} (\widehat{x}_q - x_q)\right]\right\|^2 - \left\|\widetilde{W}_k\right\|^2 - \left\|\widetilde{C}_k\right\|^2 - \left\|\widetilde{B}_k\right\|^2 \\
&= \eta_k^2 e^2(k) \left(\left\|Z(k)^T\right\|^2 + 2\left\|D_z^T\right\|^2\right) - 2\eta_k \|e(k)\| \left\|\widetilde{W}_k Z(k)^T + D_z^T \overline{C}_k E + D_z^T \overline{B}_k E\right\| \\
&= \eta_k^2 e^2(k) \left(\|Z\|^2 + 2\|D_z\|^2\right) - 2\eta_k \|e(k)[e(k) - \zeta(k)]\| \\
&\leq -\eta_k e^2(k) \left[1 - \eta_k \left(\|Z\|^2 + 2\|D_z\|^2\right)\right] + \eta\zeta^2(k) \\
&\leq -\pi e^2(k) + \eta\zeta^2(k)
\end{aligned} \tag{4.18}$$

donde π se define como

$$\pi = \frac{\eta}{1 + \max_k \left(\|Z\|^2 + 2\|D_z\|^2\right)}$$

Debido a

$$\begin{aligned}
n\left[\min(\widetilde{w}_i^2) + \min\left(\widetilde{c}_{ji}^2\right) + \min\left(\widetilde{b}_{ji}^2\right)\right] &\leq L_k \\
\leq n\left[\max(\widetilde{w}_i^2) + \max\left(\widetilde{c}_{ji}^2\right) + \max\left(\widetilde{b}_{ji}^2\right)\right]
\end{aligned}$$

donde $n\left[\min(\widetilde{w}_i^2) + \min\left(\widetilde{c}_{ji}^2\right) + \min\left(\widetilde{b}_{ji}^2\right)\right]$ y $n\left[\max(\widetilde{w}_i^2) + \max\left(\widetilde{c}_{ji}^2\right) + \max\left(\widetilde{b}_{ji}^2\right)\right]$ son $\mathcal{K}_\infty$-funciones, y $\pi e^2(k)$ es una $\mathcal{K}_\infty$-función, $\eta\zeta^2(k)$ es una $\mathcal{K}$-función. De (4.14) y (4.17) sabemos que V_k es la función de $e(k)$ y $\zeta(k)$, entonces L_k admite una función suave ISS-Lyapunov como en la *Definiciónn 2*. Del *Teorema 1*, la dinámica de la identificación del error es entrada-a estado (input-to-state stable). Debido a que la entrada "INPUT" $\zeta(k)$ está limitada y la dinámica es ISS, el estado $e(k)$ está limitado.

(5.9) puede escribirse como:

$$\Delta L_k \leq -\pi e^2(k) + \eta\zeta^2(k) \leq \pi e^2(k) + \eta\overline{\zeta} \tag{4.19}$$

Sumando (4.19) desde 1 hasta T, y usando $L_T > 0$ y L_1 es una constante, obtenemos

$$\begin{aligned}
L_T - L_1 &\leq -\pi \textstyle\sum_{k=1}^T e^2(k) + T\eta\overline{\zeta} \\
\pi \textstyle\sum_{k=1}^T e^2(k) &\leq L_1 - L_T + T\eta\overline{\zeta} \leq L_1 + T\eta\overline{\zeta}
\end{aligned}$$

(4.16) se establece. ■

4.4. Simulación

En esta sección se escoge un típico sistema caótico, el modelo de Lorenz, para demostrar las habilidades de las redes neuronales recurentes. El modelo de Lorentz se usa para descripciones fluidas especialmente para algunas caracterizaciones de la dinámica de la atmósfera. El modelo sin controlar está dado por:

$$\begin{aligned} \dot{x}_1 &= -\beta x_1 + x_2 x_3 \\ \dot{x}_2 &= \omega(x_3 - x_2) \\ \dot{x}_3 &= -x_1 x_2 + \rho x_2 - x_3 \end{aligned} \tag{4.20}$$

con $x_0 = [50, 10, 20]^T$. Usamos diferentes técnicas para obtener los estados de tiempo discretos del sistema (4.20). Definimos $s_1 = Ax_k$, $s_2 = A(x_k + s_1)$, $s_3 = A\left(x_k + \frac{s_1+s_2}{4}\right)$. Si $\left|\frac{s_1-2*s_3+s_2}{3}\right| \leq \frac{|x_k|}{1000}$ ó $\left|\frac{s_1-2*s_3+s_2}{3}\right| < 1$, entonces $x_{k+1} = x_k + \frac{s_1+4s_3+s_2}{6}$, $k = 0, 1, 2\cdots$, . primero usamos identificación por redes neuronales difusas sin aprendizaje de funciones de membresía premisas

$$\beta\widehat{x}(k+1) = A\widehat{x}(k) + W_1(k)\,\sigma_1[x(k)]$$

donde $\beta = 4$, $\sigma_1(\cdot) = \tanh(\cdot)$, $\widehat{x} = [\widehat{x}_1, \widehat{x}_2, \widehat{x}_3]^T$, $A = diag[,8,,8,,8]$, $W_1(k) \in \Re^{3\times 3}$. Los elementos de $W_1(0)$ son números aleatorios entre $[0, 1]$. Usamos (4.8) para actualizar pesos

$$W_1(k+1) = W_1(k) - \frac{1}{1 + \|\sigma\|^2} e(k)\,\sigma^T$$

Los resultados de identificación en línea se muestran en Figura 4.3. El Teorema 1 da un acondición **necesaria** deη para aprendizaje estable, $\eta \leq 1$. En este ejemplo, encontramos que si $\eta \geq 2{,}5$, el proceso de aprendizaje se vuelve inestable. El tiempo total de simulación es 600, hay 14 tiempos para $\beta\|e(k+1)\| < \|e(k)\|$. Luego, usamos una red neuronal con aprendizaje de las funciones de membresía premisas

$$\begin{aligned} c_{ji}(k+1) &= c_{ji}(k) - 2\eta_k z_i \tfrac{w_{pi}-\widehat{y}_p}{b} \tfrac{x_j - c_{ji}}{\rho_{ji}^2} (\widehat{x}_q - x_q) \\ \rho_{ji}(k+1) &= \rho_{ji}(k) - 2\eta_k z_i \tfrac{w_{pi}-\widehat{y}_p}{b} \tfrac{(x_j - c_{ji})^2}{\rho_{ji}^3} (\widehat{x}_q - x_q) \end{aligned} \tag{4.21}$$

La identificación en línea resultante se muestra en la Figura 4.4.. la complejidad del modelo es importante en el contexto de identificación del sistema, que corresponde al número de reglas.

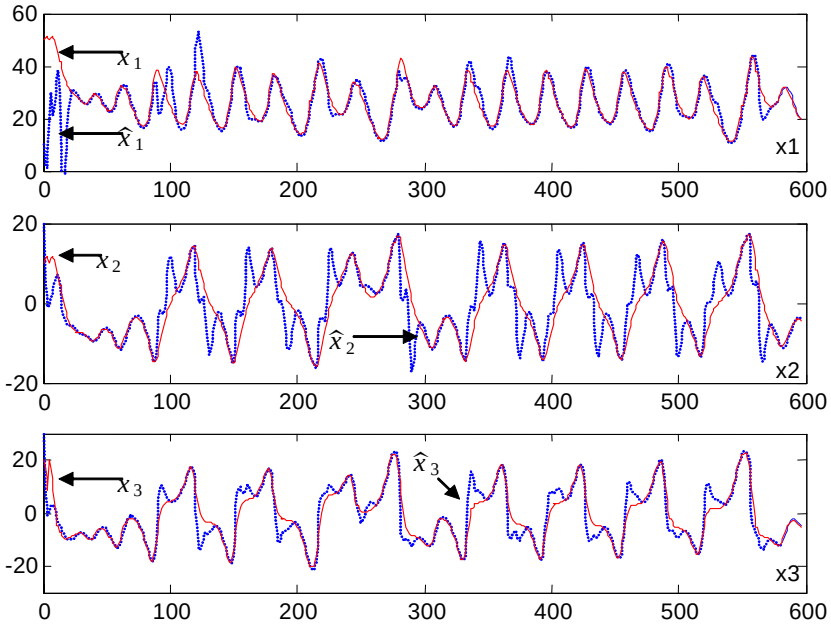

Figura 4.3: Identificación de parte IF.

En la simulación tratamos de probar con diferentes números de reglas, encontramos que después de la regla número l, mayor a 20, la aproximación de identificación ya no improvisa demasiado. El tiempo de simulación total es de 600, hay 1 tiempo para $\beta \|e(k+1)\| < \|e(k)\|$.

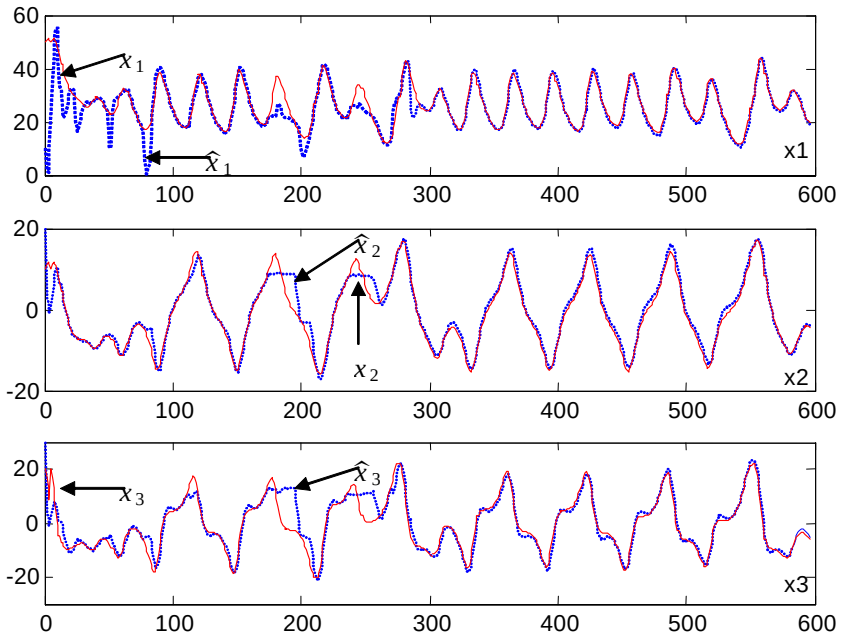

Figura 4.4: Identificación de partes IF y THEN

Capítulo 5

Identificación de estructura y parametros para redes neuronales difusas estáticas

Si un sistema es además supuesto a ser deterministico, invariante en el tiempo, solo-entrada-solo-salida (SISO), el modelo se convierte en

$$y(k) = f(y(k+1), y(k-2), \ldots, y(k-n), u((k-1), u(k-2), \ldots, u(k-m)) \quad (5.1)$$

donde $[u(k), y(k)]$ representa la pareja de entrada-salida del sistema en el tiempo k. Los números enteros positivos n y m son respectivamente el número de salidas pasadas (también llamada el orden del sistema) y el número de entradas pasadas. En la práctica m es normalmente más pequeño o igual a n. La función f puede ser una función estática que mapea las entradas y salidas pasadas a una nueva salida.

Después de derivar un sistema de inferencia difuso inicial basado en agrupamiento difuso o agrupamiento substractivo, sus parametros, por ejemplo, los centros y anchos de las funciones de membresía deben ser optimizadas. Este proceso de optimización se realiza minimizando

un índice de rendimiento

$$PI = RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\widehat{y}_i - y_i\right)^2}$$

Donde, $\widehat{y}_i$ es la salida del modelo, y_i es la salida real, y N es el número de datos. La minimización del índice de rendimiento puede ser realizada

Discutimos dos casos:

- Cada grupo corresponde a una regla
- Cada grupo corresponde a varias reglas

5.1. Una regla en cada grupo

Para cada zona j, hay una regla difusa como

$$\text{IF } x_1 \text{ is } A_1^j \text{ AND} \cdots x_n \text{ is } A_n^j \text{ THEN } y \text{ is } B^j$$

Si utilizamos fusificador singleton, implicación de Mamdani, y un defusificador promedio de centros, la salida de la j-ésima zona difusa puede ser expresada como

$$\widehat{y} = \overline{y}^j \prod_{j=1}^{n} \mu_{A_i^j}\left(x_i\right)$$

donde $\overline{y}^j$ es el centro de B^j, y $\mu_{B^j}\left(\overline{y}^j\right) = 1$ (conjunto difuso normal),

$$\mu_{A_i^j}\left(x_i\right) = \exp\left[-\left(\frac{x_i - c_i^j}{\sigma_i^j}\right)^2\right]$$

Definimos $x = \left[x_1 \cdots x_n\right]^T$, usamos la pareja de datos (x, y) para encontrar algunas FMs convenientes en la zona. Esto puede ser transformado en un problema de identificación para determinar los parametros $\overline{y}^j$, c_i^j y σ_i^j, el objetivo es encontrar tanto los valores de los centros B^j, como las FMs $A_1^j \cdots A_n^j$, tales que $\widehat{y} \to y$.

5.1.1. Entrenamiento de gradiente decendente

Definamos un indice de funcionamiento

$$J = \frac{1}{2}\left(\widehat{y} - y\right)^2$$

Definamos

$$z^j = \prod_{i=1}^{n} \exp\left[-\left(\frac{x_i - c_i^j}{\sigma_i^j}\right)^2\right] = \exp\left[-\sum_{i}^{n}\left(\frac{x_i - c_i^j}{\sigma_i^j}\right)^2\right]$$

De esta forma

$$\widehat{y} = \overline{y}^j z^j$$

Usando la regla de la cadena

$$\frac{\partial J}{\partial \overline{y}^j} = \frac{\partial J}{\partial \widehat{y}}\frac{\partial \widehat{y}}{\partial \overline{y}^j} = \left(\widehat{y} - y\right) z^j$$

El entrenamiento de gradiente decendente es

$$\overline{y}^j\left(k+1\right) = \overline{y}^j\left(k\right) - \eta\left(\widehat{y} - y\right) z^j$$

ya que $\frac{\partial \exp(x)}{\partial x} = \exp\left(x\right)$

$$\frac{\partial J}{\partial c_i^j} = \frac{\partial J}{\partial \widehat{y}}\frac{\partial \widehat{y}}{\partial z^j}\frac{\partial z^j}{\partial c_i^j} = \left(\widehat{y} - y\right)\overline{y}^j \prod_{i=1}^{n} \exp\left[-\left(\frac{x_i - c_i^j}{\sigma_i^j}\right)^2\right]\left[2\frac{\left(x_i - c_i^j\right)}{\left(\sigma_i^j\right)^2}\right]$$

De esta forma

$$c_i^j\left(k+1\right) = c_i^j\left(k\right) - 2\eta\left(\widehat{y} - y\right)\overline{y}^j z^j \frac{\left(x_i - c_i^j\right)}{\left(\sigma_i^j\right)^2}$$

$$\frac{\partial J}{\partial \sigma_i^j} = \frac{\partial J}{\partial \widehat{y}}\frac{\partial \widehat{y}}{\partial z^j}\frac{\partial z^j}{\partial \sigma_i^j} = \left(\widehat{y} - y\right)\overline{y}^j \prod_{i=1}^{n} \exp\left[-\left(\frac{x_i - c_i^j}{\sigma_i^j}\right)^2\right]\left[2\frac{\left(x_i - c_i^j\right)^2}{\left(\sigma_i^j\right)^3}\right]$$

$$\sigma_i^j\left(k+1\right) = \sigma_i^j\left(k\right) - 2\left(\widehat{y} - y\right)\overline{y}^j z^j \frac{\left(x_i - c_i^j\right)^2}{\left(\sigma_i^j\right)^3}$$

5.1.2. Aprendizaje estable

Asumimos que la pareja de datos (x, y) en la zona puede ser expresada con funciones de membresía Gaussianas como

$$y = \overline{y}^{j*} \prod_{j=1}^{n} \exp\left[-\left(\frac{x_i - c_i^{j*}}{\sigma_i^{j*}} \right)^2 \right] - \mu^j$$

donde $\overline{y}^{j*}$, c_i^{j*} y σ_i^{j*} son parámetros no-conocidos que pueden minimizar el error de modelado μ^j. En el caso de tres variables independientes, una función suave f tiene una formula de Taylor como

$$f(x_1, x_2, x_3) = \sum_{k=0}^{l-1} \frac{1}{k!} \left[\left(x_1 - x_1^0\right) \frac{\partial}{\partial x_1} + \left(x_2 - x_2^0\right) \frac{\partial}{\partial x_2} + \left(x_3 - x_3^0\right) \frac{\partial}{\partial x_3} \right]_0^k f + R_l$$

donde R_l es el resto de la formula de Taylor. Si dejamos x_1, x_2, x_3 corresponden $\overline{y}^j$, c_i^j y σ_i^j, x_1^0, x_2^0, x_3^0 corresponden $\overline{y}^{j*}$, c_i^{j*} y σ_i^{j*}

$$y_q = \sum_{i=1}^{l} w_{qi}^* \prod_{j=1}^{n} \exp\left(-\frac{\left(x_j - c_{ji}^*\right)^2}{\sigma_{ji}^{*2}} \right) / \left[\sum_{i=1}^{l} \prod_{j=1}^{n} \exp\left(-\frac{\left(x_j - c_{ji}^*\right)^2}{\sigma_{ji}^{*2}} \right) \right] - \mu_q \tag{5.2}$$

$$\widehat{y} = y + \mu^j + \left(\overline{y}^j - \overline{y}^{j*}\right) z^j + \sum_{i=1}^{n} \frac{\partial\left(\overline{y}^j z^j\right)}{\partial c_i^j} \left(c_i^j - c_i^{j*}\right) + \sum_{i=1}^{n} \frac{\partial\left(\overline{y}^j z^j\right)}{\partial \sigma_i^j} \left(\sigma_i^j - \sigma_i^{j*}\right) + R_1^j \tag{5.3}$$

donde R_1^j es el error de aproximación de segundo orden de las series de Taylor. Utilizando la regla de la cadena, tenemos

$$\begin{aligned} \frac{\partial\left(\overline{y}^j z^j\right)}{\partial c_i^j} &= \frac{\partial\left(\overline{y}^j z^j\right)}{\partial z^j} \frac{\partial z^j}{\partial c_i^j} = \overline{y}^j 2 z^j \frac{x_i - c_i^j}{\left(\sigma_i^j\right)^2} \\ \frac{\partial\left(\overline{y}^j z^j\right)}{\partial \sigma_i^j} &= \frac{\partial\left(\overline{y}^j z^j\right)}{\partial z^j} \frac{\partial z^j}{\partial \sigma_i^j} = \overline{y}^j 2 z_i \frac{\left(x_i - c_i^j\right)^2}{\left(\sigma_i^j\right)^3} \end{aligned}$$

Para simplificar la expresión utilizamos un vector de forma y omitimos j (j-ésima zona). Definimos el error de modelado como, para la k-ésima pareja de datos

$$\begin{aligned} e &= \widehat{y} - y \\ e_k &= z_k \widetilde{y}_k + 2\overline{y}_k z_k D_{1k}^T \widetilde{C}_k + 2\overline{y}_k z_k D_{2k}^T \widetilde{\Omega}_k + \zeta_k \end{aligned} \tag{5.4}$$

donde $\widetilde{y}_k = \overline{y}_k - \overline{y}_k^*$, $\zeta_k = R_1 + \mu$,

$$D_1 = \left[\frac{x_1 - c_1}{\sigma_1^2}, \cdots \frac{x_n - c_n}{\sigma_n^2}\right]^T, \quad \widetilde{C} = \left[\left(c_1 - c_1^*\right), \cdots \left(c_n - c_n^*\right)\right]^T$$
$$D_2 = \left[\frac{(x_1 - c_1)^2}{\sigma_1^3}, \cdots \frac{(x_n - c_n)^2}{\sigma_n^3}\right]^T, \quad \widetilde{\Omega} = \left[\left(\sigma_1 - \sigma_1^*\right), \cdots \left(\sigma_n - \sigma_n^*\right)\right]^T$$

