Universidad de Murcia

Facultad de Informática

Sistemas de inferencia difusos basados en redes neuroadaptativas

Autor: Nicolás Bruno [1]

Murcia, 30 de marzo de 1999

[1] Trabajo financiado por el Programa de Cooperación Interuniversitaria E.AL.98

Indice general

Indice de Figuras

Capítulo 1

Conceptos básicos

La complejidad del mundo real se ve reflejada fundamentalmente en las áreas humanísticas, como por ejemplo sistemas sociales, económicos y biológicos. Esta complejidad hace muy difícil transferir los procedimientos y técnicas de análisis, modelado y control cuantitativos utilizados en áreas tradicionales para trabajar con este tipo de sistemas. En 1964, Lofti Zadeh resume este problema mediante el conocido principio de incompatibilidad [37] que establece que "mientras la complejidad de un sistema aumenta, la habilidad para llegar a sentencias precisas y significativas acerca de su comportamiento disminuye, hasta llegar a un límite más allá del cual la precisión y la relevancia se convierten en características mutuamente exclusivas". Zadeh introduce el concepto de conjuntos difusos como un medio para representar y manipular información que no es precisa.

Ejemplo 1 *Cuantos granos de arena componen una montaña? Ciertamente que un único grano no forma una montaña. Además, si una colección de granos no forman una montaña, el sólo hecho de agregar un grano no transforma la colección en una montaña. Siguiendo este razonamiento, se puede concluir que no es posible construir montañas de arena agregando granos uno por uno.*

Ejemplo 2 *Una persona muy alta puede definirse como aquella que mida más de dos metros. Esta definición clasificará como muy alto a un individuo que mida 2.001 metros, pero no a aquel que mida 1.999 metros.*

Las extrañas conclusiones que se derivan a partir de los ejemplos anteriores están basadas en el error de considerar completamente nítido el límite entre pertenecer y no pertenecer a un conjunto.

1.1 Conjuntos difusos

El concepto de conjunto difuso generaliza la idea del conjunto ordinario (o *crisp*, de aquí en adelante), donde la pertenencia se ve como una función sobre $\{V, F\}$. Los conjuntos difusos, en cambio, permiten describir conceptos en los que el límite entre tener una propiedad y no tenerla no es completamente nítido.

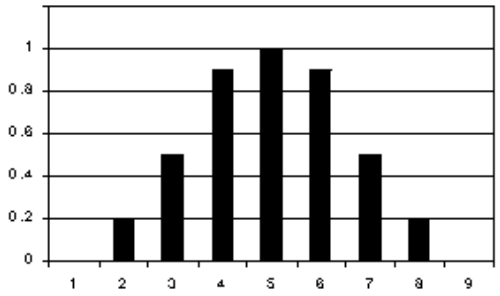

Figura 1.1: Conjunto difuso que representa la idea de "proximidad a 5"

Definición 1 *Sea X un conjunto crisp de referencia. Un* **conjunto difuso** A **sobre** X *queda definido por medio de una función característica*

$$\mu_A : X \to [0, 1]$$

Si $x \in X$, la expresión $\mu_A(x)$ se interpreta como el grado de membresía del elemento x en el conjunto difuso A. El conjunto X es llamado el **dominio de** A, *y se denota $dom(A) = X$.*

Para simplificar la notación, se puede escribir $A(x)$ en lugar de $\mu_A(x)$. Además, si el conjunto $X = \{x_1, \ldots, x_n\}$ es finito, se puede caracterizar al conjunto difuso A de la siguiente manera:

$$A = \{\mu_1/x_1, \ldots, \mu_n/x_n\}$$

donde el término μ_i/x_i expresa que $A(x_i) = \mu_i$. Esta notación puede emplearse también cuando el conjunto X sea infinito pero A posea un número finito de elementos con membresía no nula.

Ejemplo 3 *El conjunto de números naturales "cercanos a 5" puede definirse por medio del conjunto difuso sobre $\mathbb{N}$ de la figura 1.1.*

Ejemplo 4 *Un* **número difuso** *es un conjunto difuso sobre $\mathbb{R}$. Aunque en principio cualquier función de membresía es válida, existen familias de números difusos con funciones de membresía parametrizadas. Algunos ejemplos de estas funciones de membresía son:*

- *Funciones triangulares, con parámetros $\{a, b, c\}$, donde*

$$triang_{a,b,c}(x) = max\left[min\Big(\frac{x-a}{b-a}, \frac{c-x}{c-b}\Big), 0\right] \tag{1.1}$$

- *Funciones trapezoidales, con parámetros $\{a, b, c, d\}$, donde*

$$trap_{a,b,c,d}(x) = max\left[min\Big(\frac{x-a}{b-a}, 1, \frac{d-x}{d-c}\Big), 0\right] \tag{1.2}$$

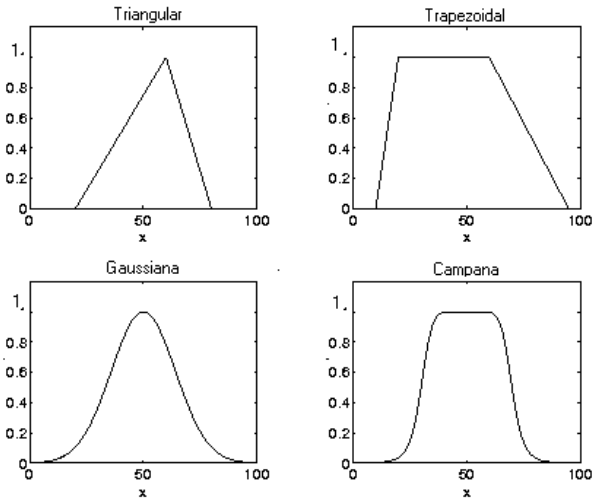

Figura 1.2: Funciones de membresía parametrizadas

- *Funciones gaussianas, con parámetros* $\{\sigma, c\}$*, donde*

$$gauss_{\sigma,c}(x) = e^{-\left(\frac{x-c}{\sigma}\right)^2} \tag{1.3}$$

- *Funciones campana, con parámetros* $\{a, b, c\}$*, donde*

$$bell_{a,b,c}(x) = \frac{1}{1 + \left|\frac{x-c}{a}\right|^{2b}} \tag{1.4}$$

En la figura 1.2 se pueden ver gráficamente las funciones $triang_{(20,60,80)}$, $trap_{(10,20,60,95)}$, $gauss_{(20,50)}$ *y* $bell_{(20,4,50)}$.

Definición 2 *Un conjunto difuso A sobre X es* **normal** *si existe al menos un elemento* $x \in X$ *tal que* $A(x) = 1$*. De otra forma, A es* **subnormal**.

Definición 3 *La* **altura** *de un conjunto difuso A sobre X se define como*

$$altura(A) = Max_{x \in X}[A(x)]$$

Definición 4 *Sea A un conjunto difuso sobre X. El* **soporte de A**, *denotado* **Sop(A)** *es el subconjunto crisp de X cuyos elementos tienen membresía no nula en A.*

$$Sop(A) = \{x \in X : A(x) > 0\}$$

Definición 5 *Sea A un conjunto difuso sobre X. El* **núcleo de A**, *denotado* **Nu(A)**, *es el subconjunto crisp de X cuyos elementos tienen membresía unitaria.*

$$Nu(A) = \{x \in X : A(x) = 1\}$$

Ejemplo 5 *El conjunto A del ejemplo 3 es normal, ya que* $A(5) = 1$*. Su soporte es el conjunto* $\{2, 3, 4, 5, 6, 7, 8\}$ *y su núcleo es el conjunto singletón* $\{5\}$.

Definición 6 *Sean A y B conjuntos difusos sobre X. Se dice que A* **es un subconjunto de** *B,*

$$A \subset B \Leftrightarrow A(x) \leq B(x)\ \forall x \in X$$

Definición 7 *Sean A y B subconjuntos difusos sobre X. A* **es igual a** *B,*

$$A = B \Leftrightarrow A \subset B \wedge B \subset A$$

Definición 8 *El* **conjunto difuso nulo sobre** *X, se denota $\emptyset_X$, o simplemente $\emptyset$. Su función de membresía es $\emptyset_X(x) = 0,\ \forall x \in X$. Por otro lado, el* **conjunto difuso universal sobre** *X se denota 1_X, o bien X, y queda caracterizado por la función de membresía $1_X(x) = 1,\ \forall x \in X$.*

Ejemplo 6 *Sea A el conjunto del ejemplo 3 y B sobre $\mathbb{N}$ el conjunto difuso $B = \{0.2/3, 0.5/5, 0.2/7\}$. Entonces se cumple que*

$$\emptyset_{\mathbb{N}} \subset B \subset A \subset 1_{\mathbb{N}}$$

1.2 Operaciones sobre conjuntos difusos

Las operaciones naturales definidas para conjuntos crisp pueden generalizarse para trabajar con conjuntos difusos. Naturalmente, existe además una gran cantidad de operadores nuevos, que no tienen correspondencia dentro de la teoría de conjuntos crisp.

1.2.1 Operadores de agregación

La unión e intersección de conjuntos crisp pueden verse desde un contexto más general como operaciones de agregación de conjuntos difusos [7]. Si A y B son dos conjuntos difusos sobre X, Se expresa la intersección de estos conjuntos como el conjunto difuso $A \cap B$ sobre X. De la misma forma la unión se expresa como el conjunto difuso $A \cup B$ sobre X. La primera consideración para definir estos operadores es que deben reducir a los operadores crisp cuando los conjuntos difusos tengan funciones de membresía sobre $\{0, 1\}$ (es decir, se correspondan con conjuntos crisp). Este requerimiento implica que los operadores deben definirse punto a punto, por lo que $(A \cup B)(x) = S(A(x), B(x))$ y $(A \cap B)(x) = T(A(x), B(x))$. Esto hace posible concentrarse en la estructura de las funciones S y T para la descripción de los operadores. Frecuentemente se utiliza la notación $a \vee b$ para denotar $S(a, b)$, y $a \wedge b$ para denotar $T(a, b)$.

Existen ciertas condiciones que permiten caracterizar a S y T, que a su vez definen la clase general de operadores de unión e intersección.

Definición 9 *Un operador $T : [0, 1]^2 \to [0, 1]$ es un operador* **T-norm** *si:*

1) $T(x, y) = T(y, x)$	*Conmutatividad*
2) $T(x, T(y, z)) = T(T(y, x), z)$	*Asociatividad*
3) $T(x, y) \geq T(x', y')$ *si* $x \geq x' \wedge y \geq y'$	*Monotonía*
4) $T(x, 1) = x$	*Elemento neutro*

Se ve que T reduce a la intersección clásica, ya que por (4) se cumple que $T(0, 1) = 0$ y $T(1, 1) = 1$. Por (1) se tiene que $T(1, 0) = T(0, 1) = 0$ y ésto junto a (3) resulta en T(0,0)=0.

Ejemplo 7 *Los operadores T-norm más frecuentemente utilizados son:*

Mínimo: $T(a,b) = min(a,b)$
Producto algebraico: $T(a,b) = ab$
Producto acotado: $T(a,b) = max(0, a+b-1)$
Producto drástico: $T(a,b) = \begin{cases} a & si\ b = 1 \\ b & si\ a = 1 \\ 0 & en\ otro\ caso \end{cases}$

Definición 10 *Un operador* $S : [0,1]^2 \to [0,1]$ *es un operador* **T-conorm** *si:*

1) $S(x,y) = T(y,x)$ *Conmutatividad*
2) $S(x, S(y,z)) = S(S(y,x), z)$ *Asociatividad*
3) $S(x,y) \geq S(x',y')$ *si* $x \geq x' \wedge y \geq y'$ *Monotonía*
4) $S(x,0) = x$ *Elemento neutro*

De la misma forma se observa que las condiciones anteriores se reducen a la unión clásica.

Ejemplo 8 *Los operadores T-conorm más frecuentemente utilizados son:*

Máximo: $S(a,b) = max(a,b)$
Suma algebraica: $S(a,b) = a + b - ab$
Suma acotada: $S(a,b) = min(1, a+b)$
Suma drástica: $S(a,b) = \begin{cases} a & si\ b = 0 \\ b & si\ a = 0 \\ 1 & en\ otro\ caso \end{cases}$

Definición 11 *Sea T un operador T-norm y S un operador T-conorm. Se definen los operadores de unión e intersección a partir de S y T respectivamente de la siguiente manera. Para todo conjunto crisp X,*

$$(A \cup_S B)(x) = S(A(x), B(x))$$

$$(A \cap_T B)(x) = T(A(x), B(x))$$

para todo $x \in X$ y todo par de conjuntos difusos A y B sobre X.

Es interesante observar que algunas propiedades de la teoría clásica de conjuntos no son válidas al trabajar con conjuntos difusos. En particular, la ley del tercero excluído ($A \cup \overline{A} = X$) y el principio de no contradicción ($A \cap \overline{A} = \emptyset$) no necesariamente se cumplen. Una idea clave en la teoría de conjuntos difusos es que si A es un concepto o idea, A será difuso cuando éste no se pueda distinguir claramente de su complemento.

Aunque a veces se pueden elegir operadores T-norm y T-conorm en forma arbitraria, frecuentemente la elección de estos operadores no es completamente independiente. Esta conexión se pone de manifiesto al introducir el operador de negación o complemento de conjuntos difusos.

Definición 12 *Un operador* $N : [0,1] \to [0,1]$ *es un operador* **de negación** *si:*

1) $N(1) = 0; N(0) = 1$ *Límite*
2) $N(x) \leq N(y) si\ x > y$ *Orden inverso*
3) $N(N(x)) = x$ *Involución*

Ejemplo 9 *Los operadores de negación más frecuentemente utilizados son:*

Negación clásica:	$N(x) = 1 - x$
Negación de Sugeno:	$N(x) = \frac{1-x}{1+sx} \quad s > -1$
Negación de Yager:	$N(x) = (1 - x^w)^{1/w} \quad w > 0$

La introducción de un operador de negación permite definir una ley general de De Morgan que relaciona operadores T-norm y T-conorm.

Lema 1 *Sea N un operador de negación y T un operador T-norm arbitrario. Entonces, el operador S, definido como*

$$S(x, y) = N(T(N(x), N(y)))$$

es un operador T-conorm, y se dice que es el **operador dual** *de T.*

La siguiente tabla resume los operadores duales más comunes cuando el operador de negación N elegido es el standard.

T-norm	**T-conorm**
Mínimo	Máximo
Producto algebraico	Suma algebraica
Producto acotado	Suma acotada
Producto drástico	Suma drástica

Existen muchas otras formas de definir estos operadores, e incluso en [18] se pueden encontrar familias parametrizadas de operadores duales.

Definición 13 *Los operadores* **standard** *de unión, intersección y complemento son el máximo, el mínimo y la negación clásica, respectivamente.*

A lo largo del texto, a menos que se indique lo contrario se utilizarán los operadores standard.

Ejemplo 10 *La figura 1.3 muestra el resultado de aplicar los operadores standard de unión, intersección y complemento sobre conjuntos difusos.*

1.2.2 Operadores de promedio

En muchos casos el tipo de agregación requerido entre conjuntos difusos requerido no es la conjunción pura de los operadores T-norm ni la disyunción pura de los operadores T-conorm, sino alguna mezcla entre estos dos extremos.

Definición 14 *Una función $G : [0,1]^n \to [0,1]$ es un* **operador de agregación promedio n-dimensional** *si satisface las siguientes condiciones:*

1. *Conmutatividad. El orden de los argumentos es indistinto.*
2. *Monotonía. $G(a_1, \ldots, a_n) \geq G(b_1, \ldots, b_n)$ si $a_i \geq b_i \ \forall i$*
3. *Idempotencia. $G(a^*, \ldots, a^*) = a^*$*

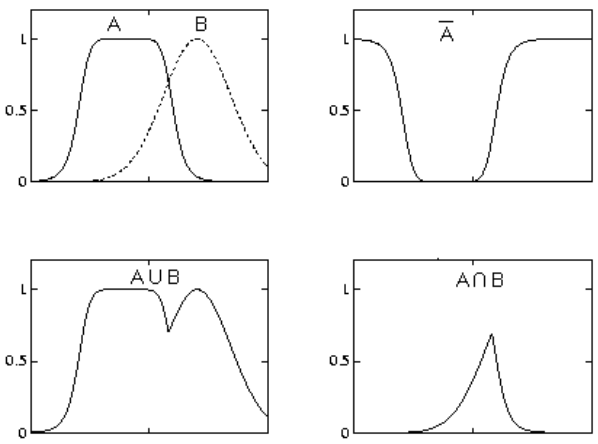

Figura 1.3: Operadores standard de unión, intersección y complemento

Una propiedad de los operadores de agregación promedio es que sus valores siempre se encuentran entre el mínimo y el máximo de sus argumentos. En efecto, si $a^* = Max_i(a_i)$, se tiene que $G(a_1, \ldots, a_n) \leq G(a^*, \ldots, a^*) = a^*$ (aplicando primero monotonía y luego idempotencia). De forma similar, si $a^* = Min_i(a_i)$, se tiene que $G(a_1, \ldots, a_n) \geq a^*$.

Ejemplo 11 *Una familia de operadores de agregación promedio generalizada se define como*

$$G_\alpha(a_1, \ldots, a_n) = \left[\frac{a_1^\alpha + \ldots + a_n^\alpha}{n}\right]^{1/\alpha} \quad \alpha \in (-\infty, \infty)$$

Es interesante notar que el operador G_α reduce a funciones conocidas para selecciones particulares de α.

- *Para $\alpha \to -\infty/\infty$ el operador se acerca a Min/Max, respectivamente,.*
- *Para $\alpha = 1$ se obtiene el promedio aritmético*

$$G_1(a_1, \ldots, a_n) = \frac{a_1 + \ldots + a_n}{n}$$

- *Para $\alpha = -1$ se obtiene la media armónica:*

$$G_{-1}(a_1, \ldots, a_n) = \frac{n}{\frac{1}{a_1} + \ldots + \frac{1}{a_n}}$$

- *Para $\alpha \to 0$, G_α tiende a la media geométrica*

$$\lim_{\alpha \to 0} G_\alpha(a_1, \ldots, a_n) = (a_1 \ldots a_n)^{1/n}$$

Operadores OWA

La familia de operadores OWA [32] permite ajustar fácilmente el grado de conjunción y disyunción implícito en una agregación.

Definición 15 *Un* **operador OWA n-dimensional** *es un mapeo* $f : \mathbb{R}^n \to \mathbb{R}$ *asociado a un vector* $w = [w_1, \ldots, w_n]$ *donde se verifica:*

1. $w_i \in [0, 1]$
2. $\sum_i w_i = 1$
3. $f(a_1, \ldots, a_n) = \sum_i w_i b_i$
   donde b_i *es el i-ésimo elemento más grande entre los* a_i*.*

Cabe destacar que el ordenamiento implícito entre los a_i introduce una componente no lineal en el proceso de agregación.

