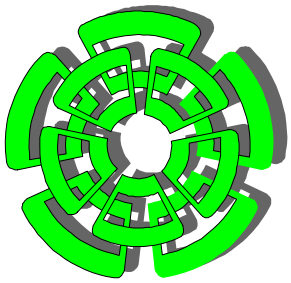

CENTRO DE INVESTIGACIÓN Y DE ESTUDIOS AVANZADOS DEL INSTITUTO POLITÉCNICO NACIONAL

DEPARTAMENTO DE CONTROL AUTOMÁTICO

Modelado con selección de estructura automática usando SVM difusos

TESIS QUE PRESENTA EL:

M. en C. Julio César Tovar Rodríguez.

Para obtener el grado de
Doctor en Ciencias en la Especialidad de Control Automático

DIRECTOR DE TESIS:
Dr. Wen Yu Liu

México, D.F., 2008

Índice general

Índice de figuras

Capítulo 1

Introducción

La mayoría de los sistemas actuales a controlar presentan algún grado de no linealidad, incertidumbres en sus entradas como en su estructura, son variantes en el tiempo, etc.. Las incertidumbres en su entrada son causadas por imprecisión en los parámetros ó desconocimiento de los mismos, mientras que las incertidumbres en su estructura se refieren a las dinámicas no modeladas, como fricción no lineal, acoplamiento de engranes, ruido en los sensores, perturbaciones externas, etc.. Las técnicas de control convencional no pueden resolver ambos problemas.

Uno de los métodos que más interes a despertado es el control inteligente, el cual está basado en redes neuronales y sistemas difusos. Las redes neuronales presentan la propiedad de aprender y pueden aproximar una función no lineal con algún grado de exactitud. Esta característica es utilizada para controlar modelos de procesos complejos y compensar incertidumbres en las entradas del sistema, sin embargo el proceso de aprendizaje degrada el comportamiento de la respuesta transitoria del sistema. La ventaja de un control difuso es que agrega conocimiento humano y experiencia apriori a la estrategia de control.

Una limitación importante que presentan los controladores difusos convencionales es que el número de reglas se incrementa exponencialmente con el número de variables involucradas. Con n variables de entradas y m conjuntos difusos (funciones de pertenencia) para cada variable, necesitan m^n reglas para construir un controlador difuso completo. Cuando n se

incrementa, la base de reglas rápidamente puede sobre cargar la memoria y puede hacer al controlador difuso difícil de implementar. Uno de los métodos desarrollados para para resolver este problema fue propuesto por Raju, Zhou and Kisner. Los sistemas jerárquicos consisten de un número de subsistemas de baja dimensionalidad conectados de una manera jerárquica. La estructura jerárquica tiene la propiedad de que el número total de reglas para los sistemas difusos se incrementa linealmente con respecto al número de variables de entrada. Para la estructura jerárquica, se observa que se definen m conjuntos difusos para cada variable de entrada, cada subsistema difuso de baja dimensionalidad consiste de m^2 reglas y el número total de reglas es $(n-1)m^2$, el cual es una función lineal del número de las variables de entrada n.

Otra metodología usada son los *Support Vector Machines* (SVM's) que fueron inventados por Vladimir Vapnik, el cual es un método para la creación de funciones de un conjunto de datos entrenados. La función puede ser una función de clasificación (la salida es binaria: es la entrada sabre una categoria) o la función puede ser una función de regresión general.

En este trabajo se anlizará la estructura de los SVM´s para la identificación en línea de sistemas no lineales, que junto con el agrupamiento en línea, redes neuronales y sistemas difusos nos ayudarán a identificar sistemas no lineales.

1.1. Motivación

La motivación para la realización de este trabajo fue encontrar una forma de generar automaticamente reglas difusas, para lo cual se utiliza la identificación en línea basado en datos (se conocen muchos trabajos donde el agrupamiento de datos se da fuera de línea). Estos se pueden ver como una caja negra (redes neuronales) o como una caja gris (reglas) en nuestro trabajo se verán estos datos como una caja gris. La generación de reglas difusas desde los datos, se pueden generar una regla por cada dato o a partir de un conjunto de datos generar una regla.

En el enfoque de agrupamiento existen dos manerar de construcción, una es obtener este agrupamiento fuera de línea y el otro método es en línea (motivo de este trabajo). Dentro

del agrupamiento en línea se generarán grupos y por cada grupo se genera una regla.

También se propone un enfoque de *Support Vector Machines* el cual genera un número de su Support Vectors en cada grupo y estos a su vez generarán el número de reglas de acuerdo a la posición de support vectors. Una vez generadas estas reglas se entrenan en línea para obtener las funciones de membresía. El kernel normal dentro de los *Support Vector Machines* es "rizado" el cual no es apropiado, en este trabajo se propone un kernel difuso.

1.2. Objetivo de la tesis

El objetivo de la tesis se enfoca principalmente en dos aspectos:

El primero es como obtener multi-modelos y como escoger estos multi-modelos. Para ello se realiza una identificación en línea de sistemas dinámicos no lineales mediantes la integración de agrupamiento en línea, lo cual generará grupos y a su vez se generá un modelo por cada grupo; y mediante el enfoque de *Support Vector Machines* difuso aplicado en cada grupo, generará un número de Support Vector, los cuales se verán como el número de nodos ocultos para la utilización de redes neuronales.

El segundo objetivo es saber cuando se mostrará el cambio de modelo. Para ello se puede ver a los SVM´s dentro de una ventana, es decir, el número de Support Vectors dará el número de reglas y esto a su vez generará el sistema difuso (modelado difuso).

Después del entrenamiento de parámetro se utiliza el enfoque multimodelo con *Support Vector Machines* para realizar diversos esquemas de identificación para sistemas dinámicos no lineales, para resolver el problema de reglas neuro-difusas convencionales.

1.3. Estructura de la tesis

El **capítulo uno** menciona los aspectos generales del control inteligente, la motivación que se tuvo para el análisis de sistemas que presentan la utilización de SVM´s difusos para la identificación en línea de sistemas no lineales y los objetivos que se plantearon para su estudio.

El **capítulo dos** hace una breve reseña sobre la identificación vía *Support Vector Machines* y multimodelo, la estructura y el proceso de aprendizaje que hace posible su interrelación con el medio que lo rodea.

El **capítulo tres** estudia el modelado vía agrupamiento en línea para datos entrada/salida así como las características más importantes de la extracción de reglas difusas mediante SVM y funciones de membresía entrenadas.

En el **capítulo cuatro** se detalla el funcionamiento del kernel difuso y la formulación de *support vector machines* difusos (FSVM).

El **capítulo cinco** muestra la capacidad de las redes neuronales tipo RBF y la seleción de multimodelos para aproximar funciones, se realizán análisis de estabilidad para asegurar la convergencia de los pesos en el algoritmo de aprendizaje.

El **capítulo seis** muestra los diferentes tipos de multi redes neuronales para identificación de sistemas no lineales y la selección de estructura automática vía support vector machine difuso

El **capítulo siete** se dan las conclusiones, se presenta el trabajo futuro y el estudio de otros tópicos de interés relacionados con los sistemas neuro-difusos.

1.4. Publicaciones

Revista

1. Julio César Tovar, Wen Yu, Nonlinear System Modeling via On-line Clustering and Fuzzy Support Vector Machine, *International Journal of Modelling, Identification and Control,* Vol. 3, No. 4, pp.405-420, 2008.

2. Julio César Tovar, Wen Yu, Multiple fuzzy neural networks for nonlinear system identification wi-ésimo automated structure selection, *Fuzzy Sets and Systems,* submitted.

Capítulo de libro

1. Julio César Tovar, Wen Yu, Xiaoou Li, Fuzzy Modeling via On-line Clustering and Support Vector Machine, *Advances Intelligent Computing -ésimoeories and Applications*, Srpinger-Verlgag, CCIS 2, pp.294-303, 2007.

2. Julio César Tovar, Wen Yu, On-Line Modeling Via Fuzzy Support Vector Machines, *A. Gelbukh and E.F. Morales (Eds.): MICAI 2008*, LNAI 5317, Springer-Verlag Berlin Heidelberg, pp. 220–229, 2008

Congreso

1. Julio César Tovar, Wen Yu, Fuzzy neural modeling via clustering and support vector machines, *16-ésimo IEEE Conference on Control Applications, CCA'08*, Singapore, pp.24-30, 2007

2. Julio César Tovar, Wen Yu, Automated fuzzy neural networks for nonlinear system identification, *IEEE International Conference on Fuzzy Systems, Fuzzy 2008*, pp.1159-1165, HongKong, China, June 1-6 , 2008

Capítulo 2

Support Vector Machines y multimodelo

Los *Support Vector Machines* (SVM's) fueron inventados por Vladimir Vapnik, es un método para la creación de funciones de un conjunto de datos entrenados. La función puede ser una función de clasificación (la salida es binaria: es la entrada sabre una categoria) o la función puede ser una función de regresión general.

Para clasificación, los SVM's operan encontrando un hiperplano en el espacio de posibles entradas. Este hiperplano intentará separar muestras positivas de muestras negativas. Esta separación escogerá la distancia mas grande del hiperplano entre las muestras positivas y negativas mas cercanas a esta. Intuitivamente, esta hace la clasificación correcta para analizar los datos que están cerca.

El reconocimiento de patrones binarios envuelve construcciones de reglas de desición para clasificar vectores dentro de una de dos clases basadas en conjuntos de vectores entrenados cuya clasificación es conocida a priori. Los SVM´s [70] implican un mapeo de datos entrenados dentro de espacios característico de dimensión grande. Un hiperplano (superficie de decisión) es cuando se construye este espacio característico que bisecciona las dos categorías y maximiza el margen de separarción entre si misma y sus puntos quedando cerca de esta (llamados support vector). Esta superficie de decisión puede entonces usarse como base para

clasificación de vectores de clasificación desconocida.

Las ventajas principales del enfoque los SVM´s son las siguientes:

· El implemento de losSVM´s a una forma de estructura de riesgo de minimización [70]. El intento de encontrar un compromiso entre la minimización de riesgo empírico y la prevención de sobreajuste.

· El problema es un problema de programación cuadráctica convexa. No existe un mínimo global y el problema se resuelve usando técnicas de programación cuadrática.

· El clasificador resultante puede ser especificado completamente en términos de sus support vectors y su función tipo kernel.

2.1. Support Vector Machine

Es una introducción al análisis del discriminante del kernel Fisher, el análisis de los componentes del kernel principal (PCA). Una posible formalización de esta tarea es la de estimar una función $f : \mathbb{R}^n \to \{-1, +1\}$, usando un par de datos entrada-salida generados independientes distribuidos identicamente (i.i.d.) de acuerdo a una distribución de probabilidad desconocida $P(X, y)$

$$(X_1, y_1), \ldots, (X_{m,}, y_m) \in \mathbb{R}^n \times Y, \qquad Y = \{-1, +1\}$$

La mejor función f que se puede obtener es una minimización del error esperado (riesgo)

$$R[f] = \int l(f(x), y)\, dP(x, y)$$

donde l denota una función de perdida escogida adecuadamente, $l(f(x), y) = \Theta(-yf(x))$, donde $\Theta(z) = 0$ para $z < 0$ y $\Theta(z) = 1$ en otro caso. (Ver Figura 2.1.)

2.1.1. Margenes y Dimensión VC (Vapnik-Chervonenkis)

Si el muestreo de entrenamiento es separable por el hiperplano, se escoge la función de la siguente manera:

$$f(x) = (w \cdot x) + b$$

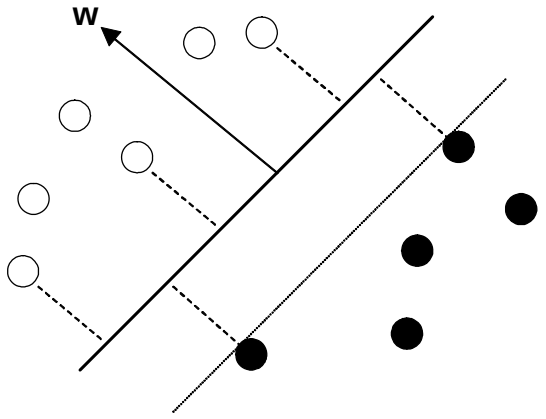

Figura 2.1: Separación de los Support Vector Machine.

Para clases de hiperplanos mostrado en la figura 2.1, su dimensión VC puede ser acotada en términos de otra cantidad, llamada *margen*. El margen se define como la distancia mínima de una muestra a la superfice de resolución. El margen a su vez se puede medir por la longitud del vector w, y como se supuso que el muestreo de entrenamiento es separable se puede reescalar w y b tal que los puntos cercanos al hiperplano satisfase $|(w \cdot x_i) + b| = 1$. Si se consideran dos muestras x_i y x_2 de diferentes clases con $(w \cdot x_1) + b = 1$ y $(w \cdot x_i) + b = -1$, entonces el margen esta dado por la distancia de esos dos puntos, medido perpendicular al hiperplano, i.e., $w/\left\|w\right\| \cdot (x_1 - x_2) = 2/\left\|w\right\|$. Este resultado liga la dimensión VC de las clases de hiperplanos separables al margen o la longitud del vector de peso w respectivamente.Se puede expresar como:

$$h \leq \Lambda^2 R^2 + 1 \text{ y } \left\|w\right\|_2 \leq \Lambda$$

donde R es el radio de la bola más pequeña alrededor del dato. [41]

Perceptron de Rosenblatt

El primer algoritmo interativo para el aprendizaje de clasificacción lineal fue propuesto por Frank Rosenblatt en 1956 para el perceptron. Se utiliza en procedimientos *en línea y manejo del error (on-line* y *mistake-driven)*, los cuales comienzan con un vector de los pesos w_0 (por lo regular $w_0 = 0$) y cada vez se adaptan los puntos de entrenamiento no clasificados

por los pesos recurrentes. El algoritmo actualiza el vector de pesos y genera la pendiente (inclinación) del hiperplano.

Este procedimiento garantiza la convergencia donde existe un hiperplano que clasifica correctamente los datos clasificados.[4]

2.1.2. Condiciones Karush-Kuhn-Tucker

Las condiciones Karush-Kuhn-Tucker (KKT) juegan un papel principal tanto en la teoría y práctica de la construcción de la optimización, estas condiciones se expresan como:

$$\begin{aligned}
\frac{\partial}{\partial w_v} L_p &= w_v - \sum_i \alpha_i y_i x_{iv} = 0,\ v = 1, \ldots, d \\
\frac{\partial}{\partial b} L_p &= -\sum_i \alpha_i y_i = 0 \\
y_i \left(\mathbf{x}_i \cdot \mathbf{w} + b \right) - 1 &\geq 0,\ i = 1, \ldots, l \\
\alpha_i &\geq 0\ \forall i \\
\alpha_i \left(y_i \left(\mathbf{x}_i \cdot \mathbf{w} + b \right) - 1 \right) &= 0\ \forall i
\end{aligned}$$

La condición KKT satisface la solución para cualquier problema de optimización construida (convexa o no), con cualquier tipo de construcción.

2.1.3. Support Vector Machines Lineal

Caso separable

La etiqueta de datos separables $\{\mathbf{x}_i y_i\}$, $i = 1, \ldots, l$, $y_i \in \{1, -1\}$, $\mathbf{x}_i \in \mathbb{R}^d$. Suponer que existe algún hiperplano el cual se puede separar en positivo y negativo, es decir, un hiperplano separable. Los puntos x los cuales satisfacen $\mathbf{w} \cdot \mathbf{x} + b = 0$ sobre el hiperplano, donde $\mathbf{w}$ es normal al hiperplano. $|b| \,/\, \|\mathbf{w}\|$es la distancia perpendicular del hiperplano al origen, y $\|\mathbf{w}\|$es la norma Euclidiana de $\mathbf{w}$. Sea $d_+ \, (d_-)$ la distancia más corta desde el hiperplano separable a los datos positivos (negativos), es decir: (ver Figura 2.2.)

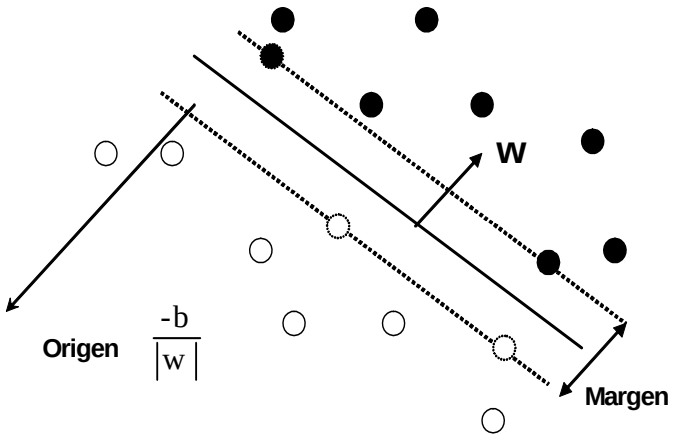

Figura 2.2: Caso separable.

$$\begin{aligned} \mathbf{x}_i \cdot \mathbf{w} + b &\geq +1 \text{ para } y_i = +1 \\ \mathbf{x}_i \cdot \mathbf{w} + b &\leq -1 \text{ para } y_i = -1 \end{aligned}$$

con $H_1 : \mathbf{x}_i \cdot \mathbf{w} + b = 1$ con $\mathbf{w}$ normal y la distancia perpendicular desde el origen $|1 - b| / \|\mathbf{w}\|$. Similarmente, $H_2 : \mathbf{x}_i \cdot \mathbf{w} + b = -1$ con $\mathbf{w}$ normal y la distancia perpendicular desde el origen $|-1 - b| / \|\mathbf{w}\|$, $d_+ = d_- = 1/\|\mathbf{w}\|$ y el margen es simplemente $2/\|\mathbf{w}\|$.

Si se introduce un multiplicador de Lagrange positivo α_1, $i = 1, \ldots, l$, uno para cada desigualdad de $y_i (\mathbf{x}_i \cdot \mathbf{w} + b) - 1 \geq 0 \ \forall i$. El Lagragiano se escribe como:

$$L_p \equiv \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{i=1}^{l} \alpha_i y_i (\mathbf{x}_i \cdot \mathbf{w} + b) + \sum_{i=1}^{l} \alpha_i$$

Los algoritmos dentro del espacio caraterístico hacen uso de la siguiente idea: via mapeo no lineal

$$\begin{aligned} \phi &: \mathbb{R}^n \to \mathcal{F} \\ x &\longmapsto \phi(x) \end{aligned}$$

los datos $x_1, \ldots, x_n \in \mathbb{R}^n$ son mapeados dentro de un espacio característico de dimensión superior $\mathcal{F}$ $((\phi(x_1), y_1), \ldots, (\phi(x_n), y_n) \in \mathcal{F} \times Y)$. Las generalizaciones encontradas son:

- Para $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$, y $d \in \mathbb{N}$ la función kernel

$$k(\mathbf{x}, \mathbf{y}) = (\mathbf{x} \cdot \mathbf{y})^d$$

  calcula el producto escalar dentro del espacio de todos los productos del vector de entradas d (nominales) de $\mathbf{x}$ y $\mathbf{y}$.

- Si k:$\mathcal{C} \times \mathcal{C} \rightarrow \mathbb{R}$ es un kernel continuo de un operador integral positivo dentro del espacio de Hilbert $L_2(\mathcal{C})$ sobre un conjunto compacto $\mathcal{C} \subset \mathbb{R}^n$, i.e.,

$$\forall\ f \in L_2(\mathcal{C}) : \int_{C^{\in}} k(x, y) f(x) f(y)\, dxdy \geq 0$$

  entonces existe un espacio $\mathcal{F}$ y un mapeo $\mathbb{R}^n \rightarrow \mathcal{F}$ tal que $k(x, y) = (\phi(x_n) \cdot \phi(y_n))$. Lo anterior se ve desde el teorema de Mercer que dice que cualquier kernel de un operador integral positivo se puede expander en sus funciones propias ψ_j $(\lambda_j > 0, N_F \leq \infty)$[41]

$$k(x, y) = \sum_{j=1}^{N_F} \lambda_j \psi_j(x) \psi_j(y)$$

El caso no separable

Se aplica cuando los datos son no separables, no se encuentra una solucion factible, es decir, la función objetivo crece arbitariamente grande.

$$\begin{aligned} \mathbf{x}_i \cdot \mathbf{w} + b &\geq +1 - \xi_i \text{ para } y_i = +1 \\ \mathbf{x}_i \cdot \mathbf{w} + b &\leq -1 + \xi_i \text{ para } y_i = -1 \\ \xi_i &\geq 0\ \forall i \end{aligned}$$

Los mulplicadores de Lagrange (problema dual) es:

$$L_D \equiv \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j$$

Sujeto:

$$\begin{aligned} 0 &\leq \alpha_i \leq C, \\ \sum_i \alpha_i y_i &= 0 \end{aligned}$$

la solución queda:

$$\mathbf{w} = \sum_{i=1}^{N_S} \alpha_i y_i \mathbf{x}_i$$

donde N_S es el número de vectores soportados y C es una cota superior.

La función de Lagrange es:

$$L_p \equiv \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_i \xi_i - \sum_i \alpha_i \{y_i (\mathbf{x}_i \cdot \mathbf{w} + b) - 1 + \xi_i\} - \sum_i \mu_i \xi_i$$

donde μ_i son los multiplicadores de lagrange introducidos para asegurar la positividad de ξ_i.

La condición KKT será (ver Figura 2.3.)

$$\begin{aligned} \frac{\partial}{\partial w_v} L_p &= w_v - \sum_i \alpha_i y_i x_{iv} = 0,\ v = 1, \ldots, d \\ \frac{\partial}{\partial b} L_p &= -\sum_i \alpha_i y_i = 0 \\ \frac{\partial}{\partial \xi_i} L_p &= C - \alpha_i - \mu_i = 0 \\ y_i (\mathbf{x}_i \cdot \mathbf{w} + b) - 1 + \xi_i &\geq 0,\ i = 1, \ldots, l \\ \xi_i &\geq 0\ \forall i \\ \alpha_i &\geq 0\ \forall i \\ \mu_i &\geq 0\ \forall i \\ \alpha_i \{y_i (\mathbf{x}_i \cdot \mathbf{w} + b) - 1 + \xi_i\} &= 0\ \forall i \\ \mu_i \xi_i &= 0 \end{aligned}$$

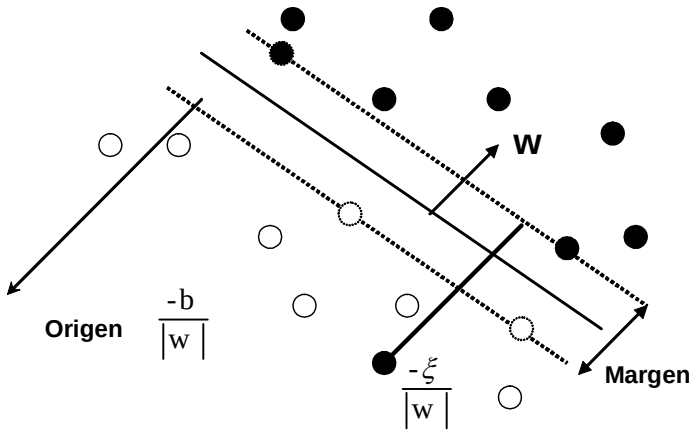

Figura 2.3: Caso no separable.

2.2. Support Vector para Identificación

Un acercamiento al procesamiento de señales digitales es el propósito del modelo compuesto por dos procesos en tiempo discreto (DTP), las cuales son las entradas y salidas linelaes, para sistemas invariantes en el tiempo (LTIS). Un LTIS usualmente se aproxima usando la función de transferencia racional que puede ser todos ceros o un promedio del movimiento (MA), o todos ceros o autoregresivo (AR), o polo-cero general (ARMA).

Los métodos de estimación ARMA clasicos presentan algunas limitaciones:

- Análisis de DPT con muestras atípicas (parte aislada) no es fácil ni inmediato, y es usualmente es logrado mediante métodos hurísticos y métodos de inspección visual.

- En términos generales, los métodos ARMA requiere una determinación previa del modelo complejo o número de parámetros dentro del modelo.

- Finalmente, las superficies de error no son convexas en algunos casos.

2.2.1. Formulación SVM-ARMA

Considerar dos DTPs $\{\mathbf{x}_n\}$ y $\{\mathbf{y}_n\}$, los cuales son las entradas y salidas respectivamente, de una LTIS racional. La ecuación diferencial correspondiente es

$$\mathbf{y}_n = \sum_{i=1}^{P} a_i y_{n-i} + \sum_{i=1}^{Q} b_j x_{n-j+1} + e_n$$

donde $\{a_n\}$ y $\{b_n\}$ son coeficientes del P AR y Q MA, respectivamente, y $\{e_n\}$ es un DTP estandar para el efecto de medición del error.

En general, los algoritmos SVM para clasificación lineal y problemas de regresión minimiza la función de costo para los residuos (CFR) que es llamada Función de pérdida de Vapnik (o $\varepsilon-$sensitivo) la cual es dada por

$$L_\varepsilon = \begin{cases} |e_n| - \varepsilon, & \text{si } |e_n| \geq \varepsilon \\ 0, & \text{si } |e_n| < \varepsilon \end{cases}$$

El uso de CFR Vapnik más una regularización L_2, incluyendo ambos métodos del modelo de coeficientes AR y MA, para crear lo que se llama modelado SVM-ARMA, esto corresponde a una minimización no construida de [29]

$$L_P\left(a_i, b_j, e_n\right) = \frac{1}{2}\left(\sum_{i=1}^{P} a_i^2 + \sum_{i=1}^{Q} b_j^2\right) + C\sum_{i=k_0}^{N} L_\varepsilon\left(e_n\right)$$

El problema dual o función de Lagrange para este problema se obtiene por introducción

de un coeficiente no negativo (multiplicador de Lagrange) [29]

$$\begin{aligned} & L_P\left(a_i, b_j, \xi_n, \xi_n^*, \alpha_n, \alpha_n^*, \beta_n, \beta_n^*\right) \\ = \; & C\sum_{n=k_0}^{N}\left(\xi_n + \xi_n^*\right) \\ & + \frac{1}{2}\left(\sum_{i=1}^{P} a_i^2 + \sum_{j=1}^{Q} b_j^2\right) - \sum_{n=k_0}^{N}\left(\beta_n\xi_n + \beta_n^*\xi_n^*\right) \\ & + \sum_{n=k_0}^{N}\alpha_n\left(y_n - \sum_{i=1}^{P} a_i y_{n-i} - \sum_{i=1}^{P} b_j x_{n-j+1} - \varepsilon - \xi_n\right) \\ & + \sum_{n=k_0}^{N}\alpha_n^*\left(-y_n + \sum_{i=1}^{P} a_i y_{n-i} + \sum_{i=1}^{P} b_j x_{n-j+1} - \varepsilon - \xi_n^*\right) \end{aligned}$$

2.2.2. Support Vector Machine Robusto

El objetivo del SVM robusto es resolver el problema de sobreajuste con las partes aisladas haciendo dos clases no separables. El algoritmo se desarrolló usando los estandares de SVM para el caso no lineal separable. Se consideran las muestras entrenadas

$$\left(\mathbf{x}_1, y_1\right)\ldots\left(\mathbf{x}_i, y_i\right), \quad \mathbf{x}_i \in \mathcal{X}, \quad y_i \in \{-1, 1\}, \quad i = 1, \ldots, l$$

Estas dos clases no pueden ser separables sin un error por un hiperplano puesto que existen patrones mal clasificados de mediciones de ruidos dentro de las muestras entrenadas. En este caso no existe una función de desición, tal que, las desigualdades

$$y_i f\left(\mathbf{x}_i\right) \geq +1, \quad i = 1, \ldots, l$$

sean verdaderas.

Dentro del entrenamiento estandar del SVM para conjunto de datos no separables, representado por el siguiente problema de optimización

$$\begin{aligned} \text{mín}\, \phi\left(\mathbf{w}\right) &= \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{l}\xi_i \\ \textit{sujeto a} \;\; y_i f\left(\mathbf{x}_i\right) &\geq +1 - \xi_i \end{aligned}$$

donde $\mathbf{w}$ es el peso de la función kernel y C es una constante para la variable débil $\{\xi_i\}_{i=1}^{l}$.

Se formula primero el problema del algoritmo robusto solo minimizando el margen del peso $\mathbf{w}$ en lugar de minimizar la suma del margen y la mal clasificación del error dentro del SVM estandar. Se introduce una nueva variable débil $\lambda D^2\left(\mathbf{x}_i, \mathbf{x}_{y_i}^*\right)$ a cambio de $\{\xi_i\}_{i=1}^{l}$ dentro del SVM entrenado

$$\begin{aligned} \text{mín}\, \phi(\mathbf{w}) &= \frac{1}{2}\mathbf{w}^T\mathbf{w} \\ sujeto\ a\ \ y_i f(\mathbf{x}_i) &\geq +1 - \lambda D^2\left(\mathbf{x}_i, \mathbf{x}_{y_i}^*\right) \end{aligned}$$

donde $\lambda \geq 0$ es un parámetro de medición preseleccionado, y $D^2\left(\mathbf{x}_i, \mathbf{x}_{y_i}^*\right)$ representa la distancia normalizada entre cada dato y el centro del respectivo, dentro del espacio kernel, la cual se calcula

$$\begin{aligned} D^2\left(\mathbf{x}_i, \mathbf{x}_{y_i}^*\right) &= \left|\phi(\mathbf{x}_i) - \phi\left(\mathbf{x}_{y_i}^*\right)\right|^2 / D_{\text{máx}}^2 \\ &= [\phi(\mathbf{x}_i)\cdot\phi(\mathbf{x}_i) - 2\phi(\mathbf{x}_i)\cdot\phi\left(\mathbf{x}_{y_i}^*\right) \\ &\quad +\phi\left(\mathbf{x}_{y_i}^*\right)\cdot\phi\left(\mathbf{x}_{y_i}^*\right)]/D_{\text{máx}}^2 \\ &= [k(\mathbf{x}_i, \mathbf{x}_i) - 2k\left(\mathbf{x}_i, \mathbf{x}_{y_i}^*\right) \\ &\quad +k\left(\mathbf{x}_{y_i}^*, \mathbf{x}_{y_i}^*\right)]/D_{\text{máx}}^2 \end{aligned}$$

donde $\{\phi(\mathbf{x}_i)\}_{i=1}^{h}$ $(h \leq l)$ denota al conjunto de las transformaciones no lineales desde el espacio de entrada al espacio característico, $k(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)\cdot\phi(\mathbf{x}_j)$ el producto interno de la función kernel; $D_{\text{máx}} = \text{máx}\left(D\left(\mathbf{x}_i, \mathbf{x}_{y_i}^*\right)\right)$ es la distancia máxima entre el centro y los datos de entrenamiento de la clase respectiva dentro del espacio kernel; y $k\left(\mathbf{x}_i, \mathbf{x}_{y_i}^*\right) = \phi(\mathbf{x}_i)\cdot\phi\left(\mathbf{x}_{j_i}^*\right)$es la función kernel formado por el dato muestra y el centro de la clase respectiva dentro del espacio kernel. Para muestras dentro de la clase $+1$, $\phi\left(\mathbf{x}_{y_i}^*\right) = \phi\left(\mathbf{x}_{+1}^*\right) = (1/n^+)\sum_{y_i=+1}\phi(\mathbf{x}_j)$, n^+ es el número de datos dentro de la clase $+1$, para datos dentro de la clase -1, $\phi\left(\mathbf{x}_{y_i}^*\right) = \phi\left(\mathbf{x}_{-1}^*\right) = (1/n^-)\sum_{y_i=+1}\phi(\mathbf{x}_j)$, n^- es el número de datos dentro de la clase -1.

La función de Lagrange

$$\begin{aligned} J(\mathbf{w}, b, \alpha) &= \frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_{i=1}^{l} \alpha_i (y_i(\mathbf{w} \cdot \phi(\mathbf{x}_i) + b) \\ &\quad -1 + \lambda D^2\left(\mathbf{x}_i, \mathbf{x}^*_{y_i}\right)) \end{aligned}$$

donde $\alpha = (\alpha_1, \ldots \alpha_l)^T$ es el multiplicador de Lagrange.

El problema dual para las muestras no separables

$$\begin{aligned} \text{máx } W(\alpha) &= \sum_{i,j=1}^{l} \alpha_i \left(1 - \lambda D^2\left(\mathbf{x}_i, \mathbf{x}^*_{y_i}\right)\right) \\ &\quad -\frac{1}{2}\sum_{i,j=1}^{l} \alpha_i \alpha_j y_i y_j k(\mathbf{x}_i, \mathbf{x}_j) \end{aligned}$$

$$\begin{aligned} sujeto\ a \sum_{i,j=1}^{l} y_i \alpha_i &= 0 \\ \alpha_i &\geq 0 \end{aligned}$$

[69]

2.3. Identificación con un enfoque multimodelo

El objetivo de esta sección es establecer algunos los conceptos básicos del enfoque multimodelo y hacer una revisión de los esquemas multimodelo existentes en tareas de identificación. También se presentan algunos esquemas de identificación utilizando redes neuronales.

La técnica multimodelo se utiliza para afrontar el problema del cambio en la región de operación que experimentan muchas plantas no lineales desconocidas. El fin de esta metodología es diseñar esquemas de identificación con capacidad de modelar la dinámica de una planta independientemente de la región de operación en que se encuentre.

El enfoque multimodelo utiliza múltiples modelos para describir el comportamiento de un sistema que opera en distintos ambientes o regiones de operación, o bien, el comportamiento

de sistemas con dinámicas muy complejas. Este enfoque inicia como un esfuerzo para mejorar el desempeño transitorio de sistemas lineales con grandes incertidumbres parámetricas y ha evolucionado gradualmente hacia el diseño de sistemas de control adaptables que en el curso del tiempo puedan actuar con rápidez y exactitud en el control de procesos no lineales con un alto grado de incertidumbres.

2.3.1. Conceptos Básicos

Un modelo puede considerarse como una representación de las partes escenciales de un sistema en una forma conveniente y puede tener diferentes formas dependiendo del problema. El método más utilizado para representar el comportamiento de un sistema es a través de un modelo matemático. En sistemas complejos se pueden utilizar modelos heurísticos y matemáticos bajo diferentes consideraciones para poder mejorar la representación del sistema.

La dinámica de un sistema puede cambiar en distintas regiones de operación. Considere que el sistema se describe por un conjunto de ecuaciones diferenciales de la forma

$$\begin{aligned} \dot{x}_t &= f\left(x_t, u_t, p\right) \\ y_t &= x_t = h\left(x_t, u_t, p\right) \end{aligned} \tag{2.1}$$

Estas ecuaciones contienen tanto a la planta como a las condiciones de operación externas. Si f y h se consideran fijos, diferentes condiciones de operación pueden expresarse con diferentes valores del vector de parámetros constantes p, o bién con diferentes funciones (f_i, h_i), $i = 1, 2, \ldots$. De esta manera fallas en el sistema, fallas en sensores y actuadores, perturbaciones externas y variaciones en los parámetros pueden llevar al sistema a diferentes regiones de operación.

Muchos sistemas complejos operan en distintos ambientes. Cuando las condiciones de operación cambian las características entrada-salida del sistema puede cambiar rápidamente o incluso discontinuamente. Si se utiliza sólo un modelo de identificación, éste tendrá que adaptarse al nuevo ambiente antes de que se tome alguna acción de control. En sistemas

lineales esta adaptación puede ser posible, sin embargo, la lentitud de la adaptación puede producir grandes errores transitorios. En sistemas no lineales de los cuales son descritos por diferentes funciones f y h en la ecuación 2.1, un modelo puede no ser suficiente para identificar los cambios en la dinámica del sistema. De aquí la necesidad de múltiples modelos tanto para identificar como para controlar adecuadamente los diferentes ambientes que se presenten.

Sin embargo, la necesidad de múltiples modelos va más lejos ya que en algunos ambientes puede ser necesario disponer de diferentes modelos cuya exactitud dependa de la región del espacio de estado en donde caen las trayectorias del sistema.

Los modelos matemáticos son necesarios para el control de procesos, predicción de comportamiento, detección de fallas, estimación de variables no medibles, así como para entender mejor el comportamiento del proceso. Sin embargo, muchos procesos reales involucran una gran cantidad de variables, son no lineales, variantes en el tiempo, el conocimiento de los fenómenos físicos y químicos involucrados a menudo es incompleto, algunas variables criticas pueden no ser medibles y algunos parámetros físicos pueden ser desconocidos.

Existen métodos para la obtención de modelos a partir de datos observados. Dentro de los métodos que se han desarrollado para identificación de sistemas no lineales estan el NARMAX, Hammerstein, Weiner y el Hammerstein-Weiner. Sin embargo, con estas estructuras a menudo es difícil representar el comportamiento del sistema en todo su intervalo de operación. Por lo anterior, son necesarios nuevos métodos de identificación de sistemas que permitan obtener modelos útiles en todo el intervalo de operación de la planta.

La identificación de sistemas con un enfoque multimodelo, también conocida como identificación basada en la región de operación, puede proporcionar una solución a éste problema. El principio fundamental de la identificación con un enfoque multimodelo consiste en dividir el espacio de entrada en regiones de operación y aproximar cada una de estas regiones con un modelo local que puede ser lineal o no lineal. Finalmente la salida del modelo global se obtiene interpolando o conmutando entre los modelos locales.

Un multimodelo describe una estructura que consiste de un conjunto de submodelos y de un mecanismo (2.4) que se encarga de combinar las salidas de estos submodelos. Por

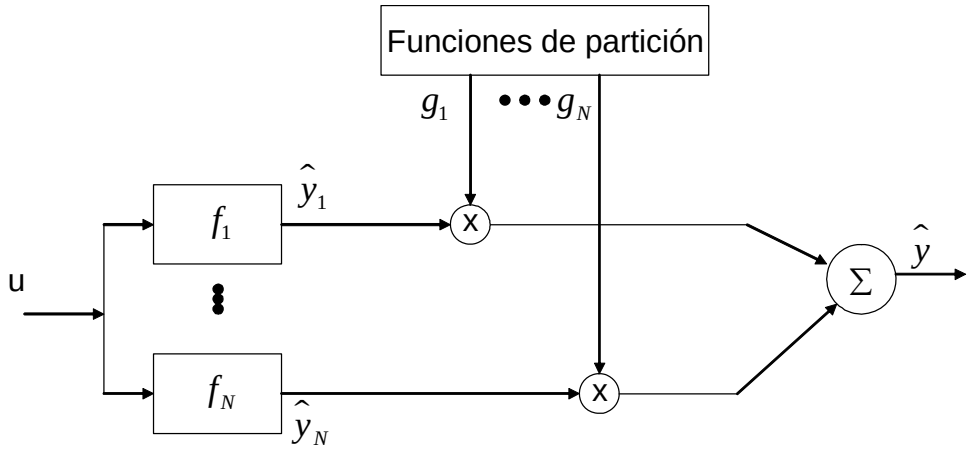

Figura 2.4: Mecanismo de selección de modelos.

ejemplo, si $\widehat{y}_i$ denota el mapeo del submodelo i, y la validez relativa del submodelo i se expresa por medio de una función de interpolación g_i, en donde $g_i \in [0, 1]$; entonces, si el submodelo i es muy exacto para una cierta región de operación, el valor de la función g_i será muy cercano a uno [24]. Considerando que el multimodelo consiste de N submodelos, la salida del multimodelo $\widehat{y}$ puede escribirse de la siguiente manera

$$\widehat{y} = F\left[\left(\widehat{y}_1, g_1\right), \left(\widehat{y}_2, g_2\right), \ldots, \left(\widehat{y}_N, g_N\right)\right] \tag{2.2}$$

Ventajas del enfoque multimodelo

En [17], se plantean algunas ventajas del enfoque multimodelo con respecto a la capacidad de aproximación que se puede lograr con una red neuronal:

• Puede ser muy difícil construir un modelo global a partir de un conjunto de datos de entrada y salida del proceso

• El enfoque multimodelo permite combinar diferentes técnicas de modelado, es decir, modelos lineales y no lineales pueden integrarse en la estructura del multimodelo. Conocimiento a *priori* (que exista sólo para ciertas regiones de operación) puede utilizarse para describir el sistema en dichas regiones.