Teorema 5.1 *Si utilizamos redes neuronales difusas de tipo-Mamdani (4.10) para identificar la planta no lineal (5.1), el siguiente algoritmo de retropropagación hace al error de identificación $e(k)$ limitado*

$$\begin{aligned} \overline{y}_{k+1} &= \overline{y}_k - \eta_k z_k e_k \\ C_{k+1} &= C_k - 2\eta_k \overline{y}_k z_k D_{1k} e_k \\ \Omega_{k+1} &= \Omega_k - 2\eta_k \overline{y}_k z_k D_{2k} e_k \end{aligned} \tag{5.5}$$

donde

$$e_k = \widehat{y}(k) - y(k) \tag{5.6}$$

$C = [c_1, \cdots c_n]^T$, $\Omega = [\sigma_1, \cdots \sigma_n]^T$, $\eta_k = \frac{\eta}{1+\Phi_k}$, $\Phi_k = z_k^2 + 4\overline{y}_k^2 z_k^2 \left\|D_{1k}\right\|^2 + 4\overline{y}_k^2 z_k^2 \left\|D_{2k}\right\|^2$, $0 < \eta \leq 1$. *El promedio del error de identificación satisface*

$$J = \lim_{T \to \infty} \sup \frac{1}{T} \sum_{k=1}^{T} e_k^2 \leq \frac{\eta}{\pi} \overline{\zeta} \tag{5.7}$$

donde $\pi = \frac{\eta}{(1+\Phi_k)^2} > 0$, $\overline{\zeta} = \max_k \left[\zeta_k^2\right]$

Demostración. Seleccionando un escalar positivo L_k definido como

$$L_k = \widetilde{y}_k^2 + \left\|\widetilde{C}_k\right\|^2 + \left\|\widetilde{\Omega}_k\right\|^2 \tag{5.8}$$

La ley de actualización (5.5) puede ser escrita como

$$\begin{aligned} \widetilde{y}_{k+1} &= \widetilde{y}_k - \eta_k z_k e_k \\ \widetilde{C}_{k+1} &= \widetilde{C}_k - 2\eta_k \overline{y}_k z_k D_{1k} e_k \\ \widetilde{\Omega}_{k+1} &= \widetilde{\Omega}_k - 2\eta_k \overline{y}_k z_k D_{2k} e_k \end{aligned}$$

Asi tenemos

$$\begin{aligned}\Delta L_k &= \left(\widetilde{y}_k - \eta_k z_k e_k\right)^2 - \widetilde{y}_k^2 + \left\|\widetilde{C}_k - 2\eta_k \overline{y}_k z_k D_{1k} e_k\right\|^2 - \left\|\widetilde{C}_k\right\|^2 + \left\|\widetilde{\Omega}_k - 2\eta_k \overline{y}_k z_k D_{2k} e_k\right\|^2 - \left\|\widetilde{\Omega}_k\right\|^2 \\ &= \eta_k^2 z_k^2 e_k^2 - 2\eta_k z_k e_k \widetilde{y}_k + 4\eta_k^2 \overline{y}_k^2 z_k^2 \left\|D_{1k}\right\|^2 e_k^2 - 4\eta_k \overline{y}_k z_k e_k D_{1k}^T \widetilde{C}_k + 4\eta_k^2 \overline{y}_k^2 z_k^2 \left\|D_{2k}\right\|^2 e_k^2 - 4\eta_k \overline{y}_k z_k e_k D_{2k}^T \widetilde{\Omega}_k \\ &= \eta_k^2 e_k^2 z_k^2 \left(1 + 4\overline{y}_k^2 \left\|D_{1k}\right\|^2 + 4\overline{y}_k^2 \left\|D_{2k}\right\|^2\right) - 2\eta_k e_k \left(z_k \widetilde{y}_k + 2\overline{y}_k z_k D_{1k}^T \widetilde{C}_k + 2\overline{y}_k z_k D_{2k}^T \widetilde{\Omega}_k\right)\end{aligned} \tag{5.9}$$

Porque $e_k = z_k \widetilde{y}_k + 2\overline{y}_k z_k D_{1k}^T \widetilde{C}_k + 2\overline{y}_k z_k D_{2k}^T \widetilde{\Omega}_k + \zeta_k$

$$\Delta L_k = \eta_k^2 e_k^2 z_k^2 \left(1 + 4\overline{y}_k^2 \left\|D_{1k}\right\|^2 + 4\overline{y}_k^2 \left\|D_{2k}\right\|^2\right) - 2\eta_k e_k \left(e_k - \zeta_k\right)$$

Porque $\eta_k > 0$

$$\begin{aligned}-2\eta_k e_k \left(e_k - \zeta_k\right) &= -2\eta_k e_k^2 + 2\eta_k e_k \zeta_k \\ &\leq -2\eta_k e_k^2 + \eta_k e_k^2 + \eta_k \zeta_k^2 \\ &= -\eta_k e_k^2 + \eta_k \zeta_k^2\end{aligned}$$

De esta forma

$$\begin{aligned}\Delta L_k &\leq \eta_k^2 e_k^2 z_k^2 \left(1 + 4\overline{y}_k^2 \left\|D_{1k}\right\|^2 + 4\overline{y}_k^2 \left\|D_{2k}\right\|^2\right) - \eta_k e_k^2 + \eta_k \zeta_k^2 \\ &= -\eta_k e_k^2 \left[1 - \eta_k \left(z_k^2 + 4\overline{y}_k^2 z_k^2 \left\|D_{1k}\right\|^2 + 4\overline{y}_k^2 z_k^2 \left\|D_{2k}\right\|^2\right)\right] + \eta_k \zeta_k^2\end{aligned}$$

Definimos $\Phi_k = z_k^2 + 4\overline{y}_k^2 z_k^2 \left\|D_{1k}\right\|^2 + 4\overline{y}_k^2 z_k^2 \left\|D_{2k}\right\|^2$, elegimos η_k como $\eta_k = \frac{\eta}{1+\Phi_k}$, $0 < \eta < 1$

$$\begin{aligned}\Delta L_k &\leq -\tfrac{\eta}{1+\Phi_k} e_k^2 \left[1 - \tfrac{\eta}{1+\Phi_k} \Phi_k\right] + \eta_k \zeta_k^2 \\ &\leq -\tfrac{\eta}{1+\Phi_k} e_k^2 \left[1 - \tfrac{1}{1+\Phi_k} \Phi_k\right] + \eta_k \zeta_k^2 \\ &\leq -\pi e_k^2 + \eta \zeta_k^2\end{aligned} \tag{5.10}$$

donde π es definida en (5.7).

Porque

$$n\left[\text{mín}\left(\widetilde{y}_k^2\right) + \text{mín}\left(\widetilde{c}_k\right) + \text{mín}\left(\widetilde{\sigma}_k\right)\right] \leq L_k \leq n\left[\text{máx}\left(\widetilde{y}_k^2\right) + \text{máx}\left(\widetilde{c}_k\right) + \text{máx}\left(\widetilde{\sigma}_k\right)\right]$$

donde $n\left[\text{mín}\left(\widetilde{y}_k^2\right) + \text{mín}\left(\widetilde{c}_k\right) + \text{mín}\left(\widetilde{\sigma}_k\right)\right]$ y $n\left[\text{máx}\left(\widetilde{y}_k^2\right) + \text{máx}\left(\widetilde{c}_k\right) + \text{máx}\left(\widetilde{\sigma}_k\right)\right]$ son funciones-$\mathcal{K}_\infty$, y πe_k^2 es una función-$\mathcal{K}_\infty$, además $\eta \zeta_k^2$ es una función-$\mathcal{K}$. De (5.6) y (5.8) conocemos que L_k es función de e_k y ζ_k, de esta forma L_k admite una función suave ISS-Lyapunov como en

la *Definicion 2*. Del *Teorema 1*, la dinamica de la identificación del error es entrada a estado estable (input-to state stable). Debido a que la "ENTRADA"ζ_k se limita y la dinámica es ISS, el "ESTADO"e_k se limita, (5.10) puede ser reescrita como:

$$\Delta L_k \leq -\pi e_k^2 + \eta \zeta_k^2 \leq \pi e_k^2 + \eta \overline{\zeta} \tag{5.11}$$

donde $\overline{\zeta} = \underset{k}{\text{máx}} \left[\zeta_k^2\right]$. Resumiendo (5.11) de 1 hasta T, y usando $L_T > 0$ y siendo L_1 una constante, obtenemos:

$$\begin{array}{c} L_T - L_1 \leq -\pi \sum_{k=1}^{T} e_k^2 + T\eta\overline{\zeta} \\ \pi \sum_{k=1}^{T} e_k^2 \leq L_1 - L_T + T\eta\overline{\zeta} \leq L_1 + T\eta\overline{\zeta} \end{array}$$

(5.7) es establecida (se establece). ■

Comentario 5.1 *Comparando el entrenamiento de gradiente decendente*

$$\begin{array}{l} \overline{y}^j (k+1) = \overline{y}^j (k) - \eta (\widehat{y} - y) z^j \\ c_i^j (k+1) = c_i^j (k) - 2\eta (\widehat{y} - y) \overline{y}^j z^j \frac{\left(x_i - c_i^j\right)}{\left(\sigma_i^j\right)^2} \\ \sigma_i^j (k+1) = \sigma_i^j (k) - 2 (\widehat{y} - y) \overline{y}^j z^j \frac{\left(x_i - c_i^j\right)^2}{\left(\sigma_i^j\right)^3} \end{array} \tag{5.12}$$

y el aprendizaje estable

$$\begin{array}{l} \overline{y}_{k+1} = \overline{y}_k - \eta_k z_k e_k \\ C_{k+1} = C_k - 2\eta_k \overline{y}_k z_k D_{1k} e_k \\ \Omega_{k+1} = \Omega_k - 2\eta_k \overline{y}_k z_k D_{2k} e_k \end{array} \tag{5.13}$$

La única diferencia es la ganancia de aprendizaje, en (5.12) η es una constante positiva, que debe ser lo bastante pequeña para que el proceso de aprendizaje sea estable. En (5.13) el rango de aprendizaje normalizado η_k está variando en el tiempo para asegurar que el proceso de identificación sea estable. Estas ganancias de aprendizaje son de fácil decisión; no se requiere de ninguna información previa, por ejemplo, podemos seleccionar $\eta = 1$. La contradicción de la rápida convergencia y de un aprendizaje estable puede omitirse. Si seleccionamos η como una función en zona-muerta:

$$\begin{cases} \eta = 0 & if\ |e(k)| \leq \overline{\zeta} \\ \eta = \eta_0 & if\ |e(k)| > \overline{\zeta} \end{cases}$$

(4.8) es la misma que en [72] y [90]. Si un término de modificación $-\sigma$ o un término modificado de regla $-\delta$ se añade en (4.8)se convierte ya sea en [30] o en [45]. Pero todos ellos necesitan de un límite superior del error de modelado $\overline{\zeta}$. Y la identificación del error es agrandada por modificaciones robustas [24]. La ley del aprendizaje (5.12) puede asegurar que los parámetros converjan a un óptimo valor (u óptimo valor local). La ley del aprendizaje (5.13) no puede asegurar que los parámetros converjan a óptimos valores.

5.2. Varias reglas en cada grupo

En el modelo SISO NARMA [9]

$$y(k) = \Psi\left[y\left(k-1\right), y\left(k-2\right), \cdots u\left(k-1\right), u\left(k-2\right), \cdots\right] = \Psi\left[X\left(k\right)\right]$$

donde

$$X\left(k\right) = \left[y\left(k-1\right), y\left(k-2\right), \cdots u\left(k-d\right), u\left(k-d-1\right), \cdots\right]^T \tag{5.14}$$

$\Psi\left(\cdot\right)$ es una ecuación diferencial no-lineal desconocida que representa la dinámica de la planta, $u\left(k\right)$ y $y\left(k\right)$ son entrada y salida escalares medibles, d es el tiempo de retardo. Reglas difusas como $j = 1 \cdots m$

$$\mathrm{R}^j: \text{ IF } x_1 \text{ is } A_1^j \text{ AND} \cdots x_n \text{ is } A_n^j \text{ THEN } y \text{ is } B^j \tag{5.15}$$

Si usamos un fusificador singleton, las aplicaciones Mamdani, y defusificador promedio de centros, la salida pude expresarse como:

$$\widehat{y} = \left(\sum_{j=1}^{m} \overline{y}^j \prod_{i=1}^{n} \mu_{A_i^j}\left(x_i\right)\right) / \left(\sum_{j=1}^{m} \prod_{i=1}^{n} \mu_{A_i^j}\left(x_i\right)\right) \tag{5.16}$$

donde $\mu_{A_i^j}$ es la función de membresía de los conjuntos difusos A_i^j, $\overline{y}^j$ es el punto para el cual $\mu_{B^j} = 1$.Si definimos

$$\phi_j = \prod_{i=1}^{n} \mu_{A_i^j}\left(x_i\right) / \left(\sum_{j=1}^{m} \prod_{i=1}^{n} \mu_{A_i^j}\left(x_i\right)\right)$$

(5.16) que puede expresarse en forma matricial como

$$\widehat{y} = W(k)\,\Phi[X(k)] \tag{5.17}$$

$W(k) = \left[\begin{array}{ccc} \overline{y}^1 & \cdots & \overline{y}^m \end{array}\right]^T$, $\Phi[X(k)] = [\phi_1 \cdots \phi_m]^T$. Seleccionamos la función de membresía como:

$$\mu_{A_i^j}(x_i) = \exp\left[-\left(\frac{x_i - c_i^j}{\sigma_i^j}\right)^2\right]$$

5.2.1. Entrenamiento del gradiente descendente

Sean

$$z^j = \prod_{i=1}^{n} \exp\left[-\left(\frac{x_i - c_i^j}{\sigma_i^j}\right)^2\right], \qquad a = \sum_{j=1}^{m} \overline{y}^j z^j, \qquad b = \sum_{j=1}^{m} z^j$$

Entonces

$$\widehat{y} = \frac{a}{b}$$

Puede transformarse en un problema de identificación para determinar los parámetros $\overline{y}^j$, c_i^j y σ_i^j, el objetivo es encontrar el valor central de B^j, las FM $A_1^j \cdots A_n^j$, tales que $\widehat{y} \to y$. Definamos un índice de representación

$$J = \frac{1}{2}(\widehat{y} - y)^2$$

Usando la regla de la cadena

$$\frac{\partial J}{\partial \overline{y}^j} = \frac{\partial J}{\partial \widehat{y}} \frac{\partial \widehat{y}}{\partial \overline{y}_j} = (\widehat{y} - y) \frac{\partial\left(\frac{a}{b}\right)}{\partial \overline{y}^j} = \frac{(\widehat{y} - y)}{b} z^j$$

Gradiente

$$\overline{y}^j(k+1) = \overline{y}^j(k) - \eta \frac{(\widehat{y} - y)}{b} z^j$$

desde $\frac{\partial e^x}{\partial x} = e^x$

$$\begin{aligned} \frac{\partial J}{\partial c_i^j} &= \frac{\partial J}{\partial \widehat{y}} \frac{\partial \widehat{y}}{\partial z^j} \frac{\partial z^j}{\partial c_i^j} = (\widehat{y} - y) \frac{\partial\left(\frac{a}{b}\right)}{\partial z^j} \frac{\partial \prod_{i=1}^{n} \exp\left[-\left(\frac{x_i - c_i^j}{\sigma_i^j}\right)^2\right]}{\partial c_i^j} \\ &= (\widehat{y} - y)\left[\frac{\overline{y}^j}{b} - \frac{a}{b^2}\right] z^j \left[2\frac{\left(x_i - c_i^j\right)}{\left(\sigma_i^j\right)^2}\right] \end{aligned}$$

Entonces

$$c_i^j(k+1) = c_i^j(k) - 2\eta(\widehat{y}-y)\, z^j \frac{\left(\overline{y}^j - \widehat{y}\right)\left(x_i - c_i^j\right)}{b\left(\sigma_i^j\right)^2}$$

Similarmente

$$\frac{\partial J}{\partial \sigma_i^j} = \frac{\partial J}{\partial \widehat{y}}\frac{\partial \widehat{y}}{\partial z^j}\frac{\partial z^j}{\partial \sigma_i^j} = (\widehat{y}-y)\frac{\partial\left(\frac{a}{b}\right)}{\partial z^j}\prod_{i=1}^{n}\exp\left[-\left(\frac{x_i - c_i^j}{\sigma_i^j}\right)^2\right]\left[2\frac{\left(x_i - c_i^j\right)^2}{\left(\sigma_i^j\right)^3}\right]$$

$$\sigma_i^j(k+1) = \sigma_i^j(k) - 2\eta(\widehat{y}-y)\, z^j \frac{\left(\overline{y}^j - \widehat{y}\right)\left(x_i - c_i^j\right)^2}{b\left(\sigma_i^j\right)^3}$$

Definimos $e(k) = \widehat{y}(k) - y(k)$, el entrenamiento de gradiente descendente es

$$\begin{aligned}
\overline{y}^j(k+1) &= \overline{y}^j(k) - \eta z^j(k)\tfrac{1}{b(k)}e(k)\\
c_i^j(k+1) &= c_i^j(k) - 2\eta z^j(k)\frac{\left(\overline{y}(k)^j - \widehat{y}(k)\right)\left(x_i(k) - c_i^j(k)\right)}{b(k)\left(\sigma_i^j(k)\right)^2}e(k)\\
\sigma_i^j(k+1) &= \sigma_i^j(k) - 2\eta z^j(k)\frac{\left(\overline{y}(k)^j - \widehat{y}(k)\right)\left(x_i(k) - c_i^j(k)\right)^2}{b(k)\left(\sigma_i^j(k)\right)^3}e(k)
\end{aligned} \tag{5.18}$$

5.2.2. Aprendizaje estable

Asumiendo que una planta no-lineal puede expresarse en una FM Gaussiana como

$$y = \left(\sum_{j=1}^{m}\overline{y}^{j*}\prod_{j=1}^{n}\exp\left[-\left(\frac{x_i - c_i^{j*}}{\sigma_i^{j*}}\right)^2\right]\right) / \left(\sum_{j=1}^{m}\prod_{j=1}^{n}\exp\left[-\left(\frac{x_i - c_i^{j*}}{\sigma_i^{j*}}\right)^2\right]\right) - \mu$$

donde $\overline{y}^{j*}$, c_i^{j*} y σ_i^{j*} son parámetros desconocidos que pueden minimizar el error de modelado μ. En el caso de tres variables independientes, la fórmula de Taylor tiene una función suave f como

$$f(x_1, x_2, x_3) = \sum_{k=0}^{l-1}\frac{1}{k!}\left[\left(x_1 - x_1^0\right)\frac{\partial}{\partial x_1} + \left(x_2 - x_2^0\right)\frac{\partial}{\partial x_2} + \left(x_3 - x_3^0\right)\frac{\partial}{\partial x_3}\right]_0^k f + R_l$$

donde R_l es el residuo de la fórmula de Taylor. Si hacemos que x_1, x_2, x_3 correspondan $\overline{y}^j$, c_i^j y σ_i^j, x_1^0, x_2^0, x_3^0 correspondan a $\overline{y}^{j*}$, c_i^{j*} y σ_i^{j*},

$$\widehat{y} = y + \mu + \sum_{j=1}^{m}\left(\overline{y}^j - \overline{y}^{j*}\right)\frac{z^j}{b} + \sum_{j=1}^{m}\sum_{i=1}^{n}\frac{\partial \widehat{y}}{\partial c_i^j}\left(c_i^j - c_i^{j*}\right) + \sum_{j=1}^{m}\sum_{i=1}^{n}\frac{\partial \widehat{y}}{\partial \sigma_i^j}\left(\sigma_i^j - \sigma_i^{j*}\right) + R_1$$

donde R_1 es una aproximación de segundo orden del error de la serie de Taylor,

$$\frac{\partial \widehat{y}}{\partial c_i^j} = 2z^j(k)\frac{\left(\overline{y}(k)^j - \widehat{y}(k)\right)\left(x_i(k) - c_i^j(k)\right)}{b(k)\left(\sigma_i^j(k)\right)^2}$$
$$\frac{\partial \widehat{y}}{\partial \sigma_i^j} = 2z^j(k)\frac{\left(\overline{y}(k)^j - \widehat{y}(k)\right)\left(x_i(k) - c_i^j(k)\right)^2}{b(k)\left(\sigma_i^j(k)\right)^3}$$