Ejemplo 12 *Algunos operadores de agregación OWA conocidos son:*

- F^**, con* $w^* = [1, 0, \ldots, 0]$*, donde* $F^*(a_1, \ldots, a_n) = Max_i(a_i)$
- F_**, con* $w_* = [0, \ldots, 0, 1]$*, donde* $F_*(a_1, \ldots, a_n) = Min_i(a_i)$
- F_p*, con* $w_p = [1/n, \ldots, 1/n]$*, donde* $F_p(a_1, \ldots, a_n) = \frac{1}{n} \sum_{i=1}^{n} a_i$

Definición 16 *Sea* F *un operador OWA y* w *su vector asociado. La medida* **orness**$(F) : [0, 1]^n \to [0, 1]$*, se define como*

$$orness(F) = \frac{1}{n-1} \sum_{1}^{n} [(n - i)w_i]$$

El valor orness de un operador F mide cuan cercano está F del operador puro de disyunción. Puede verse que $orness(F^*) = 1$, $orness(F_*) = 0$, y $orness(F_p) = 0.5$. Similarmente, una medida **andness** se puede definir como

$$andness(F) = 1 - orness(F)$$

Teorema 1 *Sean* w *y* w' *dos vectores n-dimensionales OWA asociados a los operadores* F *y* F' *respectivamente, tales que:*

- $w_i = w'_i \ \forall \ i \neq \{j, k\}$
- $w_j = w'_j + \Delta$
- $w_k = w'_k - \Delta$

donde $\Delta > 0, j > k$*. Entonces* $orness(F) \geq orness(F')$

Este teorema, cuya demostración consiste en aplicar la definición 16, muestra que al mover pesos hacia índices menores en el vector w aumenta la medida *orness*, mientras que al hacerlo hacia índices mayores, aumenta la medida *andness*.

1.2.3 Otros operadores

Debido a que los conjuntos difusos son una generalización de los conjuntos crisp, existen algunos operadores sobre los primeros que no tienen un correlato entre los segundos. A continuación se definen los operadores más importantes:

Definición 17 *Sea A un conjunto difuso sobre X. Se define A^α como el conjunto difuso sobre X tal que:*

$$A^\alpha(x) = A(x)^\alpha \quad \forall x \in X$$

Se puede ver que si $\alpha > 1$, entonces $A^\alpha \subset A$, y la operación se denomina **dilatación**. Por otro lado, si $\alpha < 1$, entonces $A^\alpha \supset A$ y la operación se denomina **concentración**.

Definición 18 *Si A es un conjunto difuso sobre X, la* **intensificación** *de A, denotada $I(A)$, se define como*

$$[I(A)](x) = \begin{cases} A(x)^2 & si\ 0 \leq A(x) \leq 0.5 \\ A(x)^{\frac{1}{2}} & en\ otro\ caso. \end{cases}$$

Definición 19 *Sean A y B dos conjuntos difusos sobre X. La* **suma acotada entre** *A* **y** *B, denotada $A \oplus B$ se define como el conjunto difuso sobre X tal que*

$$(A \oplus B)(x) = Min[1, A(x) + B(x)] \ \forall x \in X$$

Definición 20 *Sea A un conjunto difuso sobre X y $\alpha \in [0,1]$. Se define el conjunto difuso αA sobre X como*

$$(\alpha A)(x) = \alpha(A(x)) \quad \forall x \in X$$

Definición 21 *Sea A un conjunto difuso sobre X y $\alpha \in [0,1]$. Se define el conjunto A_α como el subconjunto (crisp) de X que contiene todos los elementos de X para los cuales $A(x) \geq \alpha$,*

$$A_\alpha = \{x \in X : A(x) \geq \alpha\}$$

Estos conjuntos permiten representar un conjunto difuso utilizando (posiblemente infinitos) conjuntos crisp como se enuncia en el siguiente lema:

Lema 2 *Sea A un conjunto difuso sobre X. Entonces:*

$$A = \cup_{\alpha \in [0,1]} \alpha A_\alpha$$

1.3 Relaciones difusas

Muchas relaciones reales envuelven alguna imprecisión o graduación y no son estrictamente booleanas. Este tipo de relaciones son usualmente mejor expresadas como relaciones difusas.

Definición 22 *Sean $X_1, \ldots, X_n$ conjuntos crisp. Una* **relación difusa** *sobre $X_1, \ldots, X_n$ es un conjunto difuso sobre el producto cartesiano $X_1 \times \ldots \times X_n$.*

Definición 23 *Sean $A_1, \ldots, A_n$ conjuntos difusos sobre $X_1, \ldots, X_n$. El* **producto cartesiano (o cruzado)** *$A_1 \times \ldots \times A_n$ es una relación difusa T sobre $X_1 \times \ldots \times X_n$ definida como*

$$T(x_1, \ldots, x_n) = \wedge_i [A_i(x_i)]$$

Definición 24 *Sean $X = \{X_1, \ldots, X_n\}$, $Y = \{Y_1, \ldots, Y_n\}$ dos familias de conjuntos crisp (no necesariamente disjuntas), y sean las relaciones difusas R sobre X y S sobre Y. El* **join** *entre R y S, denotado $R \bowtie S$ es una relación difusa T sobre $X \cup Y$, donde para cada $\overline{z} \in dom(T)$,*

$$T(\overline{z}) = R(\overline{x}) \wedge S(\overline{y})$$

donde $\overline{x} \in dom(R_1)$, $\overline{y} \in dom(R_2)$, y sus valores coinciden con los de $\overline{z}$ en los dominios que tienen en común.

Ejemplo 13 *Sean $X_1 = \{a, b\}$, $X_2 = \{\alpha, \beta\}$ y $X_3 = \{1, 2\}$ conjuntos crisp y sean las relaciones difusas $A = \{0.3/(a,\alpha), 0.6/(a,\beta), 1/(b,\alpha), 0.2/(b,\beta)\}$ sobre $\{X_1, X_2\}$ y $B = \{0.2/(\alpha,1), 0/(\alpha,2), 0.8/(\beta,1), 0.8/(\beta,2)\}$ sobre $\{X_2, X_3\}$. Entonces, la relación difusa $C = A \bowtie B$ sobre $\{X_1, X_2, X_3\}$ está definida como:*

$$\begin{aligned} C = \{ \quad & 0.2/(a,\alpha,1), 0/(a,\alpha,2), 0.6/(a,\beta,1), 0.6/(a,\beta,2), \\ & 0.2/(b,\alpha,1), 0/(b,\alpha,2), 0.2/(b,\beta,1), 0.2/(b,\beta,2) \quad \} \end{aligned}$$

Definición 25 *Sean $X = \{X_1, \ldots, X_n\}$, $Y = \{Y_1, \ldots, Y_n\}$ dos familias de conjuntos crisp tal que $X \subset Y$, y sea A sobre X una relación difusa. La* **extensión cilíndrica de A a Y** *es una relación difusa $A^{[Y]}$ sobre Y, definida como*

$$A^{[Y]} = A \bowtie Y$$

Ejemplo 14 *Sea A la relación del ejemplo 13. Entonces*

$$\begin{aligned} A^{[X_3]} = \{ \quad & 0.3/(a,\alpha,1), \quad 0.3/(a,\alpha,2), \\ & 0.6/(a,\beta,1), \quad 0.6/(a,\beta,2), \\ & 1/(b,\alpha,1), \quad 1/(b,\alpha,2), \\ & 0.2/(b,\beta,1), \quad 0.2/(b,\beta,2) \quad \} \end{aligned}$$

Definición 26 *Sean $X = \{X_1, \ldots, X_n\}$, $Y = \{Y_1, \ldots, Y_n\}$ dos familias de conjuntos crisp (no necesariamente disjuntas), y sea R sobre X una relación difusa. La* **proyección de R sobre Y**, *denotada $\Pi_Y(R)$ es una relación difusa sobre Y, donde para todo $\overline{z} \in dom(T)$,*

$$(\Pi_Y(R))(\overline{z}) = \vee_{\overline{x} \in Q} [R(\overline{x})]$$

donde Q es el conjunto de todos los $\overline{x} \in dom(R)$ que coinciden con $\overline{z}$ en los dominios que tienen en común. Por definición, si $X \cap Y = \emptyset$, entonces $\vee_{\overline{x} \in Q} [R(\overline{x})] = 1$ y $\Pi_Y(R) = Y$.

Ejemplo 15 *Sean A, B y C las relaciones del ejemplo 13. Entonces*

$$\Pi_A(C) = \{0.2/(a,\alpha), 0.6/(a,\beta), 0.2/(b,\alpha), 0.2/(b,\beta)\}$$

$$\Pi_B(C) = \{0.2/(\alpha,1), 0/(\alpha,2), 0.6/(\beta,1), 0.6/(\beta,2)\}$$

El concepto tradicional de inclusión se puede generalizar a relaciones difusas arbitrarias, como se ve a continuación.

Definición 27 *Sean $X = \{X_1, \ldots, X_n\}$, $Y = \{Y_1, \ldots, Y_n\}$ dos familias de conjuntos crisp (no necesariamente disjuntas), y sean las relaciones difusas R sobre X y S sobre Y. Se dice que R contiene a S, denotado $S \subset R$ si las extensiones cilíndricas de R y S a $X \cup Y$ cumplen $S^{[X \cup Y]} \subset R^{[X \cup Y]}$, es decir*

$$S^{[X \cup Y]}(\overline{z}) \leq R^{[X \cup Y]}(\overline{z}) \ \forall \overline{z}$$

Ejemplo 16 *Sea A la relación del ejemplo 13 y sea D la relación difusa sobre $\{X_2, X_3\}$ definida como*

$$D = \{0.2/(\alpha, 1), 0.3/(\alpha, 2), 0.1/(\beta, 1)\}, 0.2/(\beta, 2)\}$$

Se verifica que $D \subset A$.

Lema 3 *La relación $\subset$ es reflexiva y transitiva.*

Lema 4 *Sean P, Q relaciones difusas. Se cumple que $P \bowtie Q \subset P$.*

Demostración: Sea P definida en $X = \{X_1, \ldots, X_m\}$ y Q en $Y = \{Y_1, \ldots, Y_n\}$. Sea además $Z = X \cup Y$. Entonces, $\forall\ \overline{z} \in Z$

$$(P \bowtie Q)^{[Z]}(\overline{z}) = (P \bowtie Q)(\overline{z}) = P(\overline{x}) \wedge Q(\overline{y}) \leq P(\overline{x}) = P^{[Z]}(\overline{x})$$

donde $\overline{x} \in X$, $\overline{y} \in Y$ coinciden con $\overline{z}$ en los dominios que tienen en común.

El siguiente lema liga las definiciones de *join* y *contención* y puede ser demostrado de manera similar.

Lema 5 *Sean P, A, B relaciones difusas. Se verifica que:*

$$(P \subset A) \wedge (P \subset B) \Rightarrow P \subset A \bowtie B$$

1.4 Principio de extensión

El principio de extensión juega un papel fundamental al permitir extender funciones definidas sobre conjuntos crisp a funciones sobre conjuntos difusos. Una importante aplicación del principio de extensión es un mecanismo para operar aritméticamente con números difusos.

Definición 28 *Sean $X_1, \ldots, X_n, Y$ conjuntos crisp y sea f una función arbitraria, $f : X_1 \times \ldots \times X_n \to Y$. Si $A_1, \ldots, A_n$ son conjuntos difusos sobre $X_1, \ldots, X_n$, se define $f(A_1, \ldots, A_n)$ como el conjunto difuso sobre Y que verifica:*

$$f(A_1, \ldots, A_n) = \cup_{(x_1, \ldots, x_n) \in (X_1 \times \ldots \times X_n)} \{[A_1(x_1) \wedge \ldots \wedge A_n(x_n)]/f(x_1, \ldots, x_n)\}$$

Si $B = f(A_1, \ldots, A_n)$, entonces B es el conjunto difuso sobre Y que cumple:

$$B(y) = \vee_{(x_1, \ldots, x_n) \in (X_1 \times \ldots \times X_n) : f(x_1, \ldots, x_n) = y} [A_1(x_1) \wedge \ldots \wedge A_n(x_n)]$$

Ejemplo 17 *Utilizando el principio de extensión se puede generalizar la operación* $+ : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ *para operar con conjuntos difusos de la siguiente manera: Si* A *y* B *son números difusos, entonces la suma* $C = A + B$ *es el número difuso*

$$C(z) = \vee_{(x,y)\in\mathbb{R}^2/x+y=z}[A(x) \wedge B(y)]$$

Ejemplo 18 *Sean los números (difusos)* $B = \{0.33/7, 0.66/8, 1/9, 0.5/10\}$ *y* $A = \{0.5/9, 1/10, 0.8/11, 0.6/12, 0.4/13, 0.2/14, \}$. *La suma* $C = A + B$ *obtenida mediante la aplicación del principio de extensión es*

$$C = \{0.33/16, 0.5/17, 0.66/18, 1/19, 0.8/20, 0.6/21, 0.5/22, 0.4/23, 0.2/24\}$$

Se puede ver que para universos finitos, la aritmética con números difusos involucra operaciones algebraicas similares a la convolución en los elementos del soporte. El subconjunto de números difusos L-R, definido en [8], provee una técnica simplificada y eficiente para operar aritméticamente.

1.5 Medidas de difusión

Como se menciona en la sección 1.2.1, el concepto de difusión de un conjunto está relacionado con la falta de distinción entre éste y su complemento. Por lo tanto, se puede definir una medida de difusión de un conjunto difuso en base a la distancia según alguna métrica entre éste y su complemento. Una medida de difusión FUZ debe satisfacer las siguientes condiciones [6]:

1. $FUZ(A) = 0$ si A es un conjunto crisp.
2. $FUZ(A)$ es máximo si $A(x) = \frac{1}{2}\ \forall x \in X$
3. $FUZ(A) \geq FUZ(A^*)$ si A^* satisface

   $A^*(x) \geq A(x)$ si $A(x) \geq \frac{1}{2}$

   $A^*(x) \leq A(x)$ en otro caso.

Si el conjunto base X es finito (de tamaño n), se puede considerar a cada conjunto difuso sobre X como un vector n-dimensional, y luego seleccionar alguna métrica para formular una medida de difusión.

Ejemplo 19 *Sea* $X = \{x_1, \ldots, x_n\}$ *un conjunto crisp y* A *sobre* X *un conjunto difuso. Se define una familia de métricas con parámetro* p *de la siguiente manera:*

$$D_p(A, \overline{A}) = \Big[\sum_{i=1}^{n} \big|A(x_i) - \overline{A}(x_i)\big|^p\Big]^{1/p} \quad p \in \mathbb{N}^+ \tag{1.5}$$

Sin embargo, ya que $\overline{A}(x) = 1 - A(x)$, *se tiene que*

$$D_p(A, \overline{A}) = \Big[\sum_{i=1}^{n} \big|2A(x_i) - 1\big|^p\Big]^{1/p} \quad p \in \mathbb{N}^+ \tag{1.6}$$

Basado en la ecuación 1.6, se define una clase de medidas de difusión FUZ_p*:*

$$FUZ_p(A) = 1 - \frac{D_p(A, \overline{A})}{n^{1/p}} \tag{1.7}$$

Para el caso de conjuntos X *infinitos, se pueden definir medidas de difusión reemplazando la sumatoria de la ecuación 1.6 por una integral.*

Capítulo 2

Teoría del razonamiento aproximado

La teoría del razonamiento aproximado, también conocida como teoría AR, está basada en la utilización de conjuntos difusos y proporciona un marco para razonar frente a información incierta. Esta teoría brinda un mecanismo para modelar y realizar inferencias a partir de relaciones funcionales imprecisas y forma la base de las técnicas de modelado de sistemas difusos.

2.1 Elementos de la teoría AR

La teoría AR representa sentencias lingüísticas mediante proposiciones, asignando conjuntos difusos como valores de las variables.

Definición 29 *Un* **sistema AR** *es un par* (W, X)*, donde* $W = \{W_1, \ldots, W_n\}$ *es un conjunto de variables atómicas y* $X = \{X_1, \ldots, X_n\}$ *es una familia de conjuntos crisp. Cada* X_i *se denomina el* **conjunto base** *de* W_i*, y se denota* $Base(W_i) = X_i$*.*

Definición 30 *Sea* $A = (W, X)$ *un sistema AR. Un* **vector de variables** V *es cualquier subconjunto de* W*. Una* **proposición P** *es una sentencia de la forma*

$$V \text{ is } M$$

donde $V = \{V_1, \ldots, V_k\}$ *es un vector de variables y* M *una relación difusa sobre* $Base(V_1) \times \ldots \times Base(V_k)$*.*

Definición 31 *Sea* $P = V$ *is* M *una proposición. Entonces* V *es el* **conjunto de variables de** P *y* M *es el* **valor de** P*. Se utiliza la notación:*

$$Var(P) = V$$

$$Val(P) = M$$

Definición 32 *Una proposición* $P = W$ *is* M *es* **inconsistente** *si* $M = \emptyset$*, es decir* $altura(M) = 0$*. Si* $altura(M) = 1$ *se dice que* P *es* **consistente**.

Definición 33 *Una proposición W is M es una tautología si*

$$M(x) = 1 \ \forall x \in Dom(M)$$

Lema 6 *Sean P una relación difusa y T una tautología. Se cumple que $P \subset T$.*

Demostración: Sea P definida sobre $X = \{X_1, \ldots, X_m\}$ y T definida sobre $Y = \{Y_1, \ldots, Y_n\}$. Sea además $Z = X \cup Y$. Entonces, $\forall\ \overline{x} \in X, \overline{y} \in Y, \overline{z} \in Z$, tales que $\overline{x}$ e $\overline{y}$ coinciden con $\overline{z}$ en los dominios que tienen en común, se verifica:

$$P^{[Z]}(\overline{z}) = P(\overline{x}) \leq 1 = T(\overline{y}) = T^{[Z]}(\overline{z})$$

2.2 Semántica de la teoría AR

Cuando V_i es una variable atómica, el conjunto $Base(V_i)$ puede verse como el conjunto de todos los valores que V_i puede tomar. La proposición V_i *is* A_i significa que el valor de V_i es alguno de los elementos de A_i con membresía no nula. Con esta semántica, el valor de la proposición $P_1 \bowtie P_2$ indica los posibles valores en los que P_1 y P_2 coinciden.

Bajo esta interpretación, se puede ver que para una proposición V *is* A, mientras menor sea el conjunto A, más representativo será con respecto al valor real de V, ya que habrá menos valores posibles y la indeterminación se verá reducida. Sin embargo, si A se hace demasiado pequeño (subnormal) se comienza a perder información y la base de conocimiento se vuelve inconsistente.

2.2.1 Reglas de traducción

En [37] se presenta un conjunto de reglas de traducción que permiten representar algunas sentencias lingüísticas en términos de proposiciones del sistema AR. A continuación se describen algunas de esas reglas:

Negación: La sentencia "no (V *is* A)" se traduce en la proposición "V *is* $\overline{A}$".

Conjunción: La sentencia "(V_A *is* A) y (V_B *is* B)" es transformada en la proposición "$V_A \cup V_B$ *is* $A \bowtie B$"

Disyunción: La sentencia "(V_A *is* A) o (V_B *is* B)" se reemplaza por la sentencia "no [(no V_A *is* A) y (no V_B *is* B)]" y luego ésta se traduce.