• La estructura del modelo puede interpretarse tanto cualitativamente, en términos de las regiones de operación, como cuantitativamente en términos de modelos individuales.

• Las variables de entrada relevantes para la identificación del proceso pueden variar entre regiones de operación. En el enfoque multimodelo es posible utilizar diferentes entradas en cada región de operación. Esto es una ventaja en procesos multivariables en donde distintas variables son importantes en diferentes regiones de operación. La mayoría de los procesos químicos son multivariables por naturaleza.

• La implementación de estrategias multimodelo en un modelo de control predictivo es ventajoso ya que una menor cantidad de parámetros del modelo serán relevantes en un instante de tiempo dado.

Entrenamiento de un identifiador multimodelo

El entrenamiento para un identificador multimodelo que utiliza modelos fijos para representar las regiones de operación de la planta consiste de dos partes [17]:

• Primero se debe obtener (aprender) la partición desconocida del dominio de entrada. Esta es una tarea típica de las técnicas de aprendizaje no supervisado, en donde se tiene que decidir como dividir el dominio de entrada en distintas regiones de operación sin conocer la respuesta correcta, es decir, sin la ayuda de un tutor. La división del espacio de entrada también puede basarse en el conocimiento del proceso en lugar de utilizar un algoritmo de aprendizaje no supervisado.

• En segundo lugar se debe aprender el mapeo del dominio de entrada al espacio de salida. En este caso las salidas del proceso se conocen y por tanto se utiliza un método de aprendizaje supervisado. La salida del proceso y se compara con la predicción del modelo $\widehat{y}$ con la finalidad de definir una medida del error. Los parámetros de los modelos que representan a las regiones de operación se ajustan tomando como base esta medida del error.

Las dos etapas de éste método de entrenamiento (explorar una exacta descomposición del espacio de entrada y aprender el mapeo entrada salida), pueden ocurrir en paralelo o en forma secuencial.

2.3.2. Estructuras de Identificación con un enfoque multimodelo

Existen diferentes enfoques en la aplicación del enfoque multimodelo en identificación [17]. La primer estructura utiliza conocimiento a priori del proceso para determinar los submodelos, la segunda estructura utiliza un algoritmo de agrupamiento como una etapa de preprocesamiento, el tercer enfoque consiste en utilizar una red neuronal para establecer las fronteras de las diferentes regiones de operación, y finalmente la cuarta estructura utiliza múltiples modelos y un algoritmo de selección.

1) Partición basada en conocimiento apriori

La división del espacio de entrada en regiones de operación puede hacerse si se tienen un buen conocimiento del proceso. Si se define al conjunto de puntos de operación como Z, entonces un punto de operación $z \in Z$ es un vector de variables que caracterizan el comportamiento del sistema. Las regiones de operación estan definidas como $Zi \subset Z$, y pueden entenderse como una vecindad de puntos de operación que caracterizan el comportamiento de la planta. Por ejemplo, en un proceso de neutralización la variable pH caracteriza el comportamiento del sistema, de aquí que z representa a la variable pH. En general un proceso de neutralización puede dividirse en tres regiones de operación: neutra, ácida y alcalina; entonces la región de operación neutra puede definirse como la vecindad de puntos de operación tal que, $6{,}5 < pH < 7{,}5$. Sea x el vector de entrada al modelo que consta de m variables distintas $x = (x_1, x_2, ..., x_m)$, la salida del proceso y, y la predicción del modelo $\widehat{y}$. En muchos casos el vector x es un subconjunto del punto de operación z, ya que la salida del proceso y no esta incluida en el vector de entrada al modelo x.

En este método la descomposición del espacio de entrada se basa en el conocimiento *a priori* del proceso a identificar. Diferentes regiones de operación pueden asignarse, tomando como base que se conoce muy bien el proceso, por medio de una definición de umbrales de las variables que caracterizan al proceso x_{iTR}, para cada variable x_i ($i \in [1, ..., m]$). La salida del multimodelo depende del criterio de selección o de las funciones de partición que produzcan particiones suaves o bruscas.

Divisiones suaves del espacio de entrada permiten que el modelo de una región pueda verse influenciado por regiones de operación colindantes, así uno o más submodelos pueden

ajustarse durante la fase de entrenamiento. De acuerdo con esto, uno o más submodelos contribuyen a la predicción global del multimodelo en cada instante de tiempo. Por otro lado, si se utilizan particiones bruscas del espacio de operación, se tendrá que sólo un submodelo será responsable de la predicción del multimodelo en cada región de operación.

2) Partición basada en entrenamiento no supervisado

En este caso el problema de identificación de la planta se descompone en subproblemas utilizando un algoritmo de aprendizaje no supervisado, y la identificación de los subproblemas o regiones de operación se basa en un algoritmo de aprendizaje supervisado tal como el gradiente descendiente. Algúnos de los algoritmos no supervisados: el algoritmo adaptable de agrupamiento "k-centros" [11], o el mapa auto organizado (SOM) [42], pueden representar el mecanismo de partición.

La tarea de los algoritmos de agrupamiento es descomponer el dominio de entrada (espacio de operación) en k-regiones, y luego encontrar vectores centro que representen de una manera óptima a los vectores de entrada en cada región . El algoritmo de agrupamiento "k-centros" tradicional sólo puede asegurar optimalidad local, lo cual depende de la localización inicial de los vectores centro de cada grupo (vectores de referencia). El algoritmo adaptable de agrupamiento "k-centros", supera ésta desventaja apróximando una solución de agrupamiento óptima. Cada grupo forma una región de operación para un submodelo, los cuales pueden ser identificados por modelos de redes neuronales.

En el segundo tipo de algortimo no supervisado las regiones de operación pueden clasificarse por medio de un mapa auto organizado. A diferencia del algoritmo adaptable de agrupamiento "k-medios", el mapa auto organizado define una vecindad de adaptación. Mientras que el algoritmo de agrupamiento sólo ajusta al vector de referencia ganador $c_k(t)$, el mapa auto organizado de Kohonen además ajusta a todos los vectores de referencia que estan dentro de la vecindad de adaptación. La figura 2.5 ilustra un mapa auto organizado en la clasificación de regiones de operación.

El entrenamiento consiste en aplicar un vector de entrada a la red (punto de operación z), que se reciben todas las neuronas y se calcula la salida de cada neurona. Se permite que las neuronas interactuen unas con otras y se localiza a la neurona que responda más

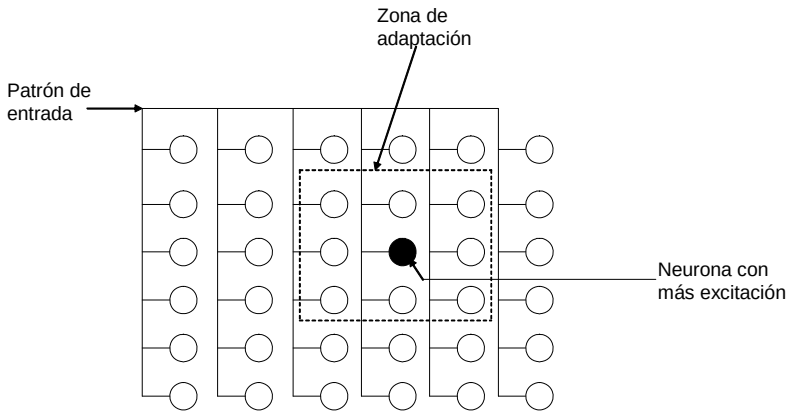

Figura 2.5: Mapa auto organizado Kohonen.

al patrón de entrada. Sólo a esta neurona y a sus vecinas, dentro de cierta distancia, se les permite ajustar sus pesos para que sean más sensibles al patrón de entrada en particular. De esta manera el entrenamiento tiene el efecto de organizar un "mapa" (espacio de puntos de operación Z), de tal forma que diferentes áreas del mapa correspodan a diferentes regiones de operación y a la vez a diferentes patrones de entrada.

3) Partición con redes de selección no lineales

Este enfoque fué introducido por Weigend [76]. La estructura de "expertos seleccionados" con una red neuronal no lineal y el algoritmo de entrenamiento tienen un enfoque probabilístico. El modelo esta formado por un cierto número de expertos y un mecanismo de selección. Cada "experto", en la forma de una red neuronal, corresponde a un submodelo. La división del espacio de entrada se realiza con una red neuronal. La red neuronal que trabaja como selector proporciona una salida a cada submodelo. La estructura del modelo de expertos con una red de selección no lineal se muestra en la figura 2.6.

Las entradas a los submodelos y las entradas a la red de selección consisten de un vector de variables medibles del dominio de entrada. Los "expertos" y la red selectora pueden compartir las mismas entradas o puede utilizarse un conjunto distinto de entradas.

La red de selección se implementa con una red neuronal directa, y sus salidas g_1, g_2 y g_3,

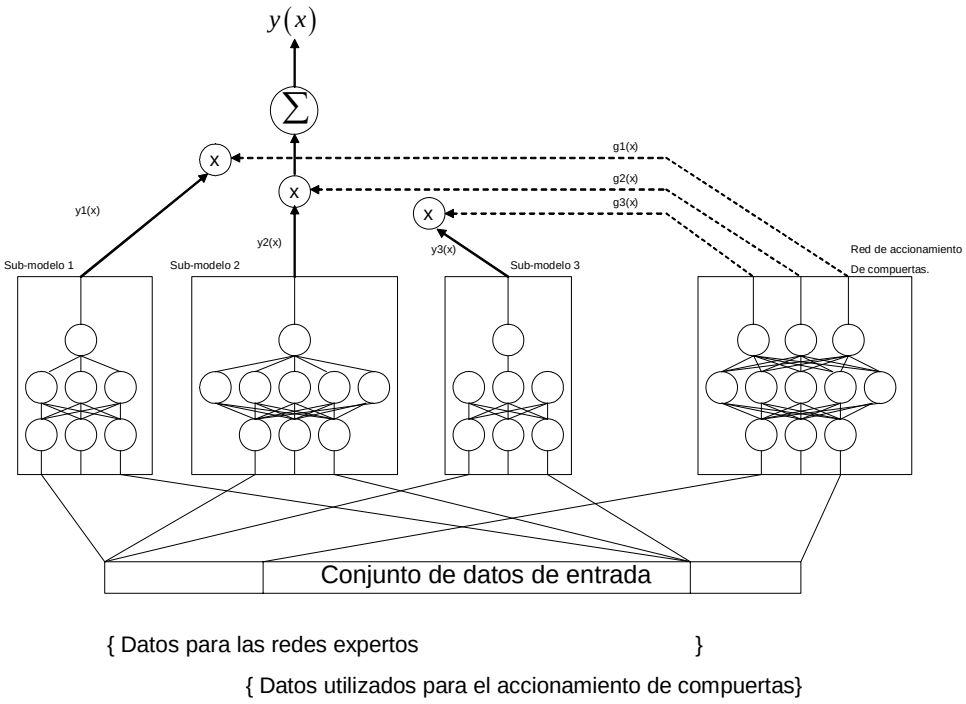

Figura 2.6: Estructura multimodelo con expertos y selector.

se combinan con las salidas de los expertos para obtener la salida del multimodelo. Debido a que las salidas de la red de selección se tratan como probabilidades estas tienen valores en el intervalo $[0, 1]$. Las neuronas de salida de la red de selección combinan la activación de las unidades ocultas ξ, con los pesos correspondientes para formar una activación intermedia s_j, de acuerdo con la siguiente expresión

$$s_j = \sum_h w_{jh}\xi_h + b_j \qquad j = (1, 2, \ldots, K) \tag{2.3}$$

Donde el subíndice h representa el h-ésimo nodo oculto. Entonces la activación intermedia es exponenciada y normalizada, lo que resulta en la salida final de la red selectora:

$$g_j = \frac{e^{s_j}}{\sum_{l=1}^{K} e^{s_j}} \tag{2.4}$$

Las salidas de la red de selección estan restringidas por $\sum_{l=1}^{K} g_j = 1$. La competencia entre los submodelos queda establecida por esta restricción. La partición utiliza fronteras suaves ya que g_j puede tomar valores en el rango $[0, 1]$. Cada vector de datos de entrada puede ser asignado suavemente a varios submodelos. La salida del j-ésimo experto, es ponderada por la j-ésima salida de la red de selección. Los expertos se pueden implementar con redes neuronales directas o recurrentes. Finalmente, no puede aplicarse un método de aprendizaje sencillo ya que no se conocen las fronteras de la regiones de operación. Este problema puede tratarse con la herramienta estadística de máxima probabilidad [17].

4) Identificación con un multimodelo y selector

Esta metodología utiliza múltiples modelos y un selector para modelar una planta que opera en distintas regiones de operación [58].

Debido a que la planta p puede operar en cualquier punto del espacio de operación es necesario determinar cual de los identificadores I_i debe seleccionarse y en que instante debe ser seleccionado. En cualquier problema de identificación y control pueden utilizarse modelos múltiples de identificación, pero sólo una señal de control puede aplicarse a la planta. Por esta razón, el criterio de selección debe establecerse en función de los errores de identificación.

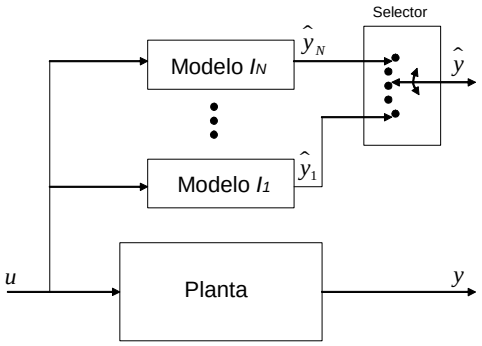

Figura 2.7: Esquema de identificación multimodelo con selector.

La estructura de un sistema capaz, al menos en teoría, de identificar la región de operación actualmente presente se muestra en la Figura 2.7.

El esquema introducido en [58] muestra N modelos de identificación denotados por $\{I_i\}_{i=1}^{N}$, con salidas correspondientes $\{\widehat{y}_i\}_{i=1}^{N}$. Cada modelo de identificación representa a la planta en una región de operación diferente. En cada instante se determina una medida de los errores de identificación $e_i \stackrel{\triangle}{=} \widehat{y}_i - y$ a trevés de un índice de desempeño $J_i(t)$, $i = 1, 2, ..., N$, y se selecciona el modelo de identificación que tenga el menor error $min_i\{J_i(t)\}$ para representar a la planta.

Cuando la planta se encuentre en la región de operación S_i, entonces el modelo I_i tendrá el menor índice de desempeño y su salida será seleccionada. Uno de los problemas de este enfoque se presenta cuando un cambio en la región de operación produce un comportamiento de la planta que no este considerado dentro de la estructura del multimodelo.

Capítulo 3

Modelado via agrupamiento en línea y SVM normal

El modelado difuso para sistemas no lineales esta basado en un conjunto de reglas SI-ENTONCES (*IF–THEN*) las cuales se usan como proposiciones lingüísticas de habla humana.[84]. La estracción de métodos de reglas difusas se pueden dividir en dos clases: 1) obtención de reglas difusas de expertos, 2) obtención de reglas difusas automáticamente de datos observados[43]. Estos procesos son también llamados identificación de la estructura para sistemas difusos. El método experto es frecuentemente abordado fuera de línea, un enfoque de prueba y error[62]. Un método común para la inicialización de la estructura es una partición uniforme de cada varaiable de la entrada, resulta una cuadricula difusa[34]. Muchos métodos de estructura de identificación estan basados sobre agrupamientos de datos, tales como agrupamiento C-difuso [21], agrupamiento montaña[54], agrupamiento substractivo[12]. Sistemas de inferencia Takagi-Sugeno son aplicados para aprendizaje en línea en [38]. El método hurístico es propuesto en [66], métodos estadístiscos y adaptables son usados in [44]. Estos enfoques requieren que todas los datos de entrada-salida estan listas antes de comenzar a identificar la planta. Estos enfoques de estructuras de identificación son fuera de línea.

La extracción de reglas difusas (estructuras de identificación) es muy importante para el modelado difuso. Enfoques rigurosos pueden ser encontrados, tales como método de redes

neuronales [7], algoritmos genéticos [62], SVD-QR [49]. *Support Vector Machine* (SVM's) es una técnica para la solución de problemas de casificación de patrones [2]. Vapnik ha definido a esto como *minimizazción de estructura de riesgo*, un principio de minimización inductivo y una cota superior del error de un modelo bajo un conjunto de datos entrenados. Los detalles de esta técnica se pueden encontrar en [15]. En SVM's, el problema de regresión se resuelve como una programación cuadráctica convexa (QP) [56]. La idea básica del regresor SVM es mapear las entradas dentro de un espacio característico de dimensión grande, luego resolviendo QP con una función de costo apropiada. Con el teorema de Mercel, un problema QP de dimensión finita en el espacio dual puede ser obtenido. El propósito de los SVM's es obtener un margen máximo separado por un hiperplano en problemas de clasificación y regresión. Los SVM's no son afectados por el tamaño de dimensionalidad [21], ofrece soluciones con una dependencia explícita sobre patrones de los datos. Existe una propiedad importante dentro de SVM, llamada separabilidad (sparseness). La solución vector es separable, la suma podrá tener un término no cero lo cual es llamado support vector. Los support vectors actúan como una llave de datos en un conjunto para clasificación y modelado.

El procedimiento del modelado difuso se da como sigue: Primero se usa el método de agrupamiento en línea el cual divide los datos entrada/salida dentro de dos grupos en el mismo intervalo de tiempo. Luego se aplica el *support vector machines* el cual produce support vectors para cada grupo. Con los support vectors, las reglas difusas son construidas y los correspondientes sistemas difusos son hechos. Finalmente, las funciones de membresía de la predicción y la consecuencia son entrenados por los datos en cada grupo. El esquema del modelado difuso en línea se muestra en la Figura 3.1.

3.1. Agrupamiento en línea

Se quiere el modelo del siguiente sistema no lineal suave de espacio de estados discretos en el tiempo

$$x(k+1) = f\left[x\left(k\right), u\left(k\right)\right], \quad y(k) = h\left[x\left(k\right)\right] \tag{3.1}$$

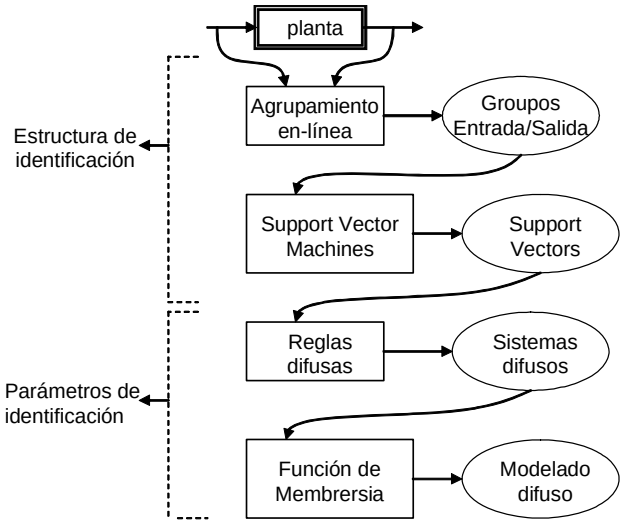

Figura 3.1: Esquema del modelado difuso en línea.

donde $u(k) \in \Re^1$ es el vector de entrada, $x(k) \in \Re^n$ es el vector de estado, y $y(\ k) \in \Re^1$ es el vector de salida, f y h son funciones suaves no lineales. (3.1) se puede escribir como

$$\begin{aligned} y(k) &= h[x(k)] = F_1[x(k)], \\ y(k+1) &= h[f[x(k), u(k)]] = F_2[x(k), u(k)] \\ y(k+n-1) &= F_n[x(k), u(k), \cdots u(k+n-2)] \end{aligned} \tag{3.2}$$

Denotando $Y(k) = [y(k), y(k+1), \cdots y(k+n-1)]^T$, $U(k) = [u(k), u(k+1), \cdots u(k+n-2)]^T$, $y(k) = F[x(k), U(k)]$, $F = [F_1 \cdots F_n]^T$. Desde (3.1) es un sistema no lineal suave, (3.2) se puede expresar como $x(k+1) = g[Y(k+1), U(k+1)]$. Esto conduce a un modelo multivariable NARMA

$$y(k) = h[x(k)] = \Psi[X(k)] \tag{3.3}$$

donde

$$X(k) = [y(k-1), y(k-2), \cdots u(k-d), u(k-d-1), \cdots]^T \tag{3.4}$$

$\Psi(\cdot)$ es una función no lineal desconocida representando las dinámicas de la planta, $u(k)$ y $y(k)$ son entradas y salidas escalares medibles, d es un retardo en el tiempo.

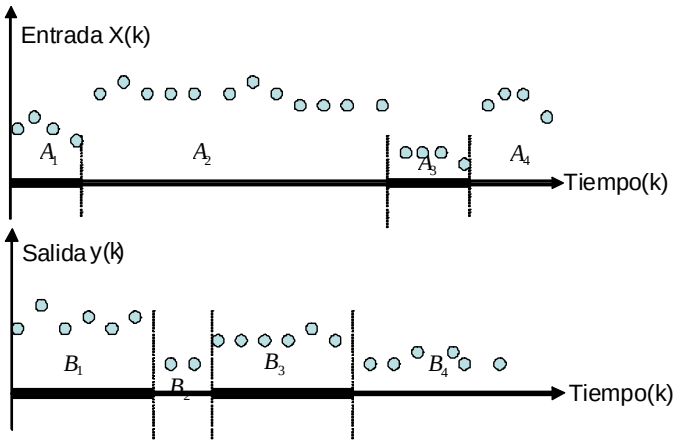

Figura 3.2: Particionamiento del espacio de entrada/salida.

El objetivo de la identificación de estructuras es la de particionar los datos de entrada y salida $[y(k), x(k)]$ del sistema no lineal y extraer reglas difusas. Se usa el siguiente ejemplo para explicar la importancia del agrupamiento en línea en el mismo tiempo indexado. Se considera la función no lineal como

$$y(k) = f[x(k)] \tag{3.5}$$

Por la norma del método de agrupamiento en línea propuesto en [1] la entrada y la salida se pueden particionar en 4 grupos, como se puede ver en la Figura 3.2. Estos grupos se pueden formar dentro de 4 reglas como "SI $x(k)$ es A_j ENTONCES $y(k)$ is B_j", $j = 1 \cdots 4$. Obviamente, para la 3rd regla: "SI $x(k)$ is A_3 ENTONCES $y(k)$ is B_3", esta no satisface la relación (3.5), porque la precondición $x(k)$ y la consequencia $y(k)$ no ocurren al mismo tiempo.

La idea básica del agrupamiento en línea es que los espacios particionados entrada y salida son acarreados fuera en el mismo tiempo indexado. Si la distancia desde el punto al centro es menor que la longitud requerida, el punto esta dentro de ese grupo. Cuando un nuevo dato llega, el centro y el grupo deberían ser cambiados de acuerdo al nuevo dato. Se da el siguiente algoritmo.

La distancia Euclidiana al tiempo k se define como

$$\begin{aligned} d_{k,x} &= \left(\sum_{i=1}^{n} \left[\frac{x_i(k) - \overline{x}_i^j}{x_{i,\text{máx}} - x_{i,\text{mín}}} \right]^2 \right)^{1/2} \\ d_{k,y} &= \left| \frac{y(k) - \overline{y}^j}{y_{\text{máx}} - y_{\text{mín}}} \right| \\ d_k &= \alpha d_{k,x} + \beta d_{k,y} \end{aligned} \tag{3.6}$$

donde $x_{i,\text{máx}} = \text{máx}_k \left\{ x_i \left(k \right) \right\}$, $x_{i,\text{mín}} = \text{mín} \left\{ x_i \left(k \right) \right\}$, $y_{\text{máx}} = \text{máx} \left\{ y \left(k \right) \right\}$, $y_{\text{mín}} = \text{mín} \left\{ y \left(k \right) \right\}$, $\overline{x}_i \left(k \right)$ y $\overline{y} \left(k \right)$ son los centros de x_i y y al tiempo k, α y β son factores positivos, normalmente se pueden escoger $\alpha = \beta = \frac{1}{2}$. Para el Grupo j, los centros son actualizados como sigue

$$\begin{aligned} \overline{x}_i^j &= \frac{1}{l_2^j - l_1^j + 1} \sum_{l=l_1^j}^{l_2^j} x_i \left(l \right) \\ \overline{y}^j &= \frac{1}{l_2^j - l_1^j + 1} \sum_{l=l_1^j}^{l_2^j} y \left(l \right) \end{aligned} \tag{3.7}$$

donde l_1^j es el primer número del Grupo j, l_2^j es el último número del Grupo j. La longitud del Grupo j es $m^j = l_2^j - l_1^j + 1$. El intervalo de tiempo del Grupo j es $\left[l_1^j, l_2^j \right]$. El proceso de la estructura de identificación se puede formar con los siguientes pasos

1. Para el primer dato G_1, $k = 1$. $y \left(1 \right)$, $x_i \left(1 \right)$ son los centros del primer grupo, $\overline{x}_i^1 = x \left(1 \right)$, $\overline{y}^j = y \left(1 \right)$, $l_1^j = l_2^j = 1$.

2. Si un nuevo dato $\left[y \left(k \right), x_i \left(k \right) \right]$ llega, $l_2^j = l_2^j + 1$, se usa (3.7) y (3.6) para calcular d_k. Si ningún dato nuevo llega, ir a 5.

3. Si $d_k \leq L$ entonces $\left[y \left(k \right), x_i \left(k \right) \right]$ esta todavía en el group G_j, ir a 2

4. Si $d_k > L$ entonces $\left[y \left(k \right), x_i \left(k \right) \right]$ esta en un nuevo grupo $j = j + 1$, el centro de G_j es $\overline{x}_i^j = x \left(k \right)$, $\overline{y}^j = y \left(k \right)$, $l_1^j = l_2^j = k$, ir a 2

5. Checar la distancias entre todos los centros $\overline{x}^j$, $\overline{y}^j$, Si $\sum_{i=1}^{n} \left[\overline{x}_i^p - \overline{x}_i^q \right]^2 + \left| \overline{y}^p - \overline{y}^q \right| \leq L$, los dos grupos G_p y G_q estan combinados dentro de un grupo.

La nueva idea del agrupamiento en línea es que los espacios particionados entrada-salida son llevados fuera en el mismo intervalo de tiempo. Existen dos razones. La primera, en el modelado de sistemas no lineales es encontrar un mapeo apropiado entre la entrada y la salida. Si se usan redes neuro difusas como modelos, las reglas tienen la forma "SI *entrada* es A ENTONCES *salida* es B". Solo cuando la entrada y la salida ocurren en el mismo intervalo de tiempo, estas corresponden al mapeo no lineal. Segundo, se propondrá un enfoque de modelado en línea basado en agrupamiento en línea. Cuando un nuevo grupo (o una nueva regla) es creado, no se quieren usar todos los datos del entrenamiento como en [1]. Si el dato tiene propiedades de tiempo, se usa el dato en el correspondiente intervalo de tiempo para entrenar la regla. Así el agrupamiento con intervalos de tiempo simplificarán los parámetros de identificación y harán el modelado en línea más fácil.

Existen tres parámetros de diseño α, β y L. α y β pueden ser vistos como los pesos del espacio sobre la entrada y la salida. Si la entrada domina la propiedad dinámica, se podría incrementar α y decrementar β. Usualmente se seleccionan $\alpha = \beta = 0{,}5$ tal que la entrada y la salida son igual de importantes. Si $\alpha = 1$, $\beta = 0$, esto empieza el agrupamiento en líneal normal. L es el umbral de la creación de nuevas reglas, esto es el valor posible más pequeño de similaridad requerido al juntar dos objetos en un grupo. Si el valor del umbral L es muy pequeño, se mantendrá algunos grupos presentes al final, y algunos singletons. Convergentemente, si el umbral L es muy grande, algunos objetos que no son muy similares estarán en las salidas en el mismo grupo. Desde $d_k = \alpha d_{k,x} + \beta d_{k,y}$, $d_{\text{máx}} = \alpha \left\| x_{\text{máx}} - x_{\text{mín}} \right\| + \left\| y_{\text{máx}} - y_{\text{mín}} \right\| \beta$. Si se quiere que el algoritmo pueda particionar grupos rigurosos, se podría dar $L < d_{\text{máx}}$, de otra manera existe solo un grupo.

Se usa el siguiente ejemplo para explicar el agrupamiento en línea. Se considera la función no lineal la cual tiene la forma [72]

$$f(x_1, x_2) = 0{,}52 + 0{,}1x_1 + 0{,}28x_2 - 0{,}6x_1x_2$$

Los datos de entrenamiento son selecionados como $x_1(k) = -1 + \frac{2k}{T}$, $x_2(k) = 1 - \frac{2k}{T}$, $k = 1, 2, \cdots T$. Se escoge $\alpha = 0{,}4$, $\beta = 0{,}6$. Se obeserva que los cambios máximos en la entrada y la salida son aproximadamente 1 y 2, así $\alpha \left\| x_{\text{máx}} - x_{\text{mín}} \right\| + \left\| y_{\text{máx}} - y_{\text{mín}} \right\| \beta = 16$, L se

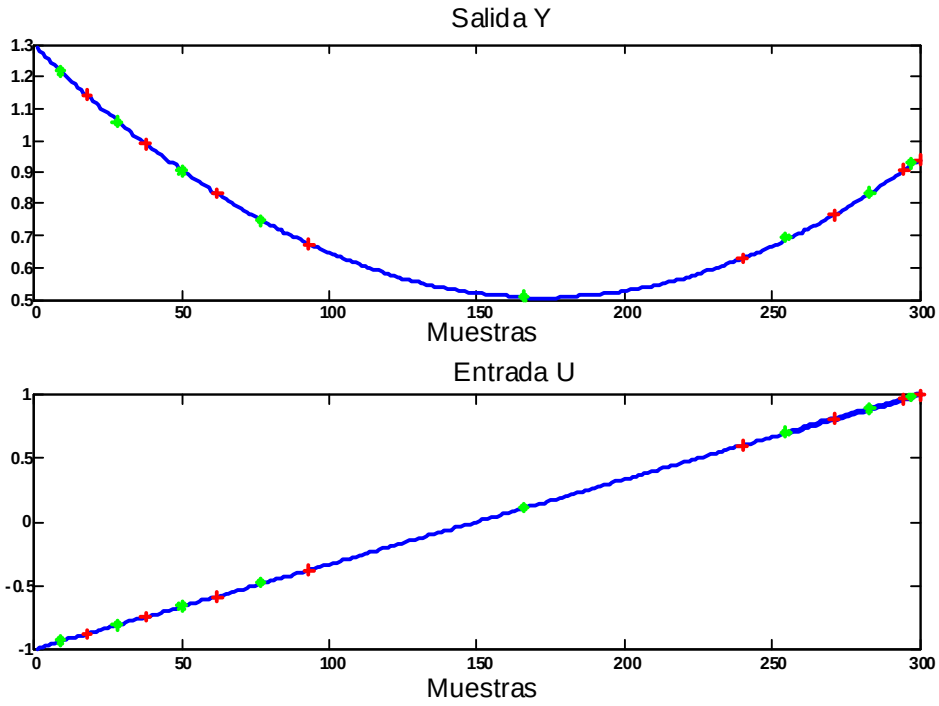

Figura 3.3: Agrupamiento en línea.

podría escoger como $L < 1{,}2$, en esta aplicación se selecciona $L = 0{,}5$. El particionamiento entrada-salida que se muestra en la Figura.3.3. Se representa el centro como "o" de cada grupo, y "+" es la frontera entre los grupos. Se puede ver que hay 7 grupos. Por ejemplo, los intervalos de tiempo del 4-ésimo grupo son $l_1^2 = 94$, $l_1^2 = 238$.

3.2. Extracción de reglas difusas por support vector machines

SVM puede separar los datos con un hiperplano en un margen máximo entre dos clases [2]. Una posible formulación de esta tarea es el diseño de una función de estimación f :

$\mathbb{R}^n \rightarrow \{-1, +1\}$, se usa una pareja de datos generado independientemente de acuerdo a una distribución desconocida $P(X, y)$

$$(X_1, y_1), \ldots, (X_{m,}, y_m) \in \mathbb{R}^n \times Y, \qquad Y = \{-1, +1\} \tag{3.8}$$

Si el entrenamiento es separable por el hiperplano, la función se escoge como

$$f(x) = (w \cdot x) + b \tag{3.9}$$

El margen en turno se puede medir por la longitud desde el vector w, tal que los puntos cerca del hiperplano satisface $|(w \cdot x) + b| = 1$.

Se usa SVM para la función de estimación. Para encontar los support vectors en el Grupo j, se usa el dato de entrada-salida $[y(k), x(k)]$, $k \in \left[l_1^j, l_2^j\right]$ para aproximar la función no lineal. Considerar la regresión en el conjunto de funciones no lineales, aplicando la trampa del kernel

$$f(x) = w^T \varphi(x) + b \tag{3.10}$$

donde la trampa del kernel $K(x, z) = \varphi^T(x)\varphi(z)$. Existen muchas posibilidades de escoger el kernel $K(x_k, x)$, solo se requiere que $K(x_k, x)$ satisfaga la condición de Mercer [15]. Por ejemplo el kernel lineal $K(x, x_k) = x_k^T x$, kernel MPL (perceptron multi capas)$K(x, x_k) = \tanh\left(k_1 x_k^T x + k_2\right)$, kernel RBF (función base radial)$K(x, x_k) = \exp\left(-\|x - x_k\|^2 / \sigma^2\right)$. La función de costo (riesgo empírico) se define como

$$R_{emp}(\theta) = \frac{1}{N} \sum_{k=1}^{N} \left| y_k - \left(w^T x_k + b\right) \right|_\epsilon$$

donde ϵ es la función de pérdida insensitiva de Vapnik, se define como

$$|y_k - f(x)|_\epsilon = \begin{cases} 0 & |y_k - f(x)| \leq \epsilon \\ |y_k - f(x)| - \epsilon & \text{de otra manera} \end{cases}$$

ϵ puede ser vista como la exactitud de la aproxiación.

El problema de optimización (original) es

$$\begin{array}{c} \text{mín } J_p = \frac{1}{2} w^T w \\ sujeto\ a\ \left| y_k - \left(w^T \varphi(x_k) + b\right) \right| \leq \epsilon \end{array} \tag{3.11}$$

Si ϵ es muy pequeña, ciertos puntos estarán fuera del tubo ϵ. Por lo tanto en suma la variables flojas ξ_k, ξ_k^* son introducidas (esto es un tanto similar al caso clasificador con superposición). El problema primordial esta modificado dentro de

$$\begin{aligned} \text{mín}\, J_p &= \tfrac{1}{2} w^T w + c \sum_{k=1}^{N} (\xi_k + \xi_k^*) \\ sujeto\ a\ \ &-(\epsilon + \xi_k^*) \leq y_k - \left(w^T x_k + b\right) \leq \epsilon + \xi_k \end{aligned}$$

La forma de Lagrange

$$\begin{aligned} L = J_p &- \sum_{k=1}^{l_2^j} \alpha_k \left[\left(w^T x_k + b\right) - y_k + (\epsilon + \xi_k)\right] \\ &- \sum_{k=1}^{N} \alpha_k^* \left[y_k - \left(w^T x_k + b\right) + (\epsilon + \xi_k^*)\right] - \sum_{k=1}^{N} (\eta_k \xi_k + \eta_k^* \xi_k^*) \end{aligned}$$

donde $\alpha_k, \alpha_k^*, \eta_k, \eta_k^* \geq 0$. La solución es el punto silla de la función Lagrange

$$\max_{\alpha_k, \alpha_k^*, \eta_k, \eta_k^*} \min_{w, b, \xi, \xi^*} L\left(w, b, \xi, \alpha, \upsilon\right)$$

El problema principal 3.11 puede ser cambiado dentro del problema dual

$$\begin{aligned} &\text{máx}_{\alpha, \alpha^*}\, J_D\left(\alpha, \alpha^*\right) = -\tfrac{1}{2} \sum_{k,l=1}^{N} \left(\alpha_k - \alpha_k^*\right)\left(\alpha_l - \alpha_l^*\right) x_k^T x_l - \epsilon \sum_{k=1}^{N} \left(\alpha_k - \alpha_k^*\right) + \sum_{k=1}^{N} y_k \left(\alpha_k - \alpha_k^*\right) \\ &sujeto\ a\ \sum_{k=1}^{N} \left(\alpha_k - \alpha_k^*\right) = 0 \end{aligned} \tag{3.12}$$

Por el paquete de programación estandar se obtiene la solución α_k^*, b se puede resolver por la condición Kuhn-Tucker. La función resultante es

$$f(x) = \sum_{k=1}^{l_2^j} \left(\alpha_k - \alpha_k^*\right) K\left(x_k, x\right) + b$$

Desde algunas $\alpha_k^* = 0$, el vector solución es separado, la suma debería ser tomado solo bajo los α_k^* no ceros (support vector), así el resultado final es

$$f(x) = \sum_{k=1}^{sv} \left(\alpha_k - \alpha_k^*\right) K\left(x_k, x\right) + b \tag{3.13}$$

donde sv es el número de support vectors, $sv_j \ll \left(l_2^j - l_1^j + 1\right)$. Se define el dato $[y(k), x(k)]$ los cuales estan en la posición de los support vector como $[x_i^*, y_i^*]$, $i = 1 \ldots sv_j$.

Desde 3.13 se sabe que los support vector son suficientes para representar la no linealidad en cada grupo. Se usan estos support vectors para construir las reglas difusas. Para el grupo j, se extraer el producto de las reglas difusas de la siguiente forma (modelo difuso Mamdani [52])

$$\text{R}^r\text{: SI } x_1^*(k) \text{ es } A_1^j \text{ y } x_2^*(k) \text{ es } A_2^j \text{ y } \cdots \ x_n^*(k) \text{ es } A_n^j \text{ ENTONCES } y^*(k) \text{ is } B^j$$

aquí $r = 1 \ldots sv_j$, $A_1^j, \cdots A_n^j$ y B^j son los conjuntos difusos estandares, $[x_i^*, y_i^*]$ es el punto de los suppor vector. Se usa las reglas SI-ENTONCES difusas sv_j para efectuar un mapeo desde un vector de entrada $X = [x_1 \cdots x_n] \in \Re^n$ a la salida $y(k)$. Las funciones de membresía son funciones Gausianas

$$\mu(x_i) = \exp\left(-\frac{(x_j - c_{ji})^2}{\sigma_{ji}^2}\right) \tag{3.14}$$

Sean $x^*(k)$ y $y^*(k)$ los centros de las funciones Gausianas, i.e., $c_{ji} = x_i^*$.