Entonces

$$\sum_{j=1}^{m}\sum_{i=1}^{n}\frac{\partial \widehat{y}}{\partial c_i^j}\left(c_i^j - c_i^{j*}\right) = \frac{2}{b(k)}D_1^T(k)\widetilde{C}(k)$$
$$\sum_{j=1}^{m}\sum_{i=1}^{n}\frac{\partial \widehat{y}}{\partial \sigma_i^j}\left(\sigma_i^j - \sigma_i^{j*}\right) = \frac{2}{b(k)}D_2^T(k)\widetilde{\Omega}(k)$$

donde

$$\widetilde{C}(k) = \left[(c_1^1 - c_1^{1*})\cdots(c_n^1 - c_n^{1*})\cdots(c_1^m - c_1^{m*})\cdots(c_n^m - c_n^{m*})\right]^T \in R^{n+m}$$
$$D_1(k) = \begin{bmatrix} z^1(k)\frac{\left(\overline{y}(k)^1 - \widehat{y}(k)\right)\left(x_1(k) - c_1^1(k)\right)}{\left(\sigma_1^1(k)\right)^2}\cdots z^1(k)\frac{\left(\overline{y}(k)^1 - \widehat{y}(k)\right)\left(x_n(k) - c_n^1(k)\right)}{\left(\sigma_n^1(k)\right)^2} \\ \cdots z^m(k)\frac{\left(\overline{y}(k)^m - \widehat{y}(k)\right)\left(x_1(k) - c_1^m(k)\right)}{\left(\sigma_1^m(k)\right)^2}\cdots z^m(k)\frac{\left(\overline{y}(k)^m - \widehat{y}(k)\right)\left(x_n(k) - c_n^m(k)\right)}{\left(\sigma_n^1(k)\right)^2} \end{bmatrix}^T$$
$$\widetilde{\Omega}(k) = \left[(\sigma_1^1 - \sigma_1^{1*})\cdots(\sigma_n^1 - \sigma_n^{1*})\cdots(\sigma_1^m - \sigma_1^{m*})\cdots(\sigma_n^m - \sigma_n^{m*})\right]^T \in R^{n+m}$$
$$D_2(k) = \begin{bmatrix} z^1(k)\frac{\left(\overline{y}(k)^1 - \widehat{y}(k)\right)\left(x_1(k) - c_1^1(k)\right)^2}{\left(\sigma_1^1(k)\right)^3}\cdots z^1(k)\frac{\left(\overline{y}(k)^1 - \widehat{y}(k)\right)\left(x_n(k) - c_n^1(k)\right)^2}{\left(\sigma_n^1(k)\right)^3} \\ \cdots z^m(k)\frac{\left(\overline{y}(k)^m - \widehat{y}(k)\right)\left(x_1(k) - c_1^m(k)\right)^2}{\left(\sigma_1^m(k)\right)^3}\cdots z^m(k)\frac{\left(\overline{y}(k)^m - \widehat{y}(k)\right)\left(x_n(k) - c_n^m(k)\right)^2}{\left(\sigma_n^m(k)\right)^3} \end{bmatrix}^T$$

$$e(k) = \widehat{y}(k) - y(k) = \frac{1}{b(k)}z^T(k)\widetilde{y}(k) + \frac{2}{b(k)}D_1^T(k)\widetilde{C}(k) + \frac{2}{b(k)}D_2^T(k)\widetilde{\Omega}(k) + \zeta(k) \quad (5.19)$$

donde $\widetilde{y}_k = \overline{y}(k) - \overline{y}^*(k)$, $\overline{y}(k) = [\overline{y}^1, \cdots, \overline{y}^m]^T$, $z(k) = [z^1 \cdots z^m]^T$, $\zeta(k) = R_1 + \mu$.

Teorema 5.2 *Si usamos una red neuronal difusa del tipoMamdani (4.10) para identificar una planta nolineal (5.1), el siguiente algoritmo de retropropagación identifica al error $e(k)$ limitado*

$$\begin{aligned} \overline{y}(k+1) &= \overline{y}(k+1) - \frac{\eta(k)}{b(k)}z(k)e(k) \\ C(k+1) &= C(k) - 2\frac{\eta(k)}{b(k)}D_1(k)e(k) \\ \Omega(k+1) &= \Omega(k) - 2\frac{\eta(k)}{b(k)}D_2(k)e(k) \end{aligned} \quad (5.20)$$

$\eta(k) = \dfrac{\eta}{1+\Phi(k)^2}$, $\Phi(k) = \|z(k)\|^2 + 4\|D_1(k)\|^2 + 4\|D_2(k)\|^2$, $0 < \eta \leq \max_k \{b^2(k)\}$. *El promedio de la identificación del error satisface*

$$J = \limsup_{T\to\infty} \frac{1}{T}\sum_{k=1}^{T} e^2(k) \leq \frac{1}{\pi}\overline{\zeta} \tag{5.21}$$

donde $\pi = \frac{\eta}{(1+\Phi(k))^2} > 0$, $\overline{\zeta} = \max_k \left[\zeta^2(k)\right]$

Demostración. Seleccionamos un escalar positivo L_k como

$$L_k = \|\widetilde{y}(k)\|^2 + \left\|\widetilde{C}(k)\right\|^2 + \left\|\widetilde{\Omega}(k)\right\|^2 \tag{5.22}$$

La ley de actualización (5.5) puede escribirse como

$$\begin{aligned}
\widetilde{y}(k+1) &= \widetilde{y}(k) - \tfrac{\eta(k)}{b(k)} z(k)\, e(k) \\
\widetilde{C}(k+1) &= \widetilde{C}(k) - 2\tfrac{\eta(k)}{b(k)} D_1(k)\, e(k) \\
\widetilde{\Omega}(k+1) &= \widetilde{\Omega}(k) - 2\tfrac{\eta(k)}{b(k)} D_2(k)\, e(k)
\end{aligned}$$

Luego, tenemos

$$\begin{aligned}
\Delta L_k &= \left\|\widetilde{y}(k) - \tfrac{\eta(k)}{b(k)} z(k)\, e(k)\right\|^2 - \|\widetilde{y}(k)\|^2 \\
&+ \left\|\widetilde{C}(k) - 2\tfrac{\eta(k)}{b(k)} D_1(k)\, e(k)\right\|^2 - \left\|\widetilde{C}(k)\right\|^2 \\
&+ \left\|\widetilde{\Omega}(k) - 2\tfrac{\eta(k)}{b(k)} D_2(k)\, e(k)\right\|^2 - \left\|\widetilde{\Omega}(k)\right\|^2 \\
&= \eta^2(k) \left\|\tfrac{z(k)}{b(k)}\right\|^2 e^2(k) - 2\eta(k)\, \tfrac{z^T(k)\widetilde{y}(k)}{b(k)} e(k) \\
&+ 4\eta^2(k) \left\|\tfrac{D_1(k)}{b(k)}\right\|^2 e^2(k) - 4\eta(k)\, \tfrac{D_1^T(k)\widetilde{C}(k)}{b(k)} e(k) \\
&+ 4\eta^2(k) \left\|\tfrac{D_2(k)}{b(k)}\right\|^2 e^2(k) - 4\eta(k)\, \tfrac{D_1^T(k)\widetilde{\Omega}(k)}{b(k)} e(k) \\
&= \tfrac{\eta^2(k)}{b^2(k)} e^2(k) \left(\|z(k)\|^2 + 4\|D_1(k)\|^2 + 4\|D_2(k)\|^2\right) \\
&- 2\eta(k)\, e(k) \left(\tfrac{1}{b(k)} z^T(k)\, \widetilde{y}(k) + \tfrac{2}{b(k)} D_1^T(k)\, \widetilde{C}(k) + \tfrac{2}{b(k)} D_1^T(k)\, \widetilde{\Omega}(k)\right)
\end{aligned} \tag{5.23}$$

Debido a que $e(k) = \frac{1}{b(k)} z^T(k)\, \widetilde{y}(k) + \frac{2}{b(k)} D_1^T(k)\, \widetilde{C}(k) + \frac{2}{b(k)} D_2^T(k)\, \widetilde{\Omega}(k) + \zeta(k)$, el último término de (5.23) es $-2\eta(k)\, e(k)\, [e(k) - \zeta(k)]$. Porque $\eta_k > 0$

$$\begin{aligned}
&-2\eta(k)\, e(k)\, [e(k) - \zeta(k)] = -2\eta(k)\, e^2(k) + 2\eta(k)\, e(k)\, \zeta(k) \\
&\leq -2\eta(k)\, e^2(k) + \eta(k)\, e^2(k) + \eta(k)\, \zeta^2(k) \\
&= -\eta(k)\, e^2(k) + \eta(k)\, \zeta^2(k)
\end{aligned}$$

Entonces

$$\begin{aligned}\Delta L_k &\leq \tfrac{\eta^2(k)}{b^2(k)} e^2(k)\left(\|z(k)\|^2 + 4\|D_1(k)\|^2 + 4\|D_2(k)\|^2\right) - \eta(k)e^2(k) + \eta(k)\zeta^2(k)\\ &= -\eta(k)e^2(k)\left[1 - \tfrac{\eta(k)}{b^2(k)}\left(\|z(k)\|^2 + 4\|D_1(k)\|^2 + 4\|D_2(k)\|^2\right)\right] + \eta_k\zeta_k^2\end{aligned}$$

Definimos $\Phi(k) = \|z(k)\|^2 + 4\|D_1(k)\|^2 + 4\|D_2(k)\|^2$, y elegimos $\eta(k)$ como $\eta(k) = \frac{\eta}{1+\Phi_k} \leq 1$. Porque $\eta \leq \min_k \{b(k)\}$, $\frac{\eta}{b^2(k)} \leq \frac{\max_k\left\{b^2(k)\right\}}{b^2(k)} \leq 1$

$$\begin{aligned}\Delta L_k &\leq -\tfrac{\eta}{1+\Phi(k)} e^2(k)\left[1 - \tfrac{\eta}{b^2(k)}\tfrac{\Phi(k)}{1+\Phi(k)}\right] + \eta(k)\zeta^2(k)\\ &\leq -\tfrac{\eta}{1+\Phi(k)} e^2(k)\left[1 - \tfrac{1}{1+\Phi(k)}\Phi(k)\right] + \eta(k)\zeta^2(k)\\ &\leq -\pi e^2(k) + \zeta^2(k)\end{aligned} \tag{5.24}$$

donde π es definido en (5.21). Debido a $\|\widetilde{y}(k)\|^2 + \left\|\widetilde{C}(k)\right\|^2 + \left\|\widetilde{\Omega}(k)\right\|^2$

$$n\left[\min\left(\widetilde{y}(k)\right) + \min\left(\widetilde{c}(k)\right) + \min\left(\widetilde{\sigma}(k)\right)\right] \leq L_k \leq n\left[\max\left(\widetilde{y}(k)\right) + \max\left(\widetilde{c}(k)\right) + \max\left(\widetilde{\sigma}(k)\right)\right]$$

donde

$$\begin{aligned}&n\left[\min\left(\widetilde{y}(k)\right) + \min\left(\widetilde{c}(k)\right) + \min\left(\widetilde{\sigma}(k)\right)\right]\\ &n\left[\max\left(\widetilde{y}(k)\right) + \max\left(\widetilde{c}(k)\right) + \max\left(\widetilde{\sigma}(k)\right)\right]\end{aligned}$$

son funciones-$\mathcal{K}_\infty$, y πe_k^2 es una función-$\mathcal{K}_\infty$, $\zeta^2(k)$ es una función-$\mathcal{K}$. De (**??**) y (5.8) sabemos que L_k es función de $e(k)$ y ζ_k, entonces L_k admite una función ISS-Lyapunov suave como en la *Definición 2*. Del *Teorema 1*, la dinámica de la identificación del error es input-to-state estable. Debido a que "INPUT" ζ_k está limitada y su dinámica es ISS, el "STATE" e_k es limitado.

(5.24) puede escribirse como

$$\Delta L_k \leq -\pi e_k^2 + \eta\zeta_k^2 \leq \pi e_k^2 + \overline{\zeta}$$

donde $\overline{\zeta} = \max_k\left[\zeta_k^2\right]$. Sumando (5.11) desde1 hasta T, y usando $L_T > 0$ y siendo L_1 una constante, obtenemos

$$\begin{aligned}L_T - L_1 &\leq -\pi\textstyle\sum_{k=1}^{T} e_k^2 + T\eta\overline{\zeta}\\ \pi\textstyle\sum_{k=1}^{T} e_k^2 &\leq L_1 - L_T + T\eta\overline{\zeta} \leq L_1 + T\eta\overline{\zeta}\end{aligned}$$

(5.7) es establecida. ■

5.3. Simulación

Realizar las fases tanto de identificación de la estructura como la estimación de parámetros para generar un sistema de inferencia difuso que sirva para modelar el comportamiento de un sistema no-lineal, parece ser un poco complejo ya que depende en gran parte de los algoritmos o procesos utilizados para generar las reglas, las funciones de membresia y para la optimización y determinación de los parametros no conocidos. Por lo tanto, en esta sección proponemos un procedimiento que nos puede ayudar a resolver este problema.

En primer lugar para realizar la extracción de las reglas y las FMs antecedentes (Fase de identificación de la estructura) planteamos la utilización del algoritmo de agrupamiento doble propuesto en la sección 4.2. Por ejemplo en la figura (5.1) se muestra la aplicación de este algoritmo para un periodo de tiempo $t = 100$.El número de reglas generado por el método de agrupamiento depende grandemente de los valores del radio (Raddi) elegido para cada una de las etapas. Para la primer etapa recomentados una valor del radio pequeño ($raddi_1 = 0{,}2$) con el fin de generar la mayor cantidad de grupos posible que nos permitan capturar la estuctura de los datos, ya que cada centro de grupo generará una reglas, el numero de reglas resultante de esta primer etapa es demasiado grande. En la segunda etapa el valor del radio seleccionado es de una importancia mayor, ya que nos permitirá reducir en gran parte el número de grupos (centros de grupo redundantes) y asu vez el número de reglas. Entre mayor sea el valor del radio, menor será el numero de reglas generadas lo que parece ser ideal, sin embrago un número pequeño de reglas puede reducir la presición del sistema de inferencia difuso ya que aumenta el error en la aproximación del sistema a modelar. Otro factor importante a considerar para la generación de las reglas, es el número de datos muestreados y almacenados en el periodo de tiempo seleccionado para el agrupamiento, ya que entre mayor sea el número de datos mayor será el numero de grupos generados en la primer etapa lo que ocasionará que se tenga que elegir un radio mayor en la segunda etapa del agrupamiento para reducir hasta donde sea posible el número de reglas (ver tabla 5.1).

Ya que estamos aplicando agrupamiento substractivo, la extracción de las FMs antecedentes se puede realizar de la manera propuesta en la sección 4.3. En la figura (5.2) se

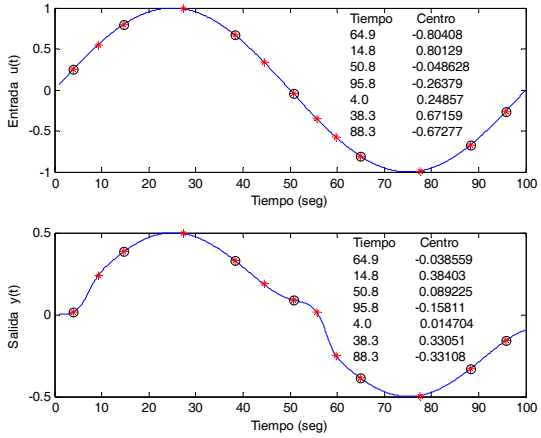

Figura 5.1: Agrupamiento Substractivo Doble: (*) Centros de grupo obtenidos en la primer etapa. (o) Centros de grupo obtenidos en la segunda etapa.

muestran las funciones de membresia extraídas de los centros de grupo obtenidos para los datos de entrada-salida de la figura (5.1). Como se puede observar en la figura, los centros fueron proyectados en dos ejes del espacio multidimensional: el tiempo y la entrada; esto debido a la correlación existente entre el tiempo y los datos de entrada-salida ya que se trata de un algoritmo de agrupamiento en línea. Mientras que el anchos de las FMs son establecidos en base al radio de vecindad r_a,definido durante el agrupamiento.

Utilizando estas funciones de membresía podemos generar reglas del tipo de (**??**), de la siguiente manera:

1. $IF\ In1$ es fm_1 and $In2$ es fm_1 THEN $f_1 = p_1 In1 + q_1 In2 + r_1$
2. $IF\ In1$ es fm_2 and $In2$ es fm_2 THEN $f_2 = p_2 In1 + q_2 In2 + r_2$
3. $IF\ In1$ es fm_3 and $In2$ es fm_3 THEN $f_3 = p_3 In1 + q_3 In2 + r_3$
4. $IF\ In1$ es fm_4 and $In2$ es fm_4 THEN $f_4 = p_4 In1 + q_4 In2 + r_4$
5. $IF\ In1$ es fm_5 and $In2$ es fm_5 THEN $f_5 = p_5 In1 + q_5 In2 + r_5$

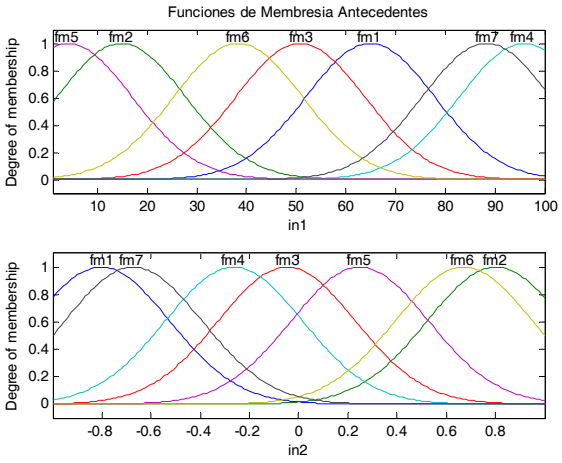

Figura 5.2: Funciones de membresía antecedentes formadas con los centros obtenidos para los datos de entada-salida mostrados en la figura (5.1)

6. $IF\ In1$ es fm_6 and $In2$ es fm_6 THEN $f_6 = p_6 In1 + q_6 In2 + r_6$

7. $IF\ In1$ es fm_7 and $In2$ es fm_7 THEN $f_7 = p_7 In1 + q_7 In2 + r_7$

En segundo lugar planteamos utilizar la estimación lineal de mínimos cuadrados para determinar la ecuación consecuente de cada una de las reglas. Por lo que se obtienen las ecuaciones siguientes:

$f_1 = -0{,}016688\ In1 + 0{,}47917\ In2 + 0{,}73453$ para la regla 1.
$f_2 = -0{,}16169\ In1 + 5{,}2218\ In2 - 4{,}9506$ para la regla 2
$f_3 = 3{,}0298\ In1 + 47{,}16\ In2 - 151{,}47$ para la regla 3
$f_4 = 1{,}5146\ In1 - 23{,}216\ In2 - 151{,}89$ para la regla 4
$f_5 = 2{,}0264\ In1 - 32{,}29\ In2 + 0{,}15269$ para la regla 5
$f_6 = -0{,}79406\ In1 - 10{,}824\ In2 + 38{,}952$ para la regla 6
$f_7 = -0{,}12208\ In1 + 3{,}4249\ In2 + 13{,}409$ para la regla 7

Con esto, el resultado final es un sistema de inferencia difuso de tipo-Sugeno de primer orden que contiene un conjunto de reglas que cubren el espacio caracteristico (ver figura 5.3).

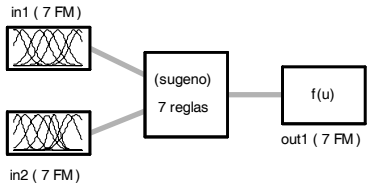

Figura 5.3: Sistema de Inferencia Difuso de Tipo Sugeno de 1er Orden: 2 Entradas, 1 Salida, 7 Reglas.

También podemos generar un sistema de inferencia difuso de tipo Mandani con reglas del tipo (5.15). Para esto, sólo tenemos que proyectar los centros de los grupo obtenidos en un eje del espacio dimensional, más la salida.