Implicación: La sentencia "Si (V_A *is* A), entonces (V_B *is* B)" se traduce en la proposición ($V_A \cup V_B$) *is* C, donde C puede ser definido de varias maneras. Las dos formas más comúnmente utilizadas son las siguientes:

1. $C = \overline{A^{[B]}} \cup B^{[A]}$
2. $C(z) = Min[1, 1 - A^{[B]}(z) + B^{[A]}(z)], \ \forall\ z \in Dom(C)$.

2.3 Deducción en AR

El mecanismo de deducción utilizado en la teoría de razonamiento aproximado es el procedimiento básico para inferir resultados partiendo de una colección de proposiciones o premisas, utilizando para ello únicamente dos reglas de inferencia:

- Regla de inferencia 1 (IR-1):

$$\frac{V_A \text{ is } A \qquad A \subset B}{V_B \text{ is } B}$$

  Esta regla está basada en el significado semántico de las proposiciones en AR y refleja el hecho que si el valor de una variable pertenece a un conjunto debe también pertenecer a cualquier otro que lo contenga.

- Regla de inferencia 2 (IR-2):

$$\frac{V_A \text{ is } A \qquad V_B \text{ is } B}{(V_A \cup V_B) \text{ is } A \bowtie B}$$

Definición 34 *Una deducción a partir de un conjunto de premisas $P_1, \ldots, P_k$ en AR es una secuencia de proposiciones $B_1, \ldots, B_n$, donde cada B_i es alguna de las siguientes:*

1. *Una premisa P_j.*
2. *Una tautología.*
3. *El resultado de aplicar IR-1 a una proposición B_j, con $j < i$.*
4. *El resultado de aplicar IR-2 a proposiciones B_{i_1}, B_{i_2}, con $i_1, i_2 < i$.*

Si existe una deducción desde las premisas $(P_1, \ldots, P_k)$ que termina en la proposición B, se dice que B es inferible desde $P_1, \ldots P_k$, y se denota

$$(P_1, \ldots, P_k) \vdash B$$

Definición 35 *Sea B una proposición.*

$$B \text{ es un } \textbf{teorema} \text{ en } AR \Leftrightarrow \emptyset \vdash B$$

Lema 7 *La aplicación de IR-1 e IR-2 a tautologías da otra tautología.*

Demostración: Aplicando las definiciones.

Teorema 2 *Los únicos teoremas de AR son tautologías.*

Demostración: Se prueba por inducción que en una deducción $B_1, \ldots, B_n$, cada B_i es una tautología. Para comenzar, B_1 es una tautología porque debe ser obtenida a partir de la opción 2 de la definición 34. Además, si $B_1, \ldots, B_{n-1}$ son tautologías, B_n también debe serlo por el lema 7.

El siguiente resultado es muy importante ya que caracteriza a todas las proposiciones inferibles a partir de una colección de premisas.

Teorema 3 $(P_1, \ldots, P_k) \vdash B \Leftrightarrow P_1 \bowtie \ldots \bowtie P_n \subset B$

Demostración:
($\Leftarrow$) Sea la siguiente deducción

$B_i = P_i$	Premisas $(i = 1 \ldots k)$
$B_{k+1} = B_1 \bowtie B_2$	IR-2
$B_{k+j} = B_{k+j-1} \bowtie B_{j+1}$	IR-2 $(j = 2 \ldots k-1)$
$B_{2k} = B$	IR-1 a B_{2k-1} (por hipótesis)

ya que por construcción, $B_{2k-1} = P_1 \bowtie \ldots \bowtie P_k$.
($\Rightarrow$) Sea $P = P_1 \bowtie \ldots \bowtie P_k$. Se prueba por inducción en la longitud de la deducción que $P \subset B_i$. Para comenzar, como B_1 es una premisa o una tautología, se tiene que $P \subset B_1$ por lemas 4 y 6. Además, supóngase $B_1, \ldots, B_{i-1}$ contienen a P. Entonces B_i puede obtenerse de cuatro maneras (definición 34). Las dos primeras opciones se prueban de manera similar que para el caso base. Para la opción 3 se utiliza la transitividad de la relación $\subset$, y para la opción 4 se utiliza el lema 5.

2.3.1 Soluciones mínimas

Sean $P_1, \ldots, P_k$ una colección de premisas y sea V un vector de variables. Si se puede inferir V *is* A desde las premisas, entonces para cualquier $B \subset A$, también puede ser inferido V *is* B, por medio de la regla IR-1. Un problema interesante es encontrar, dado un vector de variables V y un conjunto de premisas $P_1, \ldots, P_k$, una proposición V *is* G inferible desde $P_1, \ldots, P_k$ con la propiedad que cualquier otra proposición inferida V *is* H cumpla que $G \subset H$. Es decir, encontrar conjuntos mínimos de soluciones para un vector de variables dado.

Lema 8 *Sean* $X = \{X_1, \ldots, X_n\}$, $Y = \{Y_1, \ldots, Y_n\}$ *dos familias de conjuntos crisp (no necesariamente disjuntas), y sea* R *sobre* X *una relación difusa. Se cumple que*

$$R \subset \Pi_Y(R)$$

Demostración: Sea $Z = X \cap Y$. Se ve que $\forall\, \overline{x} \in X, \overline{y} \in Y, \overline{z} \in Z$, tales que $\overline{x}, \overline{y}$ coinciden con $\overline{z}$ en los dominios que tienen en común,

$$[\Pi_Y(R)]^{[Z]}(\overline{z}) = [\Pi_Y(R)](\overline{y}) = Max_Q M(\overline{x}) \geq M(\overline{x}) = M^{[Z]}(\overline{z})$$

donde Q es el conjunto de $\overline{x} \in X$ que coinciden con $\overline{y}$ en los dominios que tienen en común.
Corolario: Sean $P_1, \ldots, P_n$ una colección de premisas que verifican que $(P_1, \ldots, P_n) \vdash V$ *is* R. Entonces, para cualquier vector de variables V, se cumple que

$$(P_1, \ldots, P_n) \vdash V \text{ } is \text{ } \Pi_V(R)$$

Teorema 4 *Sean* $P_1, \ldots, P_k$ *premisas. Entonces para toda proposición* V *is* N

$$(P_1, \ldots, P_k) \vdash V \text{ } is \text{ } N \Leftrightarrow \Pi_V(P_1 \bowtie \ldots \bowtie P_k) \subset N$$

Demostración:
($\Leftarrow$) Sea $P = P_1 \bowtie \ldots \bowtie P_k$. Se sabe que $(P_1, \ldots, P_k) \vdash P$. Por lema 8 se tiene que $(P_1, \ldots, P_k) \vdash \Pi_V(P)$. Como por hipótesis $\Pi_V(P) \subset N$, se tiene que $(P_1, \ldots, P_k) \vdash V \ is \ N$.
($\Rightarrow$) Supóngase $\Pi_V(P) \not\subset N$. Esto significa que existe $\overline{x} \in Dom(N)$ tal que $N(\overline{x}) < [\Pi_V(P)](\overline{x})$. Pero por la definición de proyección, existe un punto $\overline{p} \in P$ que coincide con $\overline{x}$ en los dominios que tienen en común, y además verifica que $P(\overline{p}) = [\Pi_V(p)](\overline{x}) > N(\overline{x})$. Esto implica que $P \not\subset N$, y por el teorema 3, $P \not\vdash V \ is \ N$ en contra de la hipótesis.

Definición 36 *Sea $P_1, \ldots, P_k$ premisas y V un vector de variables. El* **valor minimal de** *V se define como*

$$minimal(V) = \Pi_V(P_1 \bowtie \ldots \bowtie P_k)$$

A continuación se presenta un resultado que complica el proceso de obtención de proposiciones minimales. En particular, no permite encontrar los valores minimales de todas las variables atómicas y combinarlos para construir los valores minimales de vectores de variables arbitrarios.

Teorema 5 *Sean V_A, V_B dos vectores de variables y sean $M_A = minimal(V_A)$ y $M_B = minimal(V_B)$. Sean, además, $F = M_A \bowtie M_B$ y $M = minimal(V_A \cup V_B)$. Entonces:*

1. $M \subset F$
2. $F \not\subset M$

Demostración: La primera propiedad se demuestra por medio de la regla de deducción IR-2. Para demostrar la segunda se utiliza el siguiente contraejemplo, donde por brevedad, se omiten los valores de membresía de los elementos:

Sea la única premisa $P = \{V_1, V_2\}$ *is* A, con $A = \{(a,1),(b,2)\}$. Se verifica que $minimal(\{V_1, V_2\}) = A$. Además, se ve que $minimal(V1) = \{a, b\}$ y $minimal(V_2) = \{1, 2\}$. Pero, $minimal(V1) \bowtie minimal(V2)$ es igual al conjunto $\{(a,1),(a,2),(b,1),(b,2)\}$, el cual es distinto de A.

Sin embargo, la proyección de conjuntos minimales sobre subconjuntos de variables conserva la condición de minimalidad como se demuestra en el siguiente teorema.

Teorema 6 *Sea $P_1, \ldots, P_n$ un conjunto de premisas y V un vector de variables tales que $minimal(V) = M$. Sea $V_A \subset V$. Entonces*

$$minimal(V_A) = \Pi_{V_A}(M)$$

Demostración: Supóngase por el absurdo que existe una proposición $V_A \ is \ R$, tal que $(P_1, \ldots, P_n) \vdash V_A \ is \ R$, y además $\Pi_{V_A}(M) \not\subset R$. Se tiene entonces que

$$\exists\, \overline{m} \in dom(M) \ : \ \Pi_{V_A}(M)(\overline{m}) > R(\overline{m})$$

Además, se cumple que $(P_1, \ldots, P_n) \vdash V \ is \ R^{[V]}$ por la regla de deducción IR-1. Por último, por la definición de proyección, existe un $\overline{n} \in (V - V_A)$ tal que

$$\Pi_{V_A}(M)(\overline{m}) = M(\overline{m}, \overline{n}) > R^{[V]}(\overline{m}, \overline{n}) = R(\overline{m})$$

Se tiene entonces que $M \not\subset R^{[V]}$, contradiciendo de ese modo la hipótesis de minimalidad de M.

Capítulo 3

La defusificación

El problema de la defusificación aparece como una etapa importante en los mecanismos de decisión basados en conjuntos difusos. En estos casos se dispone de un conjunto de alternativas $Y = \{y_i, \ldots, y_n\}$ y un conjunto difuso A sobre Y indicando el grado en que cada alternativa satisface el objetivo buscado. Un mecanismo de defusificación define la estrategia que guía, a partir de A, la selección de un elemento y^* representativo del conjunto Y.

Ejemplo 20 *Los métodos de defusificación más comúnmente utilizados son el COA(center of area) y el MOM (mean of maxima). El método COA calcula la salida y^{COA} de la siguiente manera:*

$$y^{COA} = \frac{\sum_{i=1}^{n} y_i A(y_i)}{\sum_{i=1}^{n} A(y_i)} \tag{3.1}$$

El método MOM calcula la salida y^{MOM} como el promedio entre los elementos de Y con el máximo valor de membresía:

$$y^{MOM} = \frac{1}{|Y'|} \sum_{y_i \in Y'} y_i \quad \text{con } Y' = Y_{altura(Y)} \tag{3.2}$$

3.1 El proceso de defusificación

La defusificación de un conjunto difuso puede ser vista como un proceso en dos etapas. En la primera, el conjunto difuso se convierte en una distribución de probabilidad, y luego (en la segunda etapa) tiene lugar la selección real, basada en la distribución de probabilidad obtenida. Existen diversas formas de realizar cada etapa y cada una da lugar a un mecanismo diferente de defusificación.

3.1.1 Distribuciones de probabilidad

En esta etapa se debe transformar un conjunto difuso A sobre Y en una distribución de probabilidad P sobre Y, donde $P(y_i)$ indica la probabilidad de seleccionar posteriormente al elemento y_i como el representante del conjunto Y.

Esta transformación se puede definir de una gran cantidad de maneras distintas, siempre y cuando se verifiquen las siguientes condiciones:

1) $P(y_i) = P(y_j)$ si $A(y_i) = A(y_j)$ *Identidad*
2) $P(y_i) \geq P(y_j)$ si $A(y_i) > A(y_j)$ *Monotonía*

Ejemplo 21 *Se presentan tres procedimientos para generar la distribución de probabilidad P a partir de un conjunto difuso A:*

T1 $P(y_i) = \begin{cases} 0 & si\ y_i \neq altura(A) \\ 1/m & en\ otro\ caso,\ donde\ \ m = |A_{altura(A)}| \end{cases}$

T2 $P(y_i) = \frac{A(y_i)}{\sum_j A(y_j)}$

T3 $P(y_i) = \frac{1}{|A|} \ \ \forall\ i$

A partir del ejemplo, se puede observar que existe un número infinito de procedimientos de conversión entre funciones de membresía y distribuciones de probabilidad que cumplen las condiciones pedidas. Una variable a tener en cuenta a la hora de seleccionar un procedimiento en particular, es el grado de confianza atribuido al conjunto difuso A.

Ejemplo 22 *Sea $A = \{1/y_1, 0.99/y_2\}$. Utilizando el procedimiento $T1$ se obtiene la distribución de probabilidad P donde $P(y_1) = 1$ y $P(y_2) = 0$. Es decir, $T1$ supone que el grado de confianza atribuido al conjunto es muy alto. Por otro lado, sea $B = \{1/y_1, 0.01/y_2\}$. Utilizando el procedimiento $T3$ se obtiene la distribución P con $P(y_1) = P(y_2) = 1/2$. Se puede observar que $T3$ indica una completa falta de confianza en el modelo utilizado para obtener al conjunto A.*

Utilizando medidas de entropía en las distribuciones de probabilidad [34], se demuestra que $T3$ tiene la máxima entropía posible (mínima confianza) y $T1$ tiene la mínima entropía (correspondiente a máxima confianza).

3.1.2 Procedimientos de elección

Cualquiera sea el mecanismo para obtener la distribución de probabilidad P asociada a un conjunto difuso sobre $Y = \{y_1, \ldots, y_n\}$, el resultado es una distribución de probabilidad P asociada a los elementos de Y. A continuación se presentan dos mecanismos para seleccionar un valor y^* a partir de una distribución de probabilidad P.

S1 La elección de y^* puede verse como un problema de optimización, minimizando la media del error,

$$E\{(y - y^*)^2\}$$

El mínimo es alcanzado cuando y^* es el valor esperado de la distribución,

$$y^* = E\{y\} = \sum_{y_i} y_i P(y_i)$$

S2 La selección de y^* puede obtenerse a partir de un experimento aleatorio de la siguiente manera:

1. Se divide el intervalo $[0, 1]$ en n subintervalos $R_i = [a_i, b_i]$ donde
   $a_1 = 0 \qquad b_1 = P(y_1)$
   $a_i = b_{i-1} \quad b_i = a_i + P(y_i) \qquad i > 1$
2. Se genera un número aleatorio $r \in [0, 1]$
3. Si $r \in R_i$, entonces se hace $y^* = y_i$

3.2 Algunos defusificadores

Diferentes combinaciones de transformaciones a distribuciones de probabilidad y procedimientos de elección dan como resultado distintos mecanismos de defusificación. A continuación se presentan algunas elecciones interesantes.

3.2.1 COA

La ecuación 3.1 puede ser reescrita como la aplicación del procedimiento $S1$ a la transformación $T2$:

$$y^{COA} = \sum_{i=1}^{n} y_i P(y_i) \tag{3.3}$$

$$P(y_i) = \frac{A(y_i)}{\sum_j A(y_j)} \tag{3.4}$$

3.2.2 MOM

De la misma manera, la ecuación 3.2 puede verse como la aplicación del procedimiento $S1$ a la transformación $T1$, es decir:

$$y^{MOM} = \sum_{i} y_i P(y_i) \tag{3.5}$$

$$P(y_i) = \begin{cases} 0 & \text{si } A(y_i) \neq altura(A) \\ 1/m & \text{en otro caso, donde} \quad m = |A_{altura(A)}| \end{cases} \tag{3.6}$$

3.2.3 BADD

El método de transformación $BADD$ (BAsic Defuzzification Distribution) utiliza la siguiente transformación T_{BADD} a distribuciones de probabilidad [9]:

$$P(y_i) = \frac{A(y_i)^\alpha}{\sum_j A(y_j)^\alpha} \qquad \alpha \in [0, \infty] \tag{3.7}$$

Cabe destacar que si $\alpha = 1$ se obtiene la transformación asociada al método COA, es decir $T_{BADD} \to T2$. Además, si $\alpha \to \infty$ se recupera la transformación asociada al método MOM, $T_{BADD} \to T1$. Por último, si $\alpha = 0$, se cumple que $T_{BADD} \to T3$. Más aún, se puede demostrar que las distribuciones de probabilidad asociadas a los métodos COA y MOM son casos especiales en un continuo de distribuciones de probabilidad definidos por la transformación $BADD$.

Utilizando los valores de probabilidad de la ecuación 3.7 y el procedimiento $S1$ se obtiene el método $BADD$:

$$y^{BADD} = \frac{\sum_i A(y_i)^\alpha y_i}{\sum_i A(y_i)^\alpha} \tag{3.8}$$

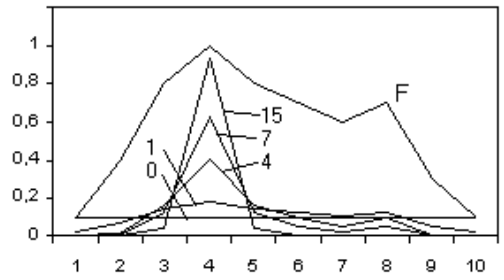

Figura 3.1: Transformación BADD para diferentes valores de α.

Ejemplo 23 *Sea F el conjunto difuso sobre $\{1,\ldots,10\}$ definido como*

$$F = \{0.1/1, 0.4/2, 0.8/3, 1/4, 0.8/5, 0.7/6, 0.6/7, 0.7/8, 0.3/9, 0.1/10\}$$

La figura 3.1 muestra la distribución de probabilidad obtenida utilizando la transformación $BADD$ para valores de $\alpha \in \{0, 1, 4, 7, 15\}$.

3.2.4 SLIDE

La transformación del método SLIDE se divide en dos etapas [33]:

1. El conjunto difuso A se transforma en otro conjunto difuso B sobre Y, mediante la regla $T_{\alpha,\beta}$:

$$B(y_i) = \begin{cases} A(y_i) & \text{si } A(y_i) \geq \alpha \\ (1-\beta)A(y_i) & \text{en otro caso} \end{cases} \tag{3.9}$$

   donde $\alpha \in [0, altura(A)]$ y $\beta \in [0, 1]$

   La transformación $T_{\alpha,\beta}$ preserva la forma y los valores de elementos con membresía mayor o igual que α y transforma linealmente el resto de los elementos.