De [72] se conoce, usando el producto inferencia, el centro promedio y el difusificador singleton, la salida del sistema difuso en el grupo j puede ser expresado como

$$\widehat{y} = \left(\sum_{j=1}^{sv_j} w_j \left[\prod_{i=1}^{n} \mu_{A_i^j}\right]\right) / \left(\sum_{j=1}^{sv_j} \left[\prod_{i=1}^{n} \mu_{A_i^j}\right]\right) = f_j[X(k)] \tag{3.15}$$

donde $\mu_{A_i^j}$ es la función de membresía de los conjuntos difusos A_i^j, w_j es el punto en el cual $\mu_{B_j} = 1$.Si se define

$$\phi_i = \prod_{i=1}^{n} \mu_{A_i^j} / \sum_{j=1}^{sv_j} \prod_{i=1}^{n} \mu_{A_i^j}$$

3.15 puede ser expresada en la forma matricial

$$\widehat{y}(k) = W(k)\, \Phi[X(k)] \tag{3.16}$$

donde el parámetro $W(k) = \left[w_1 \cdots w_{sv_j}\right] \in R^{1 \times sv_j}$, el vector de datos $\Phi[X(k)] = \left[\phi_1 \cdots \phi_{sv_j}\right]^T \in R^{sv_j \times 1}$

Por el modelo difuso Takagi-Sugeno_Kang [66],

$$\begin{array}{c} \mathrm{R}^j\text{: SI } x_1 \text{ es } A_{1i} \text{ y } x_2 \text{ es } A_{2i} \text{ y } \cdots \ x_n \text{ es } A_{ni} \\ \text{ENTONCES } \widehat{y} = p_0^j + p_1^j x_1 + \cdots p_n^j x_n \end{array} \tag{3.17}$$

donde $j = 1 \cdots sv$. La salida del sistema lógico difuso se puede expresar como

$$\widehat{y} = \sum_{i=1}^{sv} \left(p_0^i + p_1^i x_1 + \cdots p_n^i x_n \right) \phi_i \tag{3.18}$$

3.18 también puede ser expresado en la forma del tipo Mamdani 3.16,

$$\begin{array}{c} \widehat{y}\,(k) = W\,(k)\,\Phi\,[X\,(k)] \\ W\,(k) = \left[p_{10}^1 \cdots p_{10}^{sv_j} \cdots p_{1n}^1 + \cdots p_{1n}^{sv_j} \right] \\ \Phi\,[X\,(k)] = \left[\phi_1 \cdots \phi_{sv_j} \ \ x_1\phi_1 \cdots x_1\phi_{sv_j} \ \ \cdots \ \ x_n\phi_1 \cdots x_n\phi_{sv_j} \right]^T \end{array} \tag{3.19}$$

Se usa el siguiente ejemplo propuesto anteriormente para explicar la extracción de las reglas difusas. En la figura 3.4 se puede observar los 3 SVM´s que se generan para el Grupo 3. . El resultado después de usar SVM son mostrados en la siguiente tabla

Grupo	# de sv	Posición	# de datos
1	2	$1, 13$	21
2	2	$1, 14$	25
3	3	$1, 16, 32$	32
4	5	$1, 23, 74, 126, 148$	148
5	3	$1, 17, 32$	32
6	2	$12, 25$	25
7	2	$1, 6$	6

Tabla 1: SVM para función de estimación.

Por ejemplo, en el Grupo 3 existen 3 support vectors y 32 datos, las posiciones de los support vectors son $x\,(1)$, $x\,(16)$ y $x\,(32)$. Se usará la siguiente forma tipo Mamdani para la extracción de reglas difusa

$$\begin{array}{c} \text{Si } x_1 \text{ es } A_1^1 \text{ y } x_2 \text{ es } A_2^1 \text{ entonces } y \text{ es } B^1 \\ \text{Si } x_1 \text{ es } A_1^2 \text{ y } x_2 \text{ es } A_2^2 \text{ entonces } y \text{ es } B^2 \\ \text{Si } x_1 \text{ es } A_1^3 \text{ y } x_2 \text{ es } A_2^3 \text{ entonces } y \text{ es } B^3 \end{array}$$

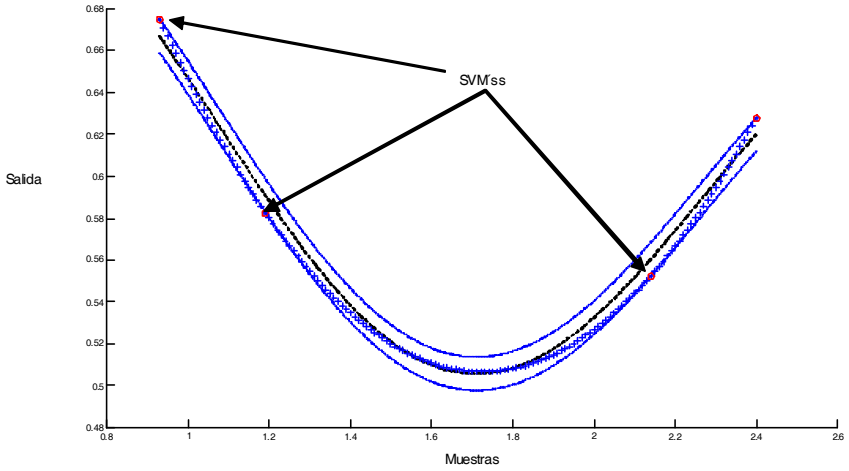

Figura 3.4: SVM´s en el Grupo 3

Las reglas difusas para el Grupo 3 son

Si x_1 es 0,77 y x_2 es 0,2 entonces y es 0,75
Si x_1 es 0,93 y x_2 es 0,32 entonces y es 0,675
Si x_1 es 1,15 y x_2 es 1,32 entonces y es 0,59

El sistema difuso para el Grupo 3 es

$$\widehat{y} = \left(\sum_{j=1}^{3} w_j \left[\mu_{A_1^j} \mu_{A_2^j} \right] \right) / \left(\sum_{j=1}^{3} \left[\mu_{A_1^j} \mu_{A_2^j} \right] \right) = f_3 \left[X \left(k \right) \right]$$

donde $\mu_{A_1^1} = \exp\left(-\frac{(x_j - 0{,}77)^2}{\sigma_{ji}^2} \right)$, $\mu_{A_2^1} = \exp\left(-\frac{(x_j - 0{,}2)^2}{\sigma_{ji}^2} \right)$, $w_1 = 0{,}75$, σ_{ji} se selecciono aleatoriamente en $(0, 1)$. El la figura 3.5 se puede observar las funciones de membresía para x_1 y x_2.

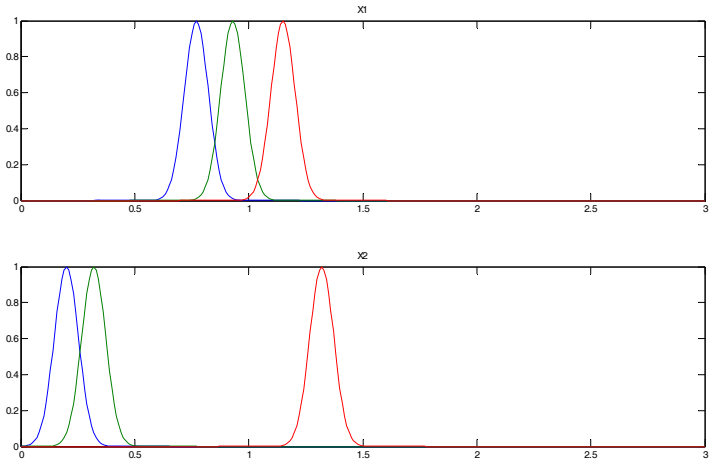

Figura 3.5: Funciones de membresía para x_1 y x_2.

3.3. Entrenamiento de funciones de membresía

Para el Grupo j, se usa el dato entrada/salida $[y(k), x(k)]$, $k \in \left[l_1^j, l_2^j\right]$ para entrenar las funciones de membresía A_i^j $(i = 1 \cdots n)$ y B^j, i.e., el parámetro de identificación de las funciones de membresía son ejecutados en la entrada/salida correspondiente en el intervalo de tiempo encontrado en la estructura de identificación. Se discuten dos casos: consecuencias de las funciones de membresía entranadas y premisa de las funciones de membresía entrenadas.

Primero se asume que la premisas de las funciones de membresía $A_{ij} \cdots A_{ni}$ son conocidas a priori, i.e., $\phi_i = \prod_{i=1}^{n} \mu_{A_i^j} / \sum_{j=1}^{l} \prod_{i=1}^{n} \mu_{A_i^j}$ es conocida (ver [26],[72]). Los modelos Mamdani (3.16) y el TSK (3.19) tienen la misma forma porque $\Phi[X(k)]$ es conocido. Se selecciona $w_j(1) = y_i^*$ como condiciones iniciales. Se usa el par de dato $[y(k), x(k)]$ para encontar algunas funciones de membresía adecuadas en la zona $k \in \left[l_1^j, l_2^j\right]$. Esto se puede transformar dentro de un problema de modelado para determinar los parámetros $\overline{y}^j$, $\overline{x}^j$ y σ_i^j tal que $\widehat{y} \to y$.

Se define el vector de error de identificación como

$$e(k) = \widehat{Y}(k) - Y(k) \tag{3.20}$$

Se usará el modelado del error $e(k)$ para entrenar el sistema difuso (3.16) tal que $\widehat{y}(k)$ puede aproximarse a $y(k)$. De acuerdo a las teorias de la función de aproximación de lógica difusa [73], el proceso de identificación no lineal (3.3) se puede expresar como

$$y(k) = W^{*}\Phi[X(k)] - \mu(k) \tag{3.21}$$

donde W^{*} son los pesos desconocidos las cuales pueden minimizar la dinámica no modelada $\mu(k)$. El error de identificación puede ser representado por (3.20) y (3.21)

$$e(k) = \widetilde{W}(k)\,\Phi[X(k)] + \mu(k) \tag{3.22}$$

donde $\widetilde{W}(k) = W(k) - W^{*}$. Solo se esta interesado en la identificación en lazo abierto, se asume que la planta (3.3) es BIBO (entrada acotada y salida acotada) estable, i.e., $y(k)$ y $u(k)$ en (3.3) son acotadas. Por la cota de la función de membresía Φ, $\mu(k)$ en (3.21) es acotada. El siguiente teorema ofrece un algoritmo de gradiente descendente estable para el modelado neuro difuso.

Teorema 3.1 *Si se usa el sistema difuso (3.15) para identificar la planta no lineal (3.3) en el grupo j, el siguiente algoritmo de gradiente descendente con un rango de aprendizaje variante en el tiempo puede identificar el error $e(k)$ acotado*

$$W(k+1) = W(k) - \eta_k e(k)\,\Phi^{T}[X(k)] \tag{3.23}$$

donde el escalar $\eta_k = \left(\eta/1 + \|\Phi[X(k)]\|^2\right)$, $0 < \eta \leq 1$. El error de identificación normalizado

$$e_N(k) = \frac{e(k)}{1 + \text{máx}_k\left(\|\Phi[X(k)]\|^2\right)}$$

satisface la siguiente realización promedio

$$\lim_{T\to\infty}\sup\frac{1}{T}\sum_{k=1}^{T}\|e_N(k)\|^2 \leq \overline{\mu} \tag{3.24}$$

donde $\overline{\mu} = \text{máx}_k\left[\|\mu\|^2\right]$.

Demostración. Se selecciona un escalar positivo L_k como

$$L_k = \left\| \widetilde{W}(k) \right\|^2 \tag{3.25}$$

Pero por la ley de actualización (3.23), se tiene

$$\widetilde{W}(k+1) = \widetilde{W}(k) - \eta_k e(k) \Phi^T [X(k)]$$

Usando las desigualdades

$$\|a - b\| \geq \|a\| - \|b\| \qquad 2\|ab\| \leq a^2 + b^2$$

para cualquier a y b. Usando (3.22) y $0 \leq \eta_k \leq \eta \leq 1$, se tiene

$$\begin{aligned}
\Delta L_k &= L_{k+1} - L_k \\
&= \left\| \widetilde{W}(k) - \eta_k e(k) \Phi^T [X(k)] \right\|^2 - \left\| \widetilde{W}(k) \right\|^2 \\
&= \left\| \widetilde{W}(k)^2 \right\| - 2\eta_k \left\| e(k) \Phi^T [X(k)] \widetilde{W}(k) \right\| \\
&\quad + \eta_k^2 \|e(k) \Phi [X(k)]\|^2 - \left\| \widetilde{W}(k) \right\|^2 \\
&= \eta_k^2 \|e(k)\|^2 \|\Phi [X(k)]\|^2 \\
&\quad -2\eta_k \|e(k) [e(k) - \mu(k)]\| \\
&\leq \eta_k^2 \|e(k)\|^2 \|\Phi [X(k)]\|^2 \\
&\quad -2\eta_k \|e(k)\|^2 + 2\eta_k \|e(k) \mu(k)\| \\
&\leq \eta_k^2 \|e(k)\|^2 \|\Phi [X(k)]\|^2 \\
&\quad -2\eta_k \|e(k)\|^2 + \eta_k \|e(k)\|^2 + \eta_k \|\mu(k)\|^2 \\
&= -\eta_k \|e(k)\|^2 \left(1 - \eta_k \left\| \Phi^T (X) \right\|^2 \right) \\
&\quad + \eta_k \|\mu(k)\|^2 .
\end{aligned} \tag{3.26}$$

Puesto que $\eta_k = \frac{\eta}{1+\|\Phi[X(k)]\|^2}$

$$\begin{aligned}
\eta_k \left(1 - \eta_k \left\|\Phi\left[X\left(k\right)\right]\right\|^2\right) &= \eta_k \left(\frac{\eta}{1+\eta_k \left\|\Phi\left[X\left(k\right)\right]\right\|^2} \right. \\
&\quad \left. \times \left\|\Phi\left[X\left(k\right)\right]\right\|^2\right) \\
&\geq \eta_k \left(1 - \eta \frac{\text{máx}_k \left(\left\|\Phi\left(k\right)\right\|^2\right)}{1+\text{máx}_k \left(\left\|\Phi\left[X\left(k\right)\right]\right\|^2\right)}\right) \\
&\geq \eta_k \left(1 - \frac{\text{máx}_k \left(\left\|\Phi\left(k\right)\right\|^2\right)}{1+\text{máx}_k \left(\left\|\Phi\left[X\left(k\right)\right]\right\|^2\right)}\right) \\
&= \frac{\eta_k}{1+\text{máx}_k \left(\left\|\Phi\left[X\left(k\right)\right]\right\|^2\right)} \\
&\geq \frac{\eta}{\left[1+\text{máx}_k \left(\left\|\Phi\left[X\left(k\right)\right]\right\|^2\right)\right]^2}
\end{aligned}$$

Así

$$\Delta L_k \leq \pi \left\|e\left(k\right)\right\|^2 + \eta \left\|\mu\left(k\right)\right\| \tag{3.27}$$

donde π se define como

$$\pi = \frac{\eta}{\left[1+\text{máx}_k \left(\left\|\Phi\left[X\left(k\right)\right]\right\|^2\right)\right]^2}$$

Porque

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

donde $n \,\text{mín}\left(\widetilde{w}_i^2\right)$ y $\text{máx}\left(\widetilde{w}_i^2\right)$ son funciones $\mathcal{K}_\infty$, y $\pi \left\|e\left(k\right)\right\|^2$ es una función $\mathcal{K}$. Así L_k admite una función ISS-Lyapunov. Por el Teorema 3.1, la dinámica del error de identificación entrada-salida es estable. De (3.22) y (3.25) se sabe que L_k es la función de $e\left(k\right)$ y $\mu\left(k\right)$. La ENTRADA corresponde al segundo término de (3.27),i.e., el error modelado $\mu\left(k\right)$. El ESTADO corresponde al primer término de (3.27), i.e., el modelado del error $e\left(k\right)$. Porque la ENTRADA $\mu\left(k\right)$ es acotada y la dinámica es ISS, el ESTADO $e\left(k\right)$ es acotado.

La ecuación (3.26) se puede reescribir como

$$\begin{aligned}
\Delta L_k &\leq -\eta \frac{\left\|e(k)\right\|^2}{\left[1+\text{máx}_k \left(\left\|\Phi[X(k)]\right\|^2\right)\right]^2} + \eta \left\|\mu\left(k\right)\right\|^2 \\
&\leq -\eta \frac{\left\|e(k)\right\|^2}{\left[1+\text{máx}_k \left(\left\|\Phi[X(k)]\right\|^2\right)\right]^2} + \eta\overline{\mu}
\end{aligned} \tag{3.28}$$

Resumiendo (3.28) desde 1 hasta T y usando $L_T > 0$ y L_1 una constante, se obtiene

$$\begin{aligned} L_T - L_1 &\leq -\eta \sum_{k=1}^{T} \|e_N(k)\|^2 + T\eta\overline{\mu} \\ \eta \sum_{k=1}^{T} \|e_N(k)\|^2 &\leq L_1 - L_T \leq L_1 + T\eta\overline{\mu} \end{aligned}$$

(3.24) se establece. ■

Comentario 3.1 *En general, un sistema difuso no puede igualar cualquier sistema no lineal exactamente. Los parámetros del sistema difuso no convergerá a sus valores óptimos. La idea de identificación en línea es forzar la salida del sistema difuso a seguir la salida de la planta. Aunque los parámetros no pueden converger a sus valores óptimos, (3.24) muestra que la normalización del error de identificación convergerá a la bola de radio $\overline{\mu}$. Si el sistema difuso (3.15) puede igualar la planta no lineal (3.3) exactamente ($\mu(k) = 0$), i.e., se puede encontrar la mejor función de membresía $\mu_{A_{ji}}$ y W^* tal que el sistema no lineal se pueda escribir como $y(k) = W^*\Phi\left[\mu_{A_{ji}}\right]$. Puesto que $\|e(k)\| > 0$, la misma ley de aprendizaje (3.23) hará asimptoticamente estable el error de identificación $\|e(k)\|$.*

$$\lim_{k\to\infty} \|e(k)\| = 0. \tag{3.29}$$

Comentario 3.2 *Normalizando el rango de aprendizaje η_k en (3.23) que es variante en el tiempo en orden para asegurar la estabilidad del error de identificación. Estos rangos de aprendizaje son mas fácil de decidir que en [73] (por ejemplo seleccionando $\eta = 1$), sin necesidad de cualquier información a priori. Los rangos de aprendisaje variantes en el tiempo pueden ser encontrados en un mismo esquema estandar adaptable [22]. Pero estos necesitan modificaciones robustas para garantizar estabilidad de la identificación. La ecuación (3.15) es similar al resultado de [51], sin embargo, los enfoques son diferentes. El algoritmo esta derivado del análisis de estabilidad (o función ISS-Lyapunov), el algoritmo de [51] fue obtenido de la minimización de la función de costo. El enfoque a la cota del error de identificación, [51] enfocado al análisis de convergencia. Es interesante ver que dos métodos diferentes pueden obtener resultados similares.*

Ahora el caso de las funciones de membresía entrenadas de consecuencia y premisia. Las condiciones iniciales son $c_{ji}(1) = x_i^*$, $w_j(1) = y_i^*$, $\sigma_{ji}(1)$ es aleatorio en $(0,1)$. Puesto que las funciones de membresía son funciones Gausianas, la salida del sistema difuso se puede expresar como

$$\widehat{y} = \frac{\sum_{i=1}^{sv_j} w_i \left[\prod_{j=1}^{n} \exp\left(-\frac{(x_j - c_{ji})^2}{\sigma_{ji}^2}\right)\right]}{\left[\sum_{i=1}^{sv_j}\prod_{j=1}^{n} \exp\left(-\frac{(x_j - c_{ji})^2}{\sigma_{ji}^2}\right)\right]} \tag{3.30}$$

Se define

$$z_i = \prod_{j=1}^{n} \exp\left(-\frac{(x_j - c_{ji})^2}{\sigma_{ji}^2}\right) \quad a = \sum_{i=1}^{sv_j} w_i z_i \quad b = \sum_{i=1}^{sv_j} z_i$$

Así $\widehat{y}_q = \frac{a_q}{b}$. Similar a 3.21, la planta no lineal 3.3 puede ser representada como

$$\widehat{y}_q = \frac{\sum_{i=1}^{sv_j} w_{qi}^* \left[\prod_{j=1}^{n} \exp\left(-\frac{\left(x_j - c_{ji}^*\right)^2}{\sigma_{ji}^2}\right)\right]}{\left[\sum_{i=1}^{sv_j}\prod_{j=1}^{n} \exp\left(-\frac{\left(x_j - c_{ji}^*\right)^2}{\sigma_{ji}^{*2}}\right)\right]} - \mu \tag{3.31}$$

donde w_{qi}^*, c_{ji}^* y σ_{ji}^{*2} son parámetros desconocidos las cuales minimizan la dinámica no modelada μ. En el caso de tres variables independientes, una función suave f tiene como fórmula de Taylor

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

donde R_l es el residuo de la fórmula de Taylor. Si x_1, x_2 y x_3 corresponden a w_{qi}^*, c_{ji}^* y σ_{ji}^{*2}, x_1^0, x_2^0 y x_3^0 corresponden a w_{qi}, c_{ji} y σ_{ji}^2

$$\begin{aligned} y + \mu = \widehat{y} + \sum_{i=1}^{l} \frac{\left(w_i^* - w_i\right) z_i}{b} + \sum_{i=1}^{l}\sum_{j=1}^{n} \frac{\partial}{\partial c_{ji}}\left(\frac{a}{b}\right)\left(c_{ji}^* - c_{ji}\right) \\ + \sum_{i=1}^{l}\sum_{j=1}^{n} \frac{\partial}{\partial \sigma_{ji}}\left(\frac{a}{b}\right)\left(\sigma_{ji}^* - \sigma_{ji}\right) + R_1 \end{aligned} \tag{3.32}$$

donde R_{1q} es el error de segundo orden de aproximación de la serie de Taylor. Usando la regla de la cadena, se obtiene

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

En la forma matricial

$$y+\mu = \widehat{y}-Z\left(k\right)\widetilde{W}-Dz\overline{C}_kE-Dz\overline{B}+R_1 \tag{3.33}$$

donde

$$\begin{aligned}
Z\left(k\right) &= \left[\frac{z_1}{b}\cdots\frac{z_1}{b}y\right]^T \qquad W=\left[w_1\ldots w_l\right] \qquad \widetilde{W}=W-W^* \\
Dz &= \left[2z_1\frac{w_{q1}-\widehat{y}_q}{b},\ldots,2z_1\frac{w_{ql}-\widehat{y}_q}{b}\right] \\
\overline{C}_k &= \left[\begin{array}{ccc} \frac{x_1-c_{11}}{\sigma_{11}^2}\left(c_{11}-c_{11}^*\right) & \cdots & \frac{x_n-c_{n1}}{\sigma_{n1}^2}\left(c_{n1}-c_{n1}^*\right)\end{array}\right] \\
\overline{B}_k &= \left[\begin{array}{ccc} \frac{\left(x_1-c_{11}\right)^2}{\sigma_{11}^3}\left(\sigma_{11}-\sigma_{11}^*\right) & \cdots & \frac{\left(x_n-c_{n1}\right)^2}{\sigma_{n1}^3}\left(\sigma_{n1}-\sigma_{n1}^*\right)\end{array}\right]
\end{aligned}$$

El error de identificación se define como $e_q=\widehat{y}_q-y_q$, usando (3.33) se tiene

$$e_q = Z\left(k\right)\widetilde{W}+Dz\overline{C}_k+Dz\overline{B}_k+R_1-\mu \tag{3.34}$$

En forma vectorial

$$e\left(k\right) = \widetilde{W}_kZ\left(k\right)+D_z\left(k\right)\overline{C}_k+D_k\left(k\right)\overline{B}_k+\zeta\left(k\right) \tag{3.35}$$

donde $\widetilde{W}_k=\left[\begin{array}{ccc} w_{11}-w_{11}^* & \cdots & w_{m1}-w_{m1}^*\end{array}\right]$, $D_z\left(k\right)=\left[\begin{array}{ccc} 2z_1\frac{w_{11}-\widehat{y}_1}{b} & \cdots & 2z_1\frac{w_{1l}-\widehat{y}_1}{b}\end{array}\right]$, $\zeta\left(k\right)=\mu-R_1$, $\mu=\left[\mu_1\cdots\mu_m\right]^T$, $R_1=\left[R_{11}-R_{1m}\right]^T$.

La cota de la función Gausiana ϕ y la planta es BIBO estable, μ y R_1 en (3.31) y (3.32) son acotadas. Así $\zeta\left(k\right)$ en (3.35) es acotado. El siguiente teorema da un algoritmo estable para redes neuro difusas tipo mamdani discretas en el tiempo.

Teorema 3.2 *Si se usa red neuro difusas tipo mamdani discretas en el tiempo (3.30) para identificar la planta no lineal (3.3), el siguiente algoritmo retropropagación (backpropagation) hace la identificación del error $e(k)$ acotado*

$$\begin{aligned} W_{k+1} &= W_k - \eta_k e(k) Z(k)^T \\ c_{ji}(k+1) &= c_{ji}(k) - 2\eta_k z_i \frac{w_{pi}-\widehat{y}_p}{b} \frac{x_j - c_{ji}}{\sigma_{ji}^3} (\widehat{y}_q - y_q) \\ \sigma_{ji}(k+1) &= \sigma_{ji}(k) - 2\eta_k z_i \frac{w_{pi}-\widehat{y}_p}{b} \frac{(x_j - c_{ji})^2}{\sigma_{ji}^3} (\widehat{y}_q - y_q) \end{aligned} \tag{3.36}$$

donde $\eta_k = \frac{\eta}{1+\|Z\|^2+2\|D_z\|^2}$, $0 < \eta \leq 1$. *El error promedio de identificación satisface*

$$J = \lim_{T\to\infty} \sup \frac{1}{T} \sum_{k=1}^{T} \|e(k)\|^2 \leq \frac{\eta}{\pi} \overline{\zeta} \tag{3.37}$$

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

Demostración. *Este es muy similar al Teorema 3.1.* ■

Para el modelo neuro difuso tipo TSK (3.37), se selecciona A_{ji} como funciones Gausianas. La salida del sistema lógico difuso se puede expresar como

$$\widehat{y} = \frac{\sum_{i=1}^{sv_j} \left(\sum_{k=0}^{n} p_k^i x_k \right) \prod_{j=1}^{n} \exp\left(-\frac{\left(x_j - c_{ji}^*\right)^2}{\sigma_{ji}^2} \right)}{\left[\sum_{i=1}^{sv_j} \prod_{j=1}^{n} \exp\left(-\frac{\left(x_j - c_{ji}^*\right)^2}{\sigma_{ji}^{*2}} \right) \right]} \tag{3.38}$$

donde $x_0 = 1$. La parte $\sum_{i=1}^{sv_j} (w_i^* - w_i) z_i / b$ en (3.32) es cambiada por

$$\sum_{i=1}^{sv_j} \left(\sum_{k=0}^{n} \left(p_{qk}^{i*} x_k - p_{qk}^i x_k \right) x_k \right) \frac{z_i}{b}.$$

Para redes neuro difusas tipo TSK se tiene lo siguiente

$$\begin{aligned} p_k^i(k+1) &= p_k^i(k) - \eta_k (\widehat{y} - y) \frac{z_i}{b} x_k \\ c_{ji}(k+1) &= c_{ji}(k) - 2\eta_k z_i \frac{w_i - \widehat{y}_p}{b} \frac{x_j - c_{ji}}{\sigma_{ji}^2} (\widehat{y} - y) \\ \sigma_{ji}(k+1) &= \sigma_{ji}(k) - 2\eta_k z_i \frac{w_{pi} - \widehat{y}_p}{b} \frac{(x_j - c_{ji})^2}{\sigma_{ji}^3} (\widehat{y} - y) \end{aligned}$$

donde $\eta_k = \frac{\eta}{1+\|Z\|^2+2\|D_z\|^2}$, $0 < \eta \leq 1$.

Se puede ver que los requerimientos de TSK tienen una diferente formulación comparada con el modelo tradicional Mamdani que es la inclusión de la ley de actualización para el $p^i_{qk}(k)$.

Cada grupo tiene un modelo difuso (3.15). Usando la idea del modelo Takagi-Sugeno [66], se pueden combinar los modelos locales en cada grupo dentro de un modelo global. Las reglas difusas tienen la siguiente forma

$$\mathrm{R}^j\text{: SI } x_{l_1^j} \leq x_1 \leq x_{l_2^j} \text{ , y } \cdots \; x_{l_1^j} \leq x_n \leq x_{l_2^j} \text{ ENTONCES } y(k) = f_j[X(k)]$$

donde $j = 1 \cdots p$, p es el número del grupo del agrupamiento en línea. Las funciones de membresía para x_i se definen como en la Figura 3.6. El modelo difuso final es

$$\widehat{y} = \left(\sum_{i=1}^{p} f_i[X(k)] \left[\prod_{j=1}^{n} \left(\mu_{A_i^j} \right) \right] \right) / \left(\sum_{i=1}^{p} \left[\prod_{j=1}^{n} \left(\mu_{A_i^j} \right) \right] \right) \tag{3.39}$$

Ahora se usa el mismo ejemplo propuesto en la sección anterior para explicar las funciones de membresia entrenadas para cada grupo. Después del entrenamiento de las funciones de membresía, las 16 funciones de membresía para x_1 se muestran en la Figura 5.2.Entonces los parámetros fijados de las funciones de membresía, usan otros 300 datos para la prueba del modelo. El dato probado es $x_1(k) = 1 - \frac{2k}{T}$, $x_2(k) = -1 + \frac{2k}{T}$, $k = 1, 2, \cdots, T$. El resultado final del modelado despues de los modelos locales ($p = 7$) se muestran en la Figura 3.8

3.4. Sistema MIMO

Considerar el siguiente sistema no lineal discreto en tiempo

$$x(k+1) = f[x(k), u(k)], \quad y(k) = h[x(k)] \tag{3.40}$$

donde $u(k) \in \Re^m$ es el vector de entradas, $x(k) \in \Re^n$ es un vector de estados, y $y(\,k) \in \Re^p$ es el vector de salidas. f y h son funciones suaves no lineales $f, h \in C^\infty$.

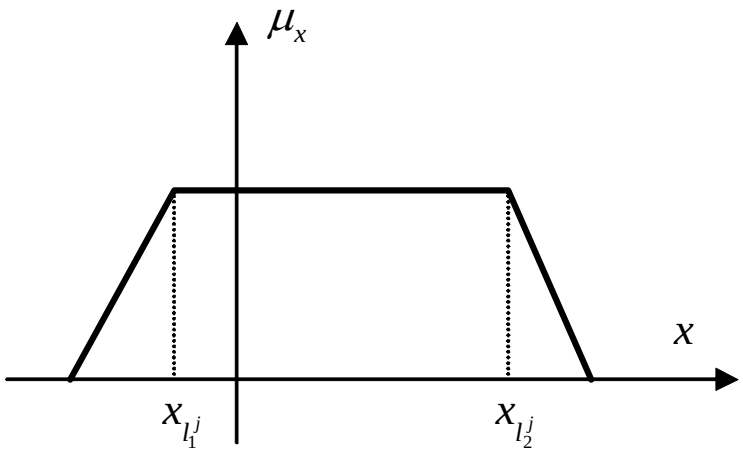

Figura 3.6: Funcion de membresia de x_i.

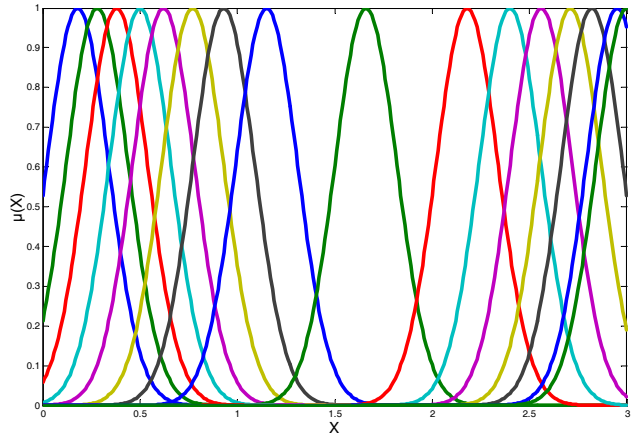

Figura 3.7: Funciones de membresia de x_1 en las 16 reglas difusas.

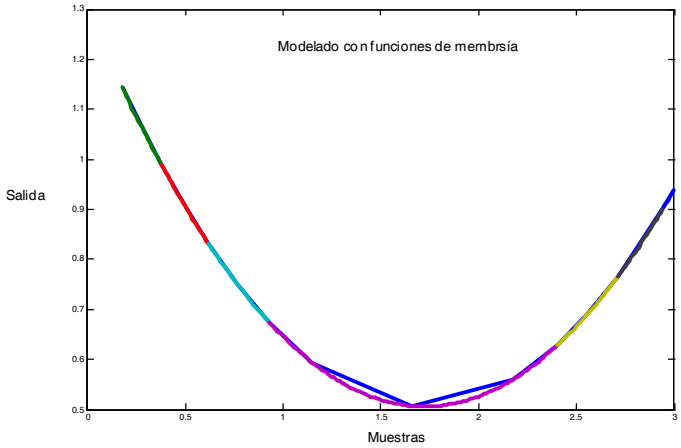

Figura 3.8: Resultado final del modelado difuso.

De (3.40) se tiene

$$\begin{aligned} &y(k) = h\left[x\left(k\right)\right] := F_1\left[x\left(k\right)\right], \quad y(k+1) = h\left[f\left[x\left(k\right), u\left(k\right)\right]\right] := F_2\left[x\left(k\right), u\left(k\right)\right] \\ &y(k+n-1) := F_n\left[x\left(k\right), u\left(k\right), u(k+1)\cdots u(k+n-2)\right] \end{aligned} \tag{3.41}$$

Denotando

$$\begin{aligned} Y(k) &= \left[y\left(k\right), y\left(k+1\right), \cdots y\left(k+n-1\right)\right]^T \\ U(k) &= \left[u\left(k\right), u\left(k+1\right), \cdots u\left(k+n-2\right)\right]^T \end{aligned}$$

así $Y(k) = F\left[x\left(k\right), U(k)\right]$, $F = \left[F_1 \cdots F_n\right]^T$. Dado que (3.40) es un sistema no lineal suave, (3.41) se puede expresar como $x(k+1) = g\left[Y(k+1), U(k+1)\right]$. Esto lleva al caso multivariable NARMA [6]

$$y(k) = h\left[x\left(k\right)\right] = \Psi\left[y\left(k-1\right), y\left(k-2\right), \cdots u\left(k-1\right), u\left(k-2\right), \cdots\right] = \Psi\left[X\left(k\right)\right] \tag{3.42}$$

donde

$$X\left(k\right) = \left[y\left(k-1\right), y\left(k-2\right), \cdots u\left(k-d\right), u\left(k-d-1\right), \cdots\right]^T \tag{3.43}$$

$\Psi\left(\cdot\right)$ 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 escalar medibles, d es un retardo en el tiempo.

Un modelo genérico difuso se presenta como una colección de reglas difusas de la siguiente forma (modelado difuso tipo Mamdani fuzzy [52])

$$\mathrm{R}^i\text{: SI } x_1 \text{ es } A_{1i} \text{ y } x_2 \text{ es } A_{2i} \text{ y } \cdots \ x_n \text{ es } A_{ni} \text{ ENTONCES } \widehat{y}_1 \text{ es } B_{1i} \text{ y} \cdots \widehat{y}_p \text{ es } B_{pi} \quad (3.44)$$

Se usa $l(i = 1, 2 \cdots l)$ reglas difusas SI-ENTONCES para realizar un mapeo desde un vector de entradas lingüistícas $X = [x_1 \cdots x_n] \in \Re^n$ hacia un vector de salidas lingüísticas $\widehat{Y}(k) = [\widehat{y}_1 \cdots \widehat{y}_m]^T$. $A_{1i}, \cdots A_{ni}$ y $B_{1i}, \cdots B_{mi}$ son conjuntos estándares difusos. Cada variable de entrada x_i tiene l_i conjuntos difusos. En el caso de conexión completa, $l = l_1 \times l_2 \times \cdots l_n$. De [72] se conoce, usando el producto inferencia, el centro promedio y el singleton difusificador, la p-ésimo salida del sistema lógico difuso se puede expresar como

$$\widehat{y}_p = \left(\sum_{i=1}^{l} w_{pi} \left[\prod_{j=1}^{n} \mu_{A_{ji}} \right] \right) / \left(\sum_{i=1}^{l} \left[\prod_{j=1}^{n} \mu_{A_{ji}} \right] \right) = \sum_{i=1}^{l} w_{pi} \phi_i \quad (3.45)$$

con $\mu_{A_{ji}}$ son las funciones de membresía de los conjuntos difusos A_{ji}, w_{pi} es el punto en el cual $\mu_{B_{pi}} = 1$. Si se define

$$\phi_i = \prod_{j=1}^{n} \mu_{A_{ji}} / \sum_{i=1}^{l} \prod_{j=1}^{n} \mu_{A_{ji}} \quad (3.46)$$

(3.45) se puede expresar en forma matricial como

$$\widehat{Y}(k) = W(k)\, \Phi[X(k)] \quad (3.47)$$

donde el parámetro $W(k) = \begin{bmatrix} w_{11} & & w_{1l} \\ & \ddots & \\ w_{m1} & & w_{ml} \end{bmatrix}$, el vector de datos $\Phi[X(k)] = [\phi_1 \cdots \phi_l]^T$.

Para el modelo difuso Takagi-Sugeno-Kang [66],

$$\mathrm{R}^i\text{: SI } x_1 \text{ es } A_{1i} \text{ y } x_2 \text{ es } A_{2i} \text{ y } \cdots \ x_n \text{ ies } A_{ni} \text{ ENTONCES } \widehat{y}_j = p^i_{j0} + p^i_{j1} x_1 + \cdots p^i_{jn} x_n \quad (3.48)$$

donde $j = 1 \cdots m$. La q-ésimo salida del sistema lógico difuso se puede expresar como

$$\widehat{y}_q = \sum_{i=1}^{l} \left(p^i_{q0} + p^i_{q1} x_1 + \cdots p^i_{qn} x_n \right) \phi_i \quad (3.49)$$

donde ϕ_i se define como en (3.46). (3.49) también se puede expresar en la forma Mamdani (3.47),

$$\widehat{Y}(k) = W(k)\,\Phi[X(k)] \tag{3.50}$$

donde $\widehat{Y}(k) = [\widehat{y}_1 \cdots \widehat{y}_m]^T$

$$W(k) = \begin{bmatrix} p_{10}^1 \cdots p_{10}^l & p_{11}^1 \cdots p_{11}^l & \cdots & p_{1n}^1 \cdots p_{1n}^l \\ \vdots & \vdots & & \vdots \\ p_{m0}^1 \cdots p_{m0}^l & p_{m1}^1 \cdots p_{m1}^l & \cdots & p_{mn}^1 \cdots p_{mn}^l \end{bmatrix}$$

$$\Phi[X(k)] = \begin{bmatrix} \phi_1 \cdots \phi_l & x_1\phi_1 \cdots x_1\phi_l & \cdots & x_n\phi_1 \cdots x_n\phi_l \end{bmatrix}^T$$

3.5. Aplicación para la mezcla de petróleo crudo

En esta sección se discutirá un proceso típico para la mezcla de petróleo crudo en PEMEX. La hoja de flujo se muestra en la Figura 3.9-(a). Los analizadores de API (*Americam Petroleum Institute*) y porcentaje de flujo son instalados en cada bloque. Esto tiene tres mezcladores (M_1, M_2 y M_3), un equipamiento de dehidratación y un tanque. La Figura 3.9-(b) describe el proceso estático de la mezcla de petróleo crudo, q_i es el porcentaje de flujo, p_i es la propiedad de $i-ésima$ materia prima, está puede ser de gravedad API (*Americam Petroleum Institute*). Se ve al proceso de mezclado como componentes múltiples de mezclado en la Figura 3.10. El proceso de modelado para la mezcla de petróleo crudo es en línea [30],[31]. Las propiedades estáticas se pueden analizar por termodinámica. Pero modelos matemáticos trabajan solo en algúnas condiciones especiales. En esta aplicación real (optimización) se obtienen solo datos entrada/salida, la lógica difusa se puede aplicar al modelo de la mezcla de petróleo crudo. Porque los nuevos datos llegan continuamente, se usará la técnica de agrupamiento en línea. Los datos llegan una vez por día. Parecierá que esto no necesita del aprendizaje en línea, porque se puede reentrenar un sistema difuso en cada día. Cuándo el sistema no lineal es cambiado, por ejemplo la prescripción de la mezcla de petróleo crudo es modificado, los datos historicos no pueden ser aplicados. Por el método bañera se tiene

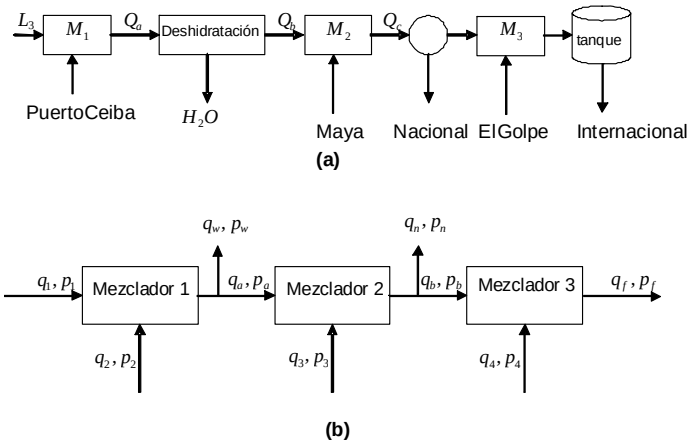

Figura 3.9: Proceso de mezclado de petróleo crudo TMDB (Terminal Marítima de Dos Bocas).

que usar el factor de olvido o mover la ventana hacia los datos recientes. El agrupamiento en línea propuesto puede evitar este tipo de problema.