En tercer lugar, para determinar los parámetros no conocidos en el modelo o para optimizar los ya existentes realizamos la fase de estimación de parámetros. Para esto, utilizando el sistema de inferencia difuso creado de manera inicial y los datos originales como datos de entrenamiento, proponemos utilizar el entrenamiento de gradiente decendente visto en la sección 4.2, para optimizar (ajustar) el centro y el ancho de cada una de las FMs. En las figuras (5.4 y 5.5) se muestran las FMs resultantes. Como se puede observar en las figuras, los parámetros de las FMs de la entrada 1 no sufrieron modificación alguna ya que están en función del tiempo; mientras que los parámetros de las FMs de la entrada 2 si fueron ajustados.

El planteamiento que realizamos en los párrafos anteriores fue aplicado al sistema representado por los datos de la figura (5.1) para diferentes periodos de tiempo (t), los resultados obtenidos son mostrados en las figuras (5.6-5.9). Como se puede observar en estos resultados, el error en la aproximación es menor cuando $t \leq 100\ seg$; sin importar el sistema de inferencia utilizado. Cuando $100\ seg < t < 250\ seg$, el error obtenido tanto con el SIF inicial como el obtenido con el SIF final, crece un poco, aumenta considerablemente; esto debido a que cuando aumenta t, aumenta grandemente el número de datos por lo que el número de reglas obtenido en la primer etapa de agrupamiento es grande y el valor del radio (raddi) para la segunda etapa resulta ya no ser adeacuado aún cuando se maneja entre el rango

recomendado

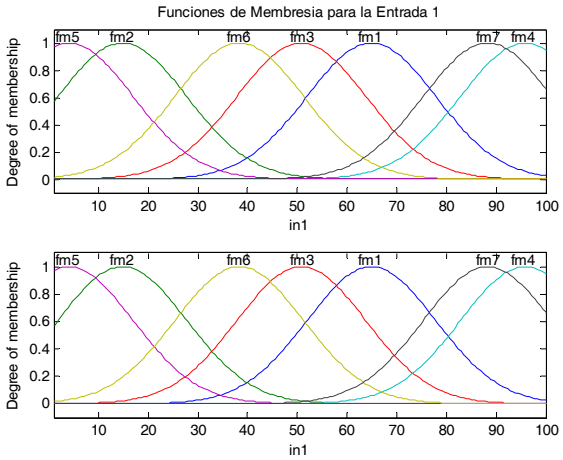

Figura 5.4: Funciones de membresía para la entrada 1: (arriba) funciones de membresía iniciales. (abajo) funciones de membresía optimizadas con gradiente descendente.

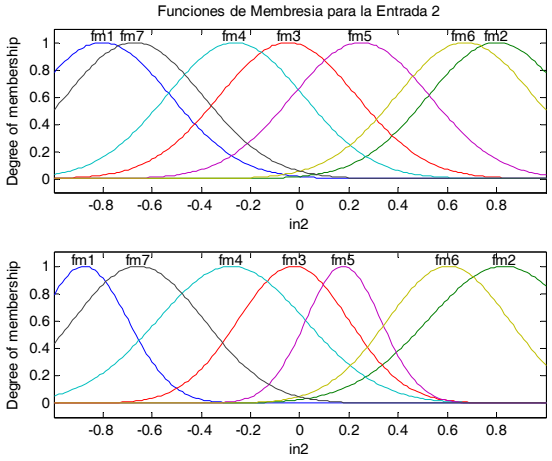

Figura 5.5: Funciones de membresía para la entrada 2: (arriba) funciones de membresía iniciales. (abajo) funciones de membresía optimizadas con gradiente descendente.

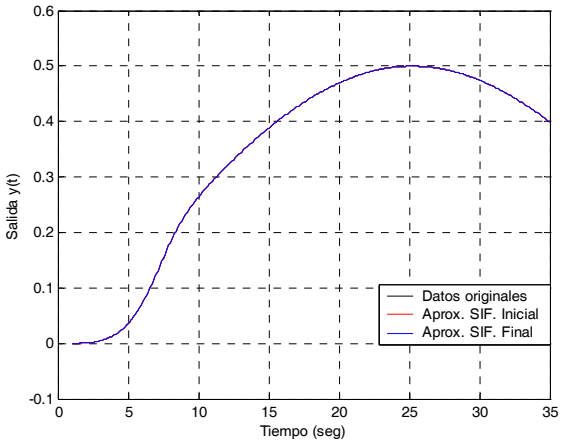

Figura 5.6: Evaluación de los Sistemas de Inferencia Difusos generados, para un periodo de tiempo de 35 seg.

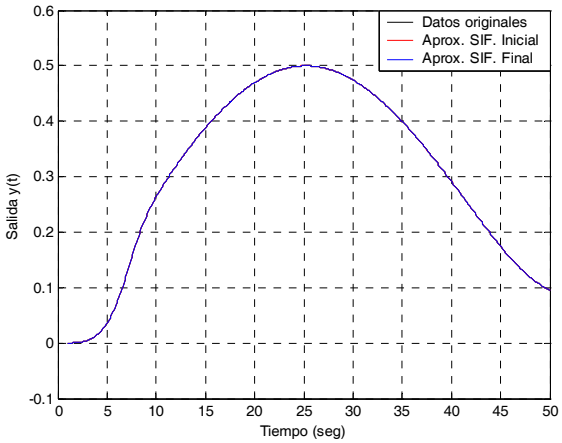

Figura 5.7: Evaluación de los Sistemas de Inferencia Difusos generados, para un periodo de tiempo de 50 seg.

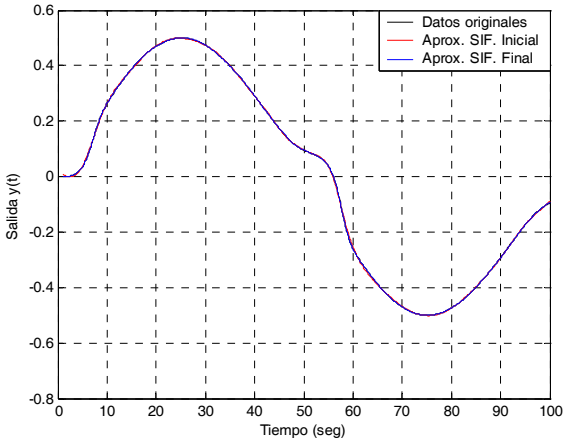

Figura 5.8: Evaluación de los Sistemas de Inferencia Difusos generados, para un periodo de tiempo de 100 seg.

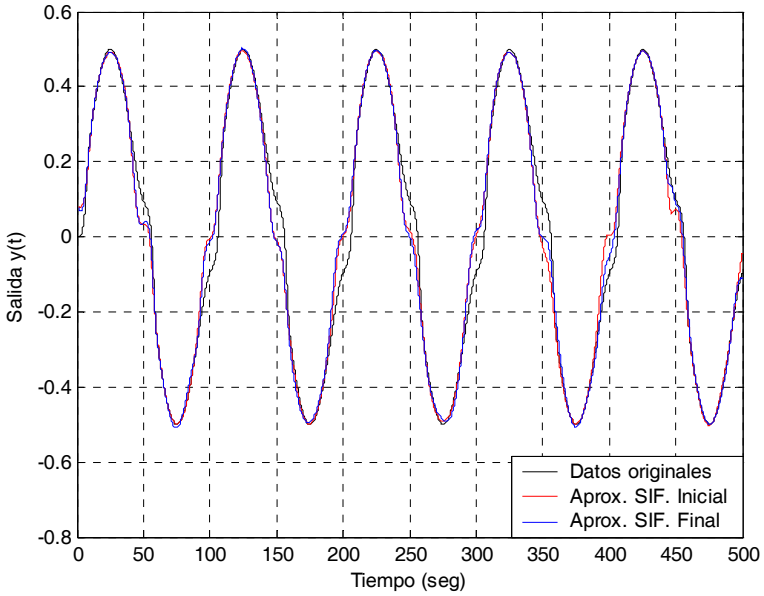

Figura 5.9: Evaluación de los Sistemas de Inferencia Difusos generados, para un periodo de tiempo de 500 seg.

T(seg)	N. Datos	Etapa 1	Etapa 2		
		N. Reglas	Raddi	Num. Reglas	Error RMS
50	500	12	0.3	12	0.000014639
			0.4	9	0.000018593
			0.5	5	0.00041764
			0.6	5	0.00046226
			0.7	5	0.00013562
			0.8	3	0.00038191
100	1000	13	0.3	13	0.00016166
			0.4	7	0.00041829
			0.5	5	0.0058765
			0.6	5	0.0064267
			0.7	3	0.015675
			0.8	2	0.024699
150	1500	20	0.3	14	0.0003881
			0.4	9	0.00068982
			0.5	5	0.0079519
			0.6	4	0.01256
			0.7	3	0.019934
			0.8	3	0.02117
200	2000	26	0.3	15	0.0010628
			0.4	8	0.0051532
			0.5	8	0.0081872
			0.6	5	0.014903
			0.7	4	0.025715
			0.8	3	0.040313
250	2500	27	0.3	18	0.0032724
			0.4	12	0.0052088
			0.5	8	0.014155
			0.6	6	0.026951
			0.7	4	0.050107
			0.8	4	0.050165

Cuadro 5.1: Table Caption

Según lo expuesto en las secciones anteriores, sobre todo por los algoritmos que se utilizan para la generación tanto de funciones de membresia como de las reglas, la *identificación de la estructura* implica encontrar las variables de entrada importantes de todas las posibles variables de entrada, especificando las funciones de membresía, la partición del espacio de entrada, y determinando el número de reglas difusas.

La *estimación de parámetros* implica la determinación de parámetros no conocidos en el modelo utilizando algún método de optimización basado tanto en la información lingüística obtenida del humano experto como de los datos numéricos obtenidos del sistema actual a ser modelado. La especificación de la estructura y la estimación de parametros están interelacionados, y ninguna de ellas puede ser identificada independientemente sin considerar al otro. En las secciones siguientes se discuten las dos fases para la identificación de un modelo difuso. Como hemos visto en las secciones anteriores, la identificación de un FS que consta basicamente de la identificación de la estructura.

Capítulo 6

Modelado via redes neuronales difusas: aplicación al sensor EMI

Actualmente los espectrómetros de movilidad ionica (EMI) estandar disponibles son instrumentos de presición costosos utilisados para detectar concentraciones de productos químicos en el aire. Han encontrado aplicaciones exitosas en la detección de gases militares, narcóticos, y contaminantes, y se han establecido como herramientas útiles de la química analítica. En estos dispositivos, los iones se inyectan en un tubo de arrastres (drift) y su movilidad es medida exactamente al medir el tiempo de su paso a través de la región de arrastre en la presencia de un campo eléctrico. Estos requieren fuentes de alimentación de alto voltaje y campos eléctricos muy uniformes. Su relativa complejidad, tamaño, y costo han limitado hasta ahora muchas de sus aplicaciones del campo, y muchas investigaciones y esfuersos de desarrollo se han concentrado en la reducción de estas limitaciones.

Una metodología más simple para el analisis de movilidad ionica ha existido desde el final del siglo pasado. Este metodo es conocido como el metodo de aspiración, y ha sido utilizado en una gran variedad de configuraciones de dispositivos y aplicaciones. Este método requiere un diseño mucho más simple y barato del instrumento (con la simplicidad de un detector de humo), aunque tiene algunas limitaciones inherentes en la forma de medida. Los dispositivos de aspiración tipicamente tienen resoluciones muy pobres de la movilidad

ionica, normalmente se ajustan en un voltaje específico y proporcionan una medida integral de la carga de movilidad ionica en la muestra, pero no ofrecen un espectro verdadero de las movilidades iónicas.

6.1. Espectrometría de Movilidad Iónica (EMI)

La EMI (o cromatografia de plasma, como fue originalmente llamada) fue introducida a finales de los 1960's como un metodo para la detección de concentraciones pequeñas de compuestos orgánicos en aire y otros gases. El término de espectrometría de movilidad ionica se refiere a los principios, la práctica y la instrumentación para analizar y caracterizar substancias químicas a través de la movilidad de sus iones cuando éstas se encuentran en estado gaseoso. Las movilidades ionicas, son determinadas de las velocidades iónicas que son medidas en un tubo de derivación con soporte electrónico (ver figura 6.1), éstas son características de las sustancias y proporcionan una manera de detectar e identificar gases [17].

En un Espectrómetro de Movilidad Iónica (ver figura 6.2), la muestra de gas se introduce de manera constante a la *región de reacción* o también llamada *cámara de ionización* del tubo de derivación, en donde las moléculas neutras de la muestra de gas son ionizadas por una *fuente de radiación*. En el proceso de ionización, las moléculas reactantes (O2, N2, y H2O) son bombardeadas por electrones de alta energía - los cuales son emitidos por la fuente radiactiva - lo que provoca que la mayoría de las moléculas se fragmenten y se conviertan en *iones reactantes*. El campo eléctrico, causa que los iones de una sola polaridad residan en la región de ionización; los iones de polaridad opuesta son sacados de la región. Los iones reactantes por un proceso de colisión sufren transferencias de carga o de protón, esto durante su fase gaseosa, dando con esto vida a unas partículas analíticas que reciben el nombre de *iones producto* (iones relativamente estables de diversas especies); este proceso tiene lugar también en la región de ionización. Para una muestra de una composición dada que es ionizada, bajo condiciones dadas de la temperatura y de la presión, los grupos de iones formados alcanzarán un equilibrio de la composición y de la cantidad que es característica

a la muestra. Para realizar el análisis de movilidad, los iones producto son inyectados a la *región de derivación* como un paquete de iones, mediante el *obturador de iones*. Cuando los iones entran a la región de derivación, se empiezan a mover bajo la influencia del campo eléctrico aplicado hasta alcanzar una velocidad constante (velocidad de derivación: v_d) que está en función del campo eléctrico y de la movilidad de estos iones en el gas derivado

$$v_d = EK$$

Donde K es la movilidad ionica y está relacionada con la sección transversal, el tamaño y la carga de la molecula del ión.

Al empezarse a mover, los iones empiezan también a separarse de acuerdo a su velocidad; es decir cuando sus movilidades empiezan a ser lo suficientemente diferentes. La derivación y separación de los iones continúa y éstos se van acercando cada vez más al *plato colector* que se encuentra al final del tubo de derivación. Cuando los iones se acercan al plato detector, son acelerados por el campo eléctrico que establece la *rejilla de apertura* (ver figura 6.2), hasta que chocan con el detector produciéndose con esto una corriente eléctrica. Finalmente, se genera un espectro cuyo trazo no es más que la corriente iónica medida por el detector en función del tiempo de derivación; que es el tiempo que le toma a un ión específico el atravesar la región de derivación

El espectrómetro de movilidad iónica proporciona un método directo para determinar experimentalmente, la distribución de la movilidad iónica de una muestra de gas ionizada; la cual es característica de la composición de la muestra de gas. Esto permite que la detección de sustancias, pueda ser realizada por medio de la identificación de una especie de iones particular a una movilidad dada; de manera tal que, la concentración de la sustancia puede ser relacionada a la corriente medida para esa movilidad particular. Esto es posible sólo si la concentración de las moléculas analíticas no excede el límite de saturación, y el gas portador permanece constante ya que, las especies de iones producto son determinadas tanto por el gas portador como por las moléculas analíticas. Por esta razón las aplicaciones prácticas de esta tecnología tiene que ser limitada a la medida de concentraciones de trazo de sustancias orgánicas en aire o gases puros.

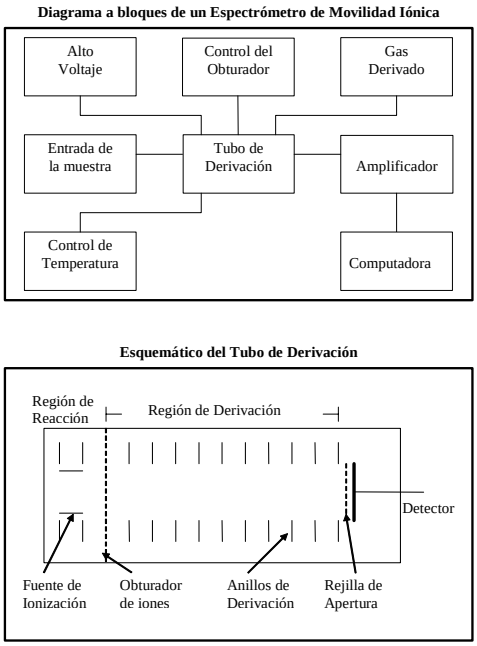

Figura 6.1: Diagrama a bloques de un Espectrómetro de Movilidad Iónica con un esquema del tubo de derivación [17]. El tubo de derivación esta compuesto de una región de reacción y una región de derivación, ambas bajo un gradiente de un campo eléctrico. Un obturador de iones es usado para contener los iones en la región de reacción y para inyectarlos a la región de derivación para su caracterización.

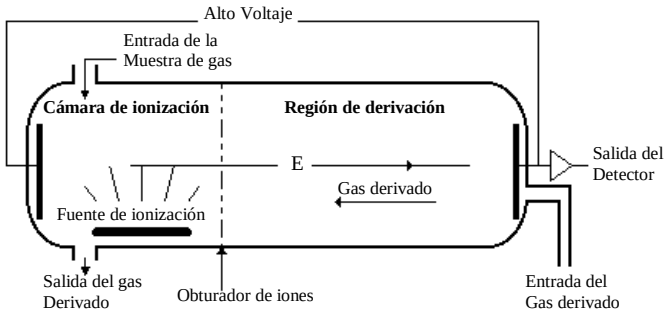

Figura 6.2: Diagrama esquemático del tubo de derivación de un espectrómetro de movilidad iónica [70]. Iones de diferente tamaño y carga arriban al detector en diferentes tiempos. El tiempo de derivación de un ión esta en función de la movilidad del ión en el gas derivado (N_2 es el más usado, pero puede ser algún otro gas), fuerza del campo eléctrico, E , (típicamente de 20 V/cm), y la longitud de la región de derivación del tubo.

En el metodo de aspiración, la muestra de gas fluye a través de un condensador o serie de condensadores o capacitores, cada uno crea un campo eléctrico predeterminado (ver figura 6.3).Los iones en la muestra son recogidos simultánea y continuamente por los distintos condensadores. Los iones que se detectan son eficazmente seleccionados por su movilidad.

Puumalainen y Paakkanen [66],[63] desarrollaron un dispositivo pequeño y de bajo costo basado en este método, conocido como condensador de aspiración plano, el cual se puede construir económicamente usando tecnología de circuitos impresos. Este dispositivo se ha utilizado con éxito para los usos militares e industriales y está comercialmente disponible de Environics, Oy, Finlandia. El dispositivo es muy semejante en cuanto a funcionamiento al espectrómetro de movilidad iónica. En el condensador, los iones de diferente movilidad iónica son separados en el espacio y medidos simultáneamente usando múltiples platos detectores en lugar de usar un solo plato como es el caso del espectrómetro de movilidad iónica; esto hace la diferencia. Una de las principales características del condensador de aspiración, es que los platos superiores de la celda y los platos detectores (platos inferiores), se encuentran ubicados de manera geométrica en forma de un capacitor o condensador; de aquí el nombre

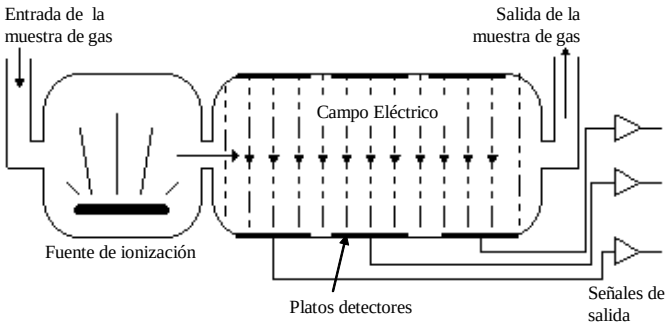

Figura 6.3: Diagrama esquemático de la celda de un condensador de aspiración plano [70]. Las moléculas de la muestra de gas son ionizadas. Los iones producto resultantes son separados por su movilidad en un campo eléctrico débil. Los iones con muy alta movilidad caen en el primer plato detector, y los iones con baja movilidad caen en el último plato detector.

dado al sensor(ver figura 6.3).