2. Se normalizan las membresías de B para obtener la distribución de probabilidad P:

$$P(y_i) = \frac{B(y_i)}{\sum_j B(y_j)} \tag{3.10}$$

Por último, combinando los valores de la distribución de probabilidad de la ecuación 3.10 con el procedimiento $S1$ se obtiene el mecanismo $SLIDE$:

$$y^{SLIDE} = \sum_i y_i P(y_i) = \frac{(1-\beta)\sum_{i\in L} A(y_i)y_i + \sum_{i\in H} A(y_i)y_i}{(1-\beta)\sum_{i\in L} A(y_i) + \sum_{i\in H} A(y_i)} \tag{3.11}$$

donde $L = \{i/A(y_i) < \alpha\}$, y $H = \{i/A(y_i) \geq \alpha\}$

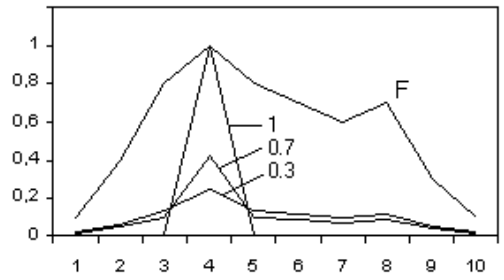

Figura 3.2: Transformación SLIDE para diferentes valores de α.

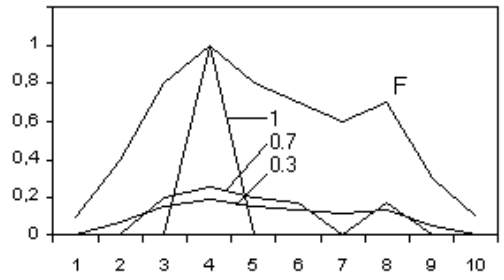

Figura 3.3: Transformación SLIDE para diferentes valores de β.

Es interesante notar que algunas elecciones de los parámetros α y β resultan en mecanismos de defusificación conocidos. En particular, si $\alpha = 0$, o $\alpha > 0$ y $\beta = 0$, $SLIDE$ coincide con COA, mientras que si $\alpha = Altura(A)$ y $\beta = 1$, $SLIDE$ coincide con MOM.

En este método, el parámetro α puede verse como un nivel de confianza, y el parámetro β como el nivel de rechazo de los valores que caen debajo del nivel de confianza α. Es interesante notar que, si se fija $\alpha = 0$, el valor y^{SLIDE} se mueve continuamente desde y^{COA} hacia y^{MOM} mientras β se mueve en forma continua en el intervalo $[0, 1]$.

Ejemplo 24 *La figura 3.2 muestra la distribución de probabilidad obtenida utilizando la transformación $SLIDE$ al conjunto F del ejemplo 23 para $\alpha = 1$ y $\beta \in \{0.3, 0.7, 1\}$. La figura 3.3 muestra la distribución obtenida para $\beta = 1$ y $\alpha \in \{0.3, 0.7, 1\}$.*

El problema del método SLIDE es que en general, no es una función lineal en β. En [34] se presenta una modificación, denominada $M-SLIDE$, que soluciona este inconveniente.

RAGE

Todos los métodos presentados anteriormente encuentran el valor esperado de la distribución de probabilidad dentro del conjunto Y de referencia. Esto puede traer inconvenientes en algunas situaciones, como en el problema de defusificación bajo restricciones. Por ejemplo, si un robot se encuentra ante un obstáculo y el análisis lleva a la conclusión que el siguiente movimiento debe realizarse o bien hacia la izquierda o bien hacia la derecha (éste sería el conjunto difuso), una técnica de defusificación utilizando el método $S1$ de valores esperados, guiará al robot directamente hacia el obstáculo.

En estos casos es conveniente utilizar el enfoque $RAGE$ (RAndom GEneration). Básicamente, el procedimiento transforma el conjunto F en una distribución de probabilidad y luego aplica el procedimiento $S2$ para obtener un valor representativo. Una característica importante del método es que siempre resulta en algún elemento y^* donde $P(y^*) \neq 0$. Esta característica hace de $RAGE$ un mecanismo muy atractivo para tomar decisiones teniendo en cuenta conjuntos de restricciones [34].

Capítulo 4

Modelado difuso

En general, la formulación de modelos complejos (especialmente, aunque no restringidos, a los sistemas humanos) con el tipo de precisión asociada a las técnicas clásicas no es posible, ni tampoco necesaria. Permitiendo, en cambio, cierta imprecisión en el modelo, se pueden formular situaciones complejas incluyendo la inherente vagueza del proceso humano al conceptualizar el mundo exterior. Los modelos difusos se basan en la codificación difusa de la información. Operan con conjuntos difusos en vez de con números. En muchos problemas reales, esta imprecisión es admisible e incluso útil ya que en esencia, la representación de la información en sistemas difusos imita los mecanismos de razonamiento aproximado de la mente humana.

4.1 Representaciones funcionales difusas

En algunas aplicaciones, como por ejemplo en control inteligente, se está interesado en la representación e implementación de relaciones funcionales complejas, no lineales y a veces hasta escasamente definidas. A modo de introducción se realizará un análisis en basado en conjuntos crisp para luego generalizar las nociones a conjuntos difusos utilizando para ello las herramientas que brinda la teoría AR.

4.1.1 El caso crisp

Sean $X_1, \ldots, X_n, Y$ conjuntos crisp. Una función $f : X_1 \times \ldots \times X_n \to Y$ asigna a cada tupla $(a_1, \ldots, a_n) \in X_1 \times \ldots \times X_n$ el valor único $f(a_1, \ldots, a_n)$. Esta función puede también ser vista como una relación $F \subset X_1 \times \ldots \times X_n \times Y$ donde se verifica

$$(a_1, \ldots, a_n, b) \in F \Leftrightarrow f(a_1, \ldots, a_n) = b \tag{4.1}$$

Si se supone, por simplicidad, que los conjuntos del dominio de f son finitos, la relación F queda caracterizada por el conjunto de tuplas:

$$\begin{array}{c} (a_1^1, \ldots, a_n^1, b^1) \\ \vdots \\ (a_1^t, \ldots, a_n^t, b^t) \end{array} \tag{4.2}$$

donde $(a_1^i, \ldots, a_n^i) \in dom(f)$ y $b^i = f(a_1^i, \ldots, a_n^i) \in Y$. De esta manera, la relación F puede ser expresada como la unión de relaciones atómicas

$$F = \cup_i (F_i) \quad i = 1..t \tag{4.3}$$

$$F_i = \{(a_1^i, \ldots, a_n^i, b^i)\} \tag{4.4}$$

Sin tener en cuenta consideraciones de eficiencia, un algoritmo que calcula el valor $f(x_1, \ldots, x_n)$ dadas las entradas `x_1, ..., x_n` es el siguiente:

`if x_1 =` a_1^1 `and ... and x_n =` a_n^1 `, then y =` b^1
`else`
`...`
`else`
`if x_1 =` a_1^t `and ... and x_n =` a_n^t `, then y =` b^t

Durante la ejecución del algoritmo se analizan en forma secuencial cada una de las reglas `IF/THEN` comparando los valores de las entradas con los datos fijos del antecedente. Si para la i-ésima regla todas las entradas `x_j` coinciden con los datos a_j^i, se asigna a la variable de salida el valor b^i. Como por definición, para cualquier valor de las entradas existe una única regla que puede aplicarse, el resultado está siempre bien definido.

4.1.2 El caso difuso

En el caso difuso, se tiene una función f definida por casos:

$$\begin{aligned} f(A_1^1, \ldots, A_n^1) &= B^1 \\ &\vdots \\ f(A_1^t, \ldots, A_n^t) &= B^t \end{aligned} \tag{4.5}$$

donde los A_i^j, al igual que los B^j son conjuntos difusos sobre X_i e Y respectivamente.

Imitando el razonamiento de la sección 4.1.1, la función f puede verse como una relación difusa $F = \cup_i (F_i)$ sobre $X_1 \times \ldots \times X_n \times Y$. Sin embargo, a diferencia del caso crisp, existen varias maneras de definir las relaciones F_i, como se verá más adelante. Lo que las diferentes formulaciones tienen en común es que las F_i se definen punto a punto en base a los conjuntos $A_1^i, \ldots, A_n^i, B^i$. De esta forma,

$$F_i(a_1, \ldots, a_n, b) = G\big[A_1^i(a_1), \ldots, A_n^i(a_n), B^i(b)\big] \tag{4.6}$$

Esto hace que el algoritmo de cálculo pueda ser reescrito (sintácticamente) de manera similar que para el caso crisp,

`if (x_1 is` A_1^1`) and ... and (x_n is` A_n^1`), then y is` B^1
`also`
`...`
`also`
`if (x_1 is` A_1^t`) and ... and (x_n is` A_n^t`), then y is` B^t

Sin embargo, la semántica es más complicada, ya que los diferentes `x_j` y A_j^i son conjuntos difusos con intersección no necesariamente nula, y el valor de verdad de cada proposición del antecedente de cada regla `IF/THEN` no está restringido a $\{V, F\}$. Para calcular el resultado en esta situación, se debe aplicar la teoría de razonamiento aproximado introducida en el capítulo 2.

4.1.3 Solución para el caso difuso

Resumiendo el análisis de la sección anterior, se tiene una familia de conjuntos crisp $\{X_1, \ldots, X_n, Y\}$ y una relación difusa $F = \cup_i(F_i)$ sobre $\{X_1, \ldots, X_n, Y\}$. Además, se tienen n conjuntos difusos V_i sobre X_i y se requiere encontrar el conjunto difuso W sobre Y correspondiente, en base a la relación difusa F.

Formalmente se puede representar esta situación mediante las siguientes $n+1$ proposiciones de la teoría AR:

$$\begin{array}{l} P_0 : \{X_1, \ldots, X_n, Y\} \ is \ F \\ P_1 : X_1 \ is \ V_1 \\ \quad \vdots \\ P_n : X_n \ is \ V_n \end{array} \tag{4.7}$$

Lo que se busca a partir de este conjunto de premisas es $minimal(Y)$, es decir, el menor conjunto difuso W sobre Y que se puede inferir a partir de las premisas $P_0, \ldots, P_n$. Según la teoría AR, este resultado puede expresarse mediante la proposición $Y \ is \ W$ donde

$$W = \Pi_Y(V_1 \bowtie \ldots \bowtie V_n \bowtie F) \tag{4.8}$$

Por la definición de proyección,

$$W(y) = \vee_{(x_1,\ldots,x_n)}(V_1 \bowtie \ldots \bowtie V_n \bowtie F)(x_1, \ldots, x_n, y) \tag{4.9}$$

Como los conjuntos $X_1, \ldots, X_n$ e Y son disjuntos, el operador $\bowtie$ se reduce al operador $\cap$ y se tiene que

$$W(y) = \vee_{x_1,\ldots,x_n}\left[V_1(x_1) \wedge \ldots \wedge V_n(x_n) \wedge F(x_1, \ldots, x_n, y)\right] \tag{4.10}$$

Por último, ya que $F = \cup_i(F_i)$, se tiene que:

$$W(y) = \vee_i W_i(y) \qquad i = 1..t \tag{4.11}$$

$$W_i(y) = \vee_{x_1,\ldots,x_n}\left[V_1(x_1) \wedge \ldots \wedge V_n(x_n) \wedge F_i(x_1, \ldots, x_n, y)\right] \tag{4.12}$$

4.2 Modelos difusos

Dado un conjunto de reglas "`also`-conectadas", de la forma:

`if (x_1 is` A_1^i`) and ... and (x_n is` A_n^i`), then y is` B^i

existen distintas elecciones para cada uno de los componentes que a su vez influyen en la ecuación 4.12. Estos son:

- Conectivo `and`. Da la interpretación del operador $\wedge$ en la ecuación 4.12, y debe ser un operador T-norm.
- Conectivo `then`. Da la interpretación de las relaciones F_i de la ecuación 4.12 por medio de la función G definida en 4.6.
- Conectivo `also`. Da la interpretación del operador $\vee$ en la ecuación 4.11 y debe ser un operador T-conorm.

Las distintas elecciones posibles dan lugar a diferentes mecanismos de modelado difuso. Las siguientes secciones describen los modelos más conocidos.

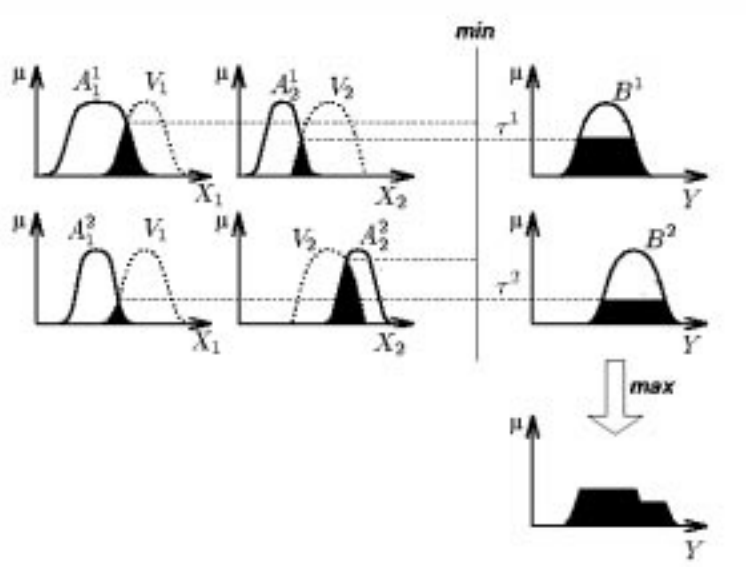

Figura 4.1: Modelo de inferencia de Mamdani con 2 entradas y 2 reglas.

4.2.1 Modelo de Mamdani

Este modelo de razonamiento [20], los operadores $\wedge$ y $\vee$ son los operadores min/max, y las relaciones F_i se definen como

$$F_i = A_1^i \cap \ldots \cap A_n^i \cap B^i$$

Por lo tanto, la ecuación 4.12 se convierte en

$$\begin{array}{rl} W_i(y) = \vee_{x_1,\ldots,x_n} & \big[V_1(x_1) \wedge \ldots \wedge V_n(x_n) \wedge \\ & \wedge\ A_1^i(x1) \wedge \ldots \wedge A_n^i(x_n) \wedge B^i(y)\big] \end{array} \tag{4.13}$$

Esta ecuación se puede reescribir como

$$\begin{array}{rl} W_i(y) = & \vee_{x_1}[V_1(x_1) \wedge A_1^i(x_1)] \\ & \wedge \ldots \wedge \\ & \vee_{x_n}[V_n(x_n) \wedge A_n^i(x_n)] \wedge B^i(y) \end{array} \tag{4.14}$$

o, en forma más concisa,

$$W_i(y) = \tau_1^i(x_1) \wedge \ldots \wedge \tau_n^i(x_n) \wedge B^i(y) \tag{4.15}$$

donde $\tau_j^i = \vee_{x_i}\big[(V_j \cap A_j^i)(x_i)\big]$ se puede ver como la relevancia de la j-ésima variable en la i-ésima regla. El valor $\tau^i = \Pi_j \tau_j^i$ se suele denominar DOF (degree of firing) de la regla i.

Ejemplo 25 *La figura 4.1 muestra gráficamente el procedimiento para obtener la salida* **y** *en el modelo de Mamdani con dos reglas* `IF/THEN` *y dos variables de entrada.*

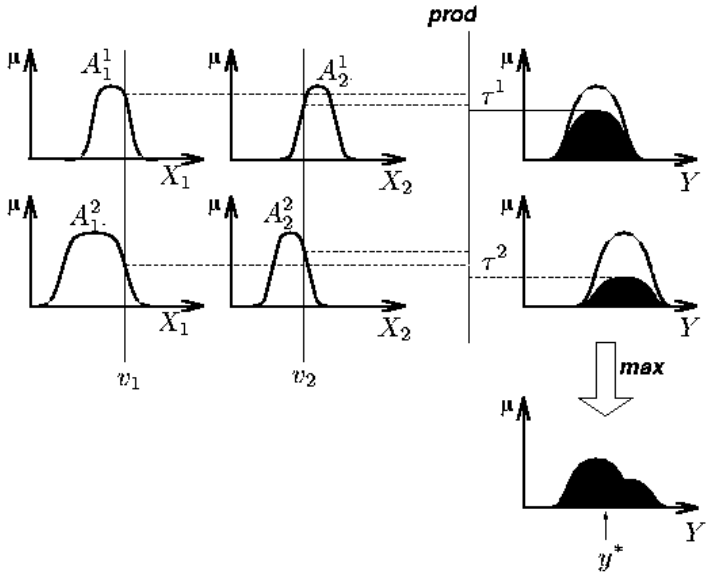

Figura 4.2: Modelo de inferencia de Larsen con 2 entradas y 2 reglas.

Usualmente, cuando las entradas `x_i` son valores crisp, `x_i`$=v_i$, las funciones de membresía de los V_i son:

$$V_i(x) = \begin{cases} 1 & \text{si } x = v_i \\ 0 & \text{en otro caso.} \end{cases} \tag{4.16}$$

y la ecuación 4.15 se reduce a

$$W_i(y) = \tau_1^i \wedge \ldots \wedge \tau_n^i \wedge B^i(y) \tag{4.17}$$

donde $\tau_j^i = A_j^i(v_i)$.

4.2.2 Modelo de Larsen

En el modelo de Larsen es muy similar al modelo de Mamdani, excepto que el operador T-norm utilizado como interpretación del `and` es el producto.

Ejemplo 26 *La figura 4.2 muestra esquemáticamente como se obtiene una salida crisp en el modelo de Larsen con dos reglas* `IF/THEN` *y dos variables de entrada crisp.*

4.2.3 Modelo TSK

Generalmente existe un conocimiento del sistema a modelar en forma de condiciones generales acerca de su estructura, como por ejemplo, leyes de conservación o ecuaciones de balance. Una alternativa propuesta para trabajar en estos casos es el modelo TSK, desarrollado por Takagi, Sugeno y Kang [29] [26], que reduce considerablemente el número de reglas de inferencia utilizadas en el modelo de

Mamdani, mediante un conjunto de reglas especiales. En estas reglas el consecuente es un valor funcional en lugar de los valores difusos B^i empleados en otros modelos. Cada regla tiene la forma

`if (x_1 is` A_1^i`) and ... and (x_n is` A_n^i`),`
`then y_i =` p_0^i `+` p_1^i `x_1 + ... +` p_n^i `x_n`

Además, en el modelo TSK las entradas `x_1, ..., x_n` son valores crisp, con `x_i` $= v_i, i = 1 \ldots n$. De esta manera, cada una de las funciones lineales de los consecuentes puede ser vista como un modelo lineal con entradas crisp `x_1 ... x_n`, salida crisp `y_i`, y parámetros crisp $p_0^i, \ldots, p_n^i$.