Se usa el siguiente modelo difuso Mandani, para j-ésima regla

$$\begin{aligned} \mathrm{R}^j\text{: SI } q_1 \text{ es } A^j_{q_1} \text{ y } p_1 \text{ es } A^j_{p_1} \text{ y } \cdots \ q_4 \text{ es } A^j_{q_4} \text{ y } p_4 \text{ es } A^j_{p_4} \\ \text{ENTONCES } q_f \text{ es } B^j_{q_f} \text{ y } p_f \text{ es } B^j_{p_f} \end{aligned} \tag{3.51}$$

1. Primero, para la mezcla de petróleo crudo, los efectos de la salida de gravedad API

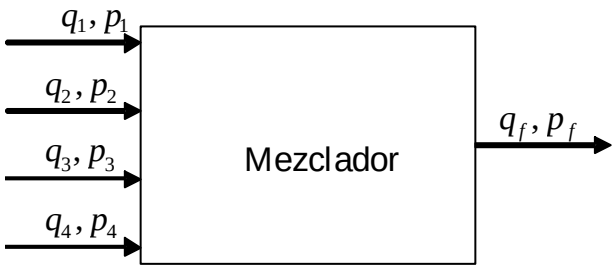

Figura 3.10: Modelo Integrado.

(*Americam Petroleum Institute*) son más que el flujo de entrada para el particionamiento, se selecciona $\alpha = 0{,}4$. Así con un conocimiento a priori, se saben los cambios máximos en la entrada y la salida que son alrededor de 3 y 1, $\alpha \|x_{\text{máx}} - x_{\text{mín}}\| + (1-\alpha) \|x_{\text{máx}} - x_{\text{mín}}\| = 1{,}8$, Así L se podría escoger tal que $L < 1{,}8$, en esta aplicación se selecciona $L = 1{,}5$. La Figura 3.11 muestra los resultados del agrupamiento en línea para datos de dos meses. Aquí "*" representa los centros de cada grupo, "+" es la frontera entre los grupos. Existen 3 grupos en los datos de 2 meses.

2. Segundo, se usa *support vector machines* (3.11) para obtener los support vectors como en (3.13). El número de support vectors para los 3 grupos son 2,3 y 3. Así se tienen 8 reglas como en (3.51). Los centros de las funciones Gausianas de esas reglas son las posiciones de los support vector.

3. Ahora se usan los datos en cada grupo para actualizar las funciones de membresía. Por ejemplo, los intervalos de tiempo del segundo grupo son $l_1^2 = 14$, $l_2^2 = 28$, los datos de 14 a 18 son usados para entrenar las 3 reglas difusas. Los datos entrenados son de dos años grabados, datos de 730 entradas/salidas. Después del entrenamiento, las funciones de membresía $A_{q_1}^j$, $A_{p_1}^j$, $A_{q_4}^j$ y los centros de $B_{p_f}^j$ del 3rd grupo se muestran en la Figura 3.12.

4. Cuando se combinan los tres modelos difusos locales (8 reglas) dentro de un modelo global Takagi-Sugeno da la forma

$$\widehat{y} = \left(\sum_{i=1}^{3} f_i \left[x\left(k\right) \right] \left[\prod_{j=1}^{8} \left(\mu_{A_i^j} \right) \right] \right) / \left(\sum_{i=1}^{3} \left[\prod_{j=1}^{8} \left(\mu_{A_i^j} \right) \right] \right)$$

Después que la fase de entrenamiento termina, se usan 28 datos entrenados, los cuales son datos grabados de un mes de otro año. De esta manera, se asegura que la fase de prueba es independiente de la fase de entrenamiento. Los resultados del modelado final se muestran en la Figura 3.13. La primera figura muestra la fase de entrenamiento, más claramente, solo son reportados una parte de datos entrenados (de 600 a 740). El comportamiento y el error de modelado de la fase de prueba se muestran en la segunda y tercera figura.

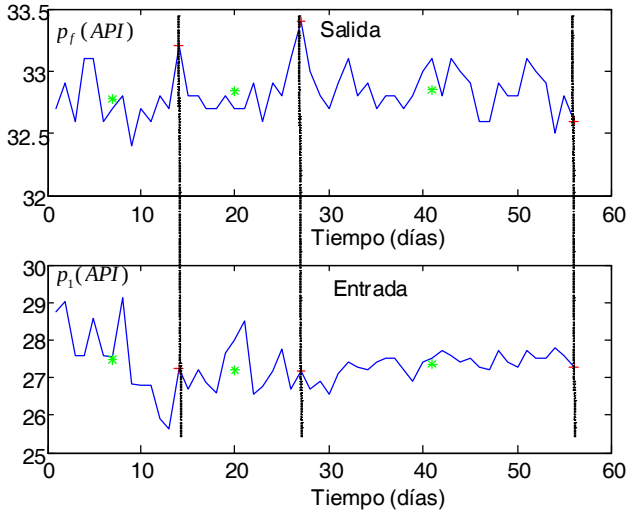

Figura 3.11: Modelado en línea para dos meses de datos ($\alpha = 0{,}4$).

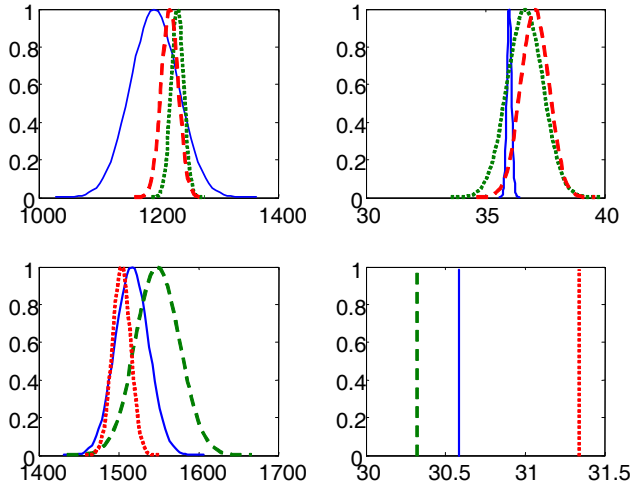

Figura 3.12: Funciones de membresía de $A^j_{q_1}$, $A^j_{p_1}$, $A^j_{q_4}$ y los centros de $B^j_{p_f}$.

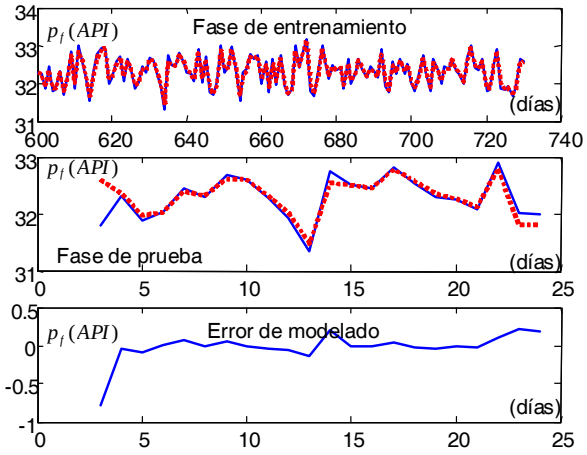

Figura 3.13: Modelado de la mezcla de petróleo crudo con agrupamiento en línea y redes neuro difusas.

Ahora se cambian los parámetros de diseño α, $\alpha = 0{,}1$. Se saben que los cambios máximos en la entrada y la salida son alrededor de 3 y 1, $\alpha \|x_{\text{máx}} - x_{\text{mín}}\| + (1-\alpha)\|x_{\text{máx}} - x_{\text{mín}}\| = 1{,}2$, Así L se podría escoger tal que $L < 1{,}2$, en esta aplicación se selecciona $L = 1$. La Figura 3.14 muestra el resultado del agrupamiento en línea para 2 meses de datos. El número de support vectors para los 6 grupos son 2,2,2,3,2,2 y 2. Se tienen 15 reglas como en (3.51). Los resultados del modelado son peores que cuando $\alpha = 0{,}4$, porque no existen suficientes datos en cada grupo para actualizar la funciones de membresía y las semánticas de las reglas difusas podrían perderse en algunas reglas.

Finalmente, se comparan el enfoque de arriba con los siguientes métodos de modelados difusos.

1. El enfoque de modelado difuso adaptable [34][72] podrían ser los métodos más populares. También usa 8 reglas difusas en la forma de (3.51). Las funciones de membresía Gausianas son seleccionadas primero aleatoreamente, entonces se usasn los mismos datos entrenados para actualizar esas funciones de membresía. Los datos de prueba

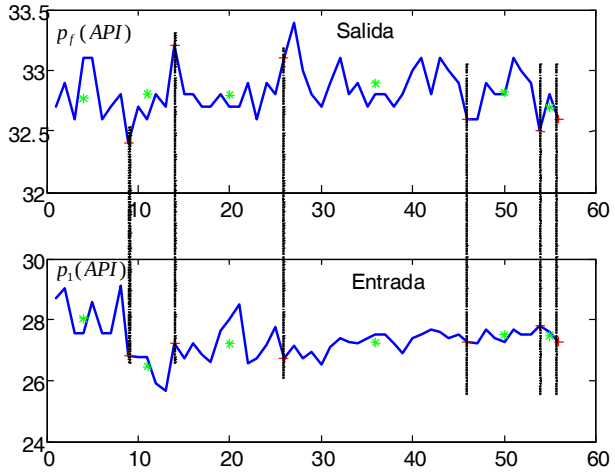

Figura 3.14: Modelado en línea para dos meses de datos ($\alpha = 0{,}1$).

son también los mismos que los anteriores.

2. El modelado difuso vía agrupamiento en línea puede reducir el número de reglas, se considera el agrupamiento sin considerar el intervalo de tiempo y cada grupo entrenado usa todos los datos como en [26], [68] y [1]. Si se usa el umbral de la salida y la entrada como 1.5. El particionamiento de la entrada y salida se muestra en la Figura 3.14 las reglas difusas son construidas, entonces todos los datos son usados para entrenarlos.

3. Otro método popular de modelado difuso es la lógica difusa con agrupamiento de datos [34][54][12]. La entrada es particionada. Con el umbral 1.5 se tienen 8 grupos en el espacio de entrada. Así se construyen 8 reglas difusas.

El modelado del error en la fase de prueba del modelado difuso de arriba se muestra en la Figura 3.51. El modelado del error cuadrático se define como $\sqrt{\frac{1}{25}\sum_{k=1}^{25} e^2(k)}$: $RMS_1 = 0{,}075$ (agrupamiento en línea con SVM), $RMS_1 = 0{,}091$ (modelado difuso con agrupamiento), $RMS_1 = 0{,}19$ (modelado difuso adaptable).

La tabla 1 tiene la comparación concreta entre nuestro enfoque (agrupamiento en línea con SVM) con otros 3 métodos de modelado difuso. Todos los 4 métodos usan modelados difusos, esto es simple si se conoce las reglas difusas. El agrupamiento en línea y el modelado difuso adaptable pueden cambiar el modelo en línea, los otros dos enfoques son usados fuera de línea. Nuestro algoritmo es mucho más complejo porque usa la optimización en SVM, así el tiempo computacional es muy grande y la exactitud es muy alta.

	Modelado adaptable difuso [34][72]	Modelado difuso vía agrupamiento en línea [36][68][1]	Modelado difuso vía agrupamiento[12]	en-línea con SVM
modelo	simple	normal	simple	simple
en linea	si	no	no	si
algoritmo	muy simple	simple	simple	normal
velocidad	muy rápido	rápido	rápido	normal
estructura	no adaptable	adaptable	adaptable	adaptable
parámetro	adaptable	adaptable	adaptable	adaptable
exactitud	normal	alto	normal	muy alto

Tabla1.Comparación de 4 métodos de modelado difuso

Capítulo 4

Modelado via agrupamiento en línea y SVM difuso

El proposito de FSVMs (fuzzy *support vector machines* 4.1) es tratar los datos entrenados con diferente importancia en el proceso de entrenamiento. La falta del término de la función de costo se minimiza, reformulando la contruccción del problema de optimización, y la construcción de la función Lagrange así que las soluciones para el hiperplano óptimo en la primera forma puede ser encontrada en la forma dual.

El SVM puede separar los datos en dos clases con un hiperplano de margen máximo [15]. Si el entrenamiento es separable para el hiperplano, la función es escogida como $f(\mathbf{x}) = (\mathbf{w} \cdot \mathbf{x}) + b$. El márgen es definido como la distancia mínima de una muestra para la superficie de la solución. El márgen en turno que podemos medir para la longitud del vector w, de forma que los puntos cercanos al hiperplano satisface $|(\mathbf{w} \cdot \mathbf{x}) + b| = 1$.

En este trabajo usamos SVM para el cálculo de la función de estimación. Para encontrar los vectores de soporte en el j de grupo, se usan los datos de entrada /salida $[y(k), \mathbf{x}(k)]$, $k \in \left[l_1^j, l_2^j\right]$ para aproximar una función no lineal. Considere la regresión en un conjunto de funciones no lineales

$$f(\mathbf{x}) = \mathbf{w}^T \varphi(\mathbf{x}) + b \tag{4.1}$$

donde el kernel es $K(\mathbf{x}, \mathbf{x}_k) = \varphi(\mathbf{x})^T \varphi(\mathbf{x}_k)$.

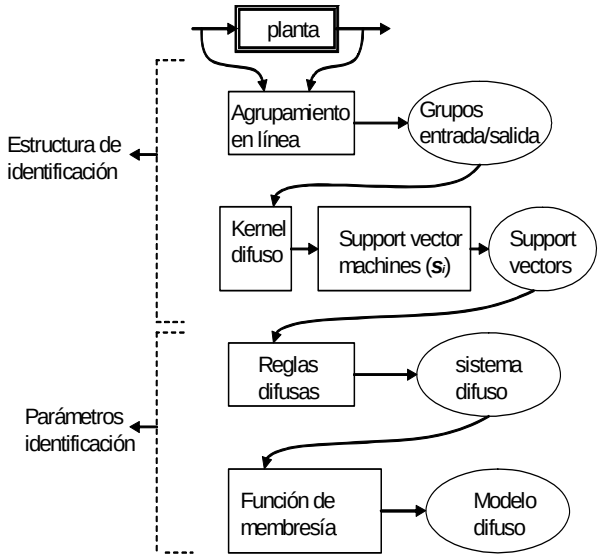

Figura 4.1: Agrupamiento en línea FSVM's y modelado difuso.

4.1. Kernel difuso

Existen muchas maneras de escoger el kernel $K(\mathbf{x}, \mathbf{x}_k)$, solo se requiere que $K(\mathbf{x}, \mathbf{x}_k)$ satisfaga la condición de Mercer [15]. Por ejemplo el kernel lineal $K(\mathbf{x}, \mathbf{x}_k) = \mathbf{x}_k^T \mathbf{x}$, el kernel MPL $K(\mathbf{x}, \mathbf{x}_k) = \tanh\left(k_1 \mathbf{x}_k^T \mathbf{x} + k_2\right)$, kernel RBF $K(\mathbf{x}, \mathbf{x}_k) = \exp\left(-\left\|\mathbf{x} - \mathbf{x}_k\right\|^2 / \sigma^2\right)$. En este trabajo, se usa el kernel difuso $K(\widehat{x}, \widehat{z})$ el cual se define como

$$K(\widehat{x}, \widehat{z}) = \begin{cases} \prod_{i,k=1}^{M} u_i(\mathbf{x}_k) \cdot u_i(\mathbf{z}_k) & \widehat{x} \text{ y } \widehat{z} \text{ ambos están en el } j-\acute{e}simo \text{ agrupamiento} \\ 0 & \text{de otra manera} \end{cases}$$

donde $\widehat{x} = [x_1, x_2, x_3, \ldots, x_M] \in R^M$ y $\widehat{z} = [z_1, z_2, z_3, \ldots, z_M] \in R^M$ son cualesquiera dos muestras entrenadas. $u_i(\mathbf{x}_k)$ es la función de membresía del j-ésimo agrupamiento.

Sea el conjunto entrenado $S = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_v, y_v)\}$ con variables interpretativas $\mathbf{x}_i$ y las clases etiquetedas correspodientes y_i, para todo $i = 1, 2, \ldots, v$ donde v es el número total de muestras entrenadas. Se asume que las muestras entrenadas estan parti-

cionadas dentro de l agrupamientos. Se puede ejecutar la siguiente permutación de muestras entrenadas:

$$\begin{array}{rl} \text{grupo } 1 & = \left\{ \left(\mathbf{x}_1^1, y_1^1\right), \ldots, \left(\mathbf{x}_{k_1}^1, y_{k_1}^1\right) \right\} \\ \text{grupo } 2 & = \left\{ \left(\mathbf{x}_1^2, y_1^2\right), \ldots, \left(\mathbf{x}_{k_2}^2, y_{k_2}^2\right) \right\} \\ & \vdots \\ \text{grupo } l & = \left\{ \left(\mathbf{x}_1^l, y_1^l\right), \ldots, \left(\mathbf{x}_{k_l}^l, y_{k_l}^l\right) \right\} \end{array} \tag{4.2}$$

donde k_g, $g = 1, 2, \ldots, l$ es el número de puntos de calidad para el $g$ht agrupamiento, así que se tiene $\sum_{g=1}^{l} k_g = v$. Entonces el kernel difuso puede ser calculado usando el conjunto entrenado en (5.6), y la obtención matricial kernel $\mathbf{K}$ se puede reescribir de la siguiente forma:

$$K = \begin{array}{cccc} \mathbf{K}_1 & 0 & \cdots & 0 \\ 0 & \mathbf{K}_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \mathbf{K}_l \end{array} \in R^{v \times v} \tag{4.3}$$

donde $\mathbf{K}_g$, $g = 1, 2, \ldots, l$ [11].

Cómo escoger la función de membresía $u_j(x_i)$ es otro problema. La función Gausiana y la función triángulo son las funciones más populares para la función de membresía de sistemas difusos.Cuándo $u_j(x_i)$ es una función Gausiana, la función kernel es

$$K(\mathbf{x}_k, \mathbf{x}_j) = \varphi(\mathbf{x}_k) \cdot \varphi(\mathbf{x}_j) = \exp\left(\frac{\|\mathbf{x}_k - \mathbf{x}_j\|^2}{2\sigma^2}\right) \tag{4.4}$$

y el kernel difuso es

$$\mathbf{K}_g = \begin{array}{cccc} K(x_1^g, x_1^g) & K(x_1^g, x_2^g) & \cdots & K\left(x_1^g, x_g^g\right) \\ K\left(x_g^g, x_1^g\right) & K(x_2^g, x_2^g) & \ddots & \vdots \\ \vdots & \ddots & \ddots & K\left(x_{k_g-1}^g, x_{k_g}^g\right) \\ K\left(x_{k_g}^g, x_1^g\right) & \cdots & K\left(x_{k_g}^g, x_{k_g-1}^g\right) & K\left(x_{k_g}^g, x_{k_g}^g\right) \end{array} \in R^{k_g \times k_g} \tag{4.5}$$

Se asume que el conjunto difuso entrenado se conoce, el siguiente paso es formular el FSVM. Se comienza con la construcción de una función de costo, el FSVM tambien quiere

maximizar el margen de separación y minimizar el error de clasificación tal que una buena capacidad de generalizacion puede ser ejecutada.

La construcción del problema de optimización del FSVM se formula como

$$\begin{aligned} \text{mín} \quad & \Phi\left(w,\xi,s\right)=\tfrac{1}{2}w^{T}w+C\sum_{i=1}^{n}s_{i}^{m}\xi_{i} \\ sujeto\ a \quad & y_{i}\left(w^{T}x_{i}+b\right)\geq 1-\xi_{i}, \quad i=1,2,\ldots,n \\ & \xi_{i}\geq 0, i=1,2,\ldots,n \end{aligned} \tag{4.6}$$

donde m influye en la falta del término defusificado en la función de costo. Ahora la función de Lagrange es

$$\begin{aligned} Q\left(w,b,\xi,\alpha,\beta,s\right)&=\tfrac{1}{2}w^{T}w+C\sum_{i=1}^{n}s_{i}^{m}\xi_{i} \\ -\sum_{i=1}^{n}\alpha_{i}\left[y_{i}\left(w^{T}x_{i}+b\right)-1+\xi_{i}\right]&-\sum_{i=1}^{n}\beta\xi_{i} \end{aligned} \tag{4.7}$$

donde α_i y β_i son multiplicadores de Lagrange no negativos. Diferenciando Q respecto a w, β y ξ_i, e igualando los resultados a cero se obtienen las tres condiciones de optimalidad siguientes

$$\begin{aligned} \tfrac{\partial Q(w,b,\xi,\alpha,\beta,s)}{\partial w}&=w-\sum_{i=1}^{n}\alpha_{i}y_{i}x_{i}=0 \\ \tfrac{\partial Q(w,b,\xi,\alpha,\beta,s)}{\partial b}&=-\sum_{i=1}^{n}\alpha_{i}y_{i}=0 \\ \tfrac{\partial Q(w,b,\xi,\alpha,\beta,s)}{\partial \xi}&=C\mu_{i}^{m}-\alpha_{i}-\beta_{i}=0 \end{aligned} \tag{4.8}$$

Substituyendo la ecuación (4.27) dentro de la ecuación (4.26), la función de Lagrange es una función de α solamente. El problema dual se vuelve

$$\begin{aligned} \text{mín} \;\; Q\left(\alpha\right)=\alpha_{i}-\tfrac{1}{2}w^{T}w+C\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_{i}\alpha_{j}y_{i}y_{j}x_{i}^{T}x_{j} \\ \sum_{i=1}^{n}\alpha_{i}y_{i}=0 \\ 0\leq\alpha_{i}\leq Cs_{i}^{m},\;\; i=1,2,\ldots,n \end{aligned} \tag{4.9}$$

Es claro que la diferencias entre SVMs y FSVMs es la cota superior de los multiplicadores de Lagrange α_i en el problema dual.

Finalmente, la condición KT de FSVMs son

$$\begin{array}{c} \alpha_i \left[y_i \left(w^T x_i + b \right) - 1 + \xi_i \right] = 0, \quad i = 1, 2, \ldots, , n \\ \left(C s_i^m - \alpha_i \right) \cdot \xi_i = 0, \quad i = 1, 2, \ldots, n \end{array} \tag{4.10}$$

Cada grupo tiene un modelo difuso (3.15). Usando la idea del modelo Takagi-Sugeno [66], se pueden combinar los modelos locales en cada grupo dentro de un modelo global. Las reglas difusas tienen la siguiente forma

$$\mathrm{R}^j\text{: SI } x_{l_1^j} \leq x_1 \leq x_{l_2^j} \text{ , y } \cdots \; x_{l_1^j} \leq x_n \leq x_{l_2^j} \text{ ENTONCES } y\left(k\right) = f_j\left[X\left(k\right)\right]$$

donde $j = 1 \cdots p$, p es el número del grupo del agrupamiento en línea. Las funciones de membresía para x_i se definen como en la Figura 4.2. El modelo difuso final es

$$\widehat{y} = \left(\sum_{i=1}^{p} f_i\left[X\left(k\right)\right] \left[\prod_{j=1}^{n} \left(\mu_{A_i^j} \right) \right] \right) / \left(\sum_{i=1}^{p} \left[\prod_{j=1}^{n} \left(\mu_{A_i^j} \right) \right] \right) \tag{4.11}$$

Ahora se usa el mismo ejemplo propuesto en la sección anterior para explicar las funciones de membresía entrenadas para cada grupo. Después del entrenamiento de las funciones de membresía, las 7 funciones de membresía para x_1 se muestran en la Figura 4.3.Entonces los parámetros fijados de las funciones de membresía, se usan otros 300 datos para la prueba del modelo. El dato probado es $x_1\left(k\right) = 1 - \frac{2k}{T}$, $x_2\left(k\right) = -1 + \frac{2k}{T}$, $k = 1, 2, \cdots, T$. El resultado final del modelado después de los modelos locales ($p = 7$) se muestran en la Figura 4.4

4.2. Support Vector Machine Difuso

La función de costo normal de la regresión no lineal se define cómo

$$R_{emp}\left(\theta\right) = \frac{1}{N} \sum_{k=l_1^j}^{l_2^j} \left| y_k - \left(\mathbf{w}^T \varphi\left(\mathbf{x}_k\right) + b \right) \right|_\epsilon$$

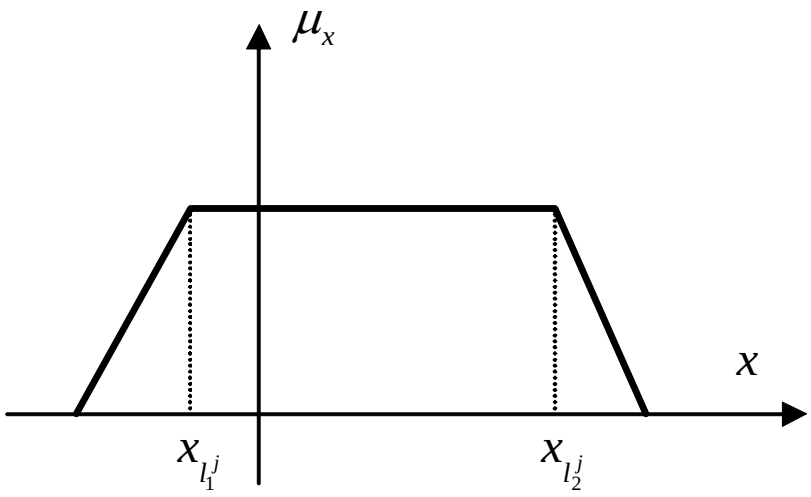

Figura 4.2: Funcion de membresia de x_i.

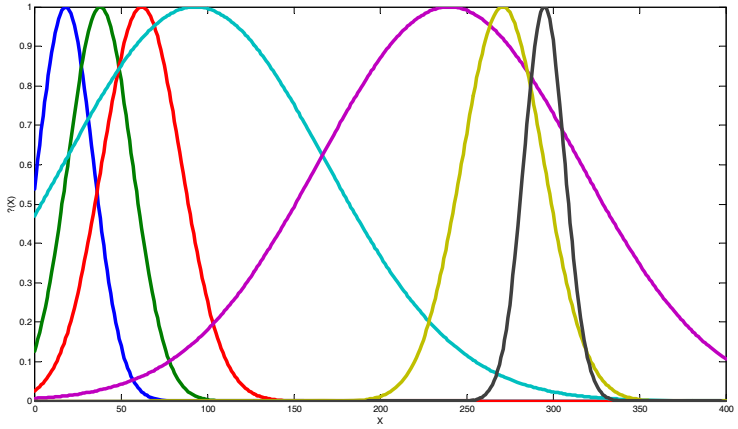

Figura 4.3: Funciones de Membresía para 7 reglas usando kernel difuso.

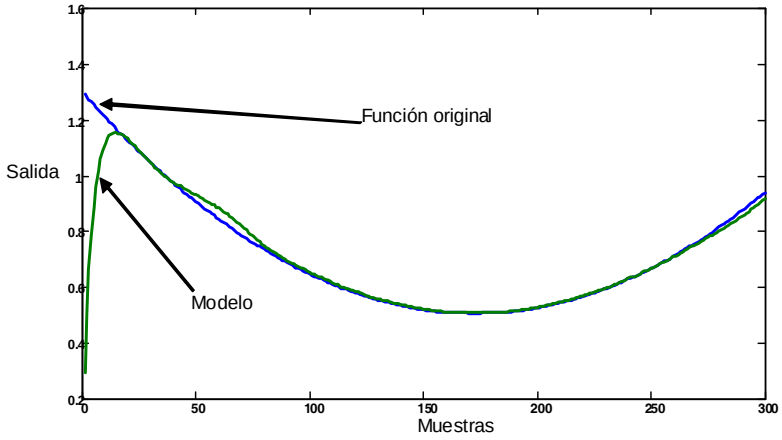

Figura 4.4: Resultado final.

donde ϵ insensitive es la función de pérdida de Vapnik, y se define como

$$|y_k - f(\mathbf{x})|_\epsilon = \begin{cases} 0 & |y_k - f(\mathbf{x})| \leq \epsilon \\ |y_k - f(\mathbf{x})| - \epsilon & \text{otra manera} \end{cases}$$

ϵ puede ser vista como la corrección de aproximación.

En este trabajo, se introduce un factor difuso s_i, que es el índice de rendimiento anterior [27]. Aquí $\sigma \leq s_i \leq 1$ y σ es un número positivo suficientemente pequeño, s_i denota el grado importante de muestras x_i para el aprendizaje óptimo del hiperplano. Se escoje μ_i como una función en forma de campana (4.12), ver Figura 4.5.

$$s_i = f(t) = \frac{1}{1 + \left|\frac{x-c}{a}\right|^{2b}} \tag{4.12}$$

El problema del hiperplano óptimo es visto como la solución para el siguiente problema:

$$\begin{aligned} &\text{mín } J_p = \tfrac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{k=1}^{n} s_k\xi_k \\ &\text{sujeto: } \left|y_k - \left(\mathbf{w}^T\varphi(\mathbf{x}_k) + b\right)\right| \leq \epsilon \end{aligned} \tag{4.13}$$

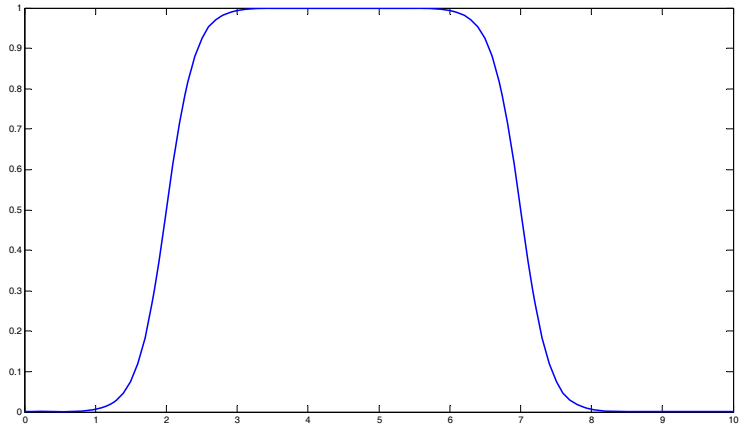

Figura 4.5: Función de membresía en forma de campana generalizada.

donde $\xi_k \geq 0$, $k = 1, 2, \ldots, n$, C es una constante, s_k más pequeña es, el efecto más pequeño que la muestra $\mathbf{x}_k$ al hiperplano óptimo. Si ε es demasiado pequeño, ciertos puntos estarán fuera de este tubo ε. Algunas variables flojas adicionales ξ_i son introducidas (es algo similar para el caso del clasificador con superposición). Para resolver el problema óptimo anterior, se puede formular la siguiente función de Lagrange:

$$L\left(w, b, \xi, a, \beta\right) = \frac{1}{2}\left\|w\right\|^2 + C\sum_{k=1}^{n} s_k\xi_k - \sum_{i=1}^{n}\alpha_k\left(y_k - \left(w\cdot\varphi\left(\mathbf{x}_k\right) + b\right) - 1 + \xi_k\right) - \sum_{k=1}^{n}\beta_k\xi_k \tag{4.14}$$

donde α_k, $\beta_k \geq 0$ son multiplicadores de Lagrange. El problema principal (4.13) puede ser cambiado dentro del problema dual por diferenciación con respecto a w, ξ y b,

$$\begin{aligned}
\tfrac{\partial L(w,b,\xi,\alpha,\beta)}{\partial w} &= w - \sum_{k=1}^{n}\alpha_k y_k\varphi\left(\mathbf{x}_k\right) = 0 \\
\tfrac{\partial L(w,b,\xi,\alpha,\beta)}{\partial b} &= -\sum_{i=1}^{n}\alpha_k y_k = 0 \\
\tfrac{\partial Q(w,b,\xi,\alpha,\beta)}{\partial \xi_k} &= s_k C - \alpha_k - \beta_k = 0
\end{aligned} \tag{4.15}$$

Aplicando estas condiciones dentro de la función de Lagrange (4.14), el problema (4.13)

puede ser transformado dentro del siguiente problema de programación cuadrática:

$$\begin{aligned} &\text{máx}\, W(\alpha) = \sum_{i=1}^{n} \alpha_i - \tfrac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j K(x_i, x_j) \\ &sujeto\ a \sum_{i=1}^{n} \alpha_i y_i = 0 \end{aligned} \tag{4.16}$$

donde $0 \leq \alpha_i \leq s_i C$, $i = 1, 2, \ldots, n$, $K(x_i, x_j) = \varphi(x_i) \cdot \varphi(x_j)$ es el kernel difuso.

Por el paquete de software estandar QP se obtiene la solución α_k^*. b se puede resolver por la siguiente condición de Kuhn-Tucker

$$\alpha_k^* \left\{ y_k \left[\mathbf{w}^T \varphi(\mathbf{x}_k) + b \right] - 1 + \xi_k \right\} = 0 \tag{4.17}$$

La función resultante es

$$f(\mathbf{x}) = \sum_{k=l_1^j}^{l_2^j} (\alpha_k - \alpha_k^*) K(\mathbf{x}_k, \mathbf{x}) + b$$

puesto que algunos $\alpha_k^* = 0$, el vector de solución es escaso, la suma sería tomada solo bajo las α_k no cero (support vector), así el resultado final es

$$f(\mathbf{x}) = \sum_{k=1}^{sv_j} (\alpha_k - \alpha_k^*) K(\mathbf{x}_k, \mathbf{x}) + b \tag{4.18}$$

donde sv_j es el número de support vectors, $sv_j \ll \left(l_2^j - l_1^j + 1\right)$. Se definen las posiciones de los support vectors como $[x_i^*, y_i^*]$, $i = 1 \cdots sv_j$.

El punto $\mathbf{x}_k$ con el correspondiente $0 \leq \alpha_k \leq s_k C$ es llamado support vector. Para SVM difuso, existen dos tipos de support vectors: uno corresponde a $0 \leq \alpha_k \leq s_k C$ que esta situado sobre el márgen del hiperplano, y el otro corresponde a $\alpha_i = s_i C$ que son mal clasificados. Así los puntos con el mismo valor de α_k en SVM difuso podrían indicar un diferente tipo de support vector en SVM difuso debido al factor s_i.

De (4.18) se conocen los support vectors los cuales son suficientes para representar la no linealidad en cada groupo. Se usan estos support vectors para construir las reglas difusas.

Para el grupo j, se extrae el producto de reglas difusas en la siguiente forma (modelo difuso Mamdani [52])

$$\mathrm{R}^j\text{: SI } x_1^*(k) \text{ es } A_1^j \text{ y } x_2^*(k) \text{ es } A_2^j \text{ y } \cdots x_n^*(k) \text{ es } A_n^j \text{ ENTONCES } y^*(k) \text{ is } B^j$$

Aquí $j = 1 \cdots sv_j$, $A_1^j, \cdots A_n^j$ y B^j son conjuntos estandares difusos, $[x_i^*, y_i^*]$ es el punto del support vector. Se usa sv_j reglas SI-ENTONCES difusas para la ejecución del mapeo desde un vector de entrada $\mathbf{x} = [x_1 \cdots x_n] \in \Re^n$ hacia una salida $y(k)$. La función de membresía Gausiana es

$$s_i(x_j) = \exp\left(-\frac{(x_j - c_{ji})^2}{\sigma_{ji}^2}\right) \tag{4.19}$$

Sean $\mathbf{x}^*(k)$ y $y^*(k)$ los centros de las funciones Gausianas, *i.e.* $c_{ji} = x_j^*$.

De [72] se conoce, usando el producto inferencia, los centros promedios y el singleton difusificador, la salida del sistema difuso en el grupo j se puede expresar como

$$\widehat{y} = \left(\sum_{j=1}^{sv_j} w_j \left[\prod_{i=1}^{n} \mu_{A_i^j}\right]\right) / \left(\sum_{j=1}^{sv_j} \left[\prod_{i=1}^{n} \mu_{A_i^j}\right]\right) = f_j[\mathbf{x}(k)] \tag{4.20}$$

donde $\mu_{A_i^j}$ son las funciones de membresía de los conjuntos difusos A_i^j, w_j es el punto al cual $\mu_{B_j} = 1$. Si se define

$$\phi_i = \prod_{j=1}^{n} \mu_{A_{ji}} / \sum_{i=1}^{sv_j} \prod_{j=1}^{n} \mu_{A_{ji}}$$

(4.20) se puede expresar en la forma matricial

$$\widehat{y}(k) = \mathbf{W}(k)\, \Phi[\mathbf{x}(k)] \tag{4.21}$$

donde el parámetro $\mathbf{W}(k) = \left[W_1 \cdots W_{sv_j}\right] \in R^{sv_j}$, el vector de datos $\Phi[\mathbf{x}(k)] = \left[\phi_1 \cdots \phi_{sv_j}\right]^T \in R^{sv_j \times 1}$.

Para el modelo difuso Takagi-Sugeno [66],

$$\mathrm{R}^j\text{: SI } x_1 \text{ es } A_{1i} \text{ y } x_2 \text{ es } A_{2i} \text{ y } \cdots x_n \text{ es } A_{ni} \text{ ENTONCES } \widehat{y} = p_0^j + p_{1x1}^j + \cdots p_{nxn}^j \tag{4.22}$$

donde $j = 1 \cdots sv$. La salida del sistema lógico difuso se puede expresar como

$$\widehat{y} = \sum_{i=1}^{sv_j} \left(p_0^i + p_{1x1}^i + \cdots p_{nxn}^i \right) \phi_i \tag{4.23}$$

(4.23) se expresa también en la forma del tipo Mamdani (4.21),

$$\widehat{y}\,(k) = \mathbf{W}\,(k)\,\Phi\left[\mathbf{x}\,(k)\right] \tag{4.24}$$

donde

$$\begin{aligned} \mathbf{W}\,(k) &= \left[p_{10}^1 \cdots p_{10}^{sv_j} \cdots p_{1n}^1 \cdots p_{1n}^{sv_j} \right] \\ \Phi\left[\mathbf{x}\,(k)\right] &= \left[\begin{array}{cccc} \phi_1 \cdots \phi_{sv_j} & x_1\phi_1 \cdots x_1\phi_{sv_j} & \cdots & x_n\phi_1 \cdots x_n\phi_{sv_j} \end{array} \right]^T \end{aligned}$$

4.3. Formulación de FSVMs

Se asume que el conjunto difuso entrenado se conoce, el siguiente paso es formular el FSVM. Se comienza con la construcción de una función de costo, el FSVM tambien quiere maximisar el margen de separación y minimizar el error de clasificación tal que una buena capacidad de generalizacion puede ser ejecutada.

La construcción del problema de optimización del FSVM se formula como

$$\begin{array}{c} \text{mín} \quad \Phi\,(w, \xi, s) = \frac{1}{2} w^T w + C \sum_{i=1}^{n} s_i^m \xi_i \\ sujeto\ a \quad y_i \left(w^T x_i + b \right) \geq 1 - \xi_i, \quad i = 1, 2, \ldots, n \\ \xi_i \geq 0, i = 1, 2, \ldots, n \end{array} \tag{4.25}$$

donde m influye defusifisidad de la falta del término defusificado en la función de costo. Ahora la función de Lagrange es

$$\begin{array}{c} Q\,(w, b, \xi, \alpha, \beta, s) = \frac{1}{2} w^T w + C \sum_{i=1}^{n} s_i^m \xi_i \\ - \sum_{i=1}^{n} \alpha_i \left[y_i \left(w^T x_i + b \right) - 1 + \xi_i \right] - \sum_{i=1}^{n} \beta \xi_i \end{array} \tag{4.26}$$

donde α_i y β_i son multiplicadores de Lagrange no negativos. Diferenciando Q respecto a w, β y ξ_i, e igualando los resultados a cero se obtienen las tres condiciones de optimalidad siguientes

$$\begin{aligned} \frac{\partial Q(w,b,\xi,\alpha,\beta,s)}{\partial w} &= w - \sum_{i=1}^{n} \alpha_i y_i x_i = 0 \\ \frac{\partial Q(w,b,\xi,\alpha,\beta,s)}{\partial b} &= -\sum_{i=1}^{n} \alpha_i y_i = 0 \\ \frac{\partial Q(w,b,\xi,\alpha,\beta,s)}{\partial \xi} &= C s_i^m - \alpha_i - \beta_i = 0 \end{aligned} \tag{4.27}$$

Substituyendo la ecuación (4.27) dentro de la ecuación (4.26), el Lagrange es una función de α solamente. El problema dual se vuelve

$$\begin{aligned} \text{mín} \quad Q(\alpha) = \alpha_i - \tfrac{1}{2} w^T w + C \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j \\ \sum_{i=1}^{n} \alpha_i y_i = 0 \\ 0 \le \alpha_i \le C s_i^m, \ i = 1, 2, \ldots, n \end{aligned} \tag{4.28}$$

Es claro que la diferencias entre SVMs y FSVMs es la cota superior de los multiplicadores de Lagrange α_i en el problema dual.