En el condensador de aspiración, una bomba mantiene constante la muestra de gas que fluye a través de todo el dispositivo. La muestra de gas es introducida a la región de ionización en donde es ionizada por una fuente radioactiva y calentada a una temperatura constante. Los iones producto formados, son entonces acarreados hacia la celda de detección a través del mismo flujo de gas. Ya en la celda de detección, la trayectoria de los iones es desviada por un campo eléctrico que es producido al aplicar una diferencia de potencial entre los platos metálicos que se encuentran en paredes opuestas de la celda. Las moléculas neutras o no ionizadas, no son afectadas por el campo eléctrico; sin embargo, las partículas cargadas son desviadas en su trayectoria y eventualmente empiezan a golpear las paredes de la celda. Los iones desviados empiezan a chocar con los platos detectores de manera tal que los iones de alta movilidad son capturados por el primer condensador y los iones de muy baja movilidad son capturados por el último. Cada uno de los condensadores, genera una corriente eléctrica que es proporcional a la carga y abundancia de los iones impactados; esta además, es función de la magnitud del campo eléctrico y de la distribución de movilidad iónica de la muestra de gas. Cuando los iones chocan con los platos detectores se desintegran y todas las moléculas

implicadas regresan al flujo del gas en su estado original.

6.2. El condensador de aspiración

En el dispositivo propuesto, solamente un electrodo de colector se utiliza para hacer las medidas. El primer electrodo, o el electrodo frontal, captura todos los iones de alta movilidad y limita por tanto el rango de iones que pueden caer en el segundo o electrodo colector. Esta configuración se domina *condensador de aspiración diferencial de primer orden*, y la corriente medida en el electrodo colector está dada por la integral ponderada de la densidad de movilidad iónica sobre un rango de movilidades definido. Esta función de ponderación que se denomina *función G* (debido a la forma de la ecuación que representa la forma de las funciones que tienen como solución una función de Green), es diferente para cada condensador de aspiración y depende de su geometría, el flujo del gas y el voltaje aplicado. Para una geometria y un flujo dados, simplemente cambiando el campo eléctrico podemos modificar la función G del condensador de la aspiración, de este modo cambiamos el rango de movilidades iónicas que son medidas.

En el sistema propuesto, el condensador de la aspiración está bajo control directo de una computadora, permitiéndonos cambiar rápida y exactamente el campo eléctrico del condensador de aspiración y hacer medidas de diferentes partes del espectro de movilidad iónica(véase fig. 6.4). Con este planteamiento, el único colector en cada voltaje es equivalente a un condensador separado en el dispositivo de Puumalainen. Las ventajas son: la flexibilidad agregada que la secuencia de la medida es controlada por el software, y no fijado por el hardware; el aumento de la SNR obtenido con solamente una medida a la vez; y el número mucho más grande de diversas medidas posibles puesto que no hay limitaciones físicas en el número de condensadores.

La limitación principal del método tradicional de aspiración es la pobre resolución de la configuración del dispositivo. En realidada, la medida es una medida integral sobre una gama relativamente amplia de las movilidades del ion. La diferencia principal y el componente central de nuestro método es que estamos haciendo medidas múltiples en diferentes voltajes

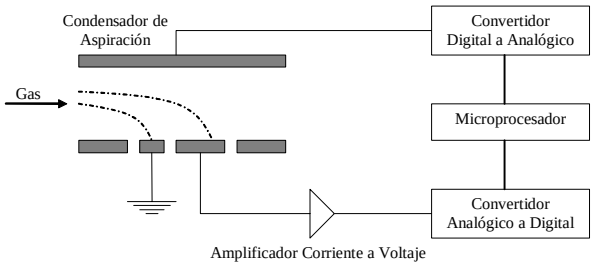

Figura 6.4: Diagrama esquemático funcional del condensador de aspiración con barrido de campo propuesto: Barriendo el potencial de desvio sobre un rango de voltajes, la curva de I(V) puede ser obtenida que son características de la distribución de la movilidad iónica de una muestra de gas ionizada.

de desvío. Con el barrido del voltaje de desvío, se obtiene una curva de corriente contra voltaje característica. Esta curva contiene toda la información del espectro de movilidad ionica, y en realidad, aplicando una transformada inversa, puede reconstruirse ese espectro. La resolución de tal acercamiento está por supuesto limitado por el número de voltajes de desviación en los cuales se hacen las medidas. Teóricamente, cualquier resolución deseada en el dominio de la movilidad iónica puede ser alcanzada simplemente aumentando el número de medidas en el dominio del voltaje.

Este acercamiento es análogo a la metodología usada para la tomografía computarizada (exploraciones de CT): Las imágenes de la radiografía no tienen ninguna resolución en la dimensión de la profundidad; sin embargo, haciendo medidas múltiples en diferentes ángulos la información tridimensional puede ser reconstruida aplicando una transformada inversa. (Estamos haciendo medidas unidimensionales para reconstruir una curva de dos dimensiones pero en vez de variar el ángulo variamos el campo eléctrico.)

6.2.1. Teoría de operación

La descripción teórica del método del condensador de aspiración fue dasarrollada por Tammet en 1967 [75]. El condensador de aspiración básico consiste de dos electrodos difer-

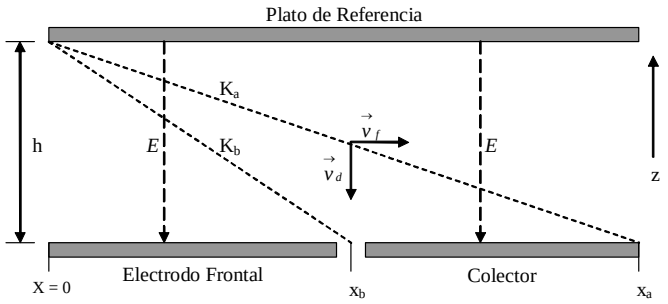

Figura 6.5: Diagrama de un condensador de aspiración diferencial de primer orden. Las trayectorias de los iones que viajan a través del condensador son desviadas por el campo eléctrico E. El electrodo frontal (precolector) captura todo ion con movilidad $K > K_b$. El electrodo colector captura todos aquellos iones con una movilidad $K > K_a$ excepto aquellos que fueron capturados por el precolector.

entes: un electrodo frontal (o precolector) y un electrodo colector con el cual se realizan las mediciones (ver figura 6.5)

Un flujo de gas previamente ionizado fluye entre los electrodos del condensador y una placa de referencia que se encuentra dispuesta en una configuración de condensador o capacitor. Los iones de una polaridad dada son desviados por el campo hacia los electrodos mediante el campo eléctrico E que se forma entre las placas, de forma perpendicular al flujo. La componente vertical de la trayectoria del ion, v_d, paralela al campo, es la velocidad de derivación (velocidad de arrastre) del ion debida a éste. La componente horizontal, v_f, paralela al flujo, es la velocidad del ion debida al frente laminar del flujo dentro del condensador. El flujo debe de mantenerse de forma laminar, puesto que de otra forma la trayectoria promedio del ion no puede ser controlada y los iones no pueden ser separados de acuerdo a su tamaño. El perfil del flujo laminar es parabólico con una velocidad de flujo máxima en el centro del canal y cercana a cero en las paredes. Sin embargo, solo nos interesa la trayectoria media para un ion de una cierta movilidad. Para este análisis se puede simplificar el modelo considerando solamente la velocidad del flujo promedio, donde f es la tasa de flujo del gas y s es él area

de la sección tranversal del canal del condensador [19].

$$\overrightarrow{v_f} = \frac{f}{s}$$

Con esta simplificación, puede verse gráficamente en la figura (6.5) que todo ión positivo con movilidad mayor o igual a K_b se desviará lo suficiente como para ser recogido por el precolector ya que aún los iones más lejanos de los electrodos a la entrada del detector, a una distancia h sobre ellos, pueden desviarse de esta manera. El campo eléctrico se define como $E = V/h$. Si trazamos una línea desde la entrada de la cámara desde una altura h a una distancia $x = 0$ hasta la orilla lejana del precolector (x-b), podemos darnos cuenta de ello. En este caso, los iones con movilidades mayores a cierto limite, el cual llamaremos k_b, cumplirán este requisito (entre mayor sea la movilidad es más facil que un ión se desvie por el campo). Podemos hacer lo mismo para la otra placa, tomando ahora x=a, y ahora los iones que pueden ser recogidos por la placa colectora y/o el electrodo precolector serán aquellos con una movilidad mayor o igual a un límite que denominaremos k_a. Siguiendo este razonamiento se puede determinar una función que describe la eficiencia de captura de iones en cada electrodo, la cual se denomina como *función G*.

Al barrer el voltaje del condensador se produce una curva de corriente contra voltaje $I(V)$ en el electrodo de medición (el colector). Esta curva es producida como una transformación del espectro de movilidad iónica de la muestra de gas. La transformada descrita por Tammet [75] se define como:

$$I(V) = \int fG(V, K, f)D(K)\,dK \tag{6.1}$$

En la ecuación anterior $G(V, K, f)$ es la función G del electrodo de medición que describe su eficiencia de captura iónica, y es una función del voltaje de deflección V aplicado al condensador, la movilidad iónica K, y la tasa de flujo del gas portador f. $D(K)$ es la distribución de movilidad iónica o función de densidad de la muestra de gas ionizado, y es igual a la densidad de carga específica de los iones con una movilidad K.

Para el precolector, que se extiende desde x=0 hasta x=b, tendremos:

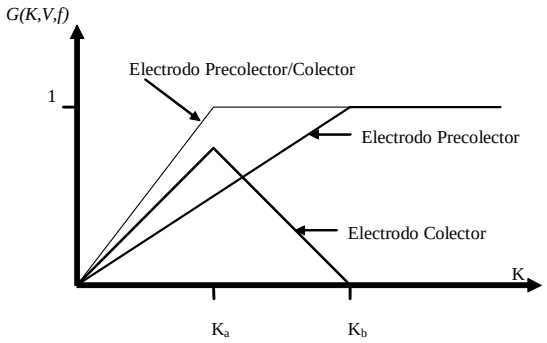

Figura 6.6: Cálculo gráfico de la función G para el electrodo colector.

$$G_b\left(K,V,f\right)=\begin{cases}\frac{K}{K_b} & \text{si} \quad K_b\geq K\geq 0\\ 1 & \text{si} \quad K\geq K_b\end{cases}$$

y para el precolector y colector juntos tendremos:

$$G_b\left(K,V,f\right)=\begin{cases}\frac{K}{K_a} & \text{si} \quad K_a\geq K\geq 0\\ 1 & \text{si} \quad K\geq K_a\end{cases}$$

y tomando en cuenta que $b<a \quad$ y $\quad K_b>K_a$ podemos calcular la función G para el colector solo:

$$G_{ab}\left(K,V,f\right)=G_a\left(K,V,f\right)-G_b\left(K,V;f\right)=\begin{cases}\frac{K}{K_a}-\frac{K}{K_b} & \text{si} \quad K_a\geq K\geq 0\\ 1-\frac{K}{K_b} & \text{si} \quad K_b\geq K\geq 0\\ 0 & \text{si} \quad K\geq K_b\end{cases} \tag{6.2}$$

y por esto es que se conoce como condensador diferencial. En la figura (6.6) se muestra una representación grafica de los calculos anteriores.

Para todas las ecuaciones anteriores:

$$K_a=\frac{hv_f}{x_aE} \qquad \text{y} \qquad K_b=\frac{hv_f}{x_bE}$$

Puede entonces observarse que la función G depende de la geometría del detector (la distancia y dimenciones de los electrodos x, x_a, y x_b, la distancia entre placas h, y la sección tranversal del canal s) la tasa de flujo del gas f (que determina la velocidad promedio $\overrightarrow{v_f}$) y el potencial de deflección aplicado V (el cual causa el campo eléctrico E). Si consideramos que las dimensiones del condensador y la tasa de flujo son constantes, podemos definir las constantes A y B como:

$$A = \frac{h^2 \overrightarrow{v_f}}{x_a} \quad \text{y} \quad B = \frac{h^2 \overrightarrow{v_f}}{x_b}$$

y entonces podemos describir la función G de un condensador de aspiración como:

$$G_{ab}(K,V) = \begin{cases} \frac{KV}{A} - \frac{KV}{B} & \text{si} \quad A \geq KV \geq 0 \\ 1 - \frac{KV}{B} & \text{si} \quad B \geq KV \geq 0 \\ 0 & \text{si} \quad KV \geq B \end{cases}$$

El maximo de la función G puede encontrarse fácilmente, este ocurre cuando KV=A En este caso:

$$G(K,V)_{\text{máx}} = 1 - \frac{A}{B} = 1 - \frac{x_b}{x_a}$$

Dado que $x_b < x_a$ y ambas son distancias positivas reales, el valor máximo de la función es siempre positivo y menor a la unidad, y solo está determinado por la longitud de las placas precolectora y colectora. Desde un punto de vista discreto, la función G puede apreciarse como dos familias de triángulos, una para cada una de las variables (V y K) (ver figura 6.7). Estas se encuentran integradas de forma que un espaciado uniforme en una de las variables produce un espaciado hiperbólico en los picos de los triángulos para la otra variable.

6.2.2. La transformada discreta de Tammet

Se desea utilizar al condensador de aspiración con un barrido de campo como un espectrómetro de movilidad iónica. Para ello, es necesario poder reconstruir la función de densidad de movilidad iónica $D(K)$ a partir de la curva $I(V)$ obtenida con el condensador, es decir,

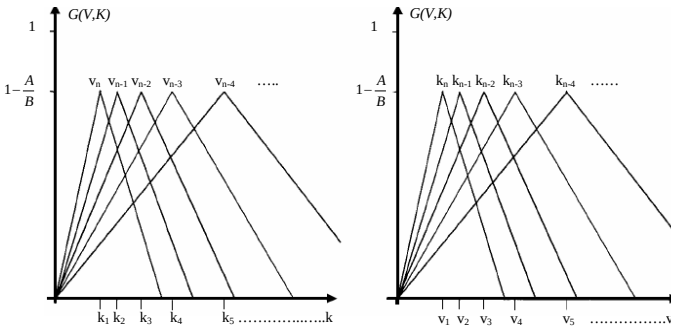

Figura 6.7: La función G vista como dos familias simétricas de triángulos. La forma de los triángulos es una función de la dimensión de los electrodos del condensador. Las dos variables independientes mantienen una relación hiperbólica.

la función inversa de la tranformada de Tammet definida en (6.1). En esta ecuación estamos definiendo cada punto en la función de densidad de corriente $I(V)$ por integración de una función G correspondiente $G(K,V)$, que mapea una función de distribución de movilidad iónica completa $D(K)$.La información completa de $D(K)$ es reflejada por cada valor de $I(V)$ pero es vista desde una perpectiva diferente.

Como es el caso para otras transformadas, la ecuación de Tammet requiere de la evaluación de las integrales asociadas, y puede ser descrita analíticamente únicamente si las expresiones involucradas son relativamente simples. Mas aún, las funciones de distribución de movilidad iónica no pueden ser descritas de manera analítica dado que son señales físicas y generalmente son obtenidas de forma discreta por un espectrómetro de movilidad iónica o algún equipo similar, y deben manipularse en forma continua. Por otro lado, la función $I(V)$ también se obtiene de forma discreta ya que los voltajes suministrados al sistema son controlados por una computadora a través de un convertidor digital-analógico.

Esto significa que se requiere una técnica adeacuada para el muestreo y una representación discreta de la ecuación de Tammet. Esta discretización permite una solución simple y eficiente desde un punto de vista computacional.

La transformada directa de Tammet en su versión discreta (DDTT) para un condensador

de aspiración con una tasa de flujo de gas constante $f = \mathrm{f}$ puede aproximarse mediante una sumatoria de Riemann [23]:

$$i\left(v_j\right) = f \sum_{i=0}^{\infty} \Delta k_i G\left(k_i, v_j\right) D\left(k_j\right)$$

donde $\Delta k_i = \left(k_i - k_{i-1}\right)$. Entonces podemos describir la ecuación para la DDTT en notación matricial:

$$i = fGd \tag{6.3}$$

y si la matriz G es cuadrada y no singular, podemos definir la transformada inversa de Tammet discreta (DITT) como:

$$fd = G^{-1}i \tag{6.4}$$

en donde i es un vector que contiene los elementos de la curva caracteristica $I\left(V\right)$,

$$i = \begin{bmatrix} I\left(v_1\right) \\ I\left(v_2\right) \\ \vdots \\ \vdots \\ I\left(v_m\right) \end{bmatrix}$$

d es un vector que contiene los elementos de la función de distribución de movilidad iónica $D\left(k_j\right)$,

$$d = \begin{bmatrix} D\left(k_1\right) \\ D\left(k_2\right) \\ \vdots \\ \vdots \\ D\left(k_n\right) \end{bmatrix}$$

y G es una matriz cuadrada no singular que contiene los elementos $\Delta k_i G\left(k_i, v_j\right)$,

$$G = \begin{bmatrix} (k_1 - k_0)\, G(k_1, v_1) & (k_2 - k_1)\, G(k_2, v_1) & \cdots & \cdots & (k_n - k_{n-1})\, G(k_n, v_1) \\ (k_1 - k_0)\, G(k_1, v_2) & (k_2 - k_1)\, G(k_2, v_2) & \cdots & \cdots & (k_n - k_{n-1})\, G(k_n, v_2) \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ (k_1 - k_0)\, G(k_1, v_m) & (k_2 - k_1)\, G(k_2, v_m) & \cdots & \cdots & (k_n - k_{n-1})\, G(k_n, v_m) \end{bmatrix}$$

donde

$$\forall i = 0, ..., n : \qquad k_i = \frac{A}{v_i} \qquad \text{y} \qquad v_i > 0$$

Esta última condición, aunque no necesaria para una tranformación directa, es esencial para la transformada inversa ya que preserva la relación hiperbólica entre los dominios del voltaje y la movilidad iónica y determina el hecho de que la matriz G posea en determinante diferente de cero y que por tanto existe una solución inversa única. Si lo que se desea es obtener una función de distribución de movilidad iónicva $D(k_i)$ uniformemente espaciada utilizando la DITT para la curva de corriente $I(v_j)$ adquirida por el condensador de aspiración, entonces se requiere utilizar un esquema de muestreo apropiado espaciado hiperbólicamente en el dominio del voltaje. De forma mas general, al aplicar la restricción descrita se puede designar la estrategia de muestreo requerida para obtener un rango y resolución deseados en cualquiera de los dominios, dadas las dimensiones del condensador y la tasa de flujo de gas empleada.

6.3. Modelado de EMI via neuro difusos

En primer lugar, podemos mencionar que el *espectrómetro de movilidad iónica* es un instrumento de presición utilizado para determinar experimentalmente, la distribución de movilidad iónica $D(K)$ de una muestra de gas ionizada; la cual es característica de la composición de la muestra de gas. La distribución de movilidad es utilizada para determinar las concentraciones y la composición de la muestra de gas. Ver figura (6.8).

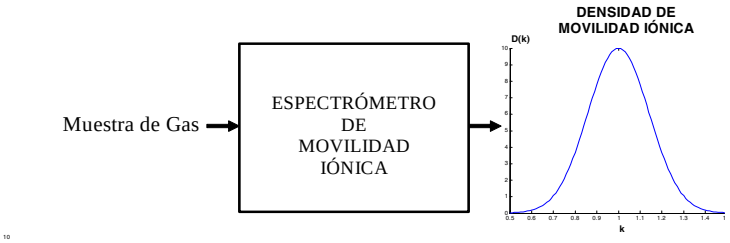

Figura 6.8: Densidad de movilidad iónica producida por el espectrómetro de movilidad iónica.

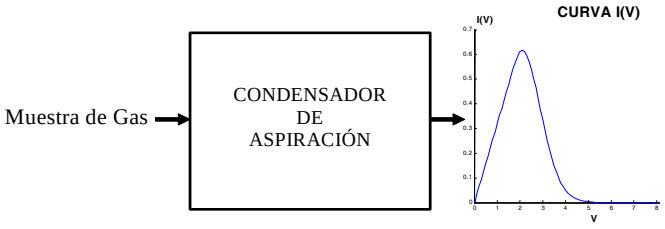

Figura 6.9: Curva de voltaje contra corriente $I(V)$ producida por el condensador de aspiración con barrido de campo.