En este modelo, el operador $\wedge$ es el producto, y el operador $\vee$ es max. Las relaciones F_i se definen como

$$F_i = A_1^i \cap \ldots \cap A_n^i \cap P^i \tag{4.18}$$

donde $P^i = p_0^i + p_1^i v_1 + \ldots + p_n^i v_n$ es un valor crisp, es decir

$$P^i(y) = \begin{cases} 1 & \text{si } y = p_0^i + p_1^i v_1 + \ldots + p_n^i v_n \\ 0 & \text{en otro caso} \end{cases} \tag{4.19}$$

Reemplazando estas interpretaciones en la ecuación 4.12, y mediante argumentos similares a los utilizados en el modelo de Mamdani, se obtiene

$$W_i(y) = \prod_j \Big(max_{x_j} [V_j(x_j) A_j^i(x_j)] \Big) P^i(y) \tag{4.20}$$

Como los `x_i`$= v_i$ son valores crisp, esta ecuación se reduce a

$$W_i(y) = A_1^i(v_1) \ldots A_n^i(v_n) P^i(y) = \tau_i P^i(y) \tag{4.21}$$

donde $\tau_i = A_1^i(v_1) \ldots A_n^i(v_n)$ es el DOF de la i-ésima regla. Se puede ver que, como cada P^i es un valor crisp, la salida W_i inferida por cada regla tiene un solo punto con membresía no nula,

$$W_i(y) = \begin{cases} \tau_i & \text{si } y = p_0^i + p_1^i v_1 + \ldots + p_n^i v_n \\ 0 & \text{en otro caso} \end{cases} \tag{4.22}$$

El mecanismo de defusificación utilizado por el modelo TSK es el COA (center of area) y por lo tanto, la salida final y^* defusificada queda expresada como:

$$y^* = \frac{\sum_{y \in Y} W(y)\ y}{\sum_{y \in Y} W(y)} = \frac{\sum_{i=1}^{t} \tau_i (p_0^i + p_1^i x_1 + \ldots + p_n^i x_n)}{\sum_{i=1}^{t} \tau_i} \tag{4.23}$$

Esta ecuación se puede escribir de la siguiente manera:

$$y^* = \sum_{i=1}^{t} \frac{\tau_i}{\sum_{j=1}^{t} \tau_j} (p_0^i + p_1^i x_1 + \ldots + p_n^i x_n) \tag{4.24}$$

Geométricamente, las reglas del modelo TSK corresponden a una aproximación de la función original mediante una combinación de funciones lineales. Cabe destacar que la simplicidad de los modelos TSK es superficial. En realidad, el conjunto de parámetros del modelo consiste no solo en los coeficientes p_j^i que aparecen en los subsistemas lineales, sino también los conjuntos difusos de los antecedentes, que incorporan fuertes componentes no lineales.

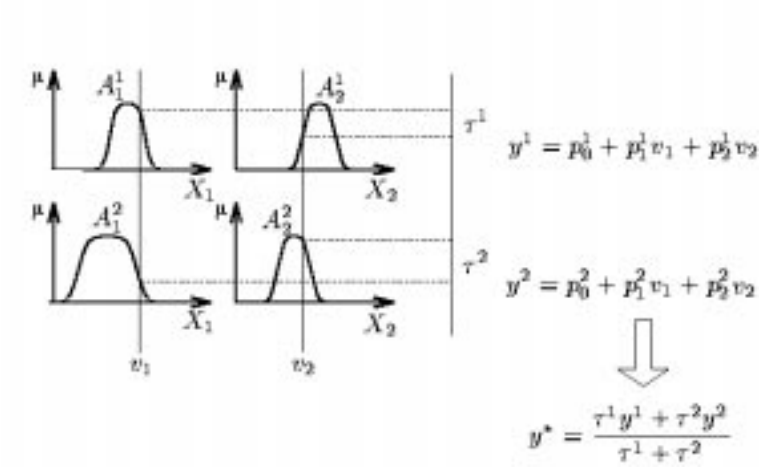

Figura 4.3: Modelo de inferencia TSK con 2 entradas y 2 reglas

Ejemplo 27 *La figura 4.3 muestra gráficamente el funcionamiento del modelo de inferencia TSK para dos entradas y dos reglas de inferencia.*

En un contexto más general, las funciones lineales de los consecuentes pueden reemplazarse por otras no lineales, obteniendo de esta manera un conjunto de reglas con la siguiente forma:

```
if (x_1 is A_1^i) and ... and (x_n is A_n^i),
  then y_i = f^i (x_1, ..., x_n)
```

4.2.4 Modelo simplificado

El modelo simplificado de razonamiento (también denominado modelo TSK de orden cero) es un caso particular del modelo TSK, cuando los coeficientes $p_1^i, \ldots, p_n^i$ valen cero. En este caso, las reglas tienen la siguiente forma (renombrando a las constantes p_0^i como simplemente p^i):

```
if (x_1 is A_1^1) and ... and (x_n is A_n^1), then y_1 =p^1
also
...
also
if (x_t is A_1^t) and ... and (x_n is A_n^t), then y_t = p^t
```

Capítulo 5

Mecanismos de Aprendizaje

Una tarea importante en el área del diseño de sistemas de inferencia es proveer una metodología para el desarrollo de modelos difusos, es decir, la obtención sistemática de un modelo difuso a partir del conocimiento del sistema real que se quiere modelar. Los primeros intentos utilizaban el denominado **enfoque directo**, en el cual se utiliza fuertemente el conocimiento de expertos humanos, extrayendo de ellos las reglas lingüísticas y funciones de membresía necesarias para construir los sistemas de inferencia difusos. Este enfoque puede resumirse en los siguientes pasos:

- Selección de las variables de entrada y salida.
- Determinación de los universos de cada variable.
- Selección de un mecanismo de razonamiento difuso.
- Determinación de los valores lingüísticos (que corresponden a conjuntos difusos) en los que se descompone el universo de cada variable.
- Construcción del conjunto de reglas lingüísticas que representan las relaciones entre las variables del sistema.
- Modificación de los parámetros del sistema para aumentar la precisión del modelo.

Lamentablemente, este esquema presenta varias dificultades, ya que el conocimiento de expertos humanos es frecuentemente incompleto, subjetivo y no sistemático. Es por ello que últimamente se han desarrollado mecanismos tendientes a automatizar la construcción del modelo difuso. En general no se conoce el funcionamiento interno del sistema real, y la información que se posee es un un conjunto de pares de entrada-salida observados, denominado **conjunto de entrenamiento**. Cada elemento de este conjunto se denomina **patrón de entrenamiento**. A partir, entonces, del conjunto de entrenamiento, la construcción del modelo difuso se puede dividir en dos etapas. En la primera se determina su estructura, encontrando un conjunto de reglas y una partición del espacio de entrada que se adecue al sistema real. En la segunda etapa se identifican los parámetros (funciones de membresía, coeficientes lineales) que describen más precisamente al sistema modelado aplicando para ello variadas técnicas de optimización.

5.1 Determinación de la estructura

Cuando no se dispone de conocimiento a priori, la identificación de la estructura se transforma en un problema muy complicado y algunas veces se deben aplicar métodos de prueba y error. Es por ello que últimamente se han desarrollado varios mecanismos tendientes a automatizar algunas etapas de esta tarea. Se pueden mencionar, entre otros, el enfoque CART [15], la utilización de estructuras de datos como k-d trees [28], el uso de métodos iterativos [27], y una gran cantidad de algoritmos de clustering, introducidos a continuación.

Los algoritmos de clustering son utilizados ampliamente, no solamente para la construcción de modelos, sino también en aplicaciones de compresión, organización y clasificación de información. Estas técnicas pueden utilizarse también para obtener las estimaciones iniciales de los parámetros de las reglas en sistemas de inferencia difuso, como se verá en la sección 5.1.6.

5.1.1 Mountain-Clustering

El método **mountain clustering** [34] [35] es un enfoque relativamente sencillo y efectivo para estimar centros de clusters en base a una medida de densidad, y está basado en el procedimiento que realiza un ser humano al visualizar clusters en un conjunto de datos. Sea $X = \{x_1, \ldots, x_n\}$ un conjunto de puntos, donde $x_i \in \mathbb{R}^m$. El método se puede describir por medio de los siguientes pasos:

1. Dividir el espacio de los datos mediante una grilla, cuyas intersecciones forman el conjunto de candidatos a cluster $V = \{v_1, \ldots, v_{p^M}\}$, donde p es el número de particiones de la grilla en cada dimensión. Es posible también dividir el espacio de forma no uniforme en cada dimensión, para aprovechar de esta forma el conocimiento previo de cada problema particular.

2. Calcular la función densidad $m : V \to \mathbb{R}$, llamada **función montaña**. La altura de la montaña en un punto $v \in V$ es igual a

$$m(v) = \sum_{i=1}^{n} e^{-\frac{||v - x_i||}{2\alpha^2}} \tag{5.1}$$

   donde α es una constante dependiente de la aplicación que determina la altura y la suavidad de la función resultante.

3. Seleccionar como cluster c_i al punto entre los candidatos v_j que tenga el valor máximo de la función de densidad (si existe más de un punto con valor máximo, se elige uno de ellos al azar).

4. Eliminar el efecto causado por el cluster identificado en el paso anterior. Para ello, alterar la función de densidad restando una componente gaussiana centrada en el cluster seleccionado y proporcional a su altura:

$$m'(v) = m(v) - m(c_i)e^{-\frac{||v - c_i||^2}{2\beta^2}} \tag{5.2}$$

   donde la constante β es normalmente más grande que α para prevenir la identificación de clusters demasiado poco espaciados.

5. Si la altura del último cluster seleccionado es mayor que una constante predeterminada δ, volver al paso 2.

5.1.2 Substractive Clustering

Aunque el método *Mountain Clustering* es relativamente simple y efectivo, la complejidad del cálculo crece exponencialmente con la dimensión del problema debido a la evaluación de la función de densidad sobre todos los puntos de la grilla. En este enfoque alternativo [4], los puntos considerados como candidatos a cluster son los propios datos y no los puntos de alguna grilla. De este modo el tiempo de computación es proporcional al número de datos e independiente de la dimensión del problema.

5.1.3 C-Means Clustering

Este método, también conocido como **K-Means Clustering** divide un conjunto de puntos $X = \{x_1, \ldots, x_n\}$ en un número preestablecido k de grupos $\{C_1, \ldots, C_k\}$, y encuentra un centro c_i dentro de cada grupo tal que se minimice una función costo J,

$$J = \sum_{i=1}^{k} J_i = \sum_{i=1}^{k} \Big(\sum_{x_k \in C_i} ||x_k - c_i||^2 \Big) \tag{5.3}$$

Los grupos se definen por medio de una matriz binaria de dimensión $c \times n$, donde $u_{i,j} = 1 \Leftrightarrow x_j \in C_i$. Si los c_i están fijos, los $u_{i,j}$ que minimizan J son:

$$u_{i,j} = \begin{cases} 1 & \text{si } ||x_j - c_i||^2 \leq ||x_j - c_k||^2, \ \forall\ k \\ 0 & \text{en otro caso.} \end{cases} \tag{5.4}$$

Por otro lado, si los $u_{i,j}$ están fijos, los centros óptimos c_i que minimizan J son el promedio de los puntos de cada grupo i:

$$c_i = \frac{1}{|C_i|} \sum_{x_k \in C_i} x_k \tag{5.5}$$

Las ecuaciones 5.4 y 5.5 son la base del siguiente algoritmo iterativo:

1. Inicializar los centros c_i aleatoriamente entre los puntos x_j.
2. Determinar la matriz U utilizando para ello la ecuación 5.4.
3. Modificar los c_i de acuerdo a la ecuación 5.5.
4. Si la función J no alcanza un nivel de tolerancia δ, volver al paso 2.

El algoritmo es iterativo y no se pueden dar garantías de convergencia. Más aún, el rendimiento del algoritmo depende de las posiciones iniciales de los clusters c_i. Es por ello recomendable reemplazar el paso 1 por algún otro método (por ejemplo, mountain clustering o substractive clustering) para encontrar buenos valores iniciales, e incluso el número total de clusters a identificar.

5.1.4 Fuzzy C-Means Clustering

Este método [1] es una mejora sobre el algoritmo descripto en la sección anterior, donde cada punto x_i pertenece en forma difusa a un grupo C_j. Por lo tanto, la matriz U ya no es binaria, sino que puede tomar valores en el intervalo $[0,1]$, sujeta a la restricción

$$\sum_{i=1}^{k} u_{i,j} = 1, \quad \forall j \tag{5.6}$$

La función J es una generalización de la ecuación 5.3:

$$J(U,\overline{c}) = \sum_{i=1}^{c} J_i = \sum_{i=1}^{c}\sum_{j=1}^{n} u_{i,j}^{m}||x_k - c_i||^2, \quad m \in [0,\infty) \tag{5.7}$$

Utilizando multiplicadores de Lagrange para las restricciones, y diferenciando las ecuaciones obtenidas se llega a las siguientes fórmulas que minimizan la función J:

$$c_i = \frac{\sum_{j=1}^{n} u_{i,j}^{m} x_j}{\sum_{j=1}^{n} u_{i,j}^{m}} \tag{5.8}$$

$$u_{i,j} = \left[\sum_{l=1}^{k}\left(\frac{||c_i - x_j||}{||c_l - x_j||}\right)^{\frac{2}{m-1}}\right]^{-1} \tag{5.9}$$

El algoritmo que determina los clusters c_i es muy similar al de la sección anterior y se presenta a continuación:

1. Inicializar los centros c_i aleatoriamente entre los puntos x_j.
2. Determinar la matriz U mediante la ecuación 5.9.
3. Modificar los clusters c_i de acuerdo a la ecuación 5.8.
4. Si la función J no alcanza un nivel de tolerancia δ, volver al paso 2.

5.1.5 Otros algoritmos

Existe una gran variedad de algoritmos de clustering, cada uno aportando alguna mejora sobre el anterior. El **PCM (Possibilistic C-Means)** [19], relaja la restricción de la ecuación 5.6, permitiendo que las membresías pertenezcan sin restricciones al intervalo $[0,1]$, de manera de aproximar mejor aquellos datos que se encuentran equidistantes de los clusters. El algoritmo **RFCM (Robust Fuzzy C-Means)** [5] intenta discriminar el ruido en los datos de entrada utilizando un cluster adicional, denominado cluster ruidoso, que engloba a todos los puntos que no pueden ser asignados a los demás clusters por encontrarse muy distantes de ellos.

5.1.6 Inicialización de reglas de inferencia difusas

Utilizando los algoritmos de clustering de las secciones anteriores se puede obtener la estructura del sistema de inferencia difuso así como, en algunos casos, las estimaciones iniciales de los conjuntos difusos A^i_j de los antecedentes y B^j de los consecuentes de las reglas `IF/THEN`, como se ve en el siguiente ejemplo.

Ejemplo 28 *Sean* $\{(p^i_1, \ldots, p^i_n, q^i),\ i = 1..N\}$*, los patrones de entrenamiento que aproximan a la función* $f : \mathbb{R}^n \to \mathbb{R}$ *y sean* $\{(c^i_1, \ldots, c^i_n, d^i),\ i = 1..t\}$*, los clusters obtenidos utilizando el procedimiento* mountain clustering*. Cada cluster puede verse como una relación lingüística de la forma:*

"Si la entrada $(x_1, \ldots, x_n)$ *está cerca de* $(c^i_1, \ldots, c^i_n)$*,*
entonces la salida inferida (y) *está cerca de* d^i*"*

Dentro del modelo de razonamiento simplificado de la sección 4.2.4, cada una de estas reglas se puede traducir en las siguientes reglas `IF/THEN`*:*

`if (x_1 is` C^i_1`) and ... and (x_n is` C^i_n`), then f_i =` d^i

donde C^i_j *es el conjunto difuso*

$$C^i_j(x) = e^{-\frac{1}{2}\left(\frac{x - c^i_j}{\sigma}\right)^2}$$

y expresa la proximidad entre la j-ésima componente del i-ésimo cluster (c^i_j) *y la j-ésima componente de la entrada* (x_j)*. El valor de* σ *se puede estimar a partir del parámetro* β *utilizado en la destrucción de la función montaña, y vale:*

$$\sigma = (2\beta)^{-\frac{1}{2}}$$

5.2 Identificación de los parámetros

Una vez que se ha fijado la estructura del sistema de inferencia, se deben ajustar los valores de los parámetros que lo componen (funciones de membresía, coeficientes lineales, etc.) de manera que el modelo pueda describir en forma más satisfactoria al sistema real. Dependiendo de la estructura particular de cada sistema de inferencia, se pueden aplicar distintos mecanismos de optimización. Por ejemplo, si existen subsistemas lineales en el mecanismo de inferencia pueden utilizarse las técnicas de la sección 5.2.1. Si las funciones involucradas son diferenciables, pueden ser de ayuda los mecanismos desarrollados en la sección 5.2.2. En cualquier caso, pueden aplicarse también los mecanismos flexibles, aunque no tan eficientes, desarrollados en la sección 5.3.

5.2.1 Estimación por cuadrados mínimos

Dentro del contexto general del problema de cuadrados mínimos, la salida y de un sistema lineal con entradas $\overline{x} = (x_1, \ldots, x_n)$ está dada por la expresión:

$$y = \theta_1 f_1(\overline{x}) + \ldots + \theta_n f_n(\overline{x}) \tag{5.10}$$

donde las f_i son funciones conocidas, y los θ_i son los parámetros a estimar.

El conjunto de entrenamiento, compuesto por pares $\{(\overline{x}_i, y_i),\ i = 1..m\}$, produce un sistema con m ecuaciones y n incógnitas:

$$A\theta = y \tag{5.11}$$

donde $a_{i,j} = f_j(x_i)$. En la mayoría de los casos, $m >> n$, es decir, el número de pares de entrenamiento es mayor que el número de parámetros a estimar. En estos casos, el sistema puede no tener solución exacta. Por lo tanto, la ecuación 5.11 se reemplaza por

$$A\theta + e = y \tag{5.12}$$

y se intenta encontrar el vector θ que minimiza la suma de los cuadrados de los errores:

$$E(\theta) = \sum_{i=1}^{m} (y_i - a_i^T \theta)^2 \tag{5.13}$$

Se puede demostrar que esta ecuación se minimiza cuando $\theta = \theta^*$, denominado el valor LSE, que satisface la ecuación normal:

$$A^T A\theta^* = A^T y \tag{5.14}$$

Si $A^T A$ es no singular, θ^* es único y está dado por

$$\theta^* = (A^T A)^{-1} A^T y \tag{5.15}$$

LSE Recursivo

Aunque la ecuación para la solución de θ se escribe en forma muy concisa, es computacionalmente costosa debido al cálculo de la inversa de $A^T A$. Más aún, la ecuación no está definida si $A^T A$ es singular. Para evitar estos problemas, se pueden emplear fórmulas iterativas que dan lugar a mecanismos más eficientes para el cálculo del LSE. De esta forma, si a_i^T es la i-ésima fila de la matriz A, θ^* se calcula por medio de las siguientes ecuaciones:

$$\begin{cases} P_{k+1} = P_k - \frac{P_k a_{k+1} a_{k+1}^T P_k}{1 + a_{k+1}^T P_k a_{k+1}} \\ \theta_{k+1} = \theta_k + P_{k+1} a_{k+1} (y_{k+1} - a_{k+1}^T \theta_k) \end{cases} \tag{5.16}$$

donde $\theta_0 = \overline{0}$, $P_0 = \gamma I$, y γ es un entero positivo grande.

Este método es iterativo y calcula progresivamente el valor $\theta^* = \theta_m$ utilizando, en cada paso, la información obtenida en los pasos anteriores.