Finalmente, la condición KT de FSVMs son

$$\begin{aligned} \alpha_i \left[y_i \left(w^T x_i + b \right) - 1 + \xi_i \right] = 0, \quad i = 1, 2, \ldots, , n \\ \left(C s_i^m - \alpha_i \right) \cdot \xi_i = 0, \quad i = 1, 2, \ldots, n \end{aligned} \tag{4.29}$$

4.4. Simulaciones

Se sabe que cada punto puede causar un sobreajuste no adaptado dentro de lo SVMs. De aquí, el concepto central del proposito del FSVM se usa para asignar a cada punto, un valor de membresía de acuerdo a sus importancia relativa en las clases. Cada punto x_i tiene un valor de membresía asignado μ_i, el conjunto entrenado parece un conjunto difuso entrenado S_f y es dada por

$$S_f = \{x_i, y_i, \mu_i\}_{i=1}^{n} \tag{4.30}$$

Para una clase positiva ($y_i = +1$) el conjunto de los valores de membresía denotados como μ_i^+, y son denotados como μ_i^- para las clases negativas ($y_i = -1$). Estos son asignados independientemente.

Se usa el siguiente ejemplo para explicar el agrupamiento en línea. Se considera la función no lineal la cual tiene la forma [32], [33]

$$f(x_1, x_2) = 0{,}52 + 0{,}1x_1 + 0{,}28x_2 - 0{,}6x_1x_2$$

Los datos de entrenamiento son selecionados como $x_1(k) = -1 + \frac{2k}{T}$, $x_2(k) = 1 - \frac{2k}{T}$, $k = 1, 2, \cdots T$. Se escoge $\alpha = 0{,}4$, $\beta = 0{,}6$. Se obeserva que los cambios máximos en la entrada y la salida son aproximadamente 1 y 2, así $\alpha \left\| x_{\text{máx}} - x_{\text{mín}} \right\| + \left\| y_{\text{máx}} - y_{\text{mín}} \right\| \beta = 16$, L se podría escoger como $L < 1{,}2$, en esta aplicación se selecciona $L = 0{,}5$. El agrupamiento entrada-salida que se muestra en la Figura.4.6. Se representa el centro como "*" de cada grupo, y "+" es la frontera entre los grupos. Se puede ver que hay 7 grupos. Por ejemplo, los intervalos de tiempo del 4-ésimo grupo son $l_1^2 = 94$, $l_1^2 = 238$.

Se usa el siguiente ejemplo propuesto anteriormente para explicar la extracción de las reglas difusas. El resultado después de usar FSVM son mostrados en la siguiente tabla

Grupo	# de sv	Posición	# de datos
1	1	1	21
2	1	1	25
3	1	1	32
4	1	1	148
5	1	32	32
6	1	25	25
7	1	6	6

Tabla 1: FSVM para función de estimación.

Ahora usando el mismo procedimiento para modelar un sistema no lineal propuesto en [57] [34][73], y usado en [32],[33].

$$y(k) = \frac{y(k-1)\, y(k-2)\, [y(k-1) + 2{,}5]}{1 + y(k-1)^2 + y(k-2)^2} + u(k-1) \tag{4.31}$$

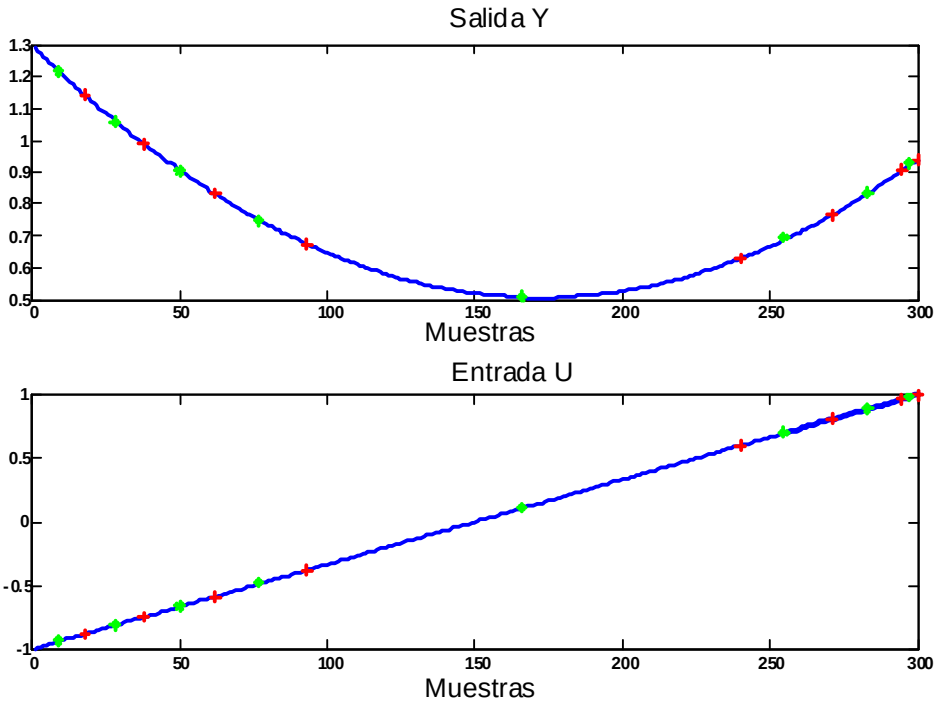

Figura 4.6: Agrupamiento en línea.

La señal de entrada se selecciona como un número aleatorio en el intervalo $[0, 1]$.

$$X(k) = [y(k-1), y(k-2), u(k-1)]^T = [x_1(k), x_2(k), x_3(k)]^T$$

Los resultados del agrupamiento en linea se muestra en la Fig.4.7.

El resultado después de FSVM se pueden ver en la siguiente Tabla

Grupo	# de sv	Posición	# de datos
1	4	$1, 4, 10, 13$	13
2	11	$5, 6, 7, 14, 15,$ $16, 24, 25, 26, 27, 32$	32
3	9	$5, 6, 7, 15, 16,$ $22, 23, 24, 28$	28
4	3	$5, 11, 16$	16

Tabla 2: SVM para función de estimación.

Por ejemplo, en el Grupo 4 existen 16 datos y 3 support vector, estos son $x(5)$, $x(11)$ y $x(16)$. Asi las reglas difusas son

Si x_1 es 0,89 y x_2 es 0,12 y x_3 is 0,3 entonces y es 345,75
Si x_1 es 0,95 y x_2 es 0,2 y x_3 es 0,2 entonces y es 379,17
Si x_1 es 1 y x_2 es 0,47 y x_3 es 0,7 entonces y es 387,2

Hay un total de 27 support vectors. Despues del entrenamiento, las funciones de membresia de x_1 en las 27 reglas difusas se muestra en la Fig.4.8.

La entrada de prueba es $u(k) = \sin(k/30) + \cos(k/20) + 2\sin(k/10)$. El resultado final del modelado difuso $\widehat{f}(x) = \sum_{j=1}^{5} \overline{z}^j \left(\prod_{i=1}^{3} \mu_{A_i^j}(x_i) \right) / \sum_{j=1}^{5} \left(\prod_{i=1}^{3} \mu_{A_i^j}(x_i) \right)$ se muestra en la Fig 4.9.

En esta sección se discute un problema típico el cual también se discutió en [34][57][73]. La planta a identificar es 4.31

La señal de entrada entrenada es seleccionada como números aleatoreos en el intervalo $[0, 1]$.

$$X(k) = [y(k-1), y(k-2), u(k-1)]^T = [x_1(k), x_2(k), x_3(k)]^T$$

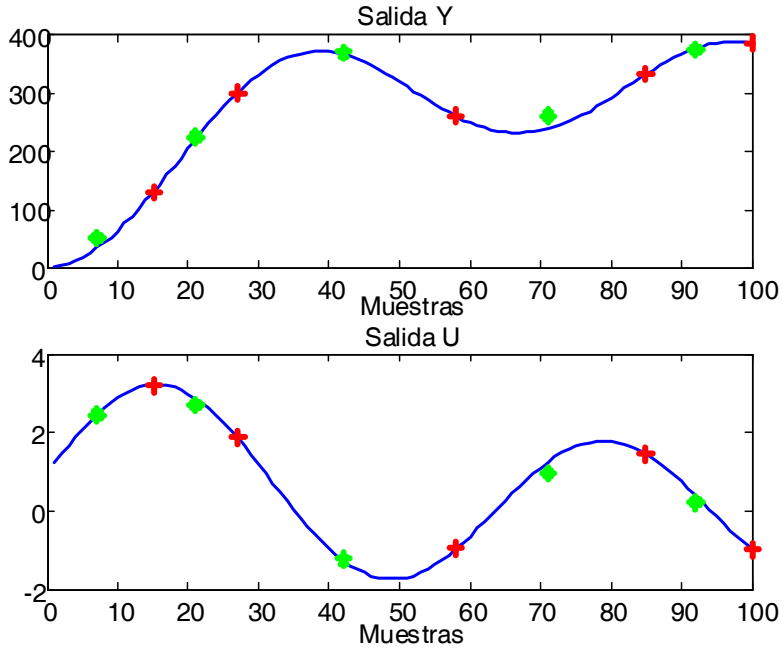

Figura 4.7: Agrupamiento en línea.

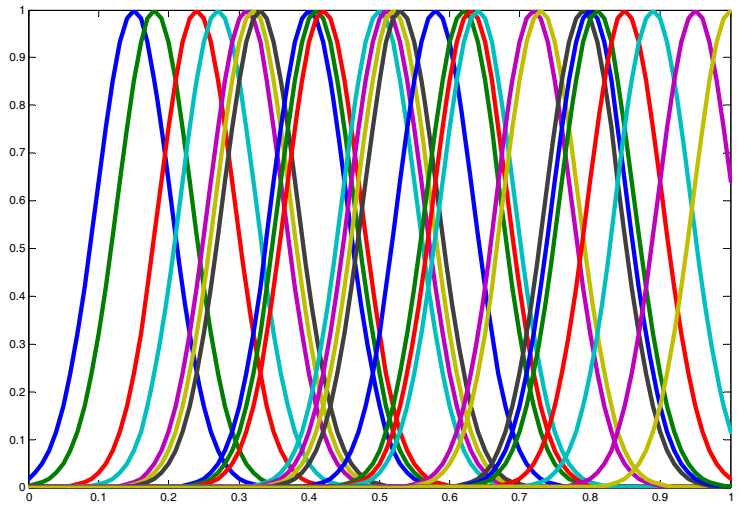

Figura 4.8: Las funciones de membresia de x_1 en las 27 reglas difusas.

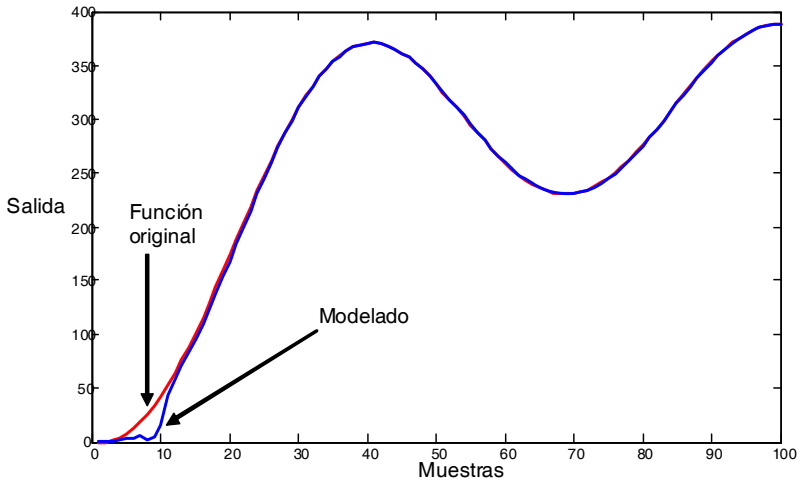

Figura 4.9: Resultado final del modelado difuso.

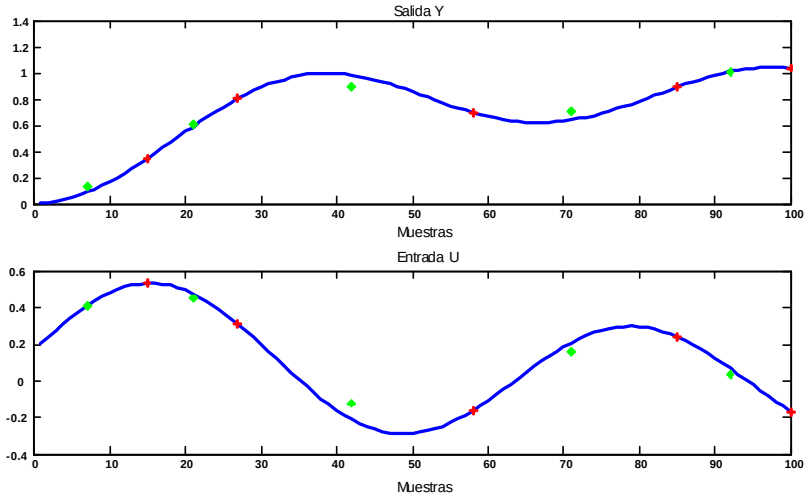

Figura 4.10: Agrupamiento en línea.

1. Se usa el siguiente modelo difuso Mamdani, para la j-ésima regla

$$\mathrm{R}^j\text{: SI } x_1(k) \text{ es } A_1^j \text{ y } x_2(k) \text{ es } A_2^j \text{ y } x_3(k) \text{ es } A_3^j \text{ ENTONCES } y(k) \text{ es } B^j \qquad (4.32)$$

   Primeramente, se usa agrupamiento en línea. Se selecciona $\alpha = 0{,}4$ conocido a priori, se conocen los cambios máximos en la entrada y la salida son alrededor de 3 y 1, $\alpha \left\|\mathbf{x}_{\text{máx}} - \mathbf{x}_{\text{mín}}\right\| + (1-\alpha)\left\|y_{\text{máx}} - y_{\text{mín}}\right\| = 1{,}8$. Así L podría ser escogida tal que $L < 1{,}8$, en esta aplicación se selecciona $L = 1{,}5$. La Figura 4.10 muestra el resultado del agrupamiento en línea para datos de dos meses. Aquí "*" representa el centro de cada grupo, "+" es el límite entre los grupos. Existen 5 grupos en $0 < t \leq 100$.

2. En segundo lugar, se usa *support vector machine* (4.13) para obtener los support vectors como en (4.18). El número de support vector para el grupo 3 son $7, 5, 8, 6$ y 8. Así se tienen 33 reglas como en (4.32). Los centros de las funciones de membresía Gausianas de estas ruglas son las posiciones de los support vectors.

3. Ahora se usan los datos en cada grupo para actualizar las funciones de membresía. Por ejemplo, los intervalos de tiempo del segundo grupo son $l_1^2 = 14$, $l_2^2 = 28$, los datos desde

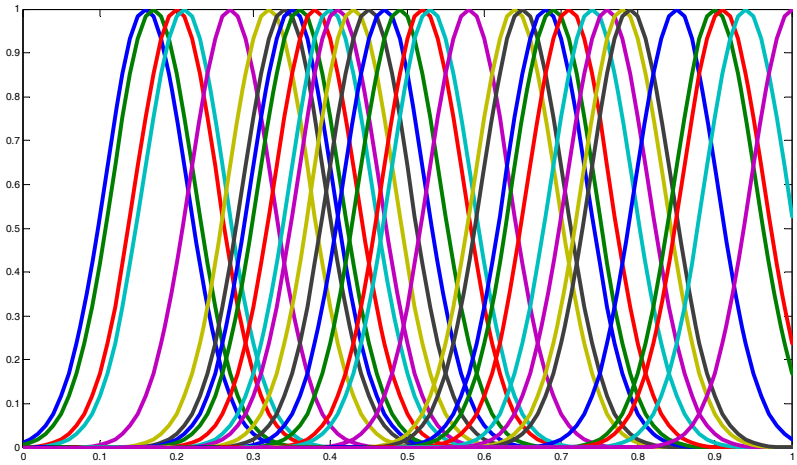

Figura 4.11: 33 funciones de membresía de x_1.

14 hasta 18 son usados para entrenar las 5 reglas difusas. Después el entrenamiento, las funciones de membresía de A_1^j, $j = 1 \cdots 33$ son mostradas en la Figura 4.11.

4. Entonces se combinan los 5 modelos locales difusos (33 reglas) dentro de un modelo global en la forma Takagi-Sugeno

$$\widehat{y} = \left(\sum_{j=1}^{5} f_j \left[\mathbf{x}\left(k\right) \right] \left[\prod_{i=1}^{33} \mu_{A_i^j} \right] \right) / \left(\sum_{j=1}^{5} \left[\prod_{i=1}^{33} \mu_{A_i^j} \right] \right)$$

Después de que la fase de entrenamiento es terminado, se usa la entrada de prueba como

$$u(k) = \sin(k/30) + \cos(k/20) + 2\sin(k/10)$$

Los resultados de modelado final son mostrados en la Figura 4.12.

Ahora se cambia el parámetro de diseño α, $\alpha = 0{,}1$. Se conocen los cambios máximos en la entrada y la salida son alrededor de 3 y 1, $\alpha \left\| \mathbf{x}_{\text{máx}} - \mathbf{x}_{\text{mín}} \right\| + (1-\alpha) \left\| y_{\text{máx}} - y_{\text{mín}} \right\| = 1{,}2$. Así L podría ser escogida tal que $L < 1{,}2$, en esta aplicación se selecciona $L = 1$. Existen 6

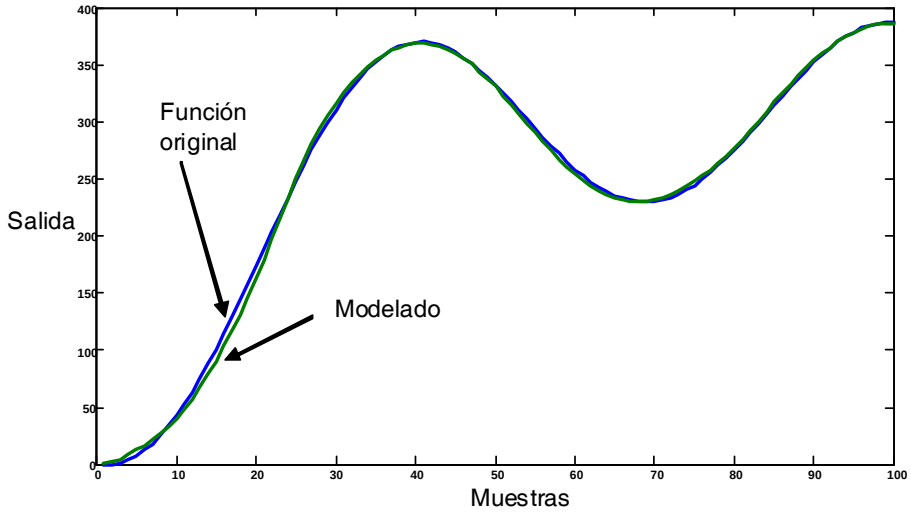

Figura 4.12: Modelado difuso.

grupos y 40 reglas como en (4.32). Los resultados del modelado son peores que con $\alpha = 0{,}4$, porque no existen suficientes datos en cada grupo para actualizar las funciones de membresía y las semánticas de las reglas difusas se perderían.

Finalmente, se comparan el enfoque antedicho con los siguientes métodos de modelado difuso.

1. El enfoque del modelado difuso adaptable [34][72] podría ser el método más popular. También se usan 33 reglas difusas en la forma de (4.32). Las funciones de membresía Gausianas son seleccionadas aleatoreamente como al principio, entonces se usan los mismos datos entrenados para actualizar esas funciones de membresía. Los datos de prueba son también los mismos antes mencionados.

2. El modelado difuso vía agrupamiento en línea pueden reducir el número de reglas, se considera el agrupamiento sin considerar el intervalo de tiempo y cada grupo entrenado usa todos los datos como en [36], [68] y [1]. Si se usan los umbrales de la salida y la

entrada como 1,5. 30 reglas difusas son construidas, entonces todos los datos son usados para entrenarlos.

3. Otro método popular para el modelado difuso es la lógica difusa con agrupamiento de datos [34][12]. La entrada es particionada. Con los umbrales 1,5, se tienen 8 grupos en el espacio de entrada. Así se construyen 8 reglas difusas. Las partes restantes son las mismas del método 1.

Capítulo 5

Modelado via multi redes neuronales

Una red tipo RBF (en su forma más básica) tiene una simple capa oculta, la capa oculta es no lineal, mientras que la capa de salida es lineal. Los argumentos de la función de activación de cada unidad oculta en una red RBF calcula la norma Euclidiana (distancia) entre el vector de entrada y el centro de la unidad. En estas redes las no linealidades localizadas decaen exponencialmente (ejemplo funciones Gausianas) construyen una aproximación local del mapeo entrada-salida no lineal, con el resultado que estas redes son capaces de aprender rápido y reduce la sensitividad a el orden de presentación de datos entrenados. La característica lineal de la capa de salida de la red RBF significa que esta red esta más relacionada al percetron de Rosenblatt que al percetron multicapa. Sin embargo, la red RBF es capas de implementar transformaciones no lineales arbitrariamente del espacio de salida.

Las redes RBF pueden ser usadas como funciones de aproximación universales. Esto consiste de una red con una simple capa oculta y una estructura similar a las redes propagación hacia atrás (back propagation). Cada unidad de capa oculta tiene un centroide c_i y un factor suave σ_i. Estas neuronas calculan la distancia entre la entrada x_i y el centroide c_i más bien el producto vectorial de los pesos y entradas. Las salidas son funciones no linelales, simétricas radial de la distancia. Por lo tanto, la salida es mucho más fuerte cuando x_i esta más cerca del valor c_i, Las redes RBF utilizan mapeos de funciones reales f_m las cuales tiene la forma

general

$$f_m(x) = \sum_{i=1}^{M} w_i K\left[(x_i - c_i)/\sigma_i\right] \tag{5.1}$$

La función K es una función kernel simétrica radial calculada por las unidades kernel M. La función exponencial Gausiana es comúnmente usadaen redes RBF

$$f_m(x) = \beta \exp\left(\sum_{i=1}^{M} \left[(x_i - c_i)/\sigma_i\right]^2\right) \tag{5.2}$$

El centroide c_i y las constantes β y σ_i tienen que ser escogidas de acuerdo al conjunto de datos entrenados. En general las funciones de activación Gausianas son superiores a las funciones sigmoide en estimación a las clases de funciones extensas.[23]

5.1. Equivalencia estructural de la red RBF y SVM

Una red RBF produce una transformación de vectores m-dimensional dentro de vectores p-dimensional por una combinacional lineal de una función base no lineal. Su estrutura se muestra en la Figura 15.1. Cada función base se expresa como sigue

$$\varphi_k(x_i) = \exp\left(-\frac{\|x_i - c_k\|^2}{2\sigma_k^2}\right) \tag{5.3}$$

donde $\|\cdot\|$ es la norma Euclidiana, x_i $(i = 1, 2, \ldots, n)$ es una entrada, c_k $(k = 1, 2, \ldots, n)$ es el ancho. La salida de la red RBF se modifica con la fórmula 5.4.

$$f(x_i) = \sum_{k=1}^{n} w_k \varphi_k(x_i) = \sum_{k=1}^{n} w_k \exp\left(-\frac{\|x_i - c_k\|^2}{2\sigma_k^2}\right) \tag{5.4}$$

donde n es el número de nodos en una capa oculta segura y w_k $(i = 1, 2, \ldots, n)$ es un peso entre la capa oculta y la capa de salida.

La función radial es usada como la función kernel. La salida de SVM se modifica con la fórmula 5.5 y la estructura se muestra en la figura 5.2

$$f(x, x_i) = \sum_{k=1}^{g} w_i k(x, x_i) + b = \sum_{k=1}^{g} w_i \exp\left(-\frac{\|x - x_i\|^2}{2\sigma_k^2}\right) + b \tag{5.5}$$

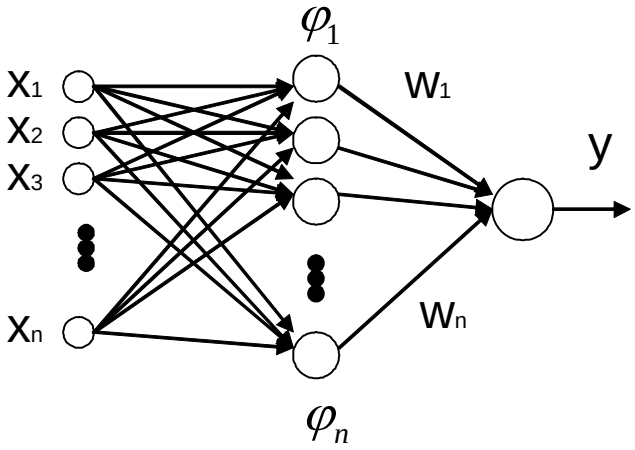

Figura 5.1: Esquema red neuronal tipo RBF.

donde x_i $(i = 1, 2, \ldots, n)$ es un support vector y g es el número de support vector. w_i $(i = 1, 2, \ldots, n)$ es el peso entre una capa oculta y una capa de salida.

Aunque los principios básicos de la red RBF y SVM son completamente diferentes, de acuerdo a la fórmula 5.4 y la fórmula 5.5 y las figuras 5.1 y 5.2, la equivalencia estructural de la red RBF y SVM pueden ser claramente vistas. Los parámetros de las redes (por ejemplo, el centro y el ancho de la función radial y el peso) son conformes.[18]

5.2. Modelado neuronal tipo RBF

Para este modelado se seguirán los siguientes pasos o procedimientos para obtener el modelado de una planta. Primero se usa el método de agrupamiento en línea la cual divide los datos de entrada/salida dentro de varios grupos en el mismo intervalo de tiempo. Después se aplica el *support vector machines* el cual produce support vector para cada grupo. Una vez obtenidos estos support vector, se construyen cada red neuronal tipo RBF para cada grupo. Finalmente se obtienen los parámetros de modelado. El esquema del modelado se muestra en la Fig. 5.3.

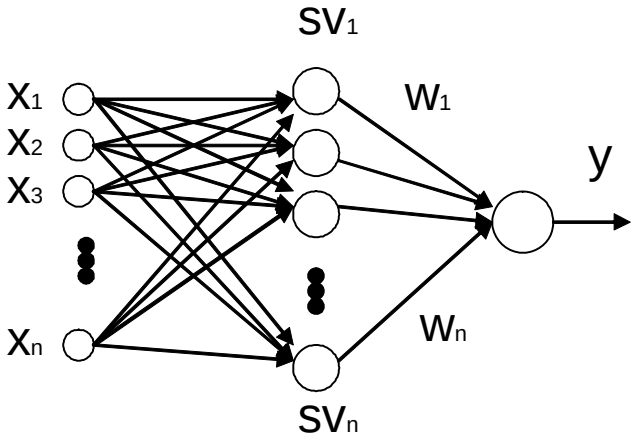

Figura 5.2: Esquema de modelado de una red RBF usando SVM como nodos de capa oculta.

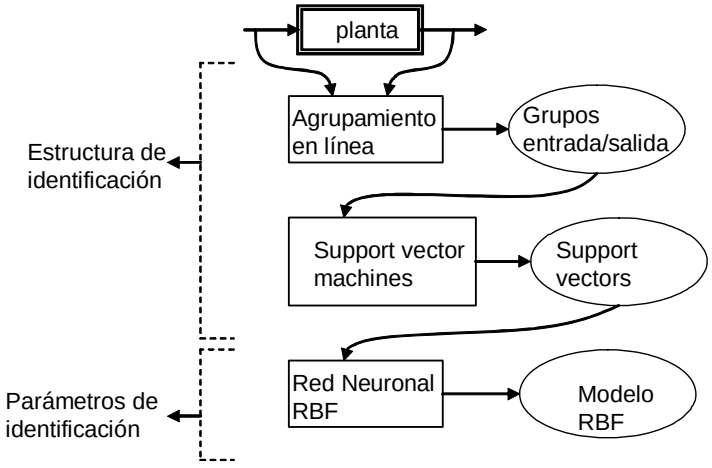

Figura 5.3: Esquema del modelado.

Como se había visto en capítulos anteriores, se quiere modelar un sistema no lineal discreto en el tiempo ahora con redes neuronales tipo RBF

$$x(k+1) = f[x(k), u(k)], \quad y(k) = h[x(k)] \tag{5.6}$$

donde $u(k) \in \mathbb{R}^m$ es el vector de entradas, $x(k) \in \mathbb{R}^n$ el vector de estado, y $y(k) \in \mathbb{R}^m$ es el vector de salida. f y h son funciones suaves no lineales.

Las redes RBF (función base-radial) corresponden a la clasificación importante ejecución de sistemas de aproximación local [39],[19]. Su principal ventaja es una simplificación grande del algoritmo de aprendizaje siguiendo la asociación de los parámetros de la red con la distribución de los datos en un espacio multidemensional. Ellos realizan la transformación no lineal de los datos del espacio de entrada hacia un espacio (característico) de mayor dimensión, haciendo el problema de reconocimiento más probable a ser linealmente separable [14].

El problema de aprendizaje de las redes función base radial es transformar a la elección del número de unidades ocultas y los parámetros de la función Gausiana (sus centros y anchos) además de la adaptación de los pesos sinápticos de las neuronas de salida lineales. Las red RBF es la implementación práctica del teorema de Cover, mostarando que el mapeo no lineal de los datos del espacio de entrada N-dimensional a el espacio oculto K-dimensional de K>N y la superposición de las señales de las neuronas ocultas con pesos propios, permita obtener la aproximación de los datos multidimensionales con exactitud arbitaria[19]. La salida de la red RBF realiza una simple operación suma de los pesos

$$y(x) = w_0 + \sum_{j=1}^{K} w_j \varphi_j(x) \tag{5.7}$$

donde las funciones de activación no lineales son asumidas como funciones Gausianas

$$\varphi_j(x) = \varphi_j(x, c_j) = \exp\left(-\frac{(x-c_j)^T (x-c_j)}{\sigma_j^2}\right) \tag{5.8}$$

con c_j el vector de centros y σ_j el parámetro ancho. Los centros usualmente son ajustados del vector de aprendizaje x_i, donde $x_i \subset \mathbb{R}^N$ es el vector de entradas del par de aprendizaje

(x_i, d_i) y d_i es el valor destinado asociado con x_i para $i = 1, 2, \ldots, p$. El parámetro ancho σ de la función Gausiana no es muy crucial y puede ser usado como un conjunto definido por el usuario como valores constantes[60].

Dentro del enfoque SVM el problema de aprendizaje de la red RBF, usado como un clasificador, el objetivo es ajustar los parámetros de la red mazimizando el margen de separación entre dos clases. Matemáticamente esta puede ser definida como el siguiente problema de minimización [71],[4]

$$\text{mín}\, \varphi_j(w, \xi) = \frac{1}{2} w^T w + C \sum_{i=1}^{p} \xi_i \tag{5.9}$$

a la funcional de restricción para $i = 1, 2, \ldots, p$

$$d_i w^T \varphi(x) \geq 1 - \xi_i \tag{5.10}$$

$$\xi_i \geq 0 \tag{5.11}$$

donde ξ_i es una variable débil no negativa y C la constante de regularización. La solución del problema es transformar la maximización de la función de los multiplicadores de Lagrange $Q(\alpha)$[71], definidas como la tarea de programación cuadrática

$$\text{máx}\, Q(\alpha) = \sum_{i=1}^{p} \alpha_i - \frac{1}{2} \sum_{i=1}^{p} \sum_{j=1}^{p} \alpha_i \alpha_j d_i d_j K(x_i, x_j) \tag{5.12}$$

con la restricción lineal

$$\sum_{i=1}^{p} \alpha_i d_i = 0, \qquad 0 \leq \alpha_i \leq C \tag{5.13}$$

La función de activación $\varphi(x)$ ha sido substituida aquí por el llamado función kernel $K(x_i, x_j) = \varphi^T(x_i)\varphi(x_j)$, la cual es usualmente la función Gausiana de la forma 5.8 para redes RBF. Resolviendo el último problema de optimización se obtiene el conjunto de multiplicadores de Lagrange α_{oi} los cuales permiten determinar la señal de salida $y(x)$ de la red en la forma

$$y(x) = \sum_{i=1}^{N_{SV}} \alpha_{oi} d_i K(x_i, x) + w_0 \tag{5.14}$$

con N_{SV} el número de support vector, que es el vector x_i asociado con los multiplicadores no ceros de Lagrange. Notar que los datos corresponden a los multiplicadores de los valores iguales a cero que no tienen influencia en la solución final. Esto es una diferencia importante a los métodos clásicos de aprendizaje donde todos los puntos tienen algun impacto sobre la solución final.

En el caso de la tarea de regresión el problema de aprendizaje se define a travéz de la llamada función ε-insensitive $L_\varepsilon\left(d, y\left(x\right)\right)$ [71],[4]. El problema de aprendizaje se define ahora como

$$\text{mín}\,\phi\left(w, \xi, \xi^{'}\right) = C\left[\sum_{i=1}^{p}\left(\xi_i + \xi_i^{'}\right)\right] + \frac{1}{2}w^T w \tag{5.15}$$

con las restricciones

$$\begin{aligned} &d_i - w^T\varphi\left(x_i\right) \leq \varepsilon + \xi_i \\ &w^T\varphi\left(x_i\right) - d_i\varepsilon + \xi_i^{'} \\ &\xi_i \geq 0, \xi_i^{'} \geq 0 \end{aligned} \tag{5.16}$$

donde ξ_i y $\xi_i^{'}$ son las variables débiles no negativas. La tarea primordial es transformar a lo que se llama problema dual con respecto a los multiplicadores de Lagrange α_i y $\alpha_i^{'}$[71],[4]

$$\begin{aligned} \text{máx}\,Q\left(\alpha\right) = &\sum_{i=1}^{p} d_i\left(\alpha_i - \alpha_i^{'}\right) - \epsilon\sum_{i=1}^{p}\left(\alpha_i + \alpha_i^{'}\right) + \\ &- \tfrac{1}{2}\sum_{i=1}^{p}\sum_{j=1}^{p}\left(\alpha_i - \alpha_i^{'}\right)\left(\alpha_j - \alpha_j^{'}\right)K\left(x_i, x_j\right) \end{aligned} \tag{5.17}$$

con la restricción lineal

$$\begin{aligned} &\sum_{i=1}^{p}\left(\alpha_i - \alpha_i^{'}\right) = 0 \\ &0 \leq \alpha_i \leq C,\ 0 \leq \alpha_i^{'} \leq C \end{aligned} \tag{5.18}$$

Resolviendo está tarea de programación cuadrática lleva a la siguiente realación para la señal de entrada $y\left(x\right)$

$$y\left(x\right) = \sum_{i=1}^{N_{SV}}\left(\alpha_{oi} - \alpha_{oi}^{'}\right)K\left(x_i, x\right) + w_0 \tag{5.19}$$

Similarmente al mode de clasificación la señal de salida de la red de regresión es dependiente solamente en los multiplicadores no ceros de Lagrange y de la función kernel Gausiana, substituyendo aquí la función de activación $\varphi(x)$[60].

El método básico de identificación de sistemas no lineales es construir un modelo matemático $\overline{y} = f(u, y)$ usando datos conocidos. Las variables de salida son estimadas y predichas optimalmente por el modelo. Donde u es entrada conocida, y es salida conocida , $\overline{y}$ es estimación óptima de la variable de salida. El método tradicional de identificación de sistemas es utilizar métodos matemáticos para obtener el modelo del sistema f en términos de datos de entrada-salida, la cual regla de entrenamiento es $e = \|\overline{y} - y\| < \varepsilon$, ε es programa de presición de identificación. Porque la regla de entrenamiento tradicional solo reduce el error de entrenamiento y no la estructura del modelo de control, la desventaja de la regla es que la habilidad de generalización no satisface aunque el modelo tenga buena presición. El algoritmo SVM adopta un riesgo estructural del principio de minimización, así la presición del entrenamiento y la complejidad del modelo alcanzan cierto balance, la complejidad se controla mejor bajo condiciones de presición garantizada, y asegura que el modelo tiene buena capacidad de generalización en el algoritmo.

El modelo del sistema basado en SVM pertenece a un modelo de caja negra, el cual esta basado solo en mediciones de entrada-salida de procesos industriales. En este procedimiento de modelado, la relación entre entradas y salidas de la planta pueden ser enfatizadas mientras que la estructura sofisticada interna es ignorada, y el mapeo ralacionado entre entrada y salida es realizada por SVM. El proceso básico y modelado de sistemas de identificación basados en SVM se muestran el la Figura 5.4.

Finalmente en el modelo del sistema no lineal, la función no lineal relacionada entre la entrada y la salida es realizada por SVM.

Existen dos tipos de parámetros durante el modelado SVM. Estos son el parámetro de regularización (como c) y el parámetro kernel (como d). El problema de selección de los valores de esos parámetros así como maximizar la prueba de error es llamado el modelo de selección del problema[53].

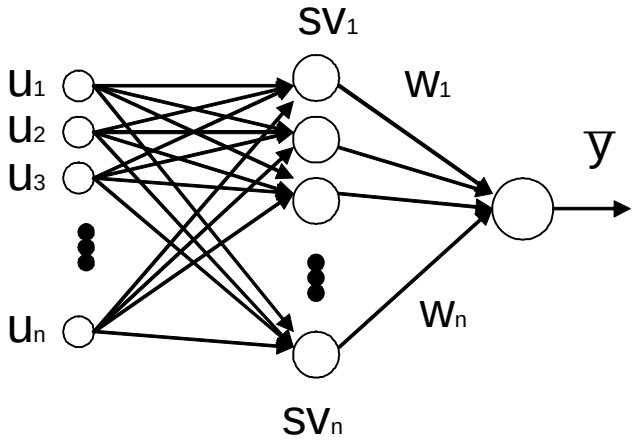

Figura 5.4: Modelo del sistema basado en SVM.

5.3. Construcción de multi modelos

La técnica multimodelo se utiliza para afrontar el problema de cambio de región de operación que experimentan muchas plantas no lineales. La razón por la cual se escoge esté enfoque es para diseñar esquemas de identificación con capacidad de modelar la dinámica de una planta utilizando en enfoque de *support vector machines* y reglas difusas.

Esté enfoque utiliza múltiples modelos para describir el comportamiento de un sistema que opera en distintos ambientes o regiones de operación. Inicia como un esfuerzo para mejorar el desempeño transitorio de sistemas lineales con grandes incertidumbres parámetricas y ha evolucionado gradualmente hacia el diseño de sistemas de control adaptables que en curso del tiempo pueden actuar con rápidez y exactitud en el control de procesos no lineales con un alto grado de incertidumbre.