En segundo lugar, el *condensador de aspiración con barrido de campo* es un instrumento que produce una curva de corriente contra voltaje $I(V)$ en su electrodo de medición. La curva contiene información sobre la distribución de movilidad iónica de la muestra de gas analizada. Ver figura (6.9).

Por otra parte, la trasformada directa de Tammet en su version discreta (DDTT)(6.3) es utilizada para transformar la distribución de movilidad iónica en una curva de corriente contra voltaje. Mientras que la transformada de Tammet inversa en su versión discreta (DITT)(6.4) es utilizada para convertir la curva de corriente contra voltaje en una distribución de movilidad ionica. Ver figura (6.10)

Por lo tanto, si deseamos utilizar al condensador de aspiración con barrido de campo como un espectrómetro de movilidad iónica, es claro que se tiene que tranformar la curva de corriente contra voltaje $I(V)$ producida por el condensador de aspiración en un espectro

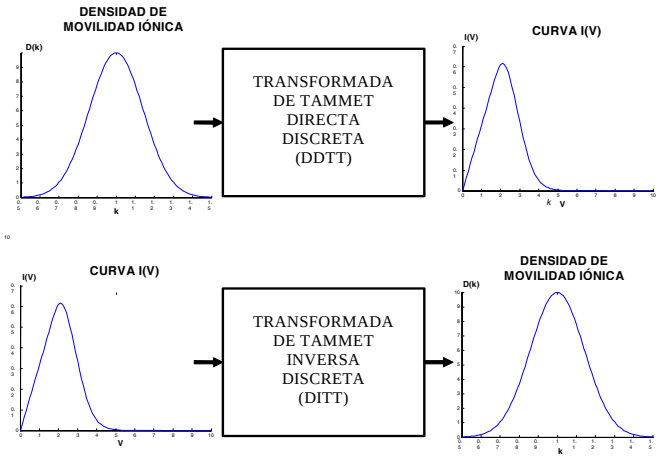

Figura 6.10: Aplicación de la transformada directa (DDTT) e inversa (DITT) de Tammet en su versión discreta.

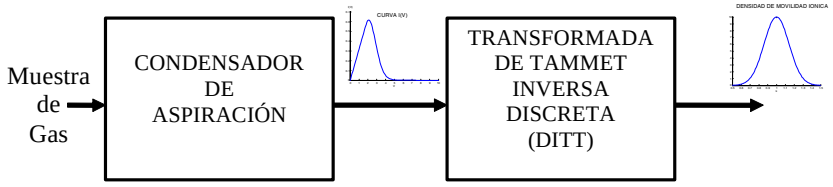

Figura 6.11: Condensador de aspiración, visto como un espectrómetro de movilidad iónica.

de movilidad ionica $D(K)$ via la transformada inversa de Tammet en su versión discreta (DITT). Ver figura (6.11)

Como se mencionó en la sección 2.4.1 la representación de sistemas no-lineales por modelos de entrada-salida son clasificados en grupos dependiendo de como ocurren las no-linealidades en el modelo y la estructura de identificación del sistema es la de tipo serie-paralelo; por lo tanto, esta será la estructura de identificación que adoptaremos para nuestra aplicación.

Para nuestra aplicación la planta será la transformada de Tammet inversa discreta, la

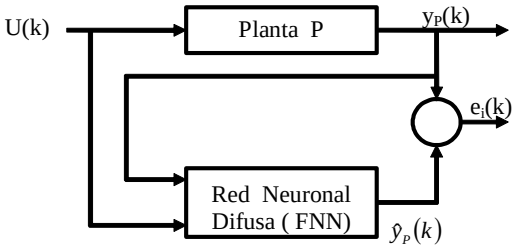

Figura 6.12: Estructura de identificación SERIE-PARALELO.

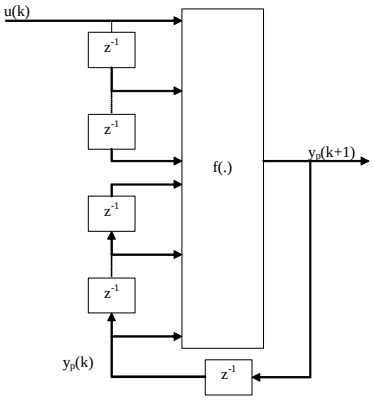

Figura 6.13:

cual será modelada utilizando un modelo de entrada-salida como el mostrado en la figura (6.13), en donde: $U(k)$ es el valor de la corriente $I(V)$ para un tiempo dado k , $y_p(k)$ es la densidad de movilidad ionica $D(k)$ en el tiempo k , $y_p(k+1)$ es la densidad de movilidad iónica deseada (salida deseada) y $f(.)$ es el sistema a identificar en este caso se trata de la transformada de Tammet inversa discreta (6.4).

Ya que la planta a identificar esta modelada mediante un modelo de entrada-salida, la red neuronal difusa utilizada en la estructura de identificación sera del tipo estatica (FFNN). Si pudieramos utilizar el modelo de estado-espacio para modelar la planta (sistema), la red neuronal difusa sería del tipo dinámica (RFNN). Los PEs utilizados en la capa oculta

de ambos tipos de redes neuronales difusas juegan un papel muy importante. Cuando un sistema linear tiene que ser identificado, utilizaremos solo PEs lineales y cuando el sistema a identificar es no-lineal loe PEs adoptados en la capa oculta son tambien no-lineales. En otras palabras, se suguiere que las redes neuronales con PEs lineales son suficientes para la identificación de sistemas lineales, mientras que los PEs no-lineales solo son necesarios si el sistema a ser identificado es no-lineal. Sin embargo, de acuerdo a nuetra intuición, las redes lineales pueden ser mejores para identificar sistemas lineales que las redes no-lineales y viceversa, lo cual conduce a pensar en redes hibridas.

Ya que la red que se utilizará en la estructura de identificación es difusa (o implementa un sistema difuso), los PEs utilizados en la capa oculta son no-lineales, ya que estos estan representados por funciones de membresía no-lineales.

6.3.1. Simulación

Para provar la capacidad de identificación de la red neuronal difusa estatica, utilizamos las transformadas de Tammet (DDTT y DITT).

En primer lugar, generamos curvas de densidad de movilidad iónica D(K) gaussianas con diferentes parametros (media, amplitud y ancho) con el fin de representar la movilidad de algunos gases anestesicos. Las curvas de corriente contra voltaje I(V) correspondientes fueron generadas via la transformada directa de Tammet DDTT (6.3). Este conjunto de graficas (ver figuras 6.14 y 6.18) fueron utilizadas como datos de entrada-salida para entrenar la FNN con el fin de identificar la DDTT. En las figuras (6.15) y (6.19) se muestran los centros de grupo generados al aplicar el algoritmo de agrupamiento substractivo doble, los parametros de la FNN: las FM's antecedentes son mostradas en las figuras (6.16) y (6.20), las reglas de

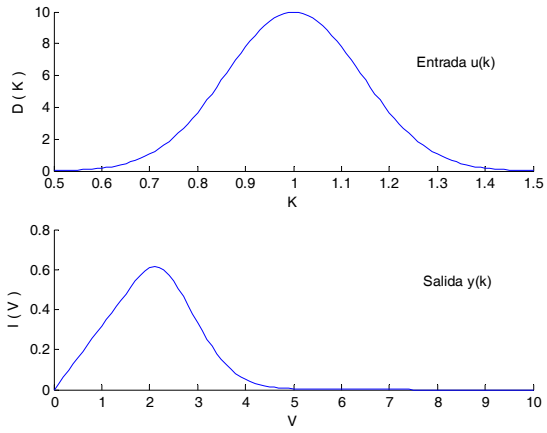

Figura 6.14: **Arriba:** Densidad de movilidad iónica gaussiana con media=1 y σ=0.2 y amplitud de 10. **Abajo:** La DDTT correspondiente

inferencia difusa son mostradas en (6.5) y (6.6).

$$
\begin{aligned}
&\text{IF } In1 \text{ es } fm_1 \text{ and } In2 \text{ es } fm_1 \text{ THEN } f_1 = \quad 0.0000408\ In1 + 0.0081467\ In2 - 0.096258\\
&\text{IF } In1 \text{ es } fm_2 \text{ and } In2 \text{ es } fm_2 \text{ THEN } f_2 = \quad 0.3909 \quad In1 - 0.099422 \quad In2 - 25.275\\
&\text{IF } In1 \text{ es } fm_3 \text{ and } In2 \text{ es } fm_3 \text{ THEN } f_3 = \quad 0.0000789\ In1 - 0.0084705\ In2 + 0.0091849\\
&\text{IF } In1 \text{ es } fm_4 \text{ and } In2 \text{ es } fm_4 \text{ THEN } f_4 = \quad 0.39719 \quad In1 + 4.1503 \quad In2 + 16.639\\
&\text{IF } In1 \text{ es } fm_5 \text{ and } In2 \text{ es } fm_5 \text{ THEN } f_5 = -0.012234 \quad In1 - 0.20824 \quad In2 + 1.4507\\
&\text{IF } In1 \text{ es } fm_6 \text{ and } In2 \text{ es } fm_6 \text{ THEN } f_6 = -0.049809 \quad In1 - 0.096992 \quad In2 + 3.9181
\end{aligned}
\tag{6.5}
$$

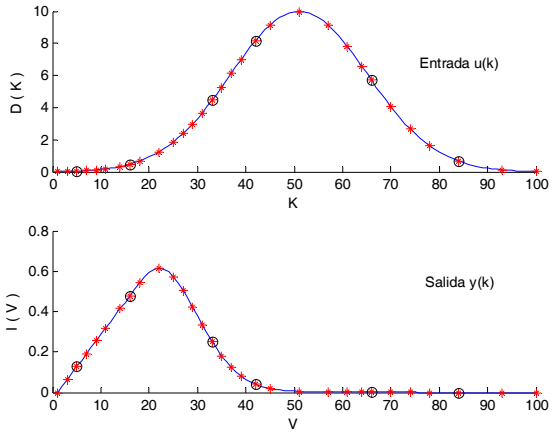

Figura 6.15: Centros de grupo generados con el algoritmo de agrupamiento substractivo doble para las graficas de la figura (6.14): $(*)$ Centros generados en la primera etapa, (o) Centros generados en la segunda etapa.

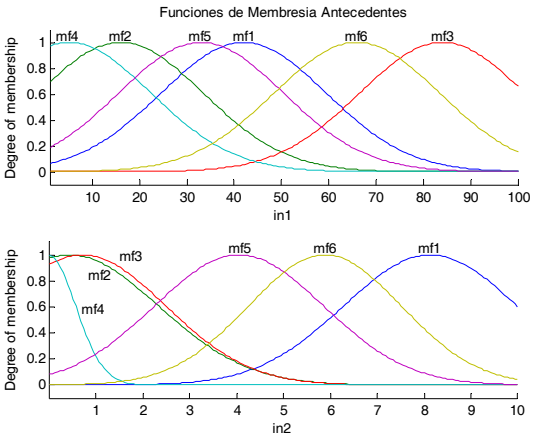

Figura 6.16: Funciones de Membresía Antecedentes de la Red Neuronal Difusa (FNN) generadas con los datos de la figura (6.14).

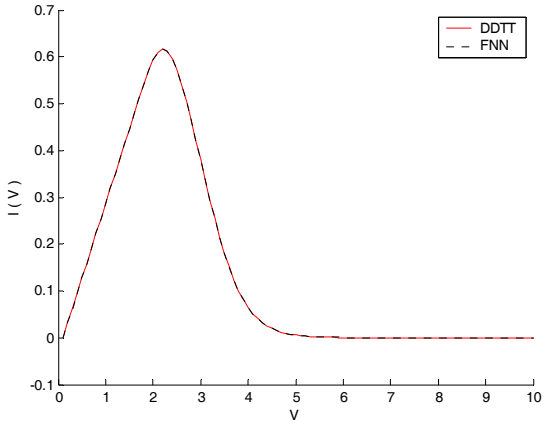

Figura 6.17: Comparación entre las curvas de corriente contra voltaje I(V) generadas tanto con la Transformada de Tammet Directa Discreta (DDTT) como con la FNN. En estas graficas se puede observar que la FNN identifica de manera exacta a la transformada directa de Tammet.

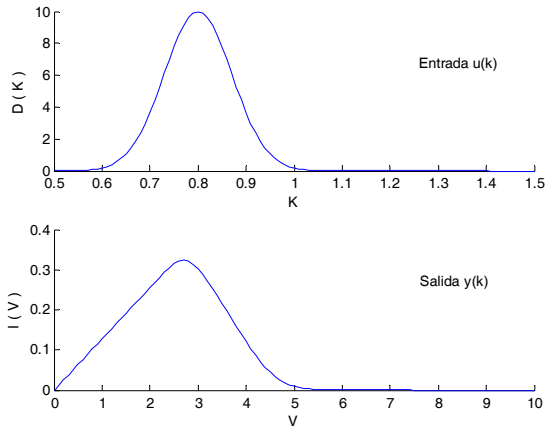

Figura 6.18: **Arriba:** Densidad de movilidad iónica gaussiana con media=0.8 y σ=0.1 y amplitud de 10. **Abajo:** La DDTT correspondiente.

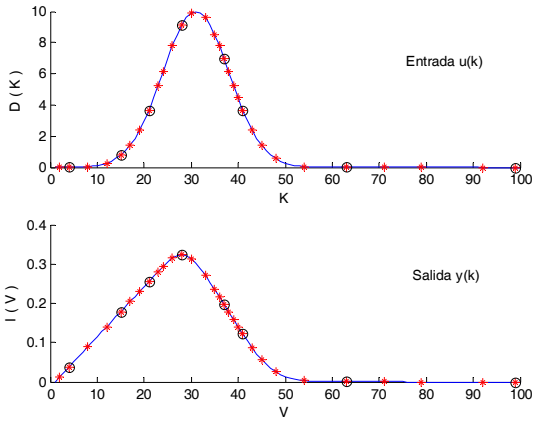

Figura 6.19: Centros de grupo generados con el algoritmo de agrupamiento substractivo doble para las graficas de la figura (6.18): $(*)$ Centros generados en la primera etapa, (o) Centros generados en la segunda etapa.

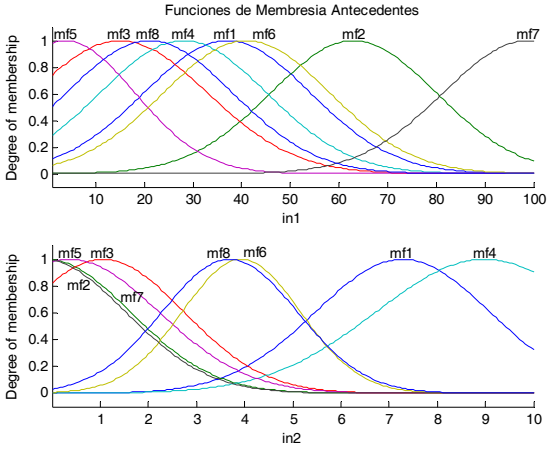

Figura 6.20: Funciones de Membresía Antecedentes de la Red Neuronal Difusa (FNN) generadas con los datos de la figura (6.18).

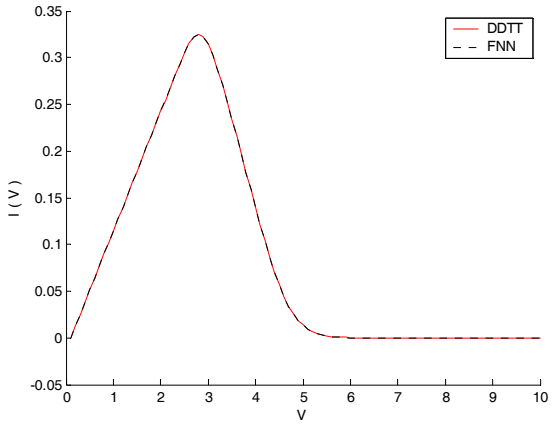

Figura 6.21: Comparación entre las curvas de corriente contra voltaje I(V) generadas tanto con la Transformada de Tammet Directa Discreta (DDTT) como con la FNN. En estas graficas se puede observar que la FNN identifica de manera exacta a la transformada directa de Tammet.

$$
\begin{aligned}
&\text{IF } In1 \text{ es } fm_1 \text{ and } In2 \text{ es } fm_1 \text{ THEN } f_1 = \text{ -0.0066815}In1\text{+ 0.034193 } In2\text{+ 0.2206}\\
&\text{IF } In1 \text{ es } fm_2 \text{ and } In2 \text{ es } fm_2 \text{ THEN } f_2 = \text{ 0.003271 } In1\text{- 0.050457 } In2\text{- 0.29528}\\
&\text{IF } In1 \text{ es } fm_3 \text{ and } In2 \text{ es } fm_3 \text{ THEN } f_3 = \text{ -0.051064 } In1\text{- 0.41822 } In2\text{+ 4.2919}\\
&\text{IF } In1 \text{ es } fm_4 \text{ and } In2 \text{ es } fm_4 \text{ THEN } f_4 = \text{ -0.014847 } In1\text{+ 0.022428 } In2\text{+ 0.52576}\\
&\text{IF } In1 \text{ es } fm_5 \text{ and } In2 \text{ es } fm_5 \text{ THEN } f_5 = \text{ -0.05063 } In1\text{- 0.55981 } In2\text{- 2.8652}\\
&\text{IF } In1 \text{ es } fm_6 \text{ and } In2 \text{ es } fm_6 \text{ THEN } f_6 = \text{ 0.079947 } In1\text{+ 0.063641 } In2\text{- 2.8634}\\
&\text{IF } In1 \text{ es } fm_7 \text{ and } In2 \text{ es } fm_7 \text{ THEN } f_7 = \text{ 0.00042838}In1\text{- 1.5062 } In2\text{- 0.047259}\\
&\text{IF } In1 \text{ es } fm_8 \text{ and } In2 \text{ es } fm_8 \text{ THEN } f_8 = \text{ -0.087293 } In1\text{+ 0.07088 } In2\text{+ 2.5318}
\end{aligned}
\tag{6.6}
$$

En segundo lugar, generamos curvas de corriente contra voltaje I(V) con el fin de representar la concentración de algunos gases anestesicos. Las curvas de densidad de movilidad iónica D(K) gaussianas correspondientes fueron generadas via la transformada inversa de Tammet DITT (6.4). Este conjunto de graficas (ver figuras 6.22 y 6.26) fueron utilizadas como datos de entrada-salida para entrenar la FNN con el fin de identificar la DITT. En las figuras (6.23)

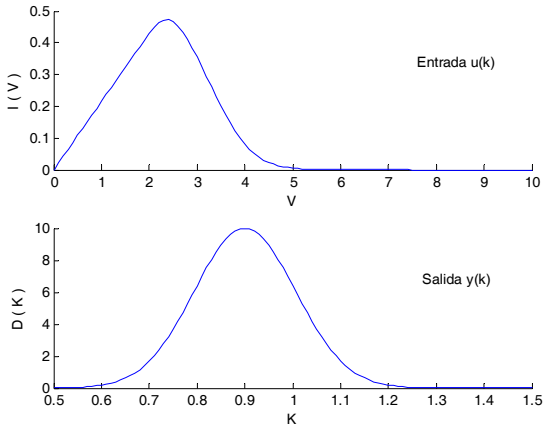

Figura 6.22: **Arriba:** Curva de corriente contra voltaja I(V), para una densidad de movilidad iónica gaussiana con una media=0.9 y σ=0.15. **Abajo:** La DITT correspondiente.

y (6.27) se muestran los centros de grupo generados al aplicar el algoritmo de agrupamiento substractivo doble, los parametros de la FNN: las FM's antecedentes son mostradas en las figuras (6.24) y (6.28), las reglas de inferencia difusa son mostradas en (6.7) y (6.8).

$$
\begin{array}{llllll}
\text{IF } In1 \text{ es } fm_1 \text{ and } In2 \text{ es } fm_1 \text{ THEN } f_1 = & \text{-22.46} & In1+\ 1177.4 & In2-\ 85.403 \\
\text{IF } In1 \text{ es } fm_2 \text{ and } In2 \text{ es } fm_2 \text{ THEN } f_2 = & \text{-0.19717} & In1-\ 3837.6 & In2+28.909 \\
\text{IF } In1 \text{ es } fm_3 \text{ and } In2 \text{ es } fm_3 \text{ THEN } f_3 = & \text{-18.374} & In1+\ 1609.7 & In2+557.42 \\
\text{IF } In1 \text{ es } fm_4 \text{ and } In2 \text{ es } fm_4 \text{ THEN } f_4 = & \text{-0.96262} & In1-\ 166.44 & In2+130.92 \\
\text{IF } In1 \text{ es } fm_5 \text{ and } In2 \text{ es } fm_5 \text{ THEN } f_5 = & \text{-3.7716} & In1+\ 3009.8 & In2+357.11 \\
\text{IF } In1 \text{ es } fm_6 \text{ and } In2 \text{ es } fm_6 \text{ THEN } f_6 = & 13.205 & In1-\ 928.43 & In2-\ 402.69 \\
\text{IF } In1 \text{ es } fm_7 \text{ and } In2 \text{ es } fm_7 \text{ THEN } f_7 = & 7.9364 & In1-\ 1547.4 & In2+715.38
\end{array}
\tag{6.7}
$$

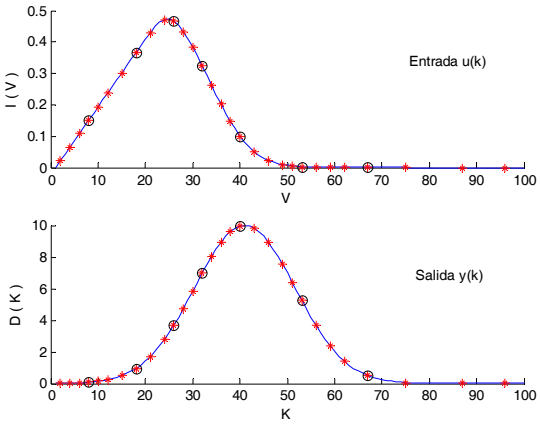

Figura 6.23: Centros de grupo generados con el algoritmo de agrupamiento substractivo doble para las graficas de la figura (6.22): $(*)$ Centros generados en la primera etapa, (o) Centros generados en la segunda etapa.