LSE Recursivo para sistemas dinámicos

En los casos en los que los pares $(\overline{x}, y)$ son obtenidos on-line, se puede introducir un factor λ, denominado **factor de olvido**, que otorga más importancia a los datos más recientes. En este caso, las ecuaciones que se deben utilizar son las siguientes:

$$\begin{cases} P_{k+1} = \frac{1}{\lambda}\left(P_k - \frac{P_k a_{k+1} a_{k+1}^T P_k}{\lambda + a_{k+1}^T P_k a_{k+1}}\right) \\ \theta_{k+1} = \theta_k + P_{k+1} a_{k+1} (y_{k+1} - a_{k+1}^T \theta_k) \end{cases} \tag{5.17}$$

donde λ generalmente pertenece al intervalo $[0.9, 1]$. Mientras menor sea λ, más rápido decae el efecto de los datos más viejos. Sin embargo, la elección de un λ demasiado pequeño causa inestabilidad numérica y debe ser evitada.

5.2.2 Minimización basada en el gradiente

El objetivo de este método es la minimización de una función diferenciable $E : \mathbb{R}^n \to \mathbb{R}$, denominada función objetivo o costo. Como en general la función E es muy compleja, no es posible resolver el problema analíticamente. El procedimiento, en cambio, es iterativo y busca sistemáticamente posibles soluciones en el espacio de entrada. La determinación de las direcciones de búsqueda se realiza en base a información proveniente de las derivadas primeras de E.

De esta manera, dado un punto $\overline{x_k} \in \mathbb{R}^n$ en el espacio de búsqueda, se determina la transición al siguiente punto $\overline{x_{k+1}}$ de acuerdo a la ecuación:

$$\overline{x_{k+1}} = \overline{x_k} + \eta_k d_k \tag{5.18}$$

donde $d_k \in \mathbb{R}^n$ es un vector que indica la nueva dirección de búsqueda y η_k es la longitud del paso, también llamado coeficiente de aprendizaje, que regula hasta donde extenderse en esa dirección. El cálculo de cada iteración, entonces, puede dividirse en dos etapas: primero se determina el vector de dirección d_k, y luego se calcula la longitud del paso η_k.

Determinación de la dirección d_k

La dirección de búsqueda en cada iteración está basada en el gradiente de la función objetivo E.

Definición 37 *Sea $E : \mathbb{R}^n \to \mathbb{R}$ una función diferenciable. El gradiente de E en el punto $\overline{x}$, denotado $\nabla E(\overline{x})$, es el vector de las derivadas primeras de E, valuado en $\overline{x}$,*

$$\nabla E(\overline{x}) = \Big[\frac{\partial E(\overline{x})}{\partial x_1}, \ldots, \frac{\partial E(\overline{x})}{\partial x_n}\Big]$$

En particular, d_k apunta en la dirección contraria al gradiente, es decir, la dirección que *localmente* hace decrecer más rápido a la función E:

$$d_k = -\nabla E(\overline{x_k}) \tag{5.19}$$

Estimación de la longitud del paso η

La determinación del parámetro η puede influir en el proceso completo de minimización. Como en general es imposible obtener una función analítica para determinar los η_k óptimos, se utilizan ciertas heurísticas que intentan aproximar estos valores. Para normalizar el parámetro η, la ecuación 5.18 se puede reescribir de la siguiente manera:

$$\overline{x_{k+1}} = \overline{x_k} - \kappa_k \frac{\nabla E(\overline{x_k})}{||\nabla E(\overline{x_k})||} \tag{5.20}$$

donde el parámetro κ_k mide la distancia euclídea entre $\overline{x_k}$ y $\overline{x_{k+1}}$.

Si κ es demasiado pequeño, el método del gradiente se aproxima mucho al mínimo, pero la convergencia es demasiado lenta. Por otro lado, si κ es muy grande, la convergencia inicial es rápida, pero el algoritmo oscila en la vecindad del mínimo. No existe un mecanismo universalmente aceptado para modificar el tamaño de κ, y esta tarea debe ser realizada utilizando heurísticas.

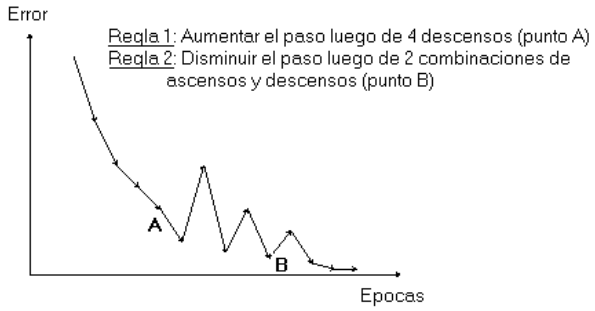

Figura 5.1: Heurísticas para modificar el paso κ

Basado en observaciones empíricas [14], el tamaño de κ puede ser modificado teniendo en cuenta las siguientes reglas:

1. Si la función E decrece durante m iteraciones consecutivas, $\kappa = (1 + p)\kappa$
2. Si la función E oscila durante n iteraciones consecutivas, $\kappa = (1 - q)\kappa$

donde los valores de los parámetros m, n, p y q utilizados en la práctica son, respectivamente, 4, 2, 0.1 y 0.1. La aplicación de estas ideas puede verse en el ejemplo de la figura 5.1.

5.3 Otros mecanismos de optimización

Existen algunos mecanismos de optimización que, a diferencia de los mencionados en las secciones anteriores, requieren que la estructura a optimizar cumpla muy pocas condiciones. Estos métodos no necesitan utilizar información acerca de las derivadas del conjunto de parámetros para minimizar la función objetivo, sino que, iterativamente, las nuevas direcciones de búsqueda incorporan componentes aleatorias y siguen ciertas reglas heurísticas basadas en conceptos simples e intuitivos. Por lo tanto, estos mecanismos son en general más lentos que los basados en otras técnicas y más complicados para estudiar analíticamente. Por otro lado, son mucho más flexibles y pueden utilizarse para optimización de problemas tanto continuos como discretos.

5.3.1 Búsqueda aleatoria

Este mecanismo explora aleatoriamente el espacio de parámetros de una función objetivo para encontrar el punto óptimo que minimiza (o maximiza) la función objetivo. Al ser extremadamente simple, el método de búsqueda aleatoria puede ser convenientemente modificado para tener en cuenta características específicas de problemas particulares. Además, el método converge al óptimo global con probabilidad 1 en un conjunto compacto, aunque en la práctica el proceso de optimización puede necesitar una cantidad de tiempo inmanejable. Si f es la función objetivo a ser minimizada, el algoritmo de búsqueda aleatoria sigue los siguientes pasos:

1. Elegir un punto inicial $\overline{x}$.
2. Agregar un vector aleatorio $\Delta\overline{x}$ al punto actual $\overline{x}$.
3. Si $f(\overline{x} + \Delta\overline{x}) < f(\overline{x})$, actualizar el punto actual a $\overline{x} + \Delta\overline{x}$.
4. Si la cota de iteraciones no ha sido alcanzada, volver al paso 2.

Algunas heurísticas que pueden utilizarse para mejorar el algoritmo se basan en las siguientes observaciones:

- Si la búsqueda en una dirección resulta en un aumento de la función objetivo, frecuentemente la dirección contraria la hace decrecer.
- Sucesivas búsquedas exitosas en una misma dirección deben inclinar las siguientes búsquedas en esa dirección, así como búsquedas no exitosas deben disminuír la probabilidad de búsqueda en esas direcciones.

En [16] se presenta un algoritmo que incluye estas heurísticas.

5.3.2 Algoritmos genéticos

Los algoritmos genéticos [12] [10] son un mecanismo de optimización estocástico basado en los conceptos de selección natural y procesos evolutivos. Estos algoritmos codifican cada punto del espacio de solución en una cadena binaria denominada **cromosoma**. Cada cromosoma tiene asociada una **medida de rendimiento** que, para problemas de maximización, es igual a la función objetivo. En todo momento se mantiene un conjunto de cromosomas o **individuos** en un conjunto denominado **población**. En cada etapa o **generación**, se construye una nueva población utilizando operadores genéticos como el **crossover** y la **mutación**. El operador de crossover se aplica a un par de cromosomas, seleccionando aleatoriamente un punto intermedio en su código genético e intercambiando sus cadenas genéticas para obtener de esa forma nuevos individuos. El operador de mutación es capaz de generar espontáneamente nuevos cromosomas con una pequeña probabilidad, usualmente intercambiando un bit del código genético, previniendo de esta forma la convergencia de la población a un mínimo local. Los miembros con mayores valores de rendimiento tienen más chances de sobrevivir y participar en las operaciones genéticas. Luego de un número determinado de generaciones, la población contiene miembros con muy buenos valores de rendimiento, de forma similar al modelo evolutivo Darwiniano.

Basado en los conceptos definidos anteriormente, se describe a continuación un algoritmo genético simple para problemas de maximización:

1. Iniciar la población con individuos generados aleatoriamente y evaluar el rendimiento de cada individuo.
2. Repetir hasta que algún criterio de terminación se cumpla:
   (a) Seleccionar dos individuos de la población con probabilidad proporcional a sus valores de rendimiento.
   (b) Aplicar el operador de crossover con probabilidad α.
   (c) Aplicar el operador de mutación con probabilidad β.
   (d) Repetir los pasos a, b y c hasta que se hayan creado suficientes individuos para la siguiente generación.

5.3.3 Simulated annealing

Esta técnica de optimización se deriva de ciertas propiedades físicas que se verifican al enfriar metales en forma controlada [17] [22]. En el método de simulated annealing, el valor de la función objetivo f que se quiere minimizar es el análogo a la energía de un sistema termodinámico. A temperaturas elevadas, se permiten evaluaciones de puntos distantes y se pueden aceptar puntos que tengan mayor energía, correspondiéndose con la situación en la que los átomos con gran movilidad tratan de orientarse con otros átomos no locales ocasionalmente aumentando la energía del sistema. A bajas temperaturas, en cambio, se evalúa la función objetivo solamente en puntos locales y es poco probable que se acepte un nuevo punto con mayor energía. Las componentes principales del algoritmo son entonces las siguientes:

Función generatriz g : Especifica la densidad de probabilidad de la diferencia entre el punto actual y el siguiente punto. En algoritmos convencionales, conocidos también como **máquinas de Boltzmann**, la función generatriz utilizada es:

$$g(\Delta\overline{x}, T) = (2\pi T)^{-n/2}\ e^{\frac{-||\Delta\overline{x}||^2}{2T}} \tag{5.21}$$

donde n es la dimensión del espacio de búsqueda.

Función de aceptación h : Luego de que un nuevo punto ha sido evaluado, la función de aceptación decide si se acepta o se rechaza el nuevo punto y generalmente se utiliza la siguiente:

$$h(\Delta E, T) = \frac{1}{1 + e^{\frac{\Delta E}{cT}}} \tag{5.22}$$

donde c es una constante dependiente del sistema y ΔE es la diferencia de energía entre el nuevo punto y el viejo.

Agenda de la temperatura : Especifica cómo decrece la temperatura T. Es una función del tiempo o del número de iteraciones y dependiente de la aplicación particular. Generalmente T se decrementa en cada iteración un pequeño porcentaje.

Utilizando estos elementos, se presenta a continuación un esquema del mecanismo general de simulated annealing:

1. Elegir un punto inicial $\overline{x}$ y una temperatura inicial T.
2. Seleccionar $\Delta\overline{x}$ de acuerdo a la función generatriz g.
3. Asignar $\overline{x} = \overline{x} + \Delta\overline{x}$ con probabilidad determinada por la función de aceptación h, donde $\Delta E = f(\overline{x} + \Delta\overline{x}) - f(\overline{x})$.
4. Reducir la temperatura T de acuerdo a la agenda de la temperatura y, si el número máximo de iteraciones no ha sido alcanzado, volver al paso 2.

Para problemas discretos, cada punto $\overline{x}$ no puede tomar valores arbitrarios, y por lo tanto, $\overline{x}+\Delta\overline{x}$ puede no ser un punto válido en el espacio de las soluciones. En su lugar, se define un **conjunto de movimientos**, igual al conjunto de los puntos legales disponibles para explorar a partir del punto $\overline{x}$ y se realiza un experimento aleatorio dentro de este conjunto para elegir el próximo punto. La definición del conjunto de movimientos es dependiente de cada problema particular.

Capítulo 6

Redes adaptativas neuro-difusas

Las redes adaptativas neuro-difusas combinan los sistemas de inferencia difusos y las redes neuronales, aprovechando las características sobresalientes de cada modelo. Por un lado, los sistemas de inferencia difusos proveen un mecanismo intuitivo y de alto nivel para representar el conocimiento mediante la utilización de reglas `IF/THEN`. Por otro lado, las redes neuronales poseen un alto grado de adaptabilidad y capacidad de aprendizaje y generalización. La construcción de herramientas que se nutren de estas dos áreas ha demostrado ser un mecanismo eficiente a la hora de modelar sistemas reales.

6.1 Redes adaptativas

Las redes adaptativas proveen un marco teórico que unifica casi todos los tipos de redes neuronales con capacidades de aprendizaje, y sus propiedades fundamentales son un elemento clave para comprender los demás paradigmas.

Definición 38 *Una* **red neuronal adaptativa** *se caracteriza por un grafo dirigido $G = (V, E)$, donde existen nodos con $fan\text{-}in$ cero (entradas) y nodos con $fan\text{-}out$ cero (salidas). Cada nodo $n \in V$ representa una unidad de procesamiento y tiene asociada una función estática denotada F_n, y cada arco $e \in E$ indica la relación causal entre los nodos conectados. El conjunto de nodos se puede dividir en dos subconjuntos, $V = (A \cup N)$. Los nodos $n \in A$ se denominan* **adaptativos**, *y sus salidas dependen, no sólo de sus entradas, sino también de parámetros modificables, denominados $\{p_1^n, p_2^n, \ldots\}$ internos al nodo. Por el contrario, los nodos $n \in N$, cuyas funciones dependen únicamente de las entradas, se denominan* **no adaptativos**.

En general, al representar gráficamente redes adaptativas, se utilizan rectángulos para los nodos adaptativos y círculos para los nodos no adaptativos.

Definición 39 *Una red adaptativa es* **feed-forward** *si no existe un ciclo en el grafo que la define. En otro caso, la red se denomina* **recurrente**.

Ejemplo 29 *Perceptrón multicapa. Un perceptrón multicapa puede modelarse fácilmente como una red adaptativa feed-forward cuyo grafo subyacente coincide con el del perceptrón. Todos los nodos de la red son adaptativos. El nodo genérico* **n** *tiene* $|n|$ *entradas* $\{I_1^n, \ldots, I_{|n|}^n\}$ *y sus parámetros son* $\{w_1^n, \ldots, w_{|n|}^n, T^n\}$ *que representan los valores de las sinapsis y el threshold del perceptrón. La función del nodo n se define como:*

$$F_n = g\left(\sum_{i=1}^{|n|} w_i^n I_i^n - T^n\right)$$

donde g la función de activación del perceptrón.

Definición 40 *La* **representación por capas** *de una red adaptativa feed-forward asigna a cada nodo de la red una capa* L_i, $i > 0$, *con la condición que para todo arco* (n_1, n_2), *la capa de* n_2 *sea estrictamente mayor que la de* n_1. *Un* **ordenamiento topológico** *de una red adaptativa feed forward es una secuencia de los nodos de la red donde se verifica que si* (n_1, n_2) *es un arco, entonces* n_1 *precede a* n_2 *en la secuencia.*

6.1.1 Mecanismo de aprendizaje

Conceptualmente, una red adaptativa feed-forward brinda un mapeo estático entre los espacios de entrada y los de salida. El objetivo al construir redes adaptativas es imitar el mapeo no lineal de algún sistema real, utilizando para ello un conjunto de entrenamiento dado. El mecanismo de aprendizaje especifica cómo deben ser modificados los parámetros de los nodos adaptativos de la red para minimizar una medida preestablecida del error entre sus salidas y las deseadas. La regla básica de aprendizaje de las redes adaptativas es la del descenso por el gradiente de la sección 5.2.2, donde el vector gradiente es calculado mediante la aplicación sucesiva de la regla de la cadena [31] [3]. Suponiendo que el conjunto de entrenamiento está compuesto por P patrones, se define una medida del error para el p-ésimo patrón $(\{x_1^p, \ldots, x_m^p\}, \{y_1^p, \ldots, y_n^p\})$ de la siguiente manera:

$$E_p = \sum_{k=1}^{n} (y_k^p - O_{s_k}^p)^2 \tag{6.1}$$

donde $\{O_{s_1}^p, \ldots, O_{s_n}^p\}$ son las salidas que produce la red cuando se le aplica la p-ésima entrada $\{x_1^p, \ldots x_m^p\}$. El objetivo entonces, es minimizar la función error, definida como $E = \sum_{p=1}^{P} E_p$.

Para sistematizar la obtención de ∇E se dan las siguientes definiciones:

Definición 41 *Sea* $R = (V, E)$ *una red adaptativa feed-forward. Se define* **la derivada indirecta del nodo** n **con respecto a la salida del nodo** m, *denotada* $\frac{\partial^+ n}{\partial m}$ *de la siguiente manera:*

$$\frac{\partial^+ n}{\partial m} = \begin{cases} 0 & \text{si } fan\text{-}in(n) = 0. \\ 1 & \text{si } n = m. \\ \sum_{y \in Y} \frac{\partial F_n}{\partial y} \frac{\partial^+ y}{\partial m} & \text{en otro caso, donde } Y = \{y \in V/(y,n) \in E\} \end{cases}$$

donde, por simplicidad, se utiliza **n** *para denotar tanto a un nodo genérico como a la variable que representa su salida.*

Se puede ver a la derivada parcial indirecta como la aplicación sucesiva de la regla de la cadena a través de la estructura de la red. En efecto, como se ve en el ejemplo de la figura 6.1(a), si se rotula cada arco (m, n) de la red con el valor funcional $\frac{\partial F_m}{\partial n}$, se puede demostrar que $\frac{\partial^+ m}{\partial n}$ es igual a la suma, sobre todos los caminos que conectan el nodo m con el nodo n, del producto de los rótulos de los arcos que forman cada camino.

La definición 41 se puede extender para el caso derivadas indirectas de funciones arbitrarias definidas a partir de algunas de las salidas de la red, y con respecto a parámetros internos de los nodos.

Definición 42 *Sea $R = (V, E)$ una red adaptativa feed forward, sea $O \subset V$ un conjunto de nodos de R y sea G una función arbitraria diferenciable que posee, entre sus variables, las salidas de los nodos de O. Se define la* **derivada parcial indirecta de G con respecto al nodo** *n de la siguiente manera:*

$$\frac{\partial^+ G}{\partial n} = \frac{\partial G}{\partial n} + \sum_{y \in O} \frac{\partial G}{y} \frac{\partial^+ y}{n} \tag{6.2}$$

Definición 43 *Sea $R = (V, E)$ una red adaptativa feed-forward y G una función diferenciable. La* **derivada indirecta de G con respecto al parámetro** *p se define de la siguiente manera:*

$$\frac{\partial^+ G}{\partial p} = \frac{\partial G}{\partial p} + \sum_{n \in N} \frac{\partial^+ G}{\partial n} \frac{\partial F_n}{\partial p}$$

donde $N = \{n \in V \ / \ p$ es parámetro de $n\}$. Debido a que los parámetros pueden compartirse por varios nodos, puede darse el caso que $|N| > 1$.