Para identificación de sistemas, el error de estimación es causada por la incertidumbre de la estructura y las incertidumbres en la estimación de los parámetros. La incertidumbre de la estructura se puede resolver cambiando múltiples modelos. La incertidumbre en la estimación de parámetro se puede resolver con la actualización de pesos de cada red neuronal. Cuando

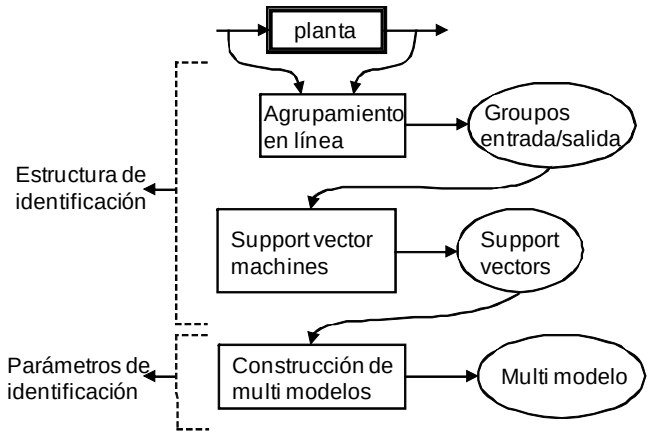

Figura 5.5: Construcción de multimodelos.

el peso de actualiza (parámetro) no se puede hacer el error de identificación muy pequeño, se debería cambiar por otra red neuronal (estructura). Cuando se tiene una red neuronal nueva, se debería usar el método de parámetros actualizados para minimizar el error de identificación hasta los parámetros (pesos) converjan. Así los nodos ocultos son arreglados mientra los pesos son actualizados. El esquema del modelado se muestra en la Fig. 5.5.

5.4. Simulación

Se tiene el problema típico el cúal también es discutido en [34][57][73] y en [32][32]. La planta a identificar es

$$y(k) = \frac{y(k-1)\,y(k-2)\,[y(k-1)+2{,}5]}{1+y(k-1)^2+y(k-2)^2} + u(k-1) \tag{5.20}$$

La señal de entrada entrenada es seleccionada como números aleatoreos en el intervalo $[0, 1]$.

$$X(k) = [y(k-1), y(k-2), u(k-1)]^T = [x_1(k), x_2(k), x_3(k)]^T$$

1) Primeramente, se usa agrupamiento en línea. Se selecciona $\alpha = 0{,}4$. como conocido a priori, se conocen los cambios máximos en la entrada y la salida son alrededor de 3 y 1,

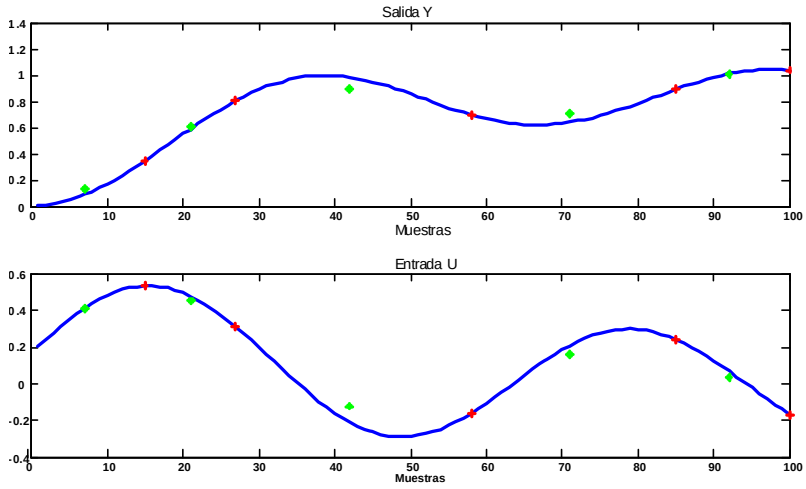

Figura 5.6: Agrupamiento en línea.

$\alpha \|\mathbf{x}_{\text{máx}} - \mathbf{x}_{\text{mín}}\| + (1-\alpha) \|y_{\text{máx}} - y_{\text{mín}}\| = 1{,}8$. Así L podría ser escogida tal que $L < 1{,}8$, en esta aplicación se selecciona $L = 1{,}5$. La Figura 5.6 muestra el resultado del agrupamiento en línea para datos de dos meses. Aquí "*" representa el centro de cada grupo, "+" es el límite entre los grupos. Exixten 5 grupos en $0 < t \leq 100$. La entrada de prueba es $u(k) = \sin(k/30) + \cos(k/20) + 2\sin(k/10)$. El resultado final del modelado neuronal tipo RBF se muestra en la Fig. 5.7

se puede observar que hasta el intervalo 100 se usan los *support vector machines* para entrenar las neuronas, y después de ese intervalo hasta 500 la red neuronal aprende por si sola.

Posteriormente utilizando la misma planta 5.20 . La señal de entrada entrenada es seleccionada como números aleatoreos en el intervalo $[0, 1]$.

$$X(k) = [y(k-1), y(k-2), u(k-1)]^T = [x_1(k), x_2(k), x_3(k)]^T$$

1) Primeramente, se usa agrupamiento en línea. Se selecciona $\alpha = 0{,}4$. como conocido a priori, se conocen los cambios máximos en la entrada y la salida son alrededor de 3 y 1,

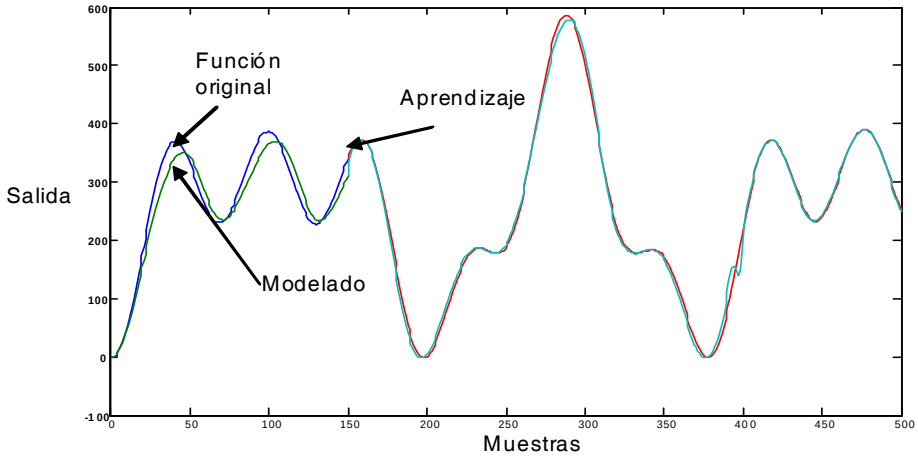

Figura 5.7: Modelado neuronal tipo RBF.

$\alpha \|\mathbf{x}_{\text{máx}} - \mathbf{x}_{\text{mín}}\| + (1-\alpha) \|y_{\text{máx}} - y_{\text{mín}}\| = 1{,}8$. Así L podría ser escogida tal que $L < 1{,}8$, en esta aplicación se selecciona $L = 1{,}5$. Figure 5.8 muestra el resultado del agrupamiento en línea para datos de dos meses. Aquí "*" representa el centro de cada grupo, "+" es el límite entre los grupos. Exixten 5 grupos en $0 < t \leq 100$.y utilizando el enfoque multimodelo se obtienen los siguientes resultados

En las gráfica superior de la Fig.5.9 se pueden observar las diferentes trayectorias de los diferentes grupos generados y los SVM´s, en la gráfica inferior se observa el intercambio de cada grupo.

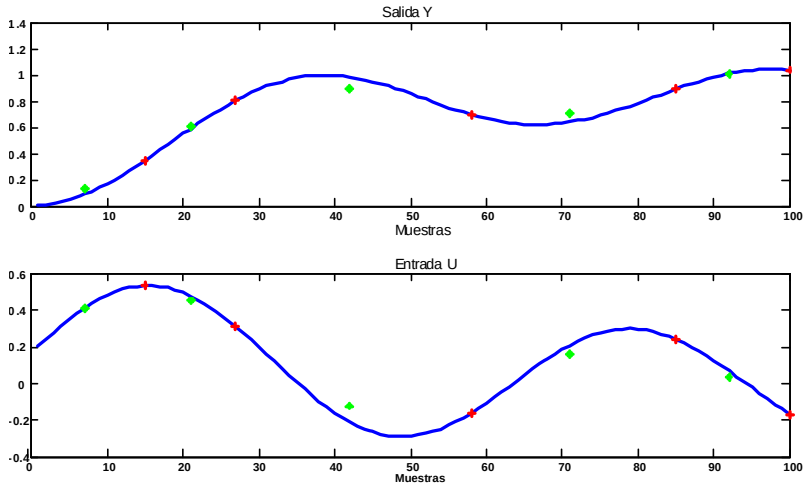

Figura 5.8: Agrupamiento en línea.

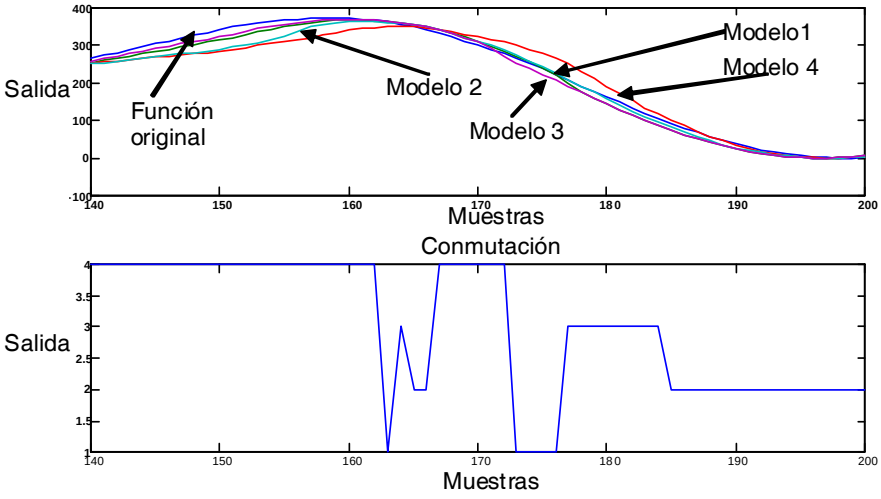

Figura 5.9: Multimodelo.

Capítulo 6

Modelado via SVM con selección de estructura automática

Las redes neuronales difusas han sido ampliamente usadas en la predicción de series de tiempo, modelado de sistemas no lineales y control [72]. En esta sección se verá el modelado de sistemas dinámicos no lineales con multiple redes neuronales difusas. El problema principal del modelado difuso es las extracción de reglas difusas, estas se pueden dividir en dos clases [43]: 1) obtención de reglas difusas de expertos, 2) obtención de reglas difusas automáticamente desde datos observados. El método experto usa el ctriterio sin sesgo [62] y la técnica entrenamiento y error, esta solo se puede aplicar fuera de línea.

El proceso de extraccion de reglas difusas para el modelado de sistemas no lineales es llamdo estructura de identificación. Un método común es la partición de los datos de entrada y salida, es también llamado reja difusa [34]. Muchos enfoques de estructura de identificación están basados en agrupamientos de datos fuera de línea, como el agrupamiento difuso C-medios [54], agrupamiento montaña [54], y agrupamiento substractivo [12]. Estos enfoques requieren que los datos esten listos antes del modelado. Existen algúnos métodos de agrupamiento en línea en la literatura. Una combinación de agrupamiento en línea y algoritmos genéticos para sistemas difusos están propuestos en [36]. En [68] el espacio de salida fue automáticamente particionado dentro de subconjuntos difusos por la teoría de resonancia

adaptiva. El agrupamiento en línea con una medición de proximidad espacial calculada recursivamente fue dada en [1]. Existe una debilidad para los métodos de agrupamiento en línea anteriores: el particionamiento de la entrada (precondición) y la salida (consecuencia) no están dentro de la cuenta de tiempo. Se usan todos los datos entrenados en cada regla. En este capítulo, un nuevo enfoque de agrupamiento en línea es propuesto. La relación de tiempo en el espacio de entrada-salida es considerado.

Al lado del enfoque agrupamiento, la extracción de reglas difusas también pueden ser realizadas por el método de redes neuronales [34], algoritmos genéticos [62], SVD-QR [10] y la técnica *support vector machines* (SVM) [15]. SVM fue usado primero para resolver problemas de clasificación de patrones. Vapnik lo definio como la estructura de minimización de riesgo el cual minimiza la cota superior del error de modelado. La idea básica del modelado SVM es el mapeo de las entradas dentro de un espacio característico de dimensión superior, entonces se resuelve la programación cuadrática (QP) con una función de costo apropiada [56]. Existe una propiedad importante dentro de SVM: la solución vector es separado. Solo las soluciones no ceros los cuales son llamados support vectors son útiles para el modelado. Aquí, se usan los suuport vectors para extraer las reglas difusas en cada grupo después del agrupamiento en línea.

A la combinación de multimodelo y redes neuronales difusas se les considera como un enfoque racional para identificación de sistemas no lineales. [40] y [48] usaron varias redes neuronales estáticas como identificador multimodelo, el algoritmo de conmutación fue realizado por una puerta de redes neuronales, pero el análisis de estabilidad no fue presentado. La identificación multimodelo y la detección de fallas usan redes neuronales estáticas presentadas en [65]. En [25], una mezcla jerárquica de combinación de métodos expertos de espacio entrada/salida es empleada. [45] propuso un controlador linealizador retroalimentado adaptivamente donde los términos no lineales son aproximados con redes neuronales múltiples. Se concluye que en sistemas en lazo cerrado son globalmente estables en el sentido que todas las señales involucradas están uniformemente acotadas. Otro tipo de redes neuronales múltiples para control adaptable son redes neuronales críticas adaptivas [77]. El método crítico adaptable determina leyes de control óptimo para un sistema adaptando dos redes neurales

consecutivamente, llamada acción red neuronal (la cúal distrubuye las señales de control) y una red neuronal crítica (la cúal "aprende" el índice de desempeño deseado para algúnas funciones asociadas con el índice de ejecución). Estas dos redes neuronales aproximan la ecuación Hamilton-Jacobi asociada con la teoría de control óptima. Durante la adaptación, ninguna de las redes necesita cualquier ' información' de una trayectoria óptima, solo el costo deseado necesita ser conocido. Esta técnica de diseño del neuro controlador no requiere entrenamiento en línea continua superando así los riesgos de inestabilidad [61].

Para los parámetros de identificación, se usan los datos para modificar las funciones de menbresía de cada regla difusa. La estabilidad de algoritmos de aprendizaje para los parámetros de identificación son muy importantes en aplicaciones. Es bien conocido que los algoritmos de identificación normal (por ejemplo, gradiente desendiente y mínimos cuadrados) son estables en condiciones idelaes. Estos podrían volverse inestables con respecto a las dinámicas no medeladas, algúnas técnicas de modificación robustas son necesarias [22]. Usando la teoría de pasividad, se demuestra con éxito que las redes neuronales con rangos de aprendizaje variantes en el tiempo son estables y robustos para cualquier incertidumbre acotada [80]. La identificación de parámetros del modelado difuso ¿tiene características similares?, En este capítulo, se muestra un rango de aprendizaje variante en el tiempo para uso común del algoritmo retro-propagación (backpropagation), se prueba que los errores de identificación están acotados.

A pesar de estas propuestas anteriores, algúnos investigadores en el pasado han llevado a cabo la realización de la selección de la estructura automatizada integralmente y la identificación de parámetros, junto con la conmutación de redes bajo redes difusos múltiples con la ejecución de identificación garantizada. Se verá una elección de estructura automatizada se propone primero dentro de un intervalo de tiempo fijo para un criterio de contrucción de la red dada. Entonces el algoritmo de actualización de los parámetros de la red se propone con el error de identificación acotado garantizado. Para cubrir con la estructura de incertidumbre, una estrategia de histérisis se desarrolla para habilitar la conmutación del identificador neuro difuso con la ejecución garantizada a lo largo de la conmutación del proceso.

El procedimiento para la selección automática de múltiples redes neuro-difusas se mues-

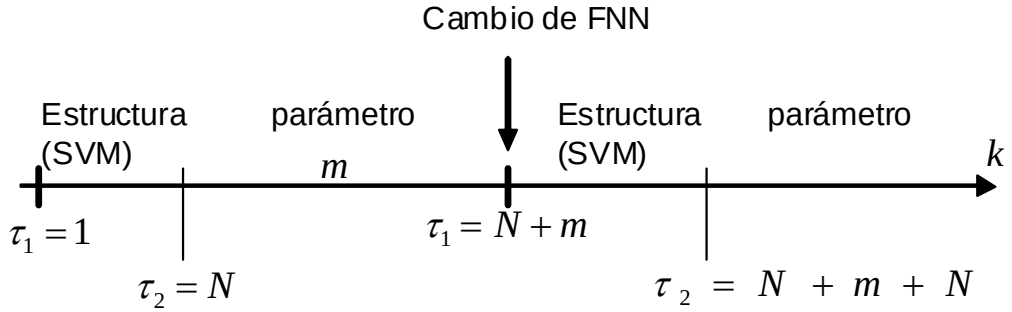

Figura 6.1: Selección automática.

tran en la figura 6.1

6.1. Multiple redes neuronales difusas para identificación de sistemas no lineales

Considerese a la siguiente planta no lineal discreta representada por

$$y(k) = f\left[\mathbf{x}\left(k\right), \theta\right] + e\left(k\right) \tag{6.1}$$

donde

$$\mathbf{x}\left(k\right) = \left[y\left(k-1\right), y\left(k-2\right), \cdots, y\left(k-n_y\right), u\left(k-d\right), u\left(k-d-1\right), \cdots, u\left(k-d-n_u\right)\right]^T \in \Re^{n\times 1} \tag{6.2}$$

y $f\left(\cdot\right)$ es una equación diferencial no lineal desconocida representando las dinámicas de la planta, $u\left(k\right)$ y $y\left(k\right)$ son escalares medibles entrada y salida, d es el retardo, θ es un vector de parámetros desconocidos asociado con una estructura de modelo apropiada, $e\left(k\right)$ es un ruido de observación acotado, n_y y n_u son las longitudes de salida y entrada, $n_y + n_u = n$. De hecho, (6.1) es un modelo NARX [8].

Un modelo difuso genérico se presenta como una colección de reglas difusas en la siguiente

forma (modelo difuso Mamdani [52])

$$\mathrm{R}^j\text{: IF } x_1 \text{ is } A_1^j \text{ and } x_2 \text{ is } A_2^j \text{ and } \cdots \; x_n \text{ is } A_n^j \text{ -ésimoEN } \widehat{y} \text{ is } B^j \tag{6.3}$$

Se usa $l(j = 1, 2, \cdots, l)$ reglas difusas IF–ésimoEN para realizar un mapeo del vector lingüístico de la entrada $\mathbf{x} = [x_1, \cdots, x_n] \in \Re^n$ a la salida $\widehat{y}$. De [72] se sabe, usando el producto inferencia, promedio de centros y el difusificador singleton, la p-ésimo salida del sistema lógico difuso se puede expresar como

$$\widehat{y} = \left(\sum_{j=1}^{l} w_j \left[\prod_{i=1}^{n} \mu_{A_i^j} \right] \right) / \left(\sum_{j=1}^{l} \left[\prod_{i=1}^{n} \mu_{A_i^j} \right] \right) = W\varphi\left[V, \mathbf{x}\left(k\right)\right] \tag{6.4}$$

donde $\mu_{A_i^j}$ es la función de membresía de los conjuntos difusos A_i^j, w_{pj} es el punto al cúal $\mu_{B_{pj}} = 1$. $W = [w_1, \cdots w_l]$, φ es la función vector de dimensión l, el elemento $\varphi_i = \prod_{i=1}^{n} \mu_{A_i^j} / \sum_{j=1}^{l} \left[\prod_{i=1}^{n} \mu_{A_i^j} \right]$, V corresponden a los parámetros de $\mu_{A_i^j}$. Generalmente las redes neuronales difusas (6.4) no puede coincidir dado un sistema no lineal (6.1) exactamente, el sistema no lineal (6.1) se puede representar como

$$y(k) = W\varphi\left[V, \mathbf{x}\left(k\right)\right] + \varepsilon\left(k\right) \tag{6.5}$$

donde $\varepsilon\left(k\right)$ se define como el error de modelado. El sistema no lineal (6.1) puede ser escrita como

$$y(k) = W^0\varphi\left[V^0, \mathbf{x}\left(k\right)\right] + \widetilde{f}\left(k\right) \tag{6.6}$$

donde $\widetilde{f}_k$ es el error modelado, V^0 y W^0 son conjuntos de parámetros conocidos escogidos por el usuario. En general, $\left|\widetilde{f}\left(k\right)\right| \geq \left|\varepsilon\left(k\right)\right|$.

De (6.6) se conoce el error modelado $\widetilde{f}_t$ depende de la estructura de la red neuronal difusa. Para algunos procesos no lineales, sus condiciones de operación varian con el tiempo o su entorno de operación es complicado, y un modelo no es suficiente para describir la planta completa. Los múltiples modelos pueden dar una mejor exactitud de identificación. Aunque una simple red neuronal difusa (6.4) puede identificar cualquier proceso no lienal (caja negra), el error de identificación puede ser grande si la estructura de la red difusa no se

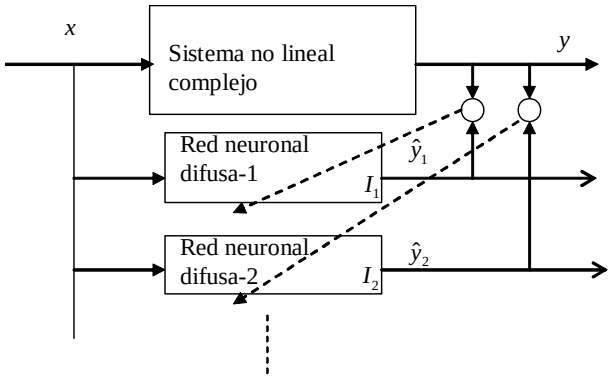

Figura 6.2: Identificador múltiple neuro difuso.

escoge apropiadamente. Generalmente hablando, no se puede encontrar la estructura de la red optima representando el sistema (6.1) bajo todas las condicones diferentes de operación. Una solución posible es usar varias redes y seleccionar la mejor por el algoritmo de conmutacióan apropiada. La estructura de los identificadores múltiples neuronales se muestran en al figura 6.2.

Aquí I_1, I_2 son los dos identificadores neuro difuso, cuyas salidas son $\widehat{y}_1, \widehat{y}_2$. Se usa una política de conmutación para escoger un identificador I_i tal que el error de identificación entre la salida de está red neuronal y la planta $(\widehat{y}_i(k) - y(k))$ se minimiza. Los identificadores múltiples neuro difusos son presentados como

$$I_\sigma : \widehat{y}_\sigma(k) = W^\sigma(k)\,\varphi\left[V^\sigma(k)\,\mathbf{x}(k)\right] \tag{6.7}$$

donde $\sigma = \{1, 2, \cdots r\}$ y r son el número total de identificadores neuro difusos.

El objetivo de la identificación múltiples redes neuro difusas es diseñar redes neuro difusas apropiadas y la política de conmutación tal que un índice de ejecución es minimizado y la conmutación infinita no ocurre.

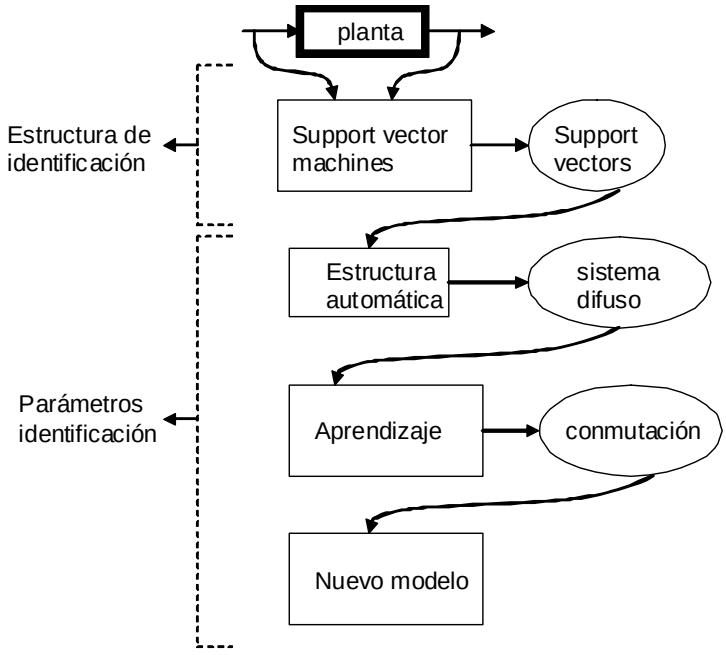

Figura 6.3: Selección de estructura automática.

6.2. Selección de estructura automática

Un SVM puede separar los datos dentro de dos clases con un hiperplano de margen máximo [15]. Si el entrenamiento es separable por el hiperplano, la función es escogida como $f(\mathbf{x}) = (\mathbf{w} \cdot \mathbf{x}) + b$. El margen se define como la distancia mínima desde una muestra hasta la superficie de resolución. El margen en turno se puede medir por la longitud del vector $\mathbf{w}$, tal que los puntos cercanos al hiperplano satisfase $|(\mathbf{w} \cdot \mathbf{x}) + b| = 1$.

Para N muestras $\{\mathbf{x}(k), y(k)\}_{k=1}^{N}$, se usa SVM para aproximar una función no lineal. Considerar la regresión en un conjunto de funciones no lineales

$$f(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x}) + b \tag{6.8}$$

donde el kernel $K(\mathbf{x}, \mathbf{x}_k) = \phi(\mathbf{x})^T \phi(\mathbf{x}_k)$. Existen muchas posibilidades para escoger el kernel $K(\mathbf{x}, \mathbf{x}_k)$, el cúal requiere la condición de Mercer [15]. Por ejemplo kernel lineal $K(\mathbf{x}, \mathbf{x}_k) = \mathbf{x}_k^T \mathbf{x}$, kernel MLP $K(\mathbf{x}, \mathbf{x}_k) = \tanh\left(k_1 \mathbf{x}_k^T \mathbf{x} + k_2\right)$, kernel RBF $K(\mathbf{x}, \mathbf{x}_k) = \exp\left(-\|\mathbf{x} - \mathbf{x}_k\|^2 / \sigma^2\right)$. En esta sección, se usará el kernel difuso el cúal se define como

$$K(\mathbf{x}, \mathbf{x}_k) = \{u(\mathbf{x}_i, \mathbf{x}_j)\}, \quad i, j = 1, \cdots N$$

donde u_i es una función de membresía. Se usa la función Gausiana,

$$u(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right) \tag{6.9}$$

La regresión es resolver el problema siguiente

$$\begin{aligned} &\text{mín } J_p = \tfrac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{k=1}^{N} s_k \xi_k \\ &sujeto\ a\ \ \left|y_k - \left(\mathbf{w}^T \varphi(\mathbf{x}_k) + b\right)\right| \leq \epsilon \end{aligned} \tag{6.10}$$

donde $\xi_k \geq 0$, $k = 1, 2, \ldots, N$, C es constante, ξ_i es la variable débil. s_k es un factor difuso. Este denota el grado de importancia de la muestras x_i para el aprendizaje del hiperplano óptimo en SVM. Se selecciona s_i como la función forma de campana $s_k = \frac{1}{1+\left|\frac{x-c}{a}\right|^{2b}}$, ver figura

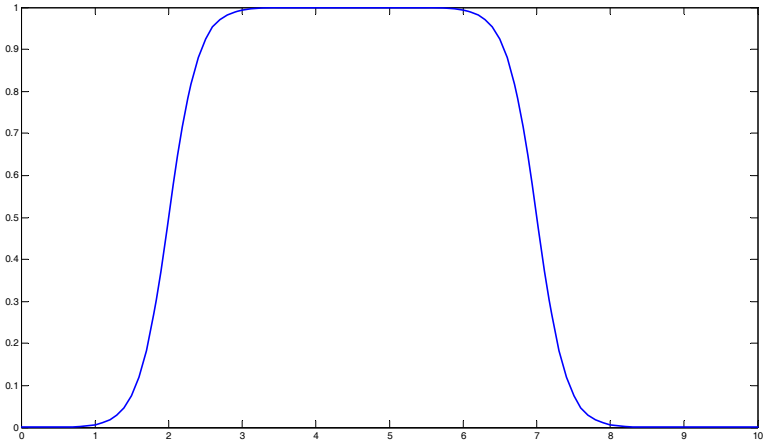

Figura 6.4: Función de membresía en forma de campana generalizada.

6.4. Así el s_k más pequeño es, el más pequeño efecto que la muestra $\mathbf{x}_k$. (6.10) puede ser transformada dentro del siguiente problema cuadráctico de programación:

$$\begin{aligned} &\text{máx}\, W(\alpha) = \sum_{i=1}^{N} \alpha_i - \tfrac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j K(x_i, x_j) \\ &\text{sujeto:} \sum_{i=1}^{N} \alpha_i y_i = 0 \end{aligned} \tag{6.11}$$

donde $0 \leq \alpha_i \leq s_i C$, $i = 1, 2, \ldots, N$. Mediante el paquete de software QP estandard se obtiene la solución α_k^*, y la función resultante es

$$f(\mathbf{x}) = \sum_{k=1}^{N} (\alpha_k - \alpha_k^*) K(\mathbf{x}_k, \mathbf{x}) + b$$

Puesto que el vector solución es escaso, la suma podría ser tomado solo bajo los donde α_k^* no ceros, y la función resultante es

$$f(\mathbf{x}) = \sum_{k=1}^{v} (\alpha_k - \alpha_k^*) K(\mathbf{x}_k, \mathbf{x}) + b \tag{6.12}$$

donde v es el número de support vectors, $v \ll N$. Se define la posición de los support vector como $\left[x_j^*, y_j^*\right]$, $j = 1 \cdots v$.

De (6.12) se sabe que los support vectors son suficientes para representar la no linealidad en cada grupo. Se usan estos support vectors para construir reglas difusas. Para cada conjunto de datos $\left[y\left(k\right), \mathbf{x}\left(k\right)\right]$, $k \in [1, N]$, se extraen las reglas del producto difuso de la forma de (6.3). Se usan los support vectors $\left[x_j^*, y_j^*\right]$, $j = 1 \cdots v$ para construir las funciones de membresía. x_j^* es el centro de la función Gausiana $\mu_{A_i^j}$, y_j^* es el centro de la función Gausiana μ_{B^j}. Al tiempo $k = N$, se usa el siguiente SVM automático para construir reglas difusas. El esquema del modelado se muestra en la Fig. 6.3.

Paso 1: Inicialización: selecionar s_k y ξ_k de acuerdo a (6.10). Además, calcular el kernel difuso de acuerdo a (6.9).

Paso 2: Encontrar los support vector: Resolver el problema dual de acuerdo a (6.11).

Paso 3: construcción reglas difusas: el support vector α_k^* se usa como los centros de las funciones de membresía, entonces el sistema difuso es (6.4).

Paso 4: Entrenamiento de la función de membresía: se discutirá en la siguiente sección

Paso 5: fase de chequeo: checar si la construcción de red podría ser cambiado. Si no, se va al Paso 4. De lo contrario, terminar el preceso de entrenamiento, ir a la fase de construcción de la red (Paso 2).

6.3. Selección de modelos

Para identificación de un sistema, el error de estimación es causado por ambas incertidumbres de estructura y de parámetros. La incertidumbre de estructura se puede resolver por la conmutación entre multiples redes neuro difusas. La incertidumbre de parámetros se resuelve por la actualización de las funciones de membresía. Cuando las funciones de membresía no pueden hacer el error de identificación mas pequeño, se debería cambiar a otra red neuro difusa. Cuando se tiene una nueva red neuro difusa, se deben usar el método de parámetros actualizados para minimizar el error de identificación hasta que los parámetros (funciones de membresía) convergan.

Se define el índice del error de desempeño $J_p(k)$ para el p–ésimo identificador neuronal como

$$J_p(k) \triangleq \alpha e^2(k) + (1-\alpha) \sum_{t=k-N_p}^{k} e^2(t) \tag{6.13}$$

donde α es un parámetro de diseño para ambos errores de pesos término corto y término largo, N_p es la longitud del p–ésimo identificador neuro difuso. Este índice de desempeño es similar como en [57].

La decisión de la estructura de conmutación (cambio) esta dirigido por el monitoreo del índice de desempeño (6.13). Para prevenir un cambio rápido arbitrariamente debido a las perturbaciones, un algoritmo de decisión de histérisis es necesario. En este capítulo, se cambiara el modelo de la red neuro difusa solo cuando las funciones de membresía son casi convergentes. Se define la función de cambio como $(k > N)$

$$\begin{array}{c} \omega(k) \triangleq \frac{1}{2}\left|\mathrm{tr}\left(W^T(k)\,W(k)\right) - \frac{1}{k-N}\sum_{t=N}^{k}\mathrm{tr}\left(W^T(t)\,W(t)\right)\right| \\ +\frac{1}{2}\left|\mathrm{tr}\left(V^T(k)\,V(k)\right) - \frac{1}{k-N}\sum_{t=N}^{k}\mathrm{tr}\left(V^T(t)\,V(t)\right)\right| \end{array}$$

donde $V^T(k) = [c_{ji}(k), \sigma_{ji}(k)]$. El algoritmo de dicisión de histéresis puede ser formulado como

$$r = \begin{cases} r+1 & \text{si } J(k) > J(k-1) + h \text{ y } \omega(k) \leq L \\ r & \text{otra manera} \end{cases} \tag{6.14}$$

donde $h > 0$ es la constante de histéresis, L es el umbral para el cambio de pesos. $r = r+1$ define la estructura de la r−-ésimo red neuronal no es apropiada para el dato entrante, se debe comenzar una red neuronal, ahora $\sigma = \{1, 2, \cdots r, r+1\}$, ver (6.7). Se escoge L tal que el algoritmo de conmutación (6.14) pueda trabajar después de los parámetros no afecten significativamente la identificación neuro difusa. El procedimiento para la selección automática de múltiples redes neuro difusas se muestra en la figura 6.5.

En general incluso para sistemas lineales conmutados invariantes en el tiempo, la estabilidad de cada componente del sistema en lazo conmutado no garantiza la estabilidad del sistema conmutado entero bajo leyes de conmutación arbitrarios. Algunas condiciones sobre la política de conmutación es muy especial: solo pueden conmutar desde el modelo r hasta

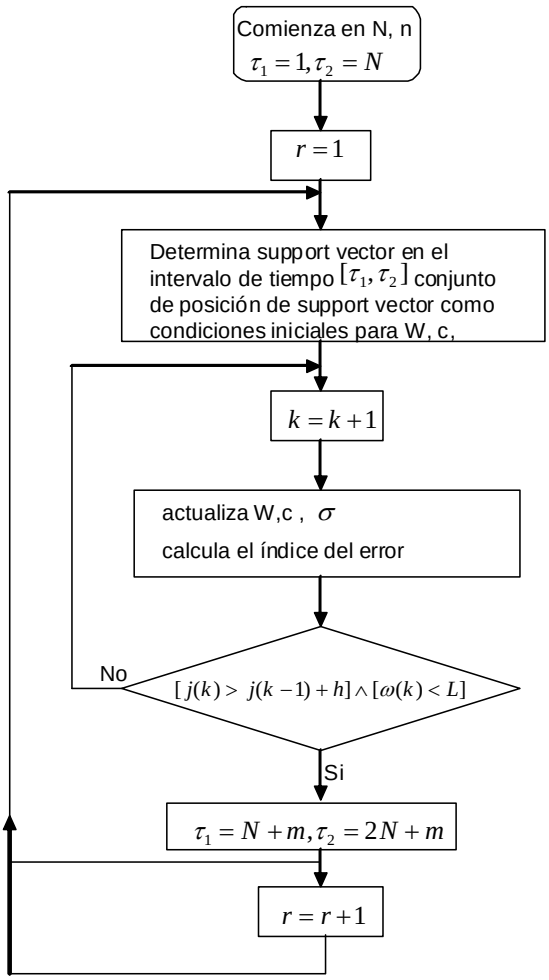

Figura 6.5: Selección automática de múltiples redes neuro difusas.

el modelo $r+1$ en el eje de tiempo. La conmutación es una secuencia. El teorema asegura que el error de identificación en cada subsistema es estable. Puesto que los subsistemas son conmutados uno por uno, se discutirá la convergencia de nuestro esquema de identificación multi neuronal.

Lemma 6.1 *(a) Para cada tiempo finito k_1, existe por lo menos un modelo $r_0 \in \sigma$ tal que el índice de desempeño $J_{r_0}(k)$ en (6.13) esta acotado en $[1, k_1)$*

(b) Para cada $i \in \sigma$ el índice de desempeño $J_i(k)$ en (6.13) tiene un límite (puede ser infinito) como $k \to k_1$. [1]

Demostración. $J_i(k)$ en (6.13) incluye dos partes: error instantáneo $e^2(k)$ y la suma de los errores en el i–ésimo modelo $\sum_{t=k-N}^{k} e^2(t)$. El teorema 1 muestra que el error de identificación en cada modelo está acotado.Para cada tiempo finito k_1, en $[1, k_1)$ hay neuro identificadores finitos que están envueltos en la identificción, (a) es establecido.

Para cada neuro identificador i, cualquier tiempo k, una constante de histéresis h y un umbral L, existe una constante r. A cada intervalo estrictamente propio $[1, k_2) \subset [1, k_1)$, r puede aumentar por 1 a lo más por un número finito de veces. Puesto que en $[1, k_1)$ existe un número finito de neuro identificadores, $J_i(k)$ tiene un límite como $k \to k_1$, $i = 1, 2, \cdots r$. (b) es establecido. ■

Teorema 6.1 *Si se usan las multi redes neuronales como en (6.7) y el algoritmo de conmutación de histérisis como en (6.14), entonces (a) existe un tiempo finito k^* después de que para todo $k > k^*$, $J_r(k)$ está acotada en $[1, K)$, $K > k^*$, (b) el proceso de conmutación de los modelos neuronales paran, i.e., r es constante.*

Demostración. (a) El lema anterior los estados que al $[1, k^*)$ existe una cota $J_i(k)$. Este teorema discute sobre $[k^*, K)$, existe una cota $J(k)$. Así en $[1, k^*)$, existe al menos σ_0 tal que J es acotada. Por otro lado, podría existir otro σ_1 tal que al $[k^*, K)$ hay neuro identificadores

[1] Sea f una función esta definida sobre un intervalo abierto conteniendo a c (excepto posiblemente a c.) EL argumento "Límite Infinito" $\lim_{x \to c} f(x) = \infty$ significa que para cada $M > 0$ existe un δ tal que $f(x) > M$ siempre que $0 < |x - c| < \delta$.

infinitos. En este caso $J_i(k)$ no es acotado. En $[1, K)$ se definen dos subconjuntos disjuntos Ω_b y Ω_u, donde J esta acotada en Ω_b y no acotada en Ω_u. Por el lema anterior, se conoce, el conjunto Ω_b no es vacío, i.e.,

$$J_i(k) \leq c < \infty \quad i \in \Omega_b, \quad k \in [0, K), \quad c > 0 \tag{6.15}$$

cuando J_i esta en Ω_u, J_i es no acotada. Se puede expresar lo anterior como

$$J_i(k) > c + h, \quad i \in \Omega_u, \quad k \in [0, K) \tag{6.16}$$

Del lema anterior se conoce cuando $k \to K$, $J_i(k)$ tiene un límite. Así existe t_1 el cual esta cerca de K tal que

$$|J_i(k) - J_i(k_1)| < h/2, \quad i \in \Omega_b, \quad k \in [k_1, K) \tag{6.17}$$

(6.17) digamos que para cada $i \in \Omega_b$, existe k_1 tal que $J_i(k)$ tiene una variación pequeña en $k \in [k_1, K)$, la cual no crece más de la mitad de h. Puesto que L se seleccional tal que el algoritmo de conmutación (6.14) no puede ser efectuado por los pesos después de que los pesos convergan. Así solo se considera el caso $\omega(k) \leq L$.

(a) Se verifica la cota de J

- Suponer que después k_1 se tienen dos neuro identificadores i y j,

$$J_j(k) \leq J_i(k) + h, \quad k \in [k_1, K)$$

de (6.14) se sabe que r no cambia después de k_1, así $J_i(k)$ satisface (6.15). Sea $k^* = k_1$, usando (6.15) se tiene

$$J \leq c + h, \quad k \in [k^*, K)$$

así J esta acotada en $[1, K)$.