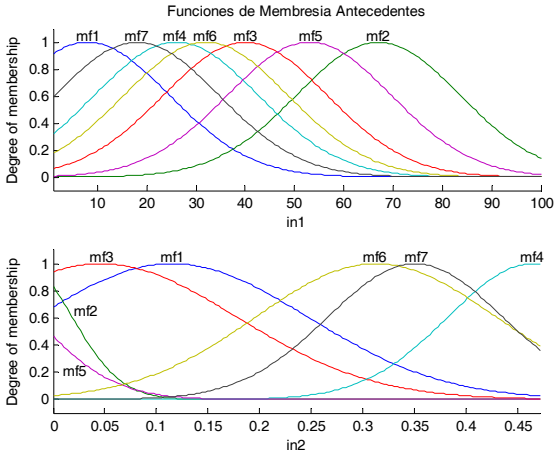

Figura 6.24: Funciones de Membresía Antecedentes de la Red Neuronal Difusa (FNN) generadas con los datos de la figura (6.22).

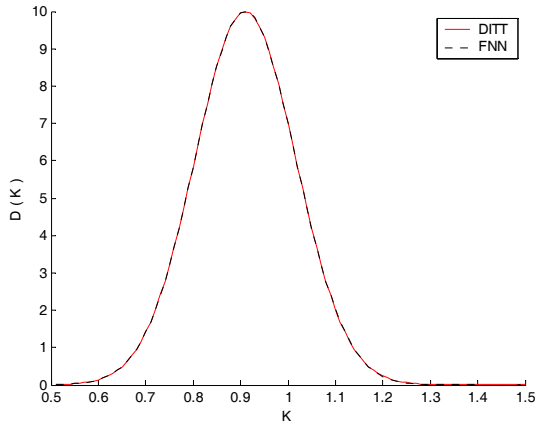

Figura 6.25: Comparación entre las curvas de Densidad de movilidad iónica D(K) generadas tanto con la Transformada de Tammet Inversa Discreta (DITT) como con la FNN. En estas graficas se puede observar que la FNN identifica de manera exacta a la transformada Inversa de Tammet.

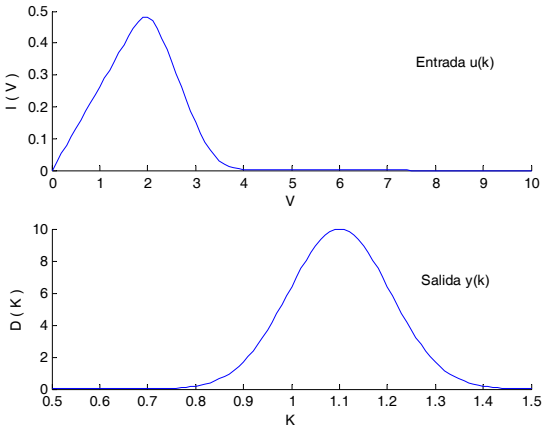

Figura 6.26: **Arriba:** Curva de corriente contra voltaja I(V), para una densidad de movilidad iónica gaussiana con una media=1.1 y σ=0.15. **Abajo:** La DITT correspondiente.

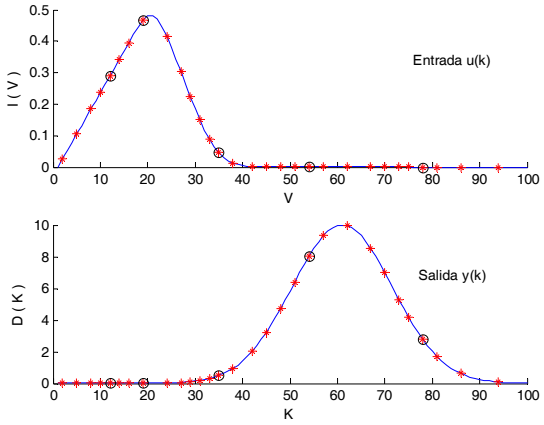

Figura 6.27: Centros de grupo generados con el algoritmo de agrupamiento substractivo doble para las graficas de la figura (6.26): $(*)$ Centros generados en la primera etapa, (o) Centros generados en la segunda etapa.

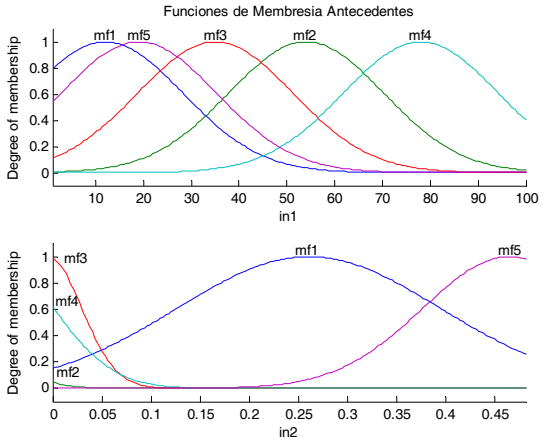

Figura 6.28: Funciones de Membresía Antecedentes de la Red Neuronal Difusa (FNN) generadas con los datos de la figura (6.26).

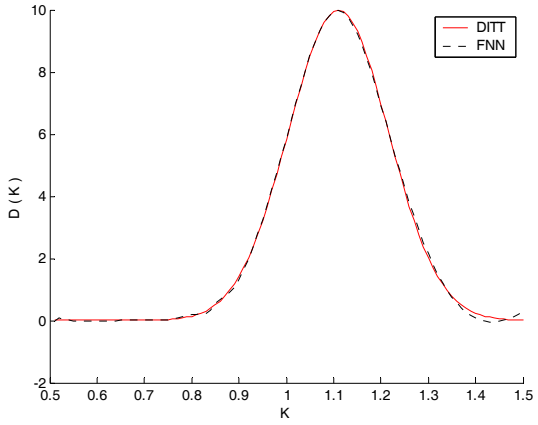

Figura 6.29: Comparación entre las curvas de Densidad de movilidad iónica D(K) generadas tanto con la Transformada de Tammet Inversa Discreta (DITT) como con la FNN. En estas graficas se puede observar que la FNN identifica de manera exacta a la transformada Inversa de Tammet.

$$
\begin{aligned}
&\text{IF } In1 \text{ es } fm_1 \text{ and } In2 \text{ es } fm_1 \text{ THEN } f_1 = \ 0.016039\ In1 + 1.5129\ \ In2 - 0.6217 \\
&\text{IF } In1 \text{ es } fm_2 \text{ and } In2 \text{ es } fm_2 \text{ THEN } f_2 = \ 1.2682\ \ \ In1 - \ 222.06\ \ In2 - 42.492 \\
&\text{IF } In1 \text{ es } fm_3 \text{ and } In2 \text{ es } fm_3 \text{ THEN } f_3 = -0.14468\ \ In1 + 10.283\ \ In2 + 2.0536 \\
&\text{IF } In1 \text{ es } fm_4 \text{ and } In2 \text{ es } fm_4 \text{ THEN } f_4 = \ 0.51854\ \ \ In1 + 6844.7\ In2 - 54.949 \\
&\text{IF } In1 \text{ es } fm_5 \text{ and } In2 \text{ es } fm_5 \text{ THEN } f_5 = -0.0057125 In1 + 2.306\ \ In2 - \ 0.98309
\end{aligned}
\tag{6.8}
$$

En estas simulaciones se puede observar que el numero de FM's asi como el numero de reglas de inferencia difusas dependen del numero de grupos generados por el algoritmo de agrupamiento substractivo doble. Las formas de las FM's y las reglas de inferencia difusas cambian dependiendo de los datos de entrenamiento. La FNN tiene una capacidad de identificación excelente, ya que los resultados de las simulaciones muestran que la FNN identifica de manera exacta a las transformadas de Tammet, DDTT y DITT, especificadas por (6.3) y (6.4) respectivamente (ver figuras 6.17,6.21, 6.25 y 6.29).

Gas Portador				Agente Anestesico	
$\mathbf{N_2}$	$\mathbf{N_2O}$	$\mathbf{O_2}$	$\mathbf{CO_2}$	**AA**	**AA %**
10 %	10 %	80 %	0 %	Isofluorano	1.5 %
0 %	0 %	80 %	20 %	Sevofluorano	5.0 %

Cuadro 6.1: Muestras de Gas

6.3.2. Aplicación en tiempo real

Si deseamos utilizar al condensador de aspiración con barrido de campo como un espectrómetro de movilidad iónica, se tiene que transformar la curva de corriente contra voltaje $I(V)$ producida por el condensador de aspiración en un espectro de movilidad ionica $D(K)$ via la transformada inversa de Tammet en su versión discreta (DITT) (ver figura 6.11). Por lo tanto, la aplicación consiste en identificar la transformada de Tammet inversa (DITT) utilizando la red neuronal difusa estatica (ver figura 6.12).

En primer lugar, generamos dos mezclas de gases utilizando un agente anestesico (AA) y un gas portador (ver tabla 6.1). Estas muestras fueron procesadas en el condensador de aspiración con barrido de campo para generar dos curvas de corriente contra voltaje I(V) caracteristicas de cada una de las muestras. Las curvas de I(V) fueron utilizadas para generar las curvas de Densidad de movilidad ionica D(K) de cada una de las muestras via la DITT. Las curvas de I(V) y D(K) (ver figuras 6.30 y 6.34) fueron utilizadas como datos de entrada-salida para entrenar la FNN con el fin de identificar la DITT. En las figuras (6.31) y (6.35) se muestran los centros de grupo generados al aplicar el algoritmo de agrupamiento substractivo doble, los parametros de la FNN: las FM's antecedentes son mostradas en las figuras (6.32) y (6.36), las reglas de inferencia difusa son mostradas en (6.9) y (6.10).

Los resultados son mostrados en las figuras (6.33) y (6.37), en donde se puede observar que la FNN identifica de manera exacta a la transformada de Tammet Inversa (DITT) ya que las curvas de D(K) son identicas.

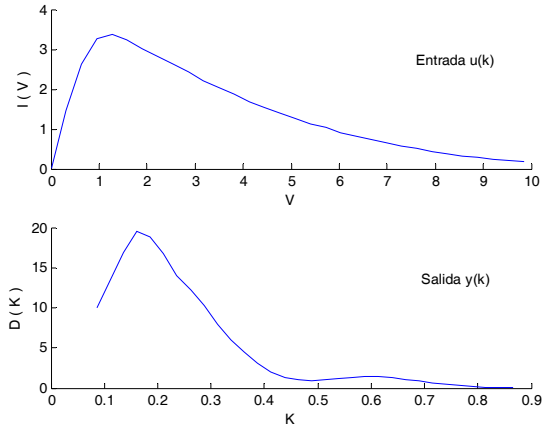

Figura 6.30: **Arriba**: Curva de corriente contra voltaje I(V) para iones positivos de Isofluorano con una concentración del 1.5 % en un gas portador de 10 % N_2, 10 % N_2O, 80 % O_2, 0 % CO_2. **Abajo:** Transformada Inversa correspondiente utilizando 32 puntos.

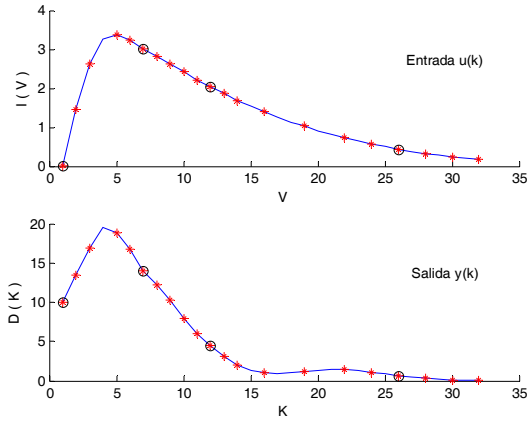

Figura 6.31: Centros de grupo generados con el algoritmo de agrupamiento substractivo doble para la prueba con Isofluorano: $(*)$ Centros generados en la primera etapa (o) Centros generados en la segunda etapa.

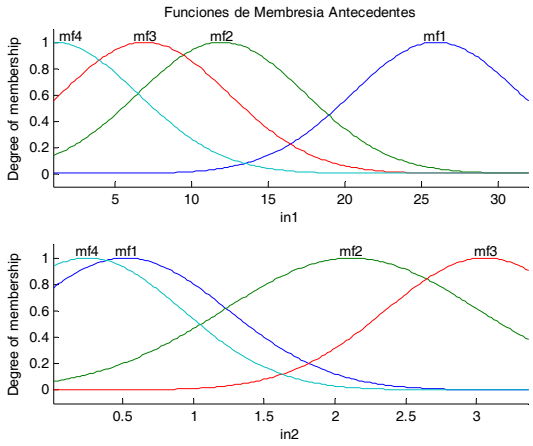

Figura 6.32: Funciones de Membresía Antecedentes de la Red Neuronal Difusa (FNN) para la prueba con Isofluorano

$$
\begin{aligned}
&\text{IF } In1 \text{ es } fm_1 \text{ and } In2 \text{ es } fm_1 \text{ THEN } f_1 = \text{ -0.094047 } In1+ \text{ 0.64911 } In2+ \text{ 2.8242} \\
&\text{IF } In1 \text{ es } fm_2 \text{ and } In2 \text{ es } fm_2 \text{ THEN } f_2 = \text{ 0.38273 } In1+ \text{ 9.6867 } In2- \text{ 19.931} \\
&\text{IF } In1 \text{ es } fm_3 \text{ and } In2 \text{ es } fm_3 \text{ THEN } f_3 = \text{-1.4797 } In1+ \text{ 2.9103 } In2+ \text{ 16.397} \\
&\text{IF } In1 \text{ es } fm_4 \text{ and } In2 \text{ es } fm_4 \text{ THEN } f_4 = \text{ 16.204 } In1+ \text{ 19.632 } In2- \text{ 6.2922}
\end{aligned} \tag{6.9}
$$

$$
\begin{aligned}
&\text{IF } In1 \text{ es } fm_1 \text{ and } In2 \text{ es } fm_1 \text{ THEN } f_1 = \text{ 0.12949 } In1+ \text{ 5.1616 } In2- \text{ 6.4545} \\
&\text{IF } In1 \text{ es } fm_2 \text{ and } In2 \text{ es } fm_2 \text{ THEN } f_2 = \text{-2.3314 } In1- \text{ 11.869 } In2+ \text{ 58.361} \\
&\text{IF } In1 \text{ es } fm_3 \text{ and } In2 \text{ es } fm_3 \text{ THEN } f_3 = \text{-0.3184 } In1+\text{4.9832 } In2+ \text{ 6.0984} \\
&\text{IF } In1 \text{ es } fm_4 \text{ and } In2 \text{ es } fm_4 \text{ THEN } f_4 = \text{ 2.3563 } In1- \text{ 9.6012 } In2+ \text{ 10.349}
\end{aligned} \tag{6.10}
$$

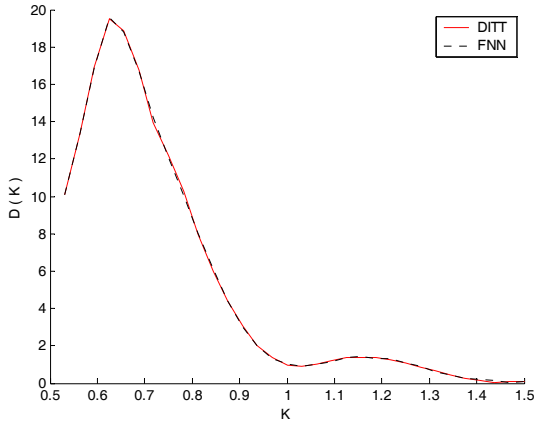

Figura 6.33: Comparación entre las curvas de Densidad de movilidad iónica D(K) para ISOFLUORANO, generadas tanto con la Transformada de Tammet Inversa Discreta (DITT) como con la FNN. En estas graficas se puede observar que la FNN identifica de manera exacta a la transformada Inversa de Tammet.

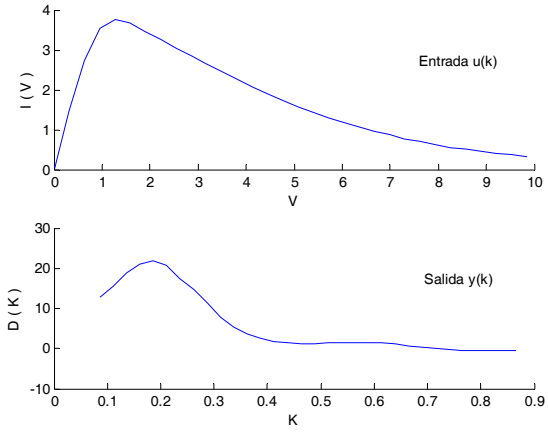

Figura 6.34: **Arriba**: Curva de corriente contra voltaje I(V) para iones positivos de Sevofluorano con una concentración del 5 % en un gas portador de 0 % N_2, 0 % N_2O, 80 % O_2, 20 % CO_2. **Abajo:** Transformada Inversa correspondiente utilizando 32 puntos.

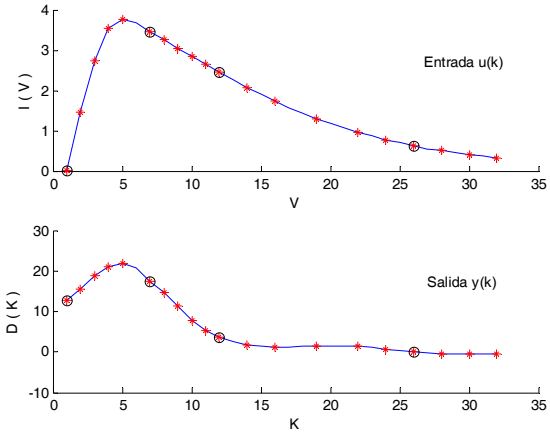

Figura 6.35: Centros de grupo generados con el algoritmo de agrupamiento substractivo doble para la prueba con Sevofluorano: (∗) Centros generados en la primera etapa (o) Centros generados en la segunda etapa.