Utilizando las definiciones anteriores se puede expresar el vector gradiente en forma concisa. Si el conjunto de parámetros a ser modificados es $\{\alpha_1, \ldots, \alpha_n\}$, se tiene que

$$\nabla E_p = \left[\frac{\partial^+ E_p}{\partial \alpha_1} \cdots \frac{\partial^+ E_p}{\partial \alpha_n} \right] \tag{6.3}$$

6.1.2 Backpropagation

La derivada indirecta, gracias a las propiedades de asociatividad, conmutatividad y distributividad de la suma y el producto, puede calcularse de varias maneras diferentes. Como se mencionó en la sección 6.1.1, una definición no recursiva de la derivada indirecta es la suma, sobre todos los caminos que conectan a los nodos involucrados, del producto de los rótulos de los arcos que forman cada camino. Se puede dar entonces, otra definición alternativa, denominada **derivada B** que, aunque es conceptualmente idéntica a la derivada indirecta, tiene grandes ventajas cuando se la utiliza en implementaciones concretas.

Definición 44 *Sea $R = (V, E)$ una red adaptativa feed-forward. Se define* **la derivada B del nodo n con respecto a la salida del nodo** *m, denotada $\frac{\partial^B n}{\partial m}$ de la siguiente manera:*

$$\frac{\partial^B n}{\partial m} = \begin{cases} 0 & \text{si } fan\text{-}out(m) = 0. \\ 1 & \text{si } n = m. \\ \sum_{y \in Y} \frac{\partial^B n}{\partial y} \frac{\partial F_y}{\partial m} & \text{en otro caso, donde } Y = \{y \in V / (m, y) \in E\} \end{cases}$$

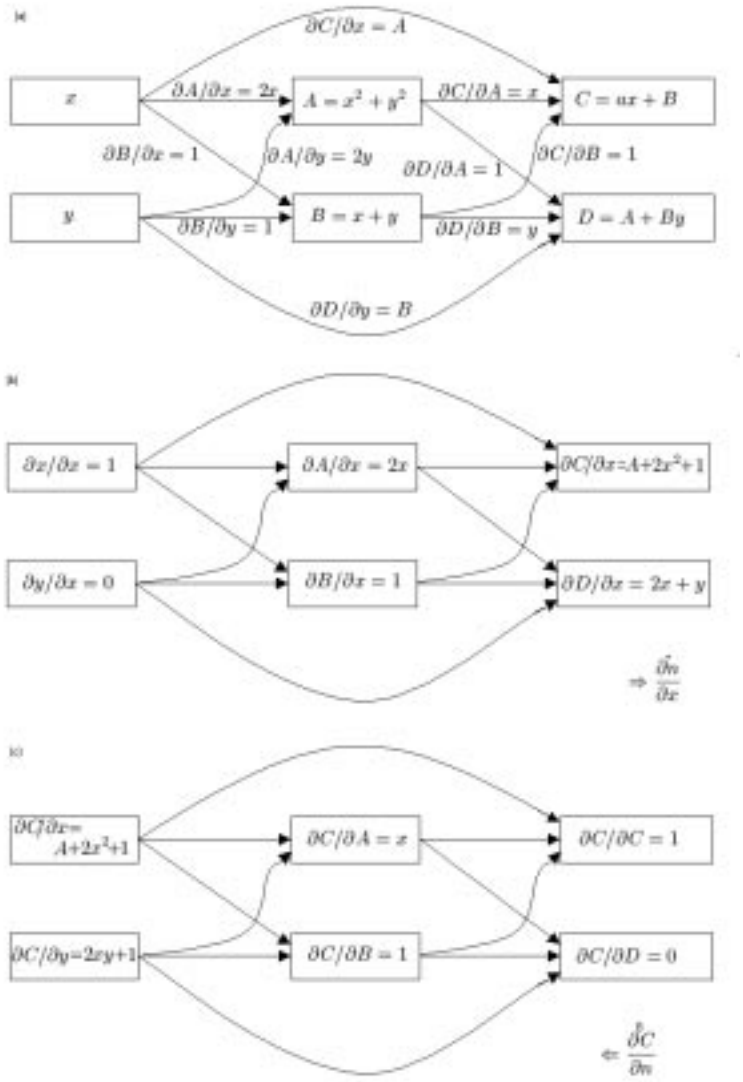

Figura 6.1: (a) Red neuronal con arcos etiquetados. (b) Cálculo secuencial de $\frac{\partial^+ n}{\partial x}$. (c) Cálculo secuencial de $\frac{\partial^B C}{\partial n}$.

Las definiciones 42 y 43 se pueden modificar para el caso de ∂^B.

Definición 45 *Sea $R = (V, E)$ una red adaptativa feed forward y sea G una función diferenciable. Se define:*

1) $\frac{\partial^B G}{\partial n} = \frac{\partial G}{\partial n} + \sum_{y \in Y} \frac{\partial^B G}{y} \frac{\partial F_y}{n}$ *con* $Y = \{y \in V/(n, y) \in E\}$

2) $\frac{\partial^B G}{\partial p} = \frac{\partial G}{\partial p} + \sum_{n \in N} \frac{\partial^B G}{\partial n} \frac{\partial F_n}{\partial p}$ *con* $N = \{n \in V/p$ *es parámetro de* $n\}$

Debido a que los parámetros pueden compartirse por varios nodos, puede darse el caso que $|N| > 1$.

En implementaciones reales, donde se necesitan calcular las derivadas parciales de la función error con respecto a los diferentes parámetros, es importante aprovechar los resultados intermedios para no repetir cálculos y degradar el rendimiento del sistema. En el siguiente ejemplo se pone de manifiesto la superioridad del enfoque ∂^B sobre el ∂^+ en este tipo de situaciones.

Ejemplo 30 *La figura 6.1(b) muestra el procedimiento utilizado para calcular las derivadas parciales de cada nodo con respecto al nodo x utilizando la noción de derivada indirecta (∂^+). Tomando un ordenamiento topológico de la red de la figura 6.1(a), como por ejemplo $[x, y, A, B, C, D]$, se calcula progresivamente la derivada parcial de cada nodo n con respecto al nodo x ($\partial^+ n/\partial x$). Como la red es feed-forward, todos los pasos utilizan información que ha sido definida. Por otra parte, en la figura 6.1(c) se utiliza la derivada B (∂^B) para calcular la derivada parcial del nodo C con respecto al resto de los nodos, realizando, en este caso, el cálculo a través de la secuencia topológica invertida $[D, C, B, A, y, x]$. Se puede ver que el resultado final en el nodo x de la figura 6.1(b) y el nodo C de la figura 6.1(c) son los mismos, como es esperado. Utilizando la derivada indirecta, es relativamente sencillo y eficiente calcular $\partial^+ n/\partial x$ para varios nodos n de salida (como por ejemplo el nodo D), ya que se puede reutilizar gran parte de la información calculada. En cambio, para obtener, por ejemplo, $\partial^+ C/\partial y$, se debe realizar nuevamente todo el procedimiento. Justamente a la inversa se comporta ∂^B, y como en general, en las redes adaptativas hay una sola medida del error y necesidad de calcular $\partial E/\partial p$ para una gran cantidad de parámetros p, en implementaciones reales se utiliza casi exclusivamente la definición ∂^B.*

Al proceso de encontrar el vector gradiente en la estructura de la red utilizando la derivada B, se lo denomina **backpropagation** [23], debido a que el vector gradiente se calcula progresivamente en la dirección opuesta a la de los arcos que componen la red.

6.2 El modelo ANFIS

El modelo ANFIS (*Adaptive Neuro-based Fuzzy Inference System*) fue desarrollado por J.R. Jang en 1993 [14] [16] y es funcionalmente equivalente a los sistemas de inferencia difusos. Las capacidades adaptativas de las redes ANFIS las hacen directamente aplicables a una gran cantidad de áreas como control adaptativo, procesamiento y filtrado de señales, clasificación de datos y extracción de características a partir de ejemplos. Una propiedad interesante del modelo, es que el conjunto de parámetros se puede descomponer para utilizar una regla de aprendizaje híbrido más eficiente que los mecanismos tradicionales.

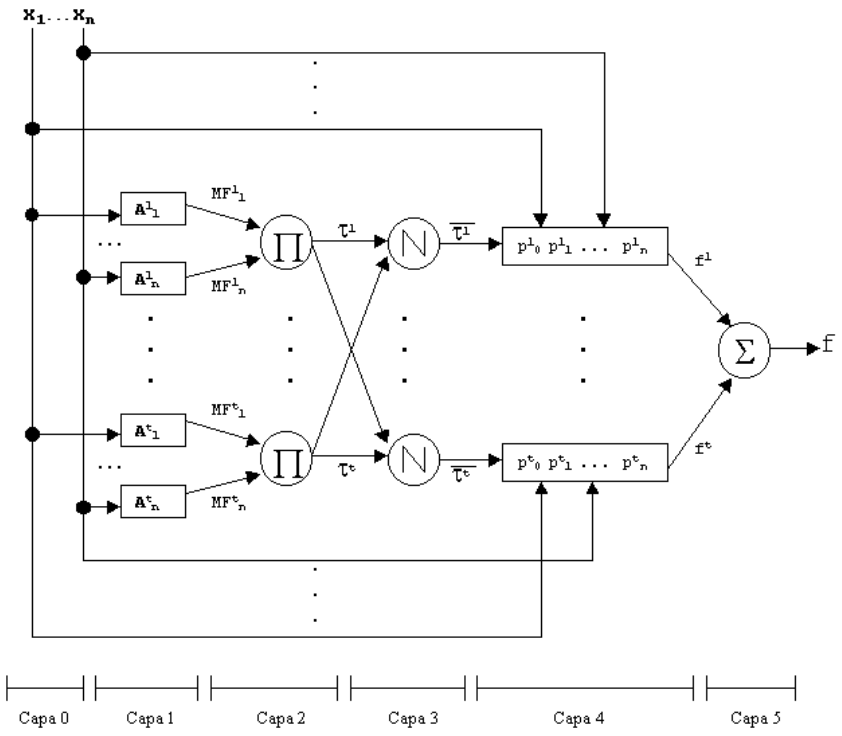

Figura 6.2: Arquitectura ANFIS

6.2.1 Arquitectura

Los diferentes modelos de inferencia difusos (sección 4.2), dan lugar a diferentes arquitecturas ANFIS. El desarrollo a continuación se corresponde con el modelo de inferencia TSK de la sección 4.2.3, debido su transparencia y eficiencia. Para arquitecturas ANFIS correspondientes o otros modelos (Mamdani o Tsukamoto), se pueden consultar [14] [16]. La clase de modelos TSK se definen mediante un conjunto de reglas con la forma:

`if (x1 is` A_1^1`) and ... and (xn is` A_n^1 `), then y1=` p_0^1 `+` $\sum_{k=1}^{n} p_k^1$ `x_k`
`also`
`...`
`also`
`if (x1 is` A_1^t`) and ... and (xn is` A_n^t `), then yt=` p_0^t `+` $\sum_{k=1}^{n} p_k^t$ `x_k`

La figura 6.2 presenta la arquitectura ANFIS equivalente al mecanismo de razonamiento utilizado para este modelo, donde los nodos situados en la misma capa realizan funciones similares, y son explicados a continuación.

- Capa 0: Corresponde a las entradas $x_1, \ldots, x_n$.
- Capa 1: Los nodos de esta capa son adaptativos, y la función MF_j^i puede definirse de varias maneras, teniendo en cuenta que debe ser una función de membresía diferenciable.

Las dos formas más comunes de definir MF_j^i son:

$$\mathrm{MF}_j^i = \frac{1}{1 + \left|\frac{x_j - c_j^i}{a_j^i}\right|^{2b_j^i}} \qquad i = 1..t, \quad j = 1..n \tag{6.4}$$

donde x_i es la entrada y $\{a_j^i, b_j^i, c_j^i\}$ es el conjunto de parámetros.

$$\mathrm{MF}_j^i = e^{-\left(\frac{x_j - c_j^i}{a_j^i}\right)^2} \qquad i = 1..t, \quad j = 1..n \tag{6.5}$$

donde x_i es la entrada y $\{a_j^i, c_j^i\}$ es el conjunto de parámetros.

- Capa 2: Cada nodo de esta capa es no adaptativo y se define la salida $\tau^i = \tau^i(\mathrm{MF}_1^i, \ldots, \mathrm{MF}_n^i)$ como el producto de sus entradas:

$$\tau^i = \prod_{j=1}^{n} \mathrm{MF}_j^i \qquad i = 1..t \tag{6.6}$$

  Cada salida τ^i corresponde al DOF de la i-ésima regla, y en general, se puede utilizar otro operador T-norm en lugar del producto.

- Capa 3: Los nodos de esta capa son no adaptativos. Se define la salida $\overline{\tau^i} = \overline{\tau^i}(\tau^1, \ldots, \tau^t)$ como la razón entre el DOF de la i-ésima regla y la suma de los DOF de todas las reglas, denominado **DOF normalizado**:

$$\overline{\tau^i} = \frac{\tau^i}{\sum_{k=1}^{t} \tau^k} \qquad i = 1..t \tag{6.7}$$

- Capa 4: Cada nodo de esta capa es adaptativo y sus parámetros son $\{p_0^i, \ldots, p_n^i\}$. La salida $f^i = f^i(\overline{\tau^i}, x_1, \ldots, x_n, p_0^i, \ldots, p_n^i)$ se corresponde con la salida parcial de la i-ésima regla,

$$f^i = \overline{\tau^i}\left(p_0^i + p_1^i x_1 \ldots + p_n^i x_n\right) \tag{6.8}$$

- Capa 5: El nodo de esta capa es no adaptativo y su salida $f = f(f^1, \ldots, f^t)$ se define como la sumatoria de las salidas parciales f^i:

$$f = \sum_{i=1}^{t} f^i \tag{6.9}$$

Frecuentemente los parámetros de los nodos de la capa 1 se denominan **parámetros de las premisas**, y los parámetros de los nodos de la capa 4 se denominan **parámetros de los consecuentes**, ya que se corresponden con las premisas y los consecuentes de las reglas `IF/THEN` del modelo de inferencia TSK.

6.2.2 Algoritmo de aprendizaje híbrido

Se puede observar en la figura 6.2 que, si los parámetros de las premisas quedan fijos, la salida del sistema se puede expresar como una combinación lineal de los parámetros de los consecuentes. La salida f puede ser reescrita entonces de la siguiente manera:

$$\begin{aligned} f \quad = \quad & \overline{\tau^1}(p_0^1 + p_1^1 x_1 + \ldots + p_n^1 x_n) \\ & + \ldots + \\ & \overline{\tau^t}(p_0^t + p_1^t x_1 + \ldots + p_n^t x_n) \end{aligned} \tag{6.10}$$

y esta ecuación puede reescribirse como:

$$\begin{aligned} f = \quad & (\overline{\tau^1})p_0^1 + (\overline{\tau^1} x_1)p_1^1 + \ldots + (\overline{\tau^1} x_n)p_n^1 \\ & + \ldots + \\ & (\overline{\tau^t})p_0^t + (\overline{\tau^t} x_1)p_1^t + \ldots + (\overline{\tau^t} x_n)p_n^t \end{aligned} \tag{6.11}$$

que es lineal en los parámetros de los consecuentes p_j^i $(i = 1..t, j = 0..n)$.

Se puede entonces combinar el descenso por el gradiente con el método de cuadrados mínimos para modificar los parámetros de la red [13]. Para ello, cada época está compuesta por una pasada hacia adelante y una pasada hacia atrás. En la pasada hacia adelante, para cada vector de entrada, se evalúa la red hasta la capa 4, y los parámetros de los consecuentes son identificados mediante el método de cuadrados mínimos. A continuación, se calculan los errores para cada par del conjunto de entrenamiento y, en la pasada hacia atrás se propagan las señales del error y los parámetros de las premisas son modificados por el mecanismo clásico de backpropagation. La siguiente tabla resume este procedimiento:

	Hacia adelante	Hacia atrás
Premisas	Fijas	Descenso por el gradiente
Consecuentes	Cuadrados mínimos	Fijos
Señales	Salidas	Errores

Cabe destacar que los parámetros de los consecuentes identificados por el método de cuadrados mínimos son óptimos bajo la condición que los parámetros de los antecedentes estén fijos. Este enfoque híbrido, por lo tanto, converge más rápidamente que el método original de backpropagation, debido a que se reducen las dimensiones del espacio de búsqueda.

Aprendizaje de antecedentes

Debido a que las redes ANFIS poseen una única salida, la medida del error para el p-ésimo patrón de entrenamiento definido en la ecuación 6.1 se reescribe de la siguiente manera:

$$E_p = (y_p - f_p)^2 \tag{6.12}$$

El método del descenso por el gradiente de la sección 5.2.2, junto con las consideraciones de implementación de la sección 6.1.2 para el caso de redes adaptativas, establecen que el cambio en cada iteración de un parámetro genérico α_j^i está regulado por la ecuación:

$$\Delta\alpha_j^i = -\eta\frac{\partial^B E}{\partial\alpha_j^i} \qquad i = 1..t, \quad j = 1..n \tag{6.13}$$

$$\text{donde} \qquad \frac{\partial^B E}{\partial\alpha_j^i} = \sum_{p=1}^{P}\frac{\partial^B E_p}{\partial\alpha_j^i} \tag{6.14}$$

En particular, si se utiliza la ecuación 6.4 para las funciones MF_j^i, entonces $\alpha_j^i \in \{a_j^i, b_j^i, c_j^i\}$. Si en cambio se utiliza la ecuación 6.5, $\alpha_j^i \in \{a_j^i, c_j^i\}$.

A continuación se calcula $\frac{\partial^B E_p}{\partial\alpha_j^i}$ donde, para mayor claridad, se omiten los subíndices p en el resto de las expresiones. Aplicando la definición 45-2, se tiene:

$$\frac{\partial^B E_p}{\partial\alpha_j^i} = \frac{\partial^B E_p}{\partial\mathrm{MF}_j^i}\frac{\partial\mathrm{MF}_j^i}{\partial\alpha_j^i} \quad i = 1..t, \quad j = 1..n \tag{6.15}$$

donde $\frac{\partial\mathrm{MF}_j^i}{\partial\alpha_j^i}$ depende de la función de membresía elegida. Si, por ejemplo, MF_j^i está dada por la ecuación 6.4, entonces:

$$\frac{\partial\mathrm{MF}_j^i}{\partial a_j^i} = \frac{-2\ \omega\ b_j^i}{a_j^i} \tag{6.16}$$

$$\frac{\partial\mathrm{MF}_j^i}{\partial b_j^i} = \begin{cases} -2\ \omega\ \ln\left|\frac{x_j - c_j^i}{a_j^i}\right| & \text{si } x_j \neq c_j^i \\ 0 & \text{en otro caso.} \end{cases} \tag{6.17}$$

$$\frac{\partial\mathrm{MF}_j^i}{\partial c_j^i} = \begin{cases} \frac{-2\ \omega\ b}{x_j - c_j^i} & \text{si } x_j \neq c_j^i \\ 0 & \text{en otro caso.} \end{cases} \tag{6.18}$$

donde $\omega = \mathrm{MF}_j^i(1 - \mathrm{MF}_j^i)$.