- Suponer al tiempo $k_2 \in [k_1, K)$,

$$J_j(k_2) > J_i(k_2) + h$$

r cambia en k_2 entonces permanece constante en $k \in [k_2, K)$. Sea $k^* = k_2$

$$J_j(k_2) > J(k_2) + h > J(k_2), \quad j \in \Omega \tag{6.18}$$

Dado que Ω_b es no vacío, existe una cota J antes que k_2, tal que

$$c > J_j(k_2)$$

Por consiguiente (6.18), J esta acotada en $[1, K)$.

(b) Se revisará el caso de cambio infinito.

- Si $j \in \Omega_b$, para cualquier $k \in [k_2, K)$, donde k_2 esta cerca de K, se tiene

$$J(k) - J_j(k) = [J(k) - J(k_2)] + [J(k_2) - J_j(k_2)] + [J_j(k_2) - J_j(k)]$$

Usando (6.18) se tiene $J(k_2) - J_j(k_2) \leq 0$, así

$$\begin{aligned} J(k) - J_j(k) &\leq [J(k) - J(k_2)] + [J_j(k_2) - J_j(k)] \\ &\leq |J(k) - J(k_2)| + |J_j(k_2) - J_j(k)| \end{aligned} \tag{6.19}$$

Siguiendo (6.17), (6.19) se vuelve

$$J(k) - J_j(k) \leq h, \quad k \in [k_2, K), \quad j \in \Omega_b$$

- si $j \in \Omega_u$, de lo anterior se sabe que $J(k)$ esta acotada, (6.15) and (6.16) implica que

$$J(k) - J_j(k) \leq c - (c + h) \leq h, \quad k \in [k_2, K), \quad j \in \Omega_u,$$

Así para todo $j \in \Omega$,

$$J(k) \leq J_j(k) + h, \quad k \in [k_2, K) \tag{6.20}$$

(6.14) y (6.20) implica que no más conmutaciones ocurrirán en $[k_2, K)$, así r es constante en $[k^*, K)$, esto es, el cambio de los modelos neuronales parará. ■

Comentario 6.1 *La prueba de este teorema es diferente del Lema de Conmutación de Histéresis en [55]. Primeramente, se usa multi redes neuro difusas como modelos de identificación. Así no es necesario "asumir el lazo abierto". Estas condiciones son provistas en el Lema 6.1. Segundamente, el algoritmo de conmutación de histéresis se cambia de acuerdo a la propiedad especial de neuro identificadores, así la prueba también considera las condiciones de los pesos.*

6.4. Simulación

Se tiene el problema típico el cual también se discutió en [34][57][73] y en [32][33]. La planta a identificar es

$$y(k) = \frac{y(k-1)\,y(k-2)\,[y(k-1)+2{,}5]}{1+y(k-1)^2+y(k-2)^2} + u(k-1) \tag{6.21}$$

La señal de entrada entrenada es seleccionada como números aleatoreos en el intervalo $[0, 1]$.

$$X(k) = [y(k-1), y(k-2), u(k-1)]^T = [x_1(k), x_2(k), x_3(k)]^T$$

1) Primeramente, se usa agrupamiento en línea. Se selecciona $\alpha = 0{,}4$. como conocido a priori, se conocen los cambios máximos en la entrada y la salida son alrededor de 3 y 1, $\alpha \|\mathbf{x}_{\text{máx}} - \mathbf{x}_{\text{mín}}\| + (1-\alpha)\|y_{\text{máx}} - y_{\text{mín}}\| = 1{,}8$. Así L podría ser escogida tal que $L < 1{,}8$, en esta aplicación se selecciona $L = 1{,}5$. Figure 6.6 muestra el resultado del agrupamiento en línea para datos de dos meses. Aquí "*" representa el centro de cada grupo, "+" es el límite entre los grupos. Exixten 5 grupos en $0 < t \leq 100$. La entrada de prueba es $u(k) = \sin(k/30) + \cos(k/20) + 2\sin(k/10)$. El resultado final del modelado neuronal tipo RBF se muestra en la Fig. 6.7

se puede observar que hasta el intervalo 100 se usan los *support vector machines* para entrenar las neuronas, y después de ese intervalo hasta 500 la red neuronal aprende por si sola.

Posteriormente utilizando la misma planta

$$y(k) = \frac{y(k-1)\,y(k-2)\,[y(k-1)+2{,}5]}{1+y(k-1)^2+y(k-2)^2} + u(k-1) \tag{6.22}$$

La señal de entrada entrenada es seleccionada como números aleatoreos en el intervalo $[0, 1]$.

$$X(k) = [y(k-1), y(k-2), u(k-1)]^T = [x_1(k), x_2(k), x_3(k)]^T$$

1) Primeramente, se usa agrupamiento en línea. Se selecciona $\alpha = 0{,}4$. como conocido a priori, se conocen los cambios máximos en la entrada y la salida son alrededor de 3 y 1,

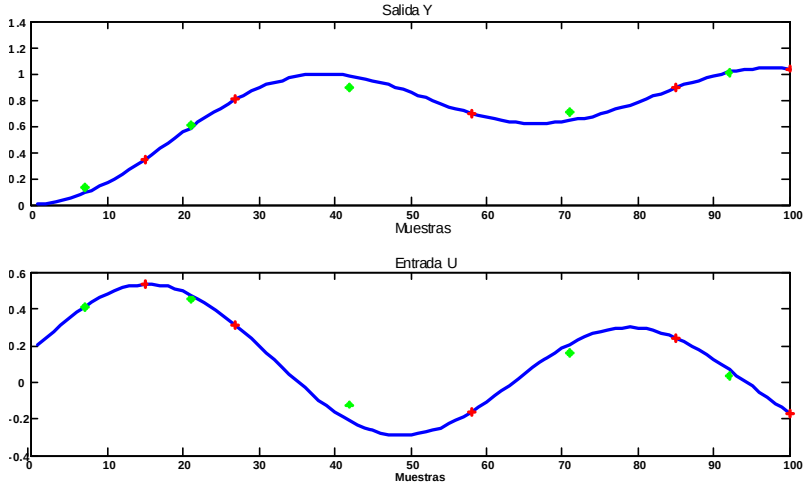

Figura 6.6: Agrupamiento en línea.

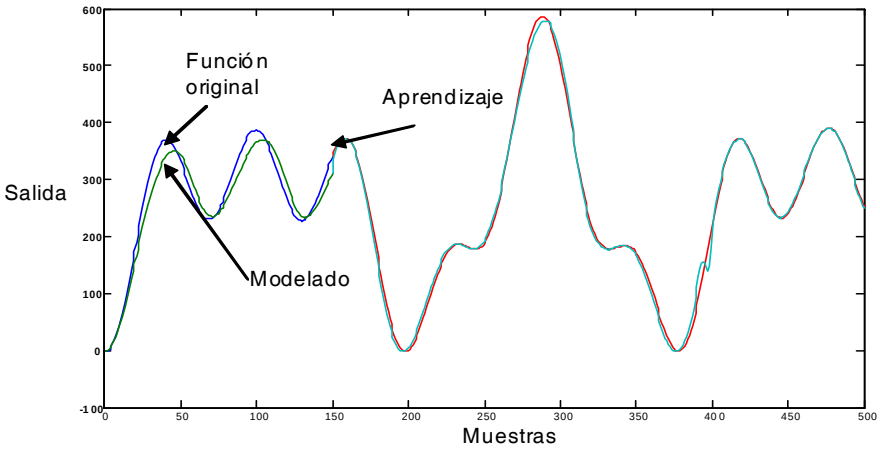

Figura 6.7: Modelado neuronal tipo RBF.

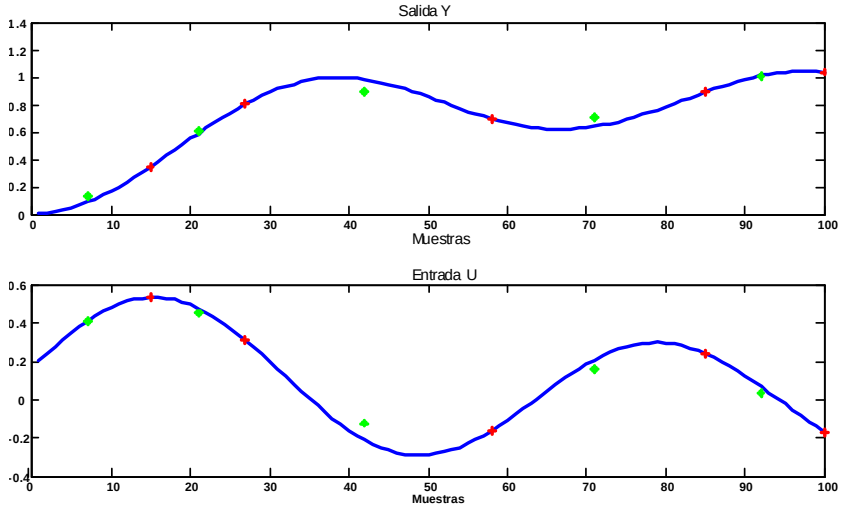

Figura 6.8: Agrupamiento en línea.

$\alpha \|\mathbf{x}_{\text{máx}} - \mathbf{x}_{\text{mín}}\| + (1-\alpha) \|y_{\text{máx}} - y_{\text{mín}}\| = 1{,}8$. Así L podría ser escogida tal que $L < 1{,}8$, en esta aplicación se selecciona $L = 1{,}5$. Figure 6.8 muestra el resultado del agrupamiento en línea para datos de dos meses. Aquí "*" representa el centro de cada grupo, "+" es el límite entre los grupos. Exixten 5 grupos en $0 < t \leq 100$.y utilizando el enfoque multimodelo se obtienen los siguientes resultados

En la gráfica superior de la Fig.6.9 se pueden observar las diferentes trayectorias de los diferentes grupos generados y los SVM´s, en la gráfica inferior se observa el intercambio de cada grupo.

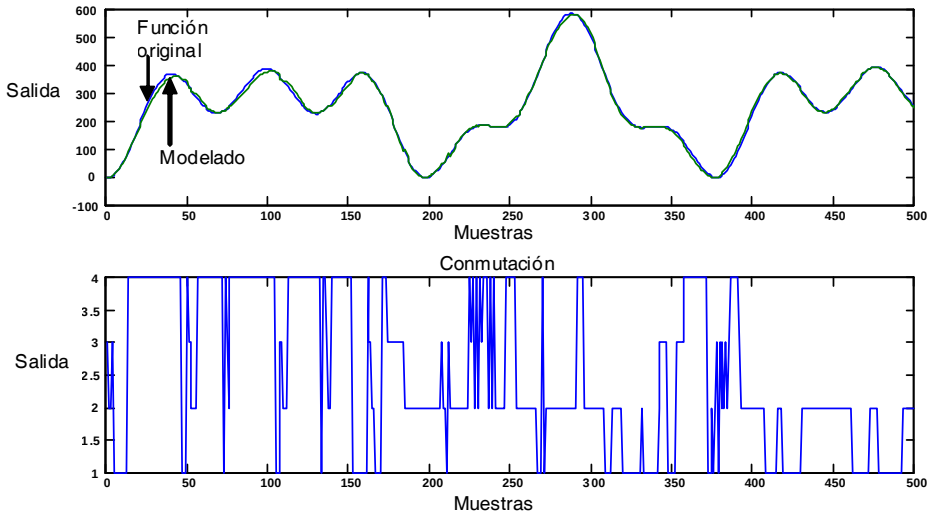

Figura 6.9: Multimodelo con selección de estructura automática.

Capítulo 7

Aplicación para horno de gas

En este capítulo se tomará un ejemplo real, el cual existe en la literatura [63], los datos se usarán para aplicar los enfoques via agrupamiento en línea y SVM normal, via agrupamiento en línea y SVM difusol, modelado via multi redes neuronales y via SVM con selección de estructura automática

7.1. Horno de gas

Se utiliza un programador el cual nos servirá para poner a funcionar el horno. Este controlador hará una secuencia de seguridad para evitar peligros.

1. Manualmente se abrirá la válvula de seguridad la cual a su vez esta controlada por el programador, al haber algún problema el controlador cerrara esta automáticamente.

2. Se accionara a un ventilador el cual funcionara por un tiempo programado para desalojar cualquier posible gas en el horno que pueda ocasionar una explosión.

3. El programador abrirá la válvula del quemador piloto y al mismo tiempo creara un arco eléctrico por medio del transformador de ignición.

4. Una fotocelda detectará si existe flama en el piloto, de detectar flama se podrá pasar al siguiente paso.

5. Una vez detectada la flama por la fotocelda el programador abrirá la válvula principal y quedando totalmente la regulación de la temperatura en manos del controlador.

6. Por medio de un sensor de temperatura se le hará llegar la señal al controlador y este a su vez abrirá o cerrara así sea necesario a una válvula de control controlada por un actuador.

El programador en todo momento esta sensando la presencia de flama, y en cualquier momento la flama se apagara, el programador cerrara la válvula de seguridad y accionara una alarma.

7.2. Resultado experimental via agrupamiento en línea y SVM normal

1. En este ejemplo se construye un modelo (conjunto de datos) para un horno de gas [3]. El conjunto de datos contiene 296 pares de puntos entrada/salida, donde la entrada u_k fue la entrada de gas codificada y la salida y_k representa la concentración de CO_2 del horno de gas. El conjunto de datos entrada/salida se muestra en la Fig.7.18. El vector de entrada esta definido por $x_k = [y_{k-1}\ y_{k-2}\ y_{k-3}\ u_{k-1}\ u_{k-2}\ u_{k-3}]^{\ T}$. Las muestras impares $[y_k, u_k]$ fueron usadas para el entrenamiento mientras las muestras iguales fueron omitidas para probar el modelo construido.Primeramente, se usa agrupamiento en línea. Se selecciona $\alpha = 0{,}4$. como conocido a priori, se conocen los cambios máximos en la entrada y la salida son alrededor de 3 y 1, $\alpha \left\|\mathbf{x}_{\text{máx}} - \mathbf{x}_{\text{mín}}\right\| + (1-\alpha) \left\|y_{\text{máx}} - y_{\text{mín}}\right\| = 1{,}8$. Así L podría ser escogida tal que $L < 1{,}8$, en esta aplicación se selecciona $L = 1{,}5$. Fig. 7.19 muestra el resultado del agrupamiento en línea para datos de dos meses. Aquí "*" representa el centro de cada grupo, "+" es el límite entre los grupos. Existen 11 grupos en $0 < t \leq 100$.

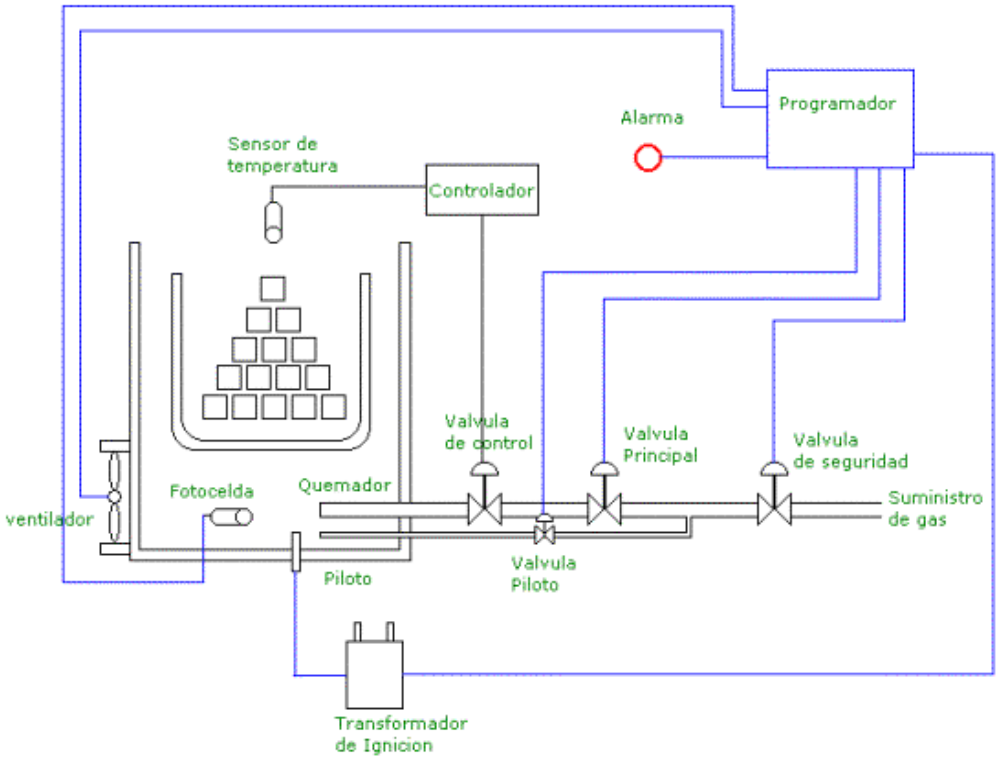

Figura 7.1: Esquema horno de gas.

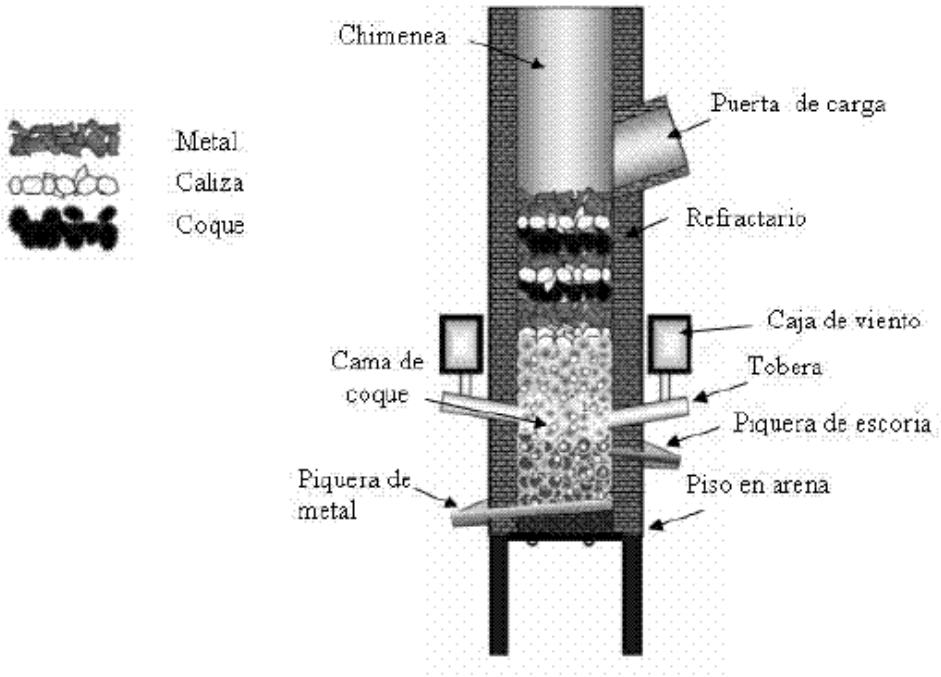

Figura 7.2: Horno de gas.

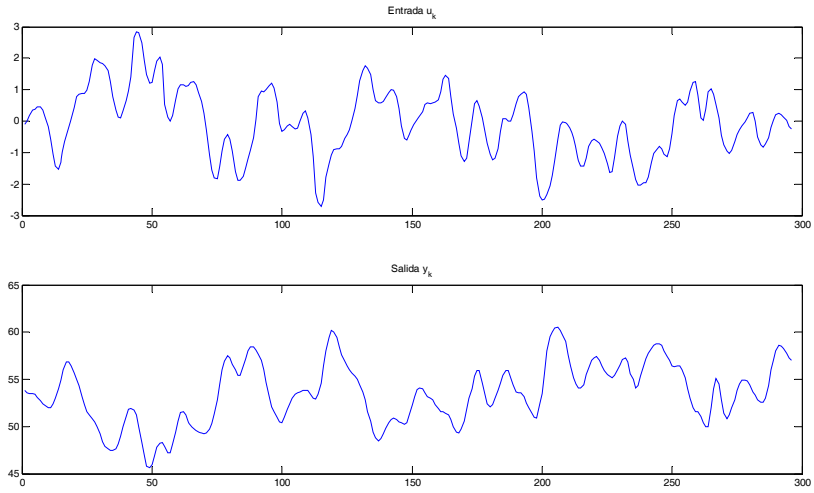

Figura 7.3: Conjunto de datos entrada/salida.

2. Segundo, se usa *support vector machines* (3.11) para obtener los support vectors como en (3.13). El número de support vectors para los 11 grupos son 3,3,3,3,5,5,7,7,3,3 y 1. Así se tienen 42 reglas como en (3.51). Los centros de las funciones Gausianas de esas reglas son las posiciones de los support vector.

3. Ahora se usan los datos en cada grupo para actualizar las funciones de membresía.

4. Cuando se combinan los 11 modelos difusos locales (42 reglas) dentro de un modelo global Takagi-Sugeno da la forma

$$\widehat{y} = \left(\sum_{i=1}^{11} f_i \left[x\left(k\right) \right] \left[\prod_{j=1}^{43} \left(\mu_{A_i^j} \right) \right] \right) / \left(\sum_{i=1}^{11} \left[\prod_{j=1}^{43} \left(\mu_{A_i^j} \right) \right] \right)$$

y aplicando la metodología de SVM normal se obtine la siguiente figura 7.5 El resultado

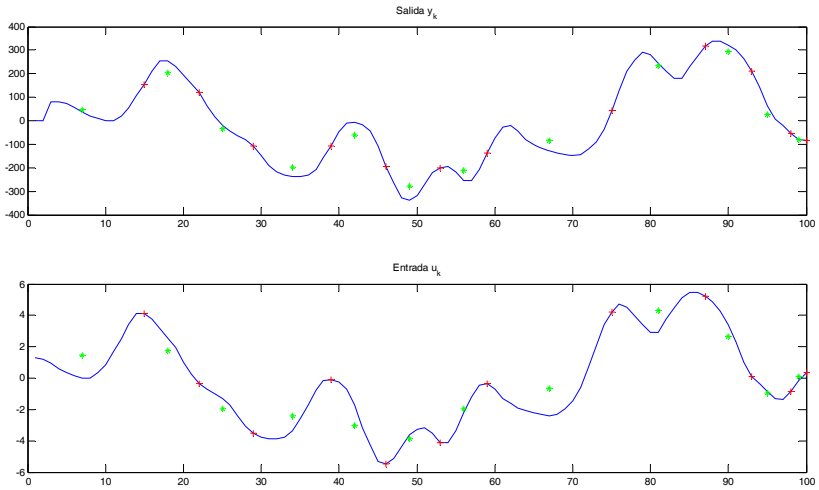

Figura 7.4: Agrupamiento en línea.

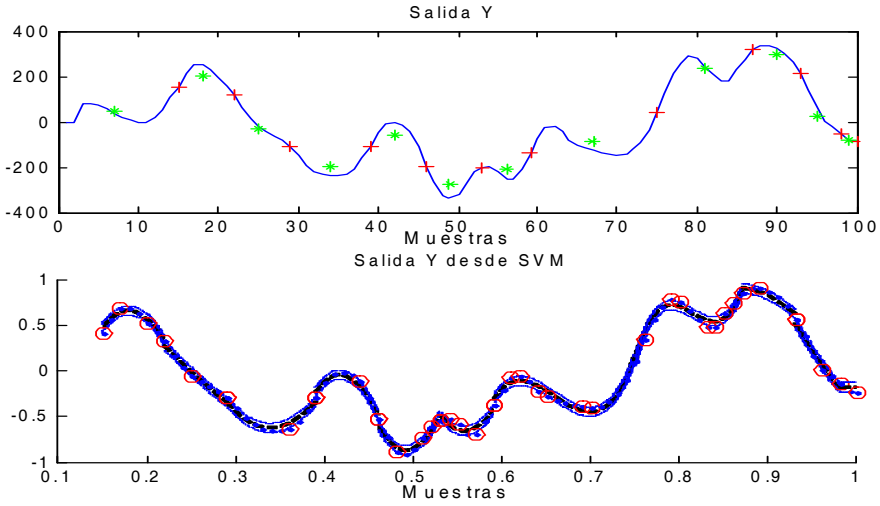

Figura 7.5: SVM

después de usar SVM son mostrados en la siguiente tabla

Grupo	# de sv	Posición	# de datos
1	3	$1, 3, 6$	8
2	3	$1, 4, 8$	8
3	3	$1, 8, 11$	11
4	3	$1, 6, 8$	8
5	5	$1, 3, 6, 7, 8$	8
6	5	$1, 2, 3, 5, 7$	7
7	7	$1, 3, 4, 6, 7, 11, 12$	17
8	7	$2, 5, 6, 9, 10, 11, 12$	13
9	3	$1, 3, 7$	7
10	3	$1, 4, 6$	6
11	1	3	3

Tabla 1: SVM para función de estimación.

Después del entrenamiento de las funciones de membresía, las 43 funciones de membresía para x_1 se muestran en la Figura 7.6.Entonces los parámetros fijados de las funciones de membresía, usan otros 296 datos para la prueba del modelo. El dato probado es $x_1(k) = 1 - \frac{2k}{T}$, $x_2(k) = -1 + \frac{2k}{T}$, $k = 1, 2, \cdots, T$. El resultado final del modelado despues de los modelos locales ($p = 43$) se muestran en la Figura 7.6

7.3. Modelado via agrupamiento en línea y SVM difuso

Cada punto puede causar sobreajuste no adaptado en SVMs. De aquí, el concepto central del proposito del FSVM es para asignar a cada punto un valor de membresía de acuerdo a sus importancia relativa en las clases. Desde cada punto x_i tiene un valor de membresía asignado μ_i, el conjunto entrenado parece un conjunto difuso entrenado S_f y es dado por

$$S_f = \{x_i, y_i, \mu_i\}_{i=1}^{n} \tag{7.1}$$

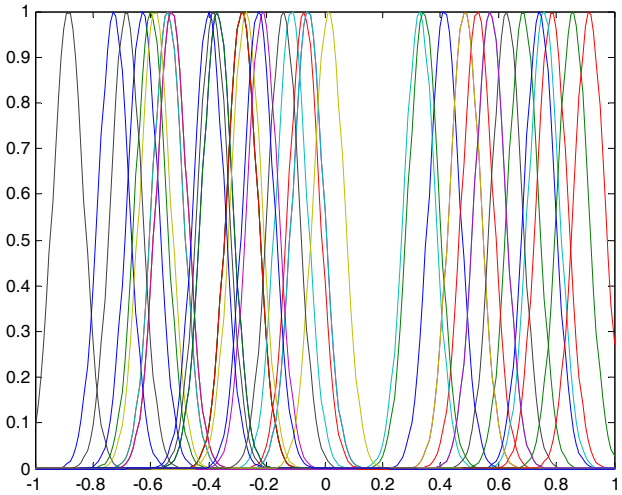

Figura 7.6: Funciones de membresia de x_1 en las 43 reglas difusas.

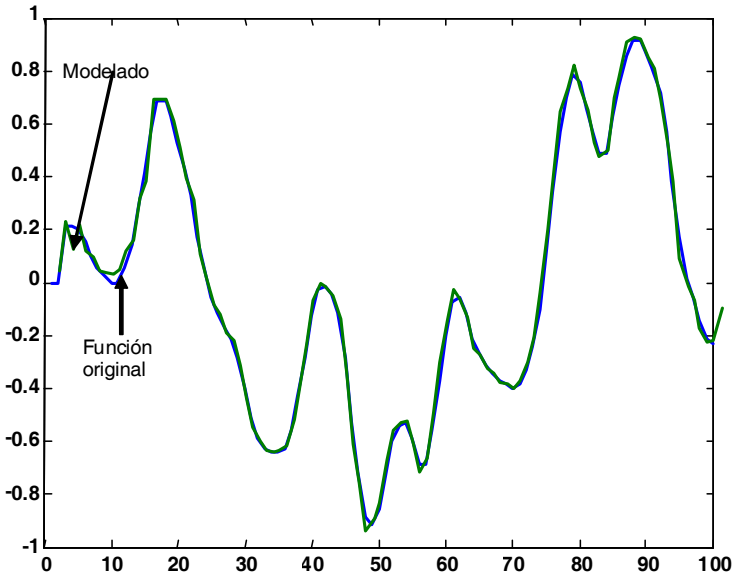

Figura 7.7:

Para una clase positiva ($y_i = +1$) el conjunto de los valores de membresía denotados como μ_i^+, y son denotados como μ_i^- para las clases negativas ($y_i = -1$). Estos son asignados independientemente.

1. En este ejemplo se construye un modelo (conjunto de datos) para un horno de gas [3]. El conjunto de datos contiene 296 pares de puntos entrada/salida, donde la entrada u_k fue la entrada de gas codificada y la salida y_k representa la concentración de CO_2 del horno de gas. El conjunto de datos entrada/salida se muestra en la Fig.7.8. El vector de entrada esta definido por $x_k = [y_{k-1}\ y_{k-2}\ y_{k-3}\ u_{k-1}\ u_{k-2}\ u_{k-3}]^T$. Las muestras impares $[y_k, u_k]$ fueron usadas para el entrenamiento mientras las muestras iguales fueron omitidas para probar el modelo construido.Primeramente, se usa agrupamiento en línea. Se selecciona $\alpha = 0{,}4$. como conocido a priori, se conocen los cambios máximos en la entrada y la salida son alrededor de 3 y 1, $\alpha \|\mathbf{x}_{\text{máx}} - \mathbf{x}_{\text{mín}}\| + (1-\alpha) \|y_{\text{máx}} - y_{\text{mín}}\| = 1{,}8$. Así L podría ser escogida tal que $L < 1{,}8$, en esta aplicación se selecciona $L = 1{,}5$. Fig. 7.9 muestra el resultado del agrupamiento en línea para datos de dos meses. Aquí "*" representa el centro de cada grupo, "+" es el límite entre los grupos. Existen 11 grupos en $0 < t \leq 100$.

2. Segundo, se usa *support vector machines* (3.11) para obtener los support vectors como en (3.13). El número de support vectors para los 11 grupos son 3,3,3,4,3,3,6,4,2,3 y 1. Así se tienen 35 reglas como en (3.51). Los centros de las funciones Gausianas de esas reglas son las posiciones de los support vector.

3. Ahora se usan los datos en cada grupo para actualizar las funciones de membresía.

4. Cuando se combinan los 11 modelos difusos locales (35 reglas) dentro de un modelo global Takagi-Sugeno da la forma

$$\widehat{y} = \left(\sum_{i=1}^{11} f_i\left[x\left(k\right)\right] \left[\prod_{j=1}^{35} \left(\mu_{A_i^j} \right) \right] \right) / \left(\sum_{i=1}^{11} \left[\prod_{j=1}^{35} \left(\mu_{A_i^j} \right) \right] \right)$$

y aplicando la metodología de SVMF se obtine la siguiente figura 7.10 El resultado

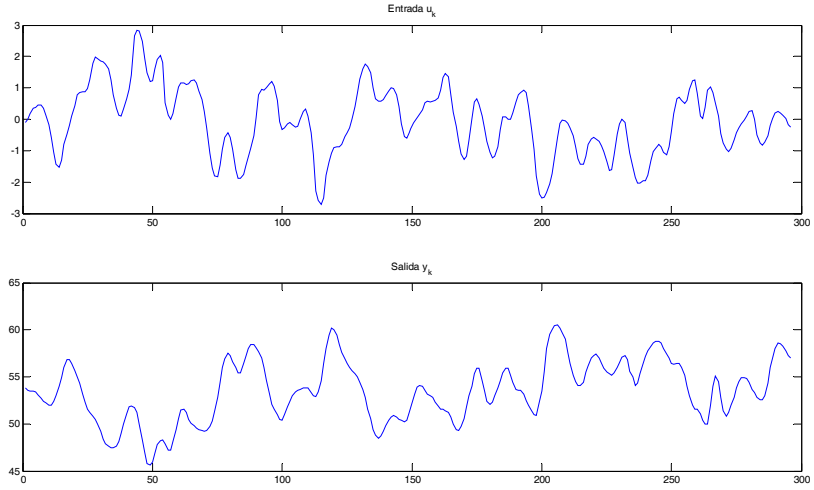

Figura 7.8: Conjunto de datos entrada/salida.

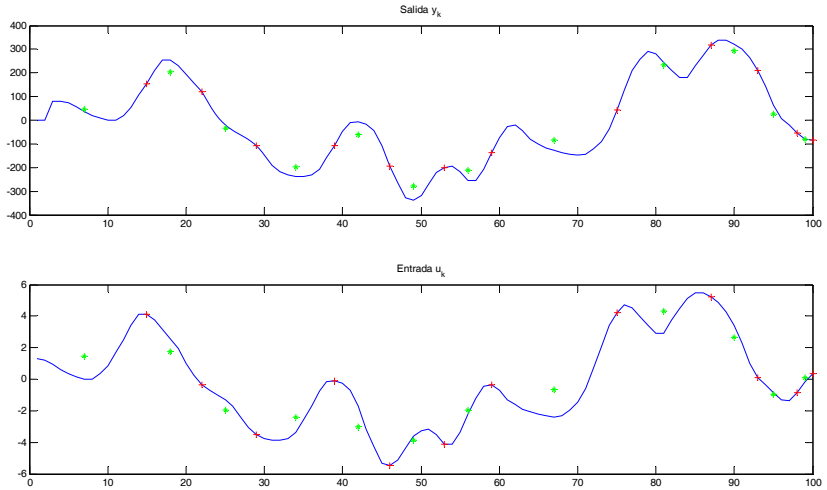

Figura 7.9: Agrupamiento en línea.

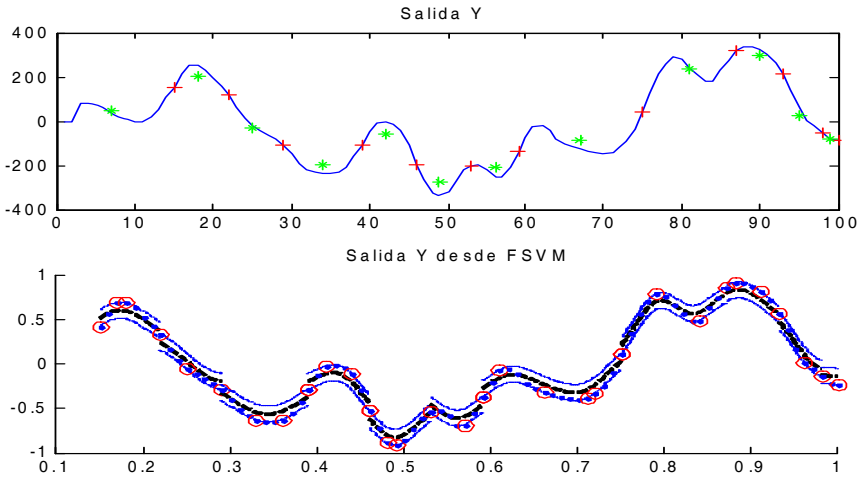

Figura 7.10: SVMF.

después de usar SVMF son mostrados en la siguiente tabla

Grupo	# de sv	Posición	# de datos
1	3	1, 3, 4	8
2	3	1, 4, 8	8
3	3	5, 8, 11	11
4	4	1, 3, 6, 8	8
5	3	1, 3, 4	8
6	3	1, 5, 7	7
7	6	1, 3, 8, 13, 14, 17	17
8	4	1, 5, 10, 13	13
9	2	2, 5	7
10	3	1, 4, 6	6
11	1	3	3

Tabla 1: SVM para función de estimación.

Después del entrenamiento de las funciones de membresía, las 35 funciones de mem-

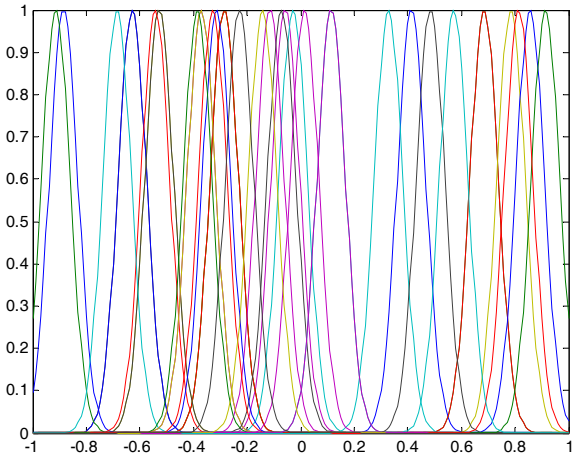

Figura 7.11: Funciones de membresia de x_1 en las 35 reglas difusas.

bresía para x_1 se muestran en la Figura 7.11.Entonces los parámetros fijados de las funciones de membresía, usan otros 296 datos para la prueba del modelo. El dato probado es $x_1(k) = 1 - \frac{2k}{T}$, $x_2(k) = -1 + \frac{2k}{T}$, $k = 1, 2, \cdots, T$. El resultado final del modelado despues de los modelos locales ($p = 35$) se muestran en la Figura 7.12

7.4. Modelado via multi redes neuronales

1. En este ejemplo se construye un modelo (conjunto de datos) para un horno de gas [3]. El conjunto de datos contiene 296 pares de puntos entrada/salida, donde la entrada u_k fue la entrada de gas codificada y la salida y_k representa la concentración de CO_2 del horno de gas. El conjunto de datos entrada/salida se muestra en la Fig.7.13. El vector de entrada esta definido por $x_k = [y_{k-1}\ y_{k-2}\ y_{k-3}\ u_{k-1}\ u_{k-2}\ u_{k-3}]^T$. Las muestras impares $[y_k, u_k]$ fueron usadas para el entrenamiento mientras las muestras iguales fueron omitidas para probar el modelo construido.Primeramente, se usa agrupamiento en línea. Se selecciona $\alpha = 0{,}4$. como conocido a priori, se conocen los cambios máximos en la en-

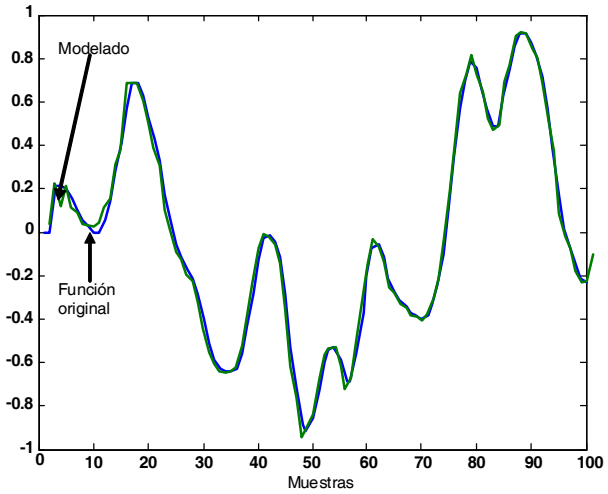

Figura 7.12: Resultado Final.

trada y la salida son alrededor de 3 y 1, $\alpha \|\mathbf{x}_{\text{máx}} - \mathbf{x}_{\text{mín}}\| + (1-\alpha) \|y_{\text{máx}} - y_{\text{mín}}\| = 1{,}8$. Así L podría ser escogida tal que $L < 1{,}8$, en esta aplicación se selecciona $L = 1{,}5$. Fig. 7.14 muestra el resultado del agrupamiento en línea para datos de dos meses. Aquí "*" representa el centro de cada grupo, "+" es el límite entre los grupos. Existen 11 grupos en $0 < t \leq 100$.

2. Segundo, se usa *support vector machines* (3.11) para obtener los support vectors como en (3.13). El número de support vectors para los 11 grupos son 3,3,3,3,3,3,4,6,3,3 y 1. Así se tienen 35 reglas como en (3.51). Los centros de las funciones Gausianas de esas reglas son las posiciones de los support vector.