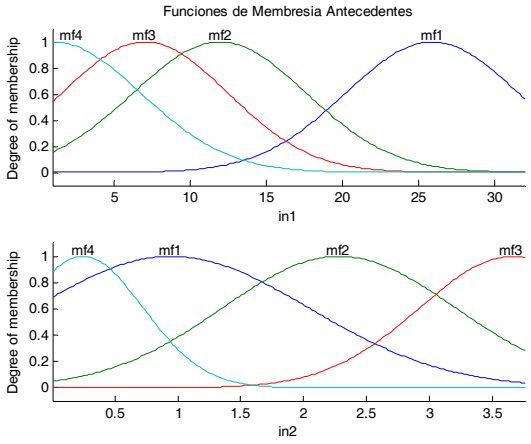

Figura 6.36: Funciones de Membresía Antecedentes de la Red Neuronal Difusa (FNN) para la prueba con Sevofluorano.

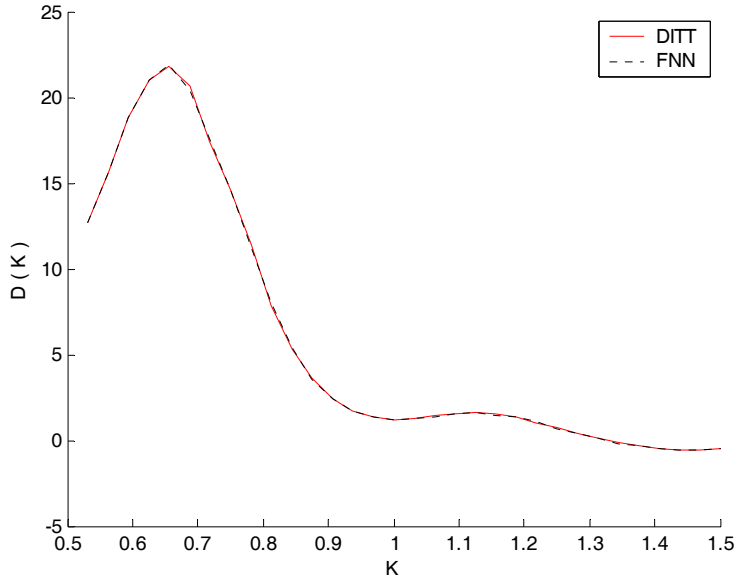

Figura 6.37: Comparación entre las curvas de Densidad de movilidad iónica D(K) para SEVOFLUORANO, generadas tanto con la Transformada de Tammet Inversa Discreta (DITT) como con la FNN. En estas graficas se puede observar que la FNN identifica de manera exacta a la transformada Inversa de Tammet.

Capítulo 7

Conclusión

Se considera que hasta el momento, el grado de avance en este trabajo es del 85 %. En cuanto al trabajo desarrollado, se puede mencionar que el algoritmo de agrupamiento on line propuesto tiene un gran potencial de funcionamiento lo cual puede comprobarse con las simulaciones realizadas hasta ahora, que muestran que este algoritmo realiza la identificación de la estructura del sistema de inferencia difuso de manera eficiente. La etapa de estimación de parámetros, por suparte, fue desarrollada utilizando una red neuronal difusa estática dando como resultado una modificación apropiada de los parámetros del sistema de inferencia difuso, lo cual se manifiesta en una buena aproximación del sistema simulado. Cabe aclarar que aún falta desarrollar la simulación en la etapa de estimación de parámetros vía una red neuronal difusa dinámica (recurrente); al concluir esta etapa se podrá realizar una comparación entre estas dos técnicas de estimación de parámetros para la identificación de sistemas.

**

Bibliografía

[1] S. Abe and M. S. Lan, Fuzzy rules extraction directly from numerical data for function approximation, *IEEE Transaction. Syst., Man, and Cybernetics*, 25 (1995), 119-129.

[2] J. S. Albus A new approach to manipulator control: cerebellar model articulation control (CMAC), *Trans. ASME, Journal of Dynamics Syst., Meas. and Control.*, Vol. 97, 220-227, 1975.

[3] M.F.Azeem, M.Hanmandlu, and N.Ahmad, Structure identification of generalized adaptive neuro-fuzzy inference systems, *IEEE Trans. Fuzzy Syst.*, Vol.11, No.6, 666-681, 2003.

[4] H. R. Berenji and P. Khedkar, Clustering in Product Space for Fuzzy Inference, *In Proceeding IEEE Int. Conf. on Neural Networks*, 1402-1407, 1993.

[5] H. R. Berenji . A Reinforcement learning-based architecture for fuzzy logic control, *Int Journal Approximate reasoning*, Vol. 6, (1992), 267-292.

[6] Thomas A. Runkler and James C. Bezdek, Alternating Cluster Estimation: A New Tool for Clustering and Function Approximation, *IEEE Transaction on Fuzzy Systems*,4(1999) 377-393.

[7] M.Brown, C.J.Harris, *Neurofuzzy Adaptive Modelling and Control*, Prentice Hall: New York , 1994.

[8] G. A. Carpenter, Neural Network Models for Patterns Recognition and Associative Memory, Neural Networks, Vol. 2, 1989.

[9] W.C.Chan, C.W.Chan, K.C.Cheung and Y.Wang, Modeling of nonlinear stochastic dynamical systems using neurofuzzy networks, *38th IEEE Conference on Decision and Control*, 2643-2648, Phoenix, USA, 1999.

[10] D.S.Chen and R.C.Jain, A robust back propagation learning algorithm for fuunction approximation, *IEEE Trans. Neural Networks*, Vol.5, No.3, 1994.

[11] M.Y.Chen and D.A.Linkensm, A systematic neuro-fuzzy modeling framework with application to material property prediction, *IEEE Trans. Syst., Man, Cybern. B*, Vol.31, 781-790, 2001.

[12] S.L. Chiu, Fuzzy Model Identification based on cluster estimation, Journal of Intelligent and Fuzzy Systems, 2(3), 1994.

[13] Earl Cox, *The Fuzzy Systems Handbook: A practitioner's guide to building, using, and maintaining fuzzy systems*, AP professional, Chestnut Hill: MA, 1994.

[14] R. N. Dave & R. Krishnapuram. Robust clustering methods: A unified view, *IEEE trans. Fuzzy System*, vol 5, pp 270-293, 1997.

[15] J. E. Dayhoff, Neural Networks Architectures: An introduction, Van Nostrand Reinhold, New York, 1990.

[16] B.Egardt, *Stability of Adaptive Controllers*, Lecture Notes in Control and Information Sciences, Vol.20, Springer-Verlag, Berlin, 1979.

[17] G.A. Eiceman, Z. Karpas, Ion-Mobility Spectrometry, CRC press, 1994.

[18] M. R. Emami and I. B. Turksen. Development of a sytematic methodology of fuzzy logic modeling, *IEEE Trans. Fuzzy System.*, vol 6, pp 346-361,Aug. 1998.

[19] Interim report on gas-sensor apparatus, CONTRAVES-Environics internal report.

[20] S.Haykin, *Neural Networks- A Comprehensive Foundation*, Macmillan College Publ. Co., New York, 1994.

[21] J.J. Hopfield, Neural networks and physical systems with emergent collective computation abilities, Proceeding of the National Academy of Sciences, Vol. 79, No. 8, 2554-2558, 1982.

[22] S.I.Horikawa, T.Furuhashi and Y.Uchikawa, On fuzzy modeling using fuzzy neual networks with the back-propagation algorithm, *IEEE Trans. Neural Networks*, Vol.3, No.5, 801-806, 1992.

[23] C. Higgins and R. Goodman (1993). Learning Fuzzy Rule-based Neural Networks for Control. *Advanced in Neural Information Processing Systems*, 5:350-357.

[24] P.A.Ioannou and J.Sun, *Robust Adaptive Control*, Prentice-Hall, Inc, Upper Saddle River: NJ, 1996.

[25] A. G. Ivakhnenko, Polynomial theory of complex systems, *IEEE Trans. Systems, Man, and Cybernetics*, Vol. 12, 364-378, 1971.

[26] Z.P.Jiang and Y.Wang, Input-to-State Stability for Discrete-Time Nonlinear Systems, *Automatica*, Vol.37, No.2, 857-869, 2001.

[27] H. Ishibuchi, K. Nozaki, N. Yamamoto, and H. Tanaka, Selecting fuzzy if-then rules for classification problems using genetic algorithms, *IEEE Trans. Fuzzy Syst.*, vol. 3, pp. 260–270, 1995.

[28] J.S.Jang, C.T.Sun and E.Mizutani, *Neuro-Fuzzy and Soft Compting: A computational approach to learning and machine intelligence*, Prentice-Hall, Upper Saddle River, NJ 07458, 1997.

[29] J. S. Jang, ANFIS: Adaptive-network-based fuzzy inference system, *IEEE Trans. Syst., Man, Cybern.*, vol. 23, pp. 665–685, 1993.

[30] S.Jagannathan and F.L.Lewis, Identification of Nonlinear Dynamical Systems Using Multilayered Neural Networks, *Automatica*, Vol.32, No.12, 1707-1712, 1996.

[31] C. Z. Janikow(1996). Exemplar based Learning in Fuzzy Decision Trees. *In Proc. IEEE Int. Conf. on Fuzzy Systems* 1996, pages 1500-1505, New Orleans.

[32] C. Z. Janikow(1998). Fuzzy Decision Trees: Issues and Methods. *IEEE Trns. Systems, Man & Cybernetics*, 28(1):1-14.

[33] C.F.Juang and C.Teng Lin, An on-Line self-constructing neural fuzzy inference network and its applications, *IEEE Trans. Fuzzy Syst.*, Vol.6, No.1, 12-32, 1983.

[34] C.F.Juang, A TSK-type recurrent fuzzy network for dynamic systems processing by neural network and genetic algorithm, *IEEE Trans. Fuzzy Syst.*, Vol.10, 155-170, 2002.

[35] N. Kasabov, Evolving fuzzy neural networks for supervised/unsupervised, *IEEE Trasc. Of systems, man and cibernetics, Part B*, Vol. 31, No. 6, 902-918, 2001.

[36] N. Kasabov, Evolving connectionist systems for en línea, knowledge-based learning: principles and applications, *IEEE Systems, Man, and Cybernetics*, 1-31. 1999.

[37] F. Klawonn and R. Kruse), *Clustering Methods in Fuzzy Control*, 1995.

[38] F. Klawonn and R. Kruse (1997). *Constructing a Fuzzy Controller from Data. Fuzzy Sets and Systems*, 85:177-193.

[39] G. Klir and B. Yuan, *Fuzzy Sets and Fuzzy Logic: Theory and applications*, Prentice Hall, Upper Saddle River, New Jersey 07458 USA, 1995.

[40] T. Kohonen, The Self-Organizing Map, Proceedings of the IEEE, Vol. 78, No. 9, 1464-1480, 1990.

[41] B. Kosko, *Neural Networks and Fuzzy Systems*, Prentice-Hall, Englewood Cliffs, 1992.

[42] C. G. Looney. A fuzzy clustering and fuzzy merging algorithm.

[43] S. Abe and M. S. Lan, A method for fuzzy rule extraction directly from numerical data and its application to pattern classification, *IEEE Transaction on Fuzzy Systems*, 3(1995) 18-28.

[44] Y.G.Leu, T.T.Lee and W.Y.Wang, Observer-based adaptive fuzzy-neural control for unknown nonlinear dynamical systems, *IEEE Trans. Syst., Man, Cybern. B*, Vol.29, 583-591, 1999.

[45] F.L.Lewis, A.Yesildirek and K.Liu, Multilayer Neural-Net Robot Controller with Guaranteed Tracking Performance, *IEEE Trans. Neural Networks*, Vol.7, No.2, 388-399, 1996.

[46] C.T.Lin and G.Lee, *Neural fuzzy systems: A neural-fuzzy synergism to intelligent systems*, Prentice-Hall Inc., NJ, 1996.

[47] C.T.Lin, A neual fuzzy control system with structure and parameter learning, *Fuzzy Sets and Systems.*, Vol.70, 183-212, 1995.

[48] C. T. Lin, *Neural Fuzzy Control Systems with Structure and Parameter Learning.* New York: World Scientific, 1994.

[49] R. P. Lippmann, An introduction to computing with neural nets, IEEE ASSP Magazine, Vol. 4, No. 2, 4-22, 1987.

[50] D.P.Mandic, A.I.Hanna and M.Razaz, A normalized gradient descent algorithm for nonlinear adaptive filters using a gradient adaptive step size, *IEEE Signal Processing Letters*, Vol.8, No.11, 295-297, 2001

[51] E.H.Mamdani, Application of fuzzy algorithms for control of simple dynamic plant, *IEE Proceedings - Control Theory and Applications*, Vol.121, No.12, 1585-1588, 1976.

[52] P.A.Mastorocostas and J.B.Theocharis, A recurrent fuzzy-neural model for dynamic system identification, *IEEE Trans. Syst., Man, Cybern. B*, Vol.32, 176-190, 2002.

[53] K.S.Narendra and S.Mukhopadhyay, Adaptive Control Using Neural Networks and Approximate Models, *IEEE Trans. Neural Networks*, Vol.8, No.3, 475-485, 1997.

[54] K.S. Narendra and K. Parthasarathy, Identification and control of dynamical systems using neural networks, *IEEE Trans. Neural Networks*, Vol.1,No.1,4-26, 1990.

[55] D. Nauck and R. Kruse, A Neural Fuzzy Controller Learning by Fuzzy Error Peopagation, *In. Proceesing. Workshop of the North American Fuzzy Information Processing Society (NAFIPS)*, 388-397, (1992).

[56] D. Nauck and R. Kruse, A Fuzzy Neural Network Learning Fuzzy Control Rules and Membership Functions by Fuzzy Error Backpropagation, *Proceedings IEEE Int. Conf on Neural Networks*, 1022-1027, 1993.

[57] D. Nauck and R. Kruse, NEFCON-I: An X-Windows based simulator for neural fuzzy controllers, *Int. Proc. IEEE Int. Computational Intelligence (WCCI'94)*, 1994.

[58] D. Nauck and R. Kruse, NEFCLASS: A Neuro-Fuzzy Approach for the Classification of Data, *Proceeding ACM Symposium on Applied Computing*, 461-465, 1995.

[59] D. Nauck and R. Kruse. Designing Neuro-Fuzzy Systems Through Backpropagation. *In W. Pedrycz, ed:Modelling: Paradigms and Practice, Kluwer*, 203-228, 1996.

[60] D. Nauck and R. Kruse. Neuro-Fuzzy Systems Research and Applications outside of Japan. I*n M. Umano, I. Hayashi and T. Furuhashi, eds: Fuzzy-Neural Networks, Soft Computing Series*, 108-134, 1996.

[61] H. T. Nguyen and M. Sugeno, *Fuzzy Systems: Modeling and control*, Kluwer Academic Publishers, Norwell, Massachusetts 02061 USA, 1998.

[62] A. Nürnberger, D. Nauck, R. Hruse, *Neuro-fuzzy control based on the NEFCON-model: recent developments*, Germany, Springer Verlag, 168-182, 1999.

[63] T. Katto, H. Paakanen, and T. Karhappa, Detection of CWA by means of aspiration condenser type IMS, Environics Oy, Mikkeli, Finland, Technical Report.

[64] R.P Paiva, *Identification neuro-fuzzy: Aspectos de interpretabilidad,* M en C. Tesis, Dept. of Informatics Engineering, Faculty of Science and technology, University of Coimbra, Portugal, 1999.

[65] D. T. Pham and L. Xing, *Neural Networks for Identification, Prediction and Contro*l, London:Springer-Verlag, 1995.

[66] P. Puumalainen, Method for detection of foreign matter in gases, U.S. Patent 5047723, Sept 10, 1991.

[67] T. A. Runkler and J. C. Besdek. Alternating cluster estimation: A new tool for clustering and function approximation. *IEEE Trans.Fuzzy Systems*, vol 7, pp 377-393, Aug 1999.

[68] S. Shrier., R.L. Barron., L. O. Gilstrap, Polynomial and neural networks: analogies and engineering applications, *Proceedings of IEEE International Conference on Neural Networks*, Vol. II, 431-439, 1987.

[69] S. M. Sulzberger, N. N. Tschichold, and S. J. Vestli, FUN: Optimization of Fuzzy Rule Based Systems Using Neural Networks. *Proceedings IEEE Int. Conf. on Neural Networks*, 312-316, 1993.

[70] E. Sacristán, Ion mobility method for inhalation anesthesia monitoring, Ph.d. Dissertation, Worcester Polythechnic Institute, Worcester, MA, 1993.

[71] J. Shing, R. Jang and Ch. T. Sun, Neuro-Fuzzy Modeling and Control, Proceedings of the IEEE, Vol. 83, No. 3, 378-406, 1995.

[72] Q.Song, Robust Training Algorithm of Multilayered Neural Networks for Identification of Nonlinear Dynamic Systems, *IEE Proceedings - Control Theory and Applications*, Vol.145, No.1, 41-46,1998

[73] M. Sugeno and T. Yasukawa. A fuzzy logic based approach to qualitative modeling. *IEEE Trans. Fuzzy System.*, vol 1, pp 7-31, Aug. 1993.

[74] T.Takagi and M.Sugeno, Fuzzy identification of systems and its applications to modeling and control, *IEEE Trans. Syst., Man, Cybern.*, Vol.15, 116-132, 1985.

[75] H. F. Tammet, The aspiration method for the determination of atmospheric-ion spectra, *Scientific notes of tartu state university*, No. 195, Trans. Air Ionization and Electroaerosols, 1967, translated from Russian Irael Program for Scientific Translations, Jerusalem 1970.

[76] H.H.Tsai and P.T.Yu, On the optimal design of fuzzy neural networks with robust learning for function approximation, *IEEE Trans. Syst., Man, Cybern. B*, Vol.30, 217-223, 2000.

[77] Rigoberto Toxqui, *Redes neuronales difusas dinámicas para identificación y control adaptable*, Tesis (M.C.), CINVESTAV-IPN, 2003.

[78] S.Wu and M.J.Er, Dynamic fuzzy neural networks- a novel approach to function approximation, *IEEE Trans. Syst., Man, Cybern. B*, Vol.30, 358-364, 2000.

[79] L.X. Wang, *A course in Fuzzy Systems and Control*, Prentice Hall PTR, Upper Saddle River, New Jersey 07458 USA, 1997.

[80] L.X.Wang, *Adaptive Fuzzy Systems and Control*, Prentice-Hall, Englewood Cliffs, New Jersey, 1994.

[81] C.H.Wang, H.L.Liu and C.T.Lin, Dynamic optimal learning rates of a certain class of fuzzy neural networks and its applications with genetic algorithm, *IEEE Trans. Syst., Man, Cybern. B*, Vol.31, 467-475, 2001.

[82] W.Y.Wang, Y.G.Leu and C.C.Hsu, Robust adaptive fuzzy-neural control of nonlinear dynamical systems using generalized projection updated law and variable structure controller, *IEEE Trans. Syst., Man, Cybern. B*, Vol.31, 140-147, 2001.

[83] W.Y.Wang, T.T.Lee, C.L.Liu and C.H.Wang, Function approximation using fuzzy neural networks with robust learning algorithm, *IEEE Trans. Syst., Man, Cybern. B*, Vol.27, 740-747, 1997.

[84] L.-X. Wang and J. M. Mendel (1992). Generation Rules by Learning from Examples. *IEEE Trans. Systems., Man., Cybernetics.*, 22(6):1414-1427.

[85] P. j. Werbos, Backpropagation through time: What it does and how to do it, Proceedings of the IEEE, Vol. 78, No. 10, 1550-1560, 1990.

[86] R.R. Yager and D.P. Filev. Approximate clustering via the mountain method, *IEEE Transactions on Systems, Man, and Cybernetics*,24:1279-1284,1994.

[87] J. Yen and R. Langari, *Fuzzy Logic: Intelligence, control, and information*, Prentice Hall, Upper Saddle River, New Jersey 07458 USA, 1998.

[88] Wen Yu, Xiaoou Li, Fuzzy identification using fuzzy neural networks with stable learning algorithms, *IEEE Transactions on Fuzzy Systems*, Vol.12, No.3, 411-420, 2004.

[89] W.Yu and X. Li, Some new results on system identification with dynamic neural networks, *IEEE Trans. Neural Networks*, Vol.12, No.2, 412-417, 2001.

[90] W.Yu, A.S. Poznyak and X.Li, Multilayer Dynamic Neural Networks for Nonlinear System On-line Identification, *International Journal of Control*, Vol.74, No.18, 1858-1864,2001.