3. Ahora se usan los datos en cada grupo para actualizar las funciones de membresía.

4. Cuando se combinan los 11 modelos difusos locales (35 reglas) dentro de un modelo

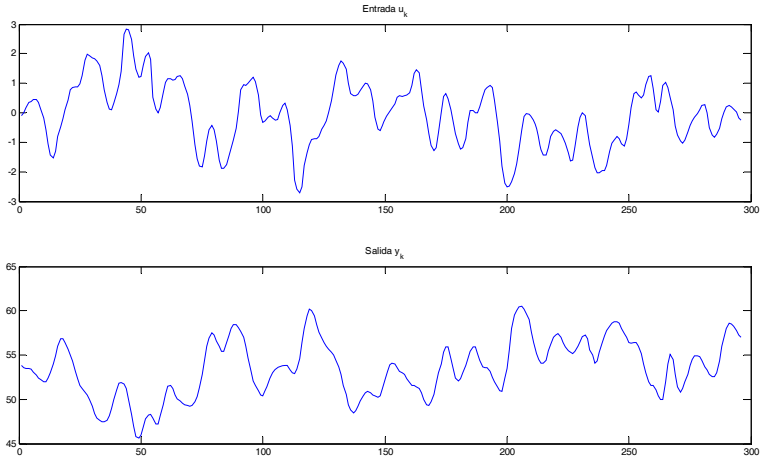

Figura 7.13: Conjunto de datos entrada/salida.

global Takagi-Sugeno da la forma

$$\widehat{y} = \left(\sum_{i=1}^{11} f_i \left[x\left(k\right) \right] \left[\prod_{j=1}^{35} \left(\mu_{A_i^j} \right) \right] \right) / \left(\sum_{i=1}^{11} \left[\prod_{j=1}^{35} \left(\mu_{A_i^j} \right) \right] \right)$$

El resultado después de usar la estructura automática son mostrados en la siguiente

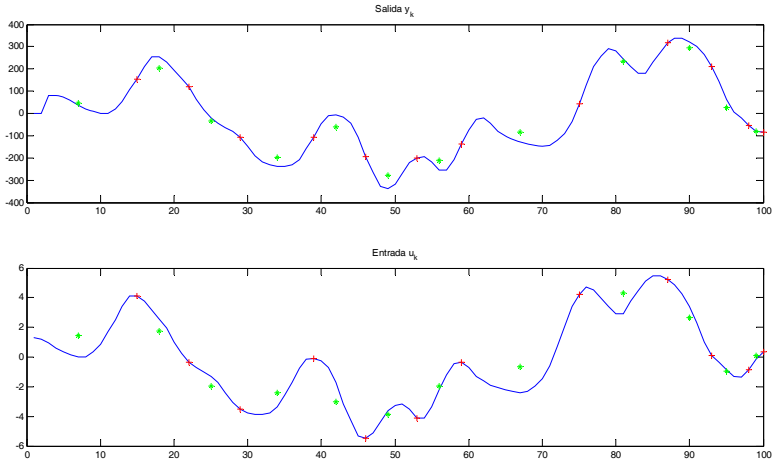

Figura 7.14: Agrupamiento en línea.

tabla

Grupo	# de sv	Posición	# de datos
1	3	1, 3, 7	8
2	3	1, 4, 8	8
3	3	1, 8, 11	11
4	3	1, 6, 8	8
5	3	1, 3, 7	8
6	3	2, 5, 7	7
7	4	1, 3, 7, 17	17
8	6	2, 5, 6, 9, 10, 12	13
9	3	2, 4, 7	7
10	3	1, 4, 6	6
11	1	3	3

Tabla 1: SVM para función de estimación.

Después del entrenamiento de las funciones de membresía, las 35 funciones de membresía para x_1 se muestran en la Figura 7.15y aplicando la metodología de multi redes

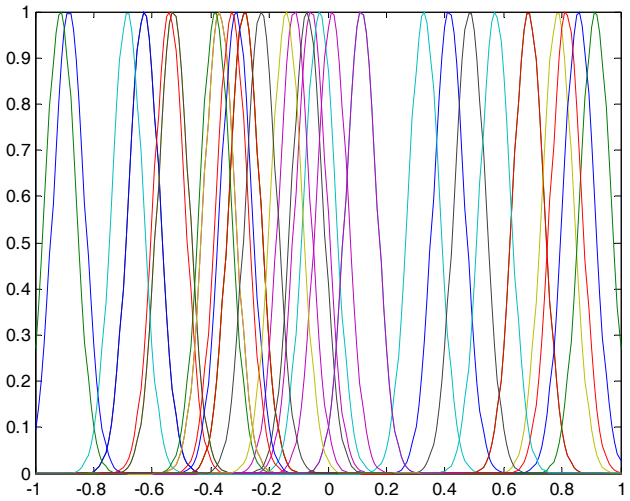

Figura 7.15: Funciones de membresia de x_1 en las 35 reglas difusas.

neuronales se obtiene la siguiente figura 7.17 y 7.16

7.5. Resultado experimental via SVM con selección de estructura automática

1. En este ejemplo se construye un modelo (conjunto de datos) para un horno de gas [3]. El conjunto de datos contiene 296 pares de puntos entrada/salida, donde la entrada u_k fue la entrada de gas codificada y la salida y_k representa la concentración de CO_2 del horno de gas. El conjunto de datos entrada/salida se muestra en la Fig.7.18. El vector de entrada esta definido por $x_k = [y_{k-1}\ y_{k-2}\ y_{k-3}\ u_{k-1}\ u_{k-2}\ u_{k-3}]^{\ T}$. Las muestras impares $[y_k, u_k]$ fueron usadas para el entrenamiento mientras las muestras iguales fueron omitidas para probar el modelo construido. Primeramente, se usa agrupamiento en línea. Se selecciona $\alpha = 0{,}4$. como conocido a priori, se conocen los cambios máximos en la entrada y la salida son alrededor de 3 y 1, $\alpha \left\|\mathbf{x}_{\text{máx}} - \mathbf{x}_{\text{mín}}\right\| + (1-\alpha) \left\|y_{\text{máx}} - y_{\text{mín}}\right\| = 1{,}8$.

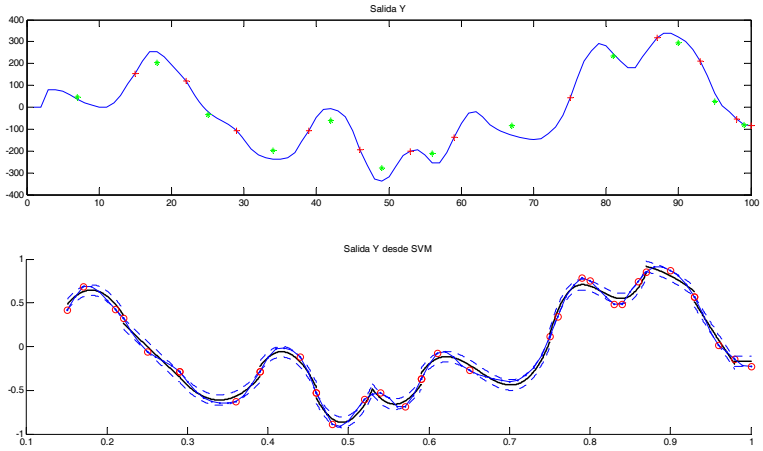

Figura 7.16: Agrupamiento en línea.

Así L podría ser escogida tal que $L < 1{,}8$, en esta aplicación se selecciona $L = 1{,}5$. Fig. 7.19 muestra el resultado del agrupamiento en línea para datos de dos meses. Aquí "*" representa el centro de cada grupo, "+" es el límite entre los grupos. Existen 11 grupos en $0 < t \leq 100$.

2. Cuando se combinan los 11 modelos difusos locales (35 reglas) dentro de un modelo global Takagi-Sugeno da la forma

$$\widehat{y} = \left(\sum_{i=1}^{11} f_i \left[x\left(k\right) \right] \left[\prod_{j=1}^{35} \left(\mu_{A_i^j} \right) \right] \right) / \left(\sum_{i=1}^{11} \left[\prod_{j=1}^{35} \left(\mu_{A_i^j} \right) \right] \right)$$

El resultado después de usar la estructura automática son mostrados en la siguiente

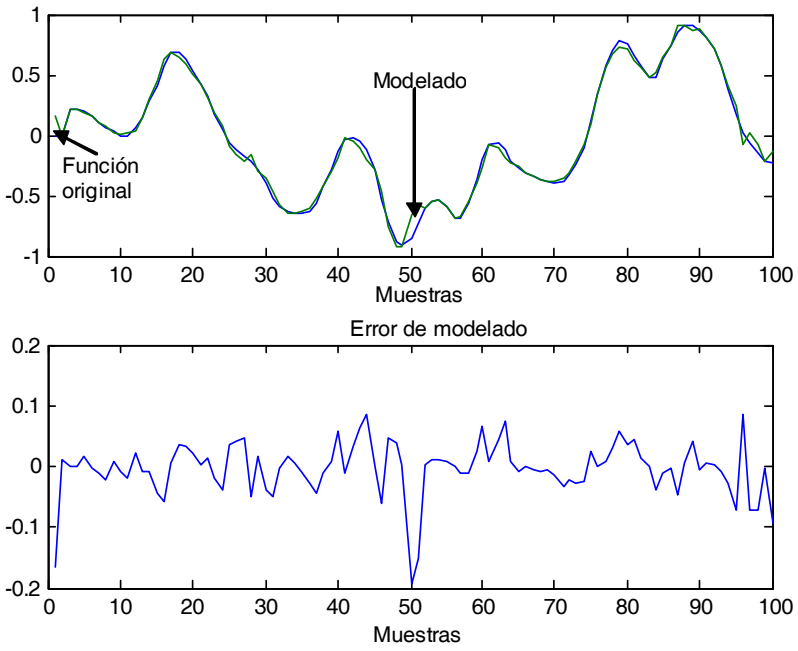

Figura 7.17: Multi redes neuronales tipo RBF.

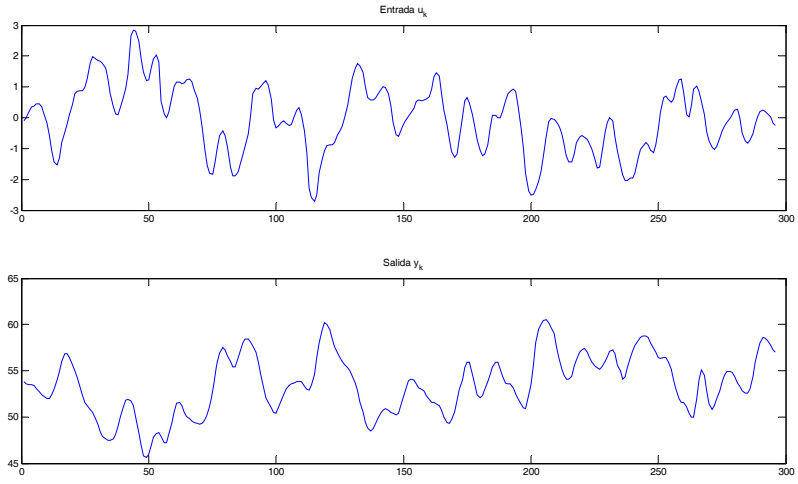

Figura 7.18: Conjunto de datos entrada/salida.

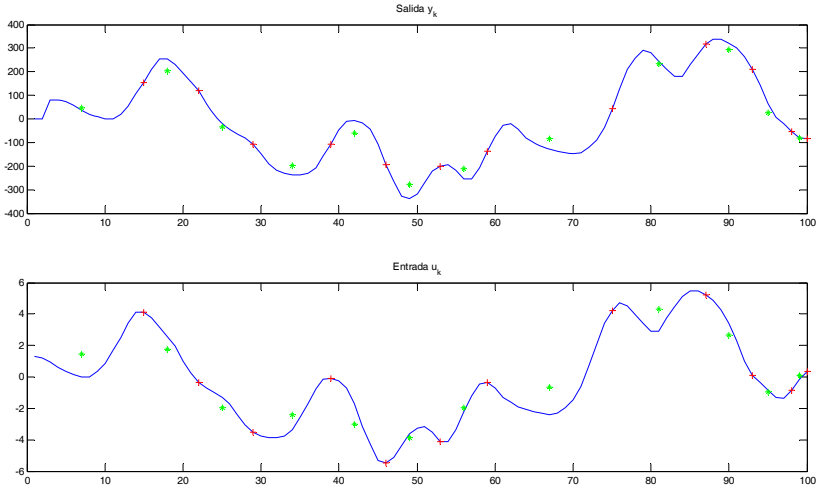

Figura 7.19: Agrupamiento en línea.

tabla

Grupo	# de sv	Posición	# de datos
1	3	$1, 3, 7$	8
2	3	$1, 4, 8$	8
3	3	$1, 8, 11$	11
4	3	$1, 6, 8$	8
5	3	$1, 3, 7$	8
6	3	$2, 5, 7$	7
7	4	$1, 3, 7, 17$	17
8	6	$2, 5, 6, 9, 10, 12$	13
9	3	$2, 4, 7$	7
10	3	$1, 4, 6$	6
11	1	3	3

Tabla 1: SVM para función de estimación.

Después del entrenamiento de las funciones de membresía, las 35 funciones de membresía para x_1 se muestran en la Figura 7.20y utilizando el enfoque multimodelo se obtienen los siguientes resultadosy

7.6. Comparación de modelos

La siguiente Tabla presenta las comparaciones concretas entre los 4 diferentes modelos. Los 4 métodos de modelado difuso usados, son simples si se conocen las reglas difusas. El agrupamiento en línea con SVM puede cambiar el modelo en línea. Los algoritmos utilizados son más complejos porque se usa la optimización en SVM, así el tiempo computacional es mucho más grande y la exactitud es mayor. Aunque el modelado difuso adaptable es muy simple, pero su estructura (regla difusa) es conocida, así el error de prueba es mucho mayor. En la figura 7.23 se ilustra los diferentes errores que se obtuvieron aplicando las diferentes metodología.

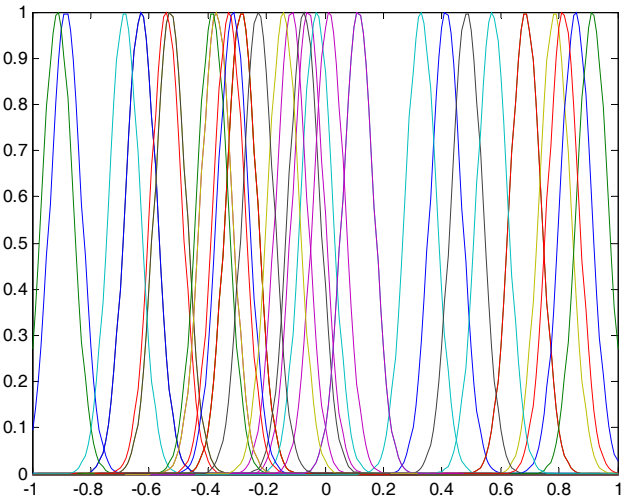

Figura 7.20: Funciones de membresia de x_1 en las 35 reglas difusas.

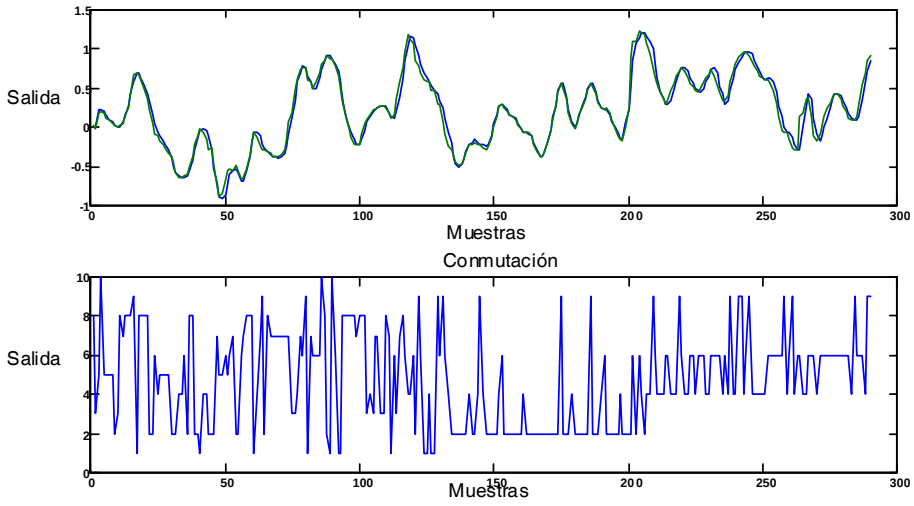

Figura 7.21: Multimodelo con selección de estructura automática no supervisada.

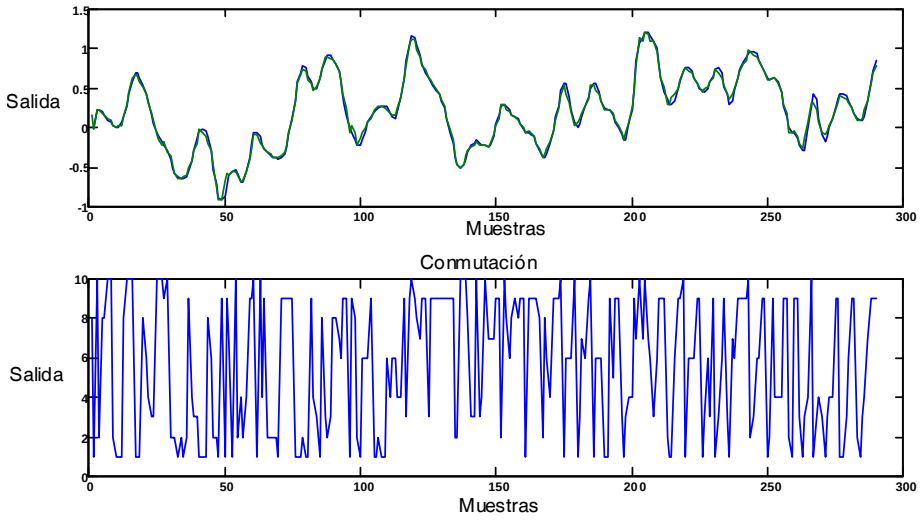

Figura 7.22: Multimodelo con selección de estructura automática supervisada.

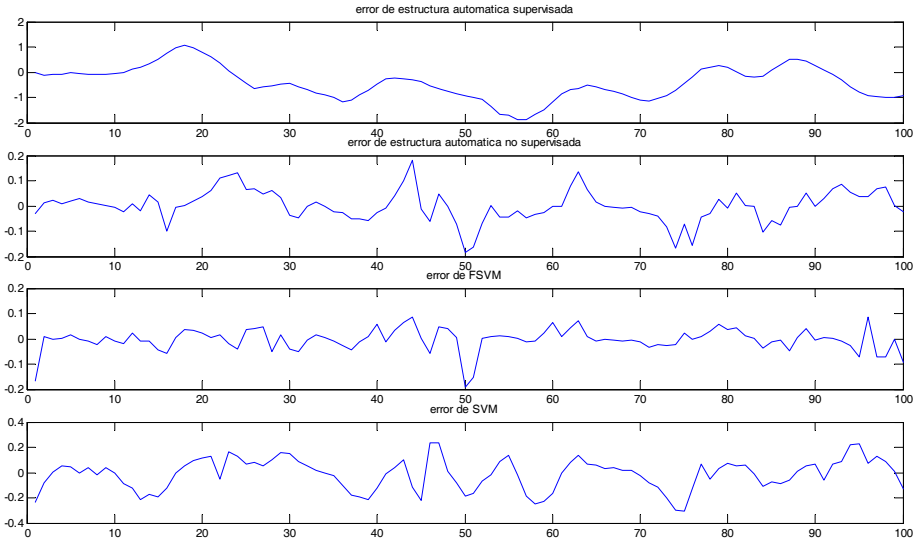

Figura 7.23: Comparasión de errores de los diferentes métodos.

	SVM Normal	SVMF	Sel. Est. Aut. no Sup.	Sel. Est. Aut. Sup.
# de datos	100	100	100	100
valor α	0.4	0.4	0.4	0.4
valor L	1.5	1.5	1.5	1.5
# de grupos	11	11	11	11
# de SV	45	45	45	45
parámetro	adaptable	adaptable	adaptable	adaptable
en línea	sí	sí	sí	sí

Tabla 1. Comparación entre métodos para el modelado.

Capítulo 8

Conclusiones

- En el capítulo 3 se propone el agrupamiento en línea que consiste en dividir los datos entrada/salida dentro de grupos en el mismo intervalo de tiempo, cuando la distancia de un punto al centro del grupo es menor que una longitud establecida, se dice que el punto esta dentro del grupo, cuando un nuevo dato llega, el centro y el grupo son cambiados de acuerdo al nuevo dato. Una vez que se obtuvieron los grupos se aplica el enfoque SVM para obtener los support vector para cada grupo, con los support vectors se construirán las reglas difusas y los correspondientes sistemas difusos para finalmente construirlas funciones de membresía de la predicción y la consecuencia para los datos de cada grupo.

- El capítulo 4 propone un método de *support vector machines* difuso (FSVM) que trata los datos entrenados del capítulo 3 para el proceso de entrenamiento modificando el kernel el cual debe satisfacer la condición de Mercer, por ejemplo existen varios tipos de kernels algunos de ellos son el kernel lineal, el kernel MPL, kernel RBF. En este capítulo se usa el kernel difuso que se puede calcular usando el conjunto entrenado, las funciones más populares para la función de membresía de sistemas difusos son la función Gausiana y la función triángulo, en este capítulo se propone la función de membresía en forma de campana.

- En el capítulo 5 se propone el modelado via multimodelo de redes neuronales tipo RBF el cual tiene una capa oculta que es no lineal y la capa de salida es lineal, aquí los support vectors servirán para construir cada red neuronal tipo RBF para cada grupo para después obtener los parámetros de identificación. El enfoque dentro del problema de SVM´s es el aprendizaje de la red, el objetivo será ajustar los parámetros de la red minimizando el margen de separación entre las dos clases. La técnica de multimodelo se utilizará para afrontar el problema de cambio de región de operación y la razón por la cual se escogió este enfoque fue para diseñar esquemas de identificación capaces de modelar la dinámica de una planta no lineal utilizando los enfoques de support vector y reglas difusas. Se sabe que los SVM´s separan los datos en dos clases dentro de un hiperplano cuyo margen es máximo. Una vez que se obtuvieron los modelados difusos y las funciones de membresía se toma la decisión de la estructura de conmutación (cambio) esta dirigido por el monitoreo del índice de desempeño. Para prevenir un cambio rápido arbitrariamente debido a las perturbaciones, un algoritmo de decisión de histérisis es necesario.

- En el capítulo 6, se observo que en el proceso de extraccion de reglas difusas para el modelado de sistemas no lineales es llamdo estructura de identificacións. A la combinación de multimodelo y redes neuronales difusas se les considera como un enfoque racional para identificación de sistemas no lineales. En este capítulo, se muestra un rango de aprendizaje variante en el tiempo para uso común del algoritmo retro-propagación (backpropagation), se prueba que los errores de identificación están acotados, el cual cambiará el modelo de la red neuro difusa solo cuando las funciones de membresía son casi iguales.

- El capítulo 7 muestra un ejemplo real de un horno de gas combinando los enfoques de los SVM, FSVM, redes neuronales, lógica difusa y multi modelo. Se puede observar que existen varias diferencias entre el SVM normal y el FSVM propuesto en este trabajo, y en enfoque multimodelo con selección de estructura automática y modelado vía multi redes neuronales, los errores son más pequeños que en el enfoque SVM normal y

FSVM. Se obeserva que el error del enfoque multimodelo con estructura de supervisión automática es mucho menor que en los otros enfoques, ya que se utiliza la misma función y valores de parámetros en los 4 casos.

8.1. Trabajo futuro

- Como trabajo futuro se pretende aplicar leyes de control a este tipo de enfoque utilizado en la elaboración de la tesis.

Bibliografía

[1] P.Angelov, *An approach for fuzzy rule-base adaptation using online clustering*, International Journal of Approximate Reasoning, Vol.35, No.3,275-289, 2004.

[2] B.E. Boser, I.M. Guyon, and V.N. Vapnik, *An training algori-ésimom for optimal margin classifier*, in Proc. 5-ésimo ACM Workshop Computacional Learning -ésimoeory, Pittsburg, PA, July 1993, pp 144-152.

[3] G. E. P. Box and G. M. Jenkins, *Time Series Analysis, Forecasting and Control*. San Francisco, CA: Holden Day Inc., 1976.

[4] Christopher J.C. Burges. *A tutorial on Support Vector Machines for Pattern Recognition.*Data Mining and knowledge Discovery, 2, 121-167. 1998.

[5] M. Brown and C. J. GHarris, *Neurofyzzy Adaptive Modeling and Control*, Upper Sadle River, NJ: Prentice-HAll, 1994.

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

[7] S.M.Chen, *Fuzzy Backward Reasoning Using Fuzzy Petri Nets*, IEEE Trans. SMC-Part B, Vol 30, No. 6, Dec. 2000.

[8] S. Chen, S. Billings, and P. Grant, *Recursive hybrid algori-ésimom for nonlinear system identification using radial basis function networks*, Int. J. Control, vol. 55, pp. 1051–1070, 1992.

[9] M.Y. Chen and D.A. Linkensm, *A systematic neuro-fuzzy modeling framework wi-ésimo application to material property prediction*, IEEE Trans. Syst. Man, Vyber. B, vol. 31, pp 781-790, Sept. 2001.

[10] J.-H.Chiang, P.-Y.Hao, *Suuport Vector Learning Mechanism for Fuzzy Rule-Based Modeling: A New Approach*, IEEE Transactions on Fuzzy Systems, Vol. 12, No. 1, February 2004.

[11] Chinrungrueng, C. and Séquin, C.H., *Optimal adaptive k-means algoritm wi-ésimo dynamic adjustment of learning*, IEEE Trans. on Neural Networks, 6, pp. 157-169, 1995.

[12] S.L.Chiu, *Fuzzy Model Identification based on cluster estimation*, Journal of Intelligent and Fuzzy Systems, Vol.2, No.3, 1994.

[13] Wei Chu, Chong Jin Ong, and S. Sa-ésimoiya Keerti. *An Improveded Conjugate Gradiant Scheme to -ésimoe Solution of Least Squares SVM.*IEEE TRANSACTION ON NEURONAL NETWORKS,Vol 16, No. 2, March 2005.

[14] T. Cover, *Geometrical and statistical properties of systems of linear inequalites*, IEEE Trans. Electronic Computers, vol. 14, pp 326-334, 1965.

[15] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines:* Cambridge Univ. Press, 2000.

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

[17] Eikens, B., and Karim.N., *Process identification wi-ésimo multiple neural network models*, International Journal of Control, 72(7/8), 576-590, 1999.

[18] Hai-Tao He, Nan Li, *An RBF Network Approach to Flatnnes Pattern Recognition based on SVM Learning*, Procecding of -ésimoe Fif-ésimo Internacional Conference on Machine Learning and Cybernectics, Dalian, pp 2959-2962, 13-16 August 2006.

[19] S. Haykin, *Neural Networks-A Comprehensive Foundations*. New York:MacMillan, 1994.

[20] S.I. Horikawa, T. Furuhashi, and Y. Uchikawa, *On fuzzy modeling using fuzzy neural networks wi-ésimo -ésimoe back-propagation algori-ésimom*, IEEE Trans. Neural Networks, vol. 3, pp. 801-806, Sept. 1993.

[21] G.F. Hughes, *On -ésimoe mean accuracy of statistical pattern recognizers*, IEEE Trans. Inform. -ésimoeory, vol. 14, no. 1, pp 55-63, 1968.

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

[23] Przemyslaw Janit and Tadeusz Lobos, *Automated Classification of Power-Quality Disturbamces Using SVM and RBF Networks*, IEEE Transactions on Power Delivery, Vol. 21, No. 3, pp 1663-1669, Jul 7 2006.

[24] Johansen, T.A., and Foss, B.A., *Multiple model approaches to modelling and control,*International Journal of Control, 72(7/8), pp. 575, 1999.

[25] M.I.Jordan and R.A.Jacobs, *Hierarchical mixtures of experts and -ésimoe em algori-ésimom,* Neural Computation, vol. 6, pp. 181–214, 1994.

[26] C.F. Juang, *Combination of on-line clustering and Q-value based GA for reinforcement fuzzy system desing*, IEEE Trans. Fuzzy Syst., Vol. 13, No.3, 289-302, 2005.

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

[28] Jie Zhou, Le-ping Xin, Gang Rong, David Zhang. *Decision fusion based cartridge identification using Support Vector Machine*. IEEE 2000.

[29] José Luis Rojo-Álvarez, Manel Martínez-Ramon, Mario de Prado-Cumplido, Antonio Artés-Rodríguez, Aníbal R. Figueiras-Vidal.*Support Vector Me-ésimood for Robust ARMA System Identification*. IEEE Transactions on Signal Processing, Vol 52, No. 1, January 2004.

[30] J. J. Rubio and Wen Yu, *Modelling of crude oil blending via discrete-time neural networks*, International Conference on Electrical and Electronics Engineering, Acapulco, México, 427-432, 2004.

[31] J. J. Rubio, *Modelación y Optimización del Mezclado de Petróleo Crudo con Redes Neuronales*, Tesis para obtener el grado de maestro en ciencias, CINVESTAV, Febrero del 2004.

[32] José de Jesús Rubio, Wen Yu, Andrés Ferreyra, *Neural Netwoks training with optimal bounded ellipsoid algorithm,* Neural Computing and Applications, ISSN: 0941-0643, September 2008.

[33] J. J. Rubio Avila, A. Ferreyra, C. Avilés Cruz, I. Vazquez, *Clustering Algorithm for Nonlinear System Identification*, WSEAS Transactions on Computers, ISSN: 1109-2750, Vol. 7, Issue 8, August 2008.

[34] J. S. Juang, *ANFIS: Adaptive-network-based fuzzy inference system*, IEEE Transactions on Systems, Man and Cybernetics, Vol. 23, 665–685, 1993.

[35] C.F. Juang, *A TSK-type recurrent fuzzy network for dinamyc systems processing by neural network and genetic algori-ésimom*, IEEE Trans. Fuzzy Syst., vol. 10, pp. 155-170, Apr. 2002.

[36] C.F.Juang, *Combination of on-line clustering and Q-value based GA for reinforcement fuzzy system design*, IEEE Transactions on Fuzzy Systems, Vol.13, No.3, 289- 302, 2005

[37] Jung-Hsien Chiang, Pei-Yi Hao. *Support Vector Learning Mechanism for Fuzzy Rule-Based Modeling: A New Approach.* IEEE Transactions on Fuzzy Systems, Vol. 12, No. 1, February 2004.

[38] N. Kasabov and Q. Song, *DENFIS: Dynamic evolving neural-fuzzy inference system and its application for time-series prediction*, IEEE Trans. Fuzzy Syst., vol 10, no. 2, 144-155, 2002.

[39] V. Kecman, *Support Vector Machines, Neural Networks and Fuzzy Logic Models*, Cambridge. MA:MIT Press, 2001.

[40] B.Kikens and M. Karim, *Process identification wi-ésimo multiple neural network models*, Int. J. Control, vol. 72, pp. 576–590, 1999.

[41] Klaus-Robert Mueller, Sebastian Mika, Gunnar Rasch, Koji Tsuda, y Bernhar Scholkopf. *An Introduction to Kernel-Based Learning Algori-ésimom.* IEEE TRANSACTION ON NEURONAL NETWORKS, Vol. 12, No. 2, March 2001.

[42] Kohonen, *Self Organizing Maps*, Heidelberg: Springler, 1995.

[43] Jacek M. Leski, *TSK-Fuzzy Modeling Based on ε-Insensitive Learning*, IEEE Trans. on Fuzzy System, vol. 13, no. 2, pp181-193, Apr 2005.

[44] C.-C. Lee, *Fuzzy logical in control systems: Fuzzy logic controller, part I*, IEEE Trans. Syst., Man, Cybern., vol. 20, pp. 404-418, Mar./Apr., 1990.

[45] C-Y.Lee and J-J.Lee, *Adaptive control for uncertain nonlinear systems based on multiple neural networks,* IEEE Trans. Syst. Man and Cybern. B, vol. 34, pp. 325–333, 2004

[46] 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, Cyber. B, vol 29, pp. 583-591, June, 1999.

[47] F.L. Lewis, A. Yesildirek and K. Liu, *Multilayer neural-net robot controller wi-ésimo guaranteed tracking performance*, IEEE Trans. Neural Networks, vol. 7, pp. 338-399, MAr. 1996.

[48] G. Lightbody and G. W. Irwin, *Nonlinear control structures based on embedded neural system models,* IEEE Transactions on Neural Networks, vol. 8, no. 3, pp. 553–567, 1997.

[49] C.-T. Lin and C-.S.G. Lee, *Neuralnetwork-based fuzzy logic control and decision system*, IEEE Trans. Comput., vol 40 pp. 1320-1336, Dic. 1991.

[50] C.T. Lin and G. Lee, *Neural Fuzzy Systems: A Neural-Fuzzy Synergism to Intelligent System*. Upper Saddle River, NJ: Prentice-Hall, 1996.

[51] D.P. Mandic, A.I. Hanna and M. Razaz, *A normalized gradient adaptive step size*, IEEE Signal Processing Lett,. vol 8, pp. 295-297, Feb 2001.

[52] E.H. Mamdani, *Application of fuzzy algori-ésimoms for control of simple dynamic plant*, Proc. Inst. Elect. Eng., vol 121, no. 12, pp. 1585-1588, 1976.

[53] Ming-Guang Zhang, Wei-Wu Yan, Zhan-Ting Yuan, *Study of nonlinear system identification based on support vector machine*, Procceding of -ésimoe -ésimoird International Conference on Machine and Cybernectics, Shanghai, 26-29 August 2004, pp 3287-3290.

[54] S.Mitra adn Y.Hayashi, *Neuro–fuzzy rule generation: survey in soft computing framework*, IEEE Transactions on Neural Networks, Vol.11, No.3, 748-769, 2000.

[55] A. Morse, D. Mayne, and G. Goodwin, *Applications of hysteresis switching in parameter adaptive control*, IEEE Trans. on Automatic Control, vol. 37, no. 9, pp. 1343-1354, Sept 1992.

[56] Klaus-Robert Mueller, Sebastian Mika, Gunnar Rasch, Koji Tsuda, y Bernhar Scholkopf, *An Introduction to Kernel-Based Learning Algori-ésimom.* IEEE TRANSACTION ON NEURONAL NETWORKS, Vol. 12, No. 2, March 2001.

[57] 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.

[58] Narendra, K. S. and Balkrishnan, *Performance Improvement in Adaptive Control Systems Using Multiple Models and Switching*, Proc. Seven-ésimo Yale Workshop on Adaptive and Learning Systems, Center for Systems Science, Yale University, New Haven, CT,

[59] K. S. Narenda and X. Cheng, *Adaptive control of discrete-time systems using multiple models*, IEEE Transactions on Neural Networks, vol. 8, no. 3, pp. 475-485, 1997.

[60] Stanislaw Osowski, Tomasz Markiewicz, *OLS Versus SVM Approach to Learning of RBF Networks*, Proceedings of International Joint Conference on Neural Networks, Montreal, Canada, pp 1051-1056, July 31- August 4, 2005.

[61] D. Prokhorov and D. Wunsch, *Adaptive critic designs*, IEEE Transactions. on Neural Networks, vol. 8, no. 5, pp. 997–1007, Sept 1997.

[62] I.Rivals and L.Personnaz, *Neural-network construction and selection in nonlinear modeling*, IEEE Transactions on Neural Networks, Vol.14, No.4, 804-820, 2003.

[63] S. Chen, X. X. Wang, and C. J. Harris, *NARX-Based Nonlinear System Identification Using Orthogonal Least Squares Basis Hunting*, IEEE Transactions on control systems technology, Vol. 16, No. 1, pp 78-84 January 2008.

[64] Shantanu Chakrabartty, Masakazu Yagi, Tadashia Shibata and Gert Cauwenberghs. *Robust Cephalometric Landmark Identification using Support Vector Machine.*ICASSP 2003.

[65] R.R.Selmic and F.L.Lewis, *Multimodel neural networks identification and failure detection of nonlinear systems*, in 40-ésimo IEEE Conference on Decision and Control, December 2001, pp. 3128 – 3133.

[66] T. Takagi and M. Sugeno, *Fuzzy identification of systems and its applications to medeling and control*, *IEEE Trans. Syst., Man. and Cybern.*, vol. 1, pp. 116-132, Jan. 1985.

[67] H.H. Tsai and P.T. Yu, *On -ésimoe optimal desing of fuuzy neural networks wi-ésimo robust learning for function approximation*, IEEE Trans. Syst., Man. Cyber. B, vol. 30, pp. 217-223, Apr. 2000.

[68] S.G.Tzafestas and K.C.Zikidis, *NeuroFAST: On-line neuro-fuzzy ART-based structure and parameter learning TSK model*, IEEE Transactions on Systems, Man and Cybernetics, Part B, Vol.31, No.5, 797-803, 2001

[69] Qing Song, Wenjie Hu, and Wenfang Xie. *Robust Support Vector Machine wi-ésimo Bullet Hole Image Classification.* IEEE TRANSACTION ON SYSTEMS, MAN AND CYBERNETICS.Vol 32, No 4, November 2002.

[70] V. Vapnik, *-ésimoe Nature of Statistical Learning -ésimoeory.* New York:Springer-Verlag, 1995.

[71] V. Vapnik, *Statistical learning -ésimoeory*, New York:Wiley, 1998.

[72] L.X.Wang, *Adaptive Fuzzy Systems and Control*, Englewood Cliffs NJ: Prentice-Hall, 1994.

[73] 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 wi-ésimo genetic algori-ésimom*, IEEE Trans. Syst., Man, Cybern. B, Vol.31, 467-475, 2001.

[74] W.Y. Wang, T.T. Lee, C.L. Liu, and C.H. Wang, *Function approximation using fuzzy neural networks wi-ésimo robust learning algori-ésimom*, IEEE Trans. Syst., Man, Cyber. B, vol. 27, pp. 740-747, Sept. 1997.

[75] W.Y. Wang, Y.G. Leu and C.C. Hsu, *Robust adaptive fuzzy-neural control of nonlinear dynamical system using generalized projection update law and variable structure controller*, IEEE Trans. Syst., Man, Cyber. B, vol. 31, pp. 140-147, 2001.

[76] Weigend, A.S., Mangeas, M. and Srivastava, A.N., *Nonlinear gated expertes for time series: discovering regime and avoiding overfittings*, International Journal of Neural Systems," 6, pp. 373-399, 1995.

[77] P. Werbos, *Approximate dynamic programming for real-time control and neural modeling*, Handbook of Intelligent Control. White and Sofge, 1992.

[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, pp. 358-364, 2000.

[79] Yongjie Zhai, Ruixin Li, DongfengWang, Pu Han. *Risk Function Based Fuzzy Judge SVM Algori-ésimom and its applications.*IEEE 2003.

[80] Wen Yu, Xiaoou Li, *Online clustering for nonlinear system modeling via fuzzy neural networks,*

[81] W.Yu, X.Li, *Some New Results on System Identification wi-ésimo Dynamic Neural Networks,* IEEE Transactions on Neural Networks, Vol.12, No.2, 412-417, 2001.

[82] W.Yu and X. Li, *Some stability properties of dynamic neural networks*, IEEE Trans. Circuits and Systems, Part I, Vol.48, No.1, 256-259, 2001.

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

[84] L.A. Zadeh, *Fuzzy sets*. Inf. Control, vol 8, pp 338-353, Aug, 1998.

[85] Daisuke Zenke. *Algori-ésimoms for -ésimoe Suppoet Vector Machine by -ésimoe Pa-ésimo Following Me-ésimood.* Doctoral Program in Policy Planning Sciences. University of Tsukuba. January 2003.

AGRADECIMIENTO

Agradezco al Consejo Nacional de Ciencia y Tecnología el apoyo brindado para cursar los estudios de Doctorado en el Centro de Investigación y de Estudios Avanzados del IPN, a través de la beca que me fue concedida durante el transcurso del mismo.

Así mismo a mis padres María Cristina Rodríguez y Luciano Tovar quienes me dieron su apoyo, a mi hermana Maricarmen Tovar, y a mis amigos con los cuales pases grandes momentos. A los cuales les dedico esta tesis.

También a mi asesor el Dr. Wen Yu Liu por haberme dado la oportunidad de haber realizado este trabajo de tesis así como a mis sinodales por haberme brindado la oportunidad.

“Solo hay dos cosas infinitas: el universo y la estupidez humana, de lo primero no estoy seguro”. Albert Einstein.