Si, en cambio, MF_j^i está dado por la ecuación 6.5, se tiene que:

$$\frac{\partial\mathrm{MF}_j^i}{\partial a_j^i} = \frac{(x_j - c_j^i)^2}{(a_j^i)^3}\mathrm{MF}_j^i \tag{6.19}$$

$$\frac{\partial\mathrm{MF}_j^i}{\partial c_j^i} = \frac{x_j - c_j^i}{(a_j^i)^2}\mathrm{MF}_j^i \tag{6.20}$$

Aplicando ahora la definición 45-1, se obtiene:

$$\frac{\partial^B E_p}{\partial\mathrm{MF}_j^i} = \frac{\partial^B E_p}{\partial\tau^i}\frac{\partial\tau^i}{\partial\mathrm{MF}_j^i} \tag{6.21}$$

$$\text{donde} \quad \frac{\partial\tau^i}{\partial\mathrm{MF}_j^i} = \prod_{k=1,k\neq j}^{n}\mathrm{MF}_k^i = \frac{\tau^i}{\mathrm{MF}_j^i} \qquad i = 1..t, \quad j = 1..n \tag{6.22}$$

$$\text{y a su vez} \quad \frac{\partial^B E_p}{\partial\tau^i} = \sum_{k=1}^{t}\frac{\partial^B E_p}{\partial\overline{\tau^k}}\frac{\partial\overline{\tau^k}}{\partial\tau^i} \tag{6.23}$$

$$\text{donde} \quad \frac{\partial \overline{\tau^k}}{\partial \tau^i} = \frac{\delta_{k,i}(\sum_{m=1}^{t} \tau^m) - \tau^k}{\left(\sum_{m=1}^{t} \tau^m\right)^2} = \frac{\delta_{k,i} - \overline{\tau^k}}{\sum_{m=1}^{t} \tau^m} \quad k = 1..t, \quad i = 1..t \tag{6.24}$$

$$\text{y} \quad \frac{\partial^B E_p}{\partial \overline{\tau^k}} = \frac{\partial^B E_p}{\partial f^k} \frac{\partial f^k}{\partial \overline{\tau^k}} \tag{6.25}$$

$$\text{donde} \quad \frac{\partial f^k}{\partial \overline{\tau^k}} = p_0^k + p_1^k x_1 + \ldots + p_n^k x_n = \wp^k \qquad k = 1..t \tag{6.26}$$

$$\text{y por último} \quad \frac{\partial^B E_p}{\partial f^k} = \frac{\partial^B E_p}{\partial f} \frac{\partial f}{\partial f^k} \tag{6.27}$$

$$\text{donde} \quad \frac{\partial f}{\partial f^k} = 1 \qquad k = 1..t \tag{6.28}$$

$$\text{y, como } fan\text{-}out(f) = 0, \quad \frac{\partial^B E_p}{\partial f} = \frac{\partial E_p}{\partial f} = -2(y - f) \tag{6.29}$$

Para calcular $\frac{\partial^B E_p}{\partial \alpha_j^i}$ se combinan todos los resultados parciales obtenidos hasta este punto. Reemplazando entonces 6.28 y 6.29 en 6.27 se obtiene:

$$\frac{\partial^B E_p}{\partial f^k} = -2(y - f) \tag{6.30}$$

Reemplazando esta ecuación, junto con 6.26, en 6.25:

$$\frac{\partial^B E_p}{\partial \overline{\tau^k}} = -2\wp^k(y - f) \tag{6.31}$$

Esta ecuación, junto con 6.24 se reemplazan en 6.23 para obtener:

$$\frac{\partial^B E_p}{\partial \tau^i} = \frac{-2(y - f)}{\sum_{m=1}^{t} \tau^m} \sum_{k=1}^{t} \wp^k (\delta_{k,i} - \overline{\tau^k}) \tag{6.32}$$

que se puede escribir, distribuyendo $\wp^k$, como

$$\frac{\partial^B E_p}{\partial \tau^i} = \frac{-2(y - f)}{\sum_{m=1}^{t} \tau^m} (\wp^i - f) \tag{6.33}$$

Reemplazando esta ecuación junto con 6.22 en 6.21:

$$\frac{\partial^B E_p}{\partial \mathrm{MF}_j^i} = \frac{-2(y - f)(\wp^i - f)\tau^i}{\mathrm{MF}_j^i \sum_{m=1}^{t} \tau^m} = \frac{-2(y - f)(\wp^i - f)\overline{\tau^i}}{\mathrm{MF}_j^i} \tag{6.34}$$

Para terminar, reemplazando esta última ecuación en 6.15, se llega a:

$$\frac{\partial^B E_p}{\partial \alpha_j^i} = -2(y - f)(\wp^i - f)\overline{\tau^i}\Delta_j^i \tag{6.35}$$

$$\text{donde} \quad \Delta_j^i = \frac{1}{\mathrm{MF}_j^i} \frac{\partial \mathrm{MF}_j^i}{\partial \alpha_j^i}$$

depende de la elección particular de las funciones de membresía MF_j^i.

Parámetros compartidos

En algunas ocasiones, los parámetros de una red pueden compartirse por varios nodos adaptativos. En particular, cada parámetro α_j^i puede pertenecer a más de un nodo de la capa 1. En estos casos, utilizando la definición 45-2, la ecuación 6.15 se generaliza a:

$$\frac{\partial^B E_p}{\partial \alpha_j^i} = \sum_{k,l} \frac{\partial^B E_p}{\partial \mathrm{MF}_l^k} \frac{\partial \mathrm{MF}_l^k}{\partial \alpha_j^i} \tag{6.36}$$

para todos los k, l tales que MF_l^k utiliza en su definición al parámetro α_j^i. Reemplazando la ecuación 6.34 en 6.36, se llega a:

$$\frac{\partial^B E}{\partial \alpha_j^i} = \sum_{k,l} \frac{\partial^B E}{\partial \mathrm{MF}_l^k} \frac{\partial \mathrm{MF}_l^k}{\partial \alpha_j^i} = -2(y - f) \sum_{k,l} (\wp^k - f) \overline{\tau^k} \Delta_{l,j}^{k,i} \tag{6.37}$$

$$\text{donde} \qquad \Delta_{l,j}^{k,i} = \frac{1}{\mathrm{MF}_l^k} \frac{\partial \mathrm{MF}_l^k}{\partial \alpha_j^i}$$

para todos los k, l tales que MF_l^k utiliza en su definición al parámetro α_j^i.

Combinación de LSE y backpropagation

Aunque la aproximación por cuadrados mínimos acelera el proceso de aprendizaje, es computacionalmente costosa. Existen algunos mecanismos alternativos para modificar los parámetros, y a continuación se listan algunos de ellos ordenados de acuerdo a su complejidad computacional:

1. Unicamente descenso por el gradiente: Todos los parámetros se modifican de acuerdo al descenso por el gradiente.

2. Descenso por el gradiente y una pasada de LSE: El método de cuadrados mínimos se aplica solamente en la primera época para estimar los valores iniciales de los consecuentes y en las siguientes etapas se modifican los parámetros por el método del gradiente.

3. Enfoque híbrido: El método propuesto en la sección 6.2.2.

4. Únicamente aproximación por cuadrados mínimos: Las salidas de la red son linealizadas con respecto a todos los parámetros y se utiliza el método de cuadrados mínimos. Este mecanismo ha sido propuesto en la literatura de redes neuronales [24] [25].

Capítulo 7

Aplicaciones

Este capítulo presenta brevemente algunos resultados de simulación obtenidos utilizando el modelo ANFIS del capítulo 6. En la sección 7.1 se aproxima la función no lineal $sinc : \mathbb{R}^2 \to \mathbb{R}$ [14]. En la sección 7.2, se utilizan mecanismos de clustering para modelar una función simple. En la sección 7.3 se aproxima una serie temporal conocida como **Box Jenkins Gas Furnace**. Finalmente, en la sección 7.4 se modela un problema real de vivienda realizando selección de variables.

7.1 Aproximación de una función no lineal

Se utiliza el modelo ANFIS para aproximar la función no lineal $sinc : \mathbb{R}^2 \to \mathbb{R}$, definida como:

$$sinc(x, y) = \frac{sin(x)sin(y)}{xy} \tag{7.1}$$

El conjunto C de entrenamiento consiste en 121 patrones dispuestos en una grilla de la siguiente forma:

$$\begin{aligned} C = \{(x, y, z) \ / \quad & x \in \{-10, -9, \ldots, 9, 10\}, \\ & y \in \{-10, -9, \ldots, 9, 10\}, \\ & z = sinc(x, y) \ \} \end{aligned} \tag{7.2}$$

El modelo ANFIS se construye utilizando 4 funciones de membresía $bell_{a,b,c}$ para cada variable de entrada, igualmente espaciadas a lo largo del rango del conjunto de entrenamiento. Los parámetros de las funciones de membresía valen:

x			y		
a_i	b_i	c_i	a_i	b_i	c_i
1/3	2	−10	1/3	2	−10
1/3	2	−3	1/3	2	−3
1/3	2	3	1/3	2	3
1/3	2	10	1/3	2	10

Estos parámetros se comparten, combinando las funciones de membresía de las variables x e y, produciendo de esta manera 16 reglas de inferencia.

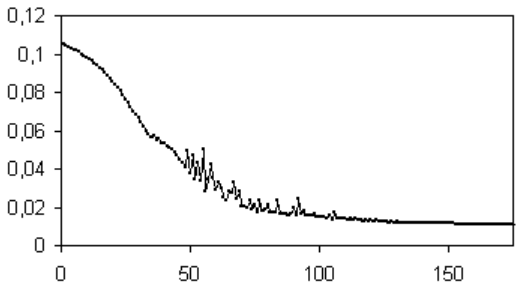

Figura 7.1: Error cuadrático medio para $sinc(x, y)$.

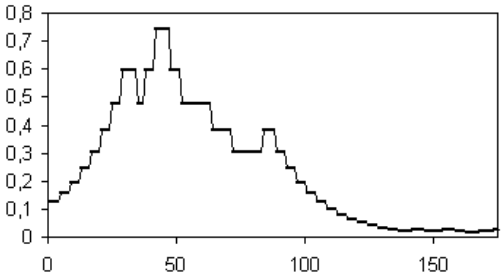

Figura 7.2: Variación del parámetro κ.

El número de coeficientes lineales de los consecuentes es 48, elevando a 72 el número total de parámetros ajustables. La figura 7.1 muestra el error cuadrático medio (ver ecuación 7.4) para el modelo ANFIS a lo largo de las 200 épocas del entrenamiento. En la figura 7.2 se puede observar la variación durante la etapa de aprendizaje del parámetro κ, correspondiente a la longitud del paso del método del gradiente en el mecanismo de aprendizaje híbrido.

7.2 Aproximación utilizando clustering

En esta sección se aproxima una función no lineal $f : \mathbb{R} \to \mathbb{R}$, utilizando para ello un mecanismo de clustering. Se ha seleccionado una función f con una sola variable de entrada para visualizar más fácilmente los clusters identificados. Esta función se define de la siguiente manera:

$$f(x) = \frac{0.9x}{1.2x^3 + x + 0.3} + \eta \tag{7.3}$$

donde η es una componente aleatoria con amplitud 0.15. La figura 7.3 muestra en forma gráfica al conjunto de entrenamiento, compuesto por 200 patrones elegidos aleatoriamente en el intervalo $[0, 1]$. El mecanismo de clustering utilizado para la identificación de la estructura del sistema de inferencia difuso es el Fuzzy C-Means Clustering de la sección 5.1.4. Para el número total de clusters y los valores iniciales se utilizan los resultados obtenidos luego de aplicar el método Substractive Clustering con parámetros $\alpha = 1$, $\beta = 2$ y $\delta = 0.01$.

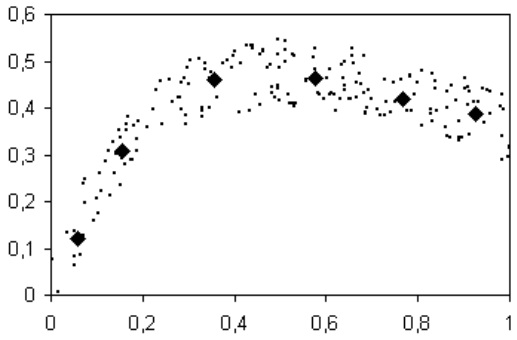

Figura 7.3: Clusters identificados para la función f.

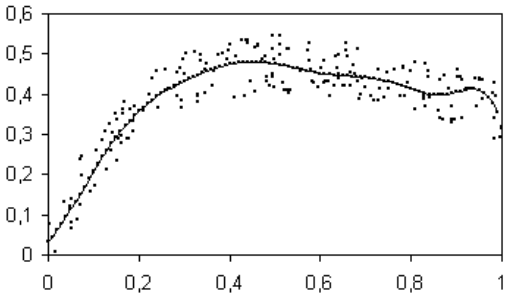

Figura 7.4: Aproximación de la función f.

De esta forma se obtienen 6 clusters que se pueden ver gráficamente en la figura 7.3 y se listan a continuación:

c_x	c_y
0.575	0.462
0.156	0.307
0.926	0.386
0.058	0.121
0.355	0.459
0.766	0.418

Luego de entrenar la red ANFIS durante 200 épocas se llega al modelo de la figura 7.4. Si se escriben los resultados obtenidos con el esquema del modelo de inferencia TSK de la sección 4.2.3, se obtiene el siguiente conjunto de reglas de inferencia `IF/THEN`:

`if` x `is` $Bell_{0.798,2.083,0.095}$, `then y =` $18.49x + 6.29$ `,also`
`if` x `is` $Bell_{0.531,2.205,0.278}$, `then y =` $2.73x - 13.97$ `,also`
`if` x `is` $Bell_{0.433,2.291,0.614}$, `then y =` $-6.02x - 9.77$ `,also`
`if` x `is` $Bell_{0.499,2.562,0.673}$, `then y =` $17.3x + 7.02$ `,also`
`if` x `is` $Bell_{1.069,2.056,0.847}$, `then y =` $-12.34x + 6.91$ `,also`
`if` x `is` $Bell_{0.961,2.048,1.007}$, `then y =` $-17.44x + 5.35$

7.3 Identificación de series temporales

El **Box-Jenkins Gas Furnace** [2] es un ejemplo muy conocido de identificación de series temporales. El conjunto de entrenamiento está formado por 296 pares de entrada-salida $[u(t), y(t)]$ observados, donde la variable $u(t)$ representa el flujo de gas de entrada al horno y la variable $y(t)$ es la concentración de CO_2 en los gases de salida. El índice considerado para evaluar el modelo ANFIS es el error cuadrático medio (E_{RMS}) definido como:

$$E_{RMS} = \sqrt{\frac{\sum_{p=1}^{N}(y_p - f_p)^2}{N}} \tag{7.4}$$

donde N es el número de patrones, y_p es la salida observada, y f_p es la salida obtenida por el modelo para el p-ésimo patrón.

Existen diferentes conjuntos de variables que pueden ser candidatos para modelar la salida $y(t)$. Varios autores [36] han elegido al conjunto de variables de entrada $\{u(t\text{-}4), y(t\text{-}1)\}$ para aproximar $y(t)$. La siguiente tabla muestra algunos resultados conocidos en la literatura y los resultados obtenidos utilizando el modelo ANFIS.

Modelo	Variables	Reglas	RMSE
Tong(77)	2	19	0.684
Pedrycz(84)	2	81	0.565
Xu(87)	2	25	0.572
Peng(88)	2	49	0.548
Sugeno(91)	6	2	0.261
Sugeno(93)	3	6	0.435
Wang(96)	2	5	0.397
GMBD(97)	2	2	0.396
ANFIS	2	2	0.387
ANFIS	6	2	0.229

El conjunto de reglas proviene directamente del mecanismo de clustering descripto en la sección 7.2. Una tabla más completa, describiendo diferentes elecciones para los conjuntos de variables de entrada y diferentes números de clusters identificados se presenta a continuación:

Clusters	2 Variables[1]	4 Variables[2]	6 Variables[3]
2	0.387	0.241	0.229
4	0.349	0.219	0.209
8	0.325	0.179	0.171
16	0.306	0.142	0.135

([1]) correspondientes a $u(t-4), y(t-1)$.
([2]) correspondientes a $u(t-4), u(t-3), y(t-2), y(t-1)$.
([3]) correspondientes a $u(t-4), u(t-3), u(t-2), y(t-3), y(t-2), y(t-1)$.

La figura 7.5 muestra las gráficas de la salida $y(t)$ original e inferida por el modelo para el caso de dos variables de entrada $\{u(t\text{-}4), y(t\text{-}1)\}$ y dos reglas de inferencia. Como ambas funciones son muy parecidas y no se pueden distinguir claramente entre sí, la figura 7.6 muestra, en otra escala, el error cuadrático

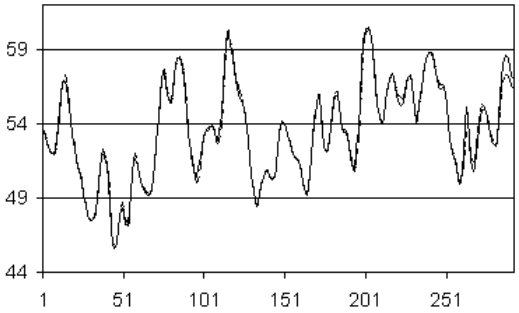

Figura 7.5: Aproximación de $y(t)$ a partir de $\{u(t-4), y(t-1)\}$.

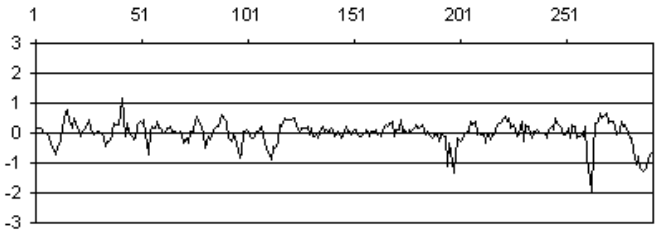

Figura 7.6: E_{RMS} entre $y(t)$ y la salida del modelo ANFIS.

medio entre las salidas observadas y las inferidas. En este caso, el modelo de inferencia TSK que se puede reconstruir a partir del modelo ANFIS obtenido luego de 250 épocas de entrenamiento se describe a continuación:

`if` $u(t-4)$ `is` $Bell_{\,0.24,2.31,-0.35}$ `and` $y(t-1)$ `is` $Bell_{\,0.47,2.21,51.05}$
`then` $y(t) = -1.33u(t-4) + 0.57y(t-1) + 22.59$
`also if` $u(t-4)$ `is` $Bell_{\,0.08,1.97,-1.13}$ `and` $y(t-1)$ `is` $Bell_{\,0.73,1.41,57.49}$
`then` $y(t) = -0.74u(t-4) + 0.67y(t-1) + 17.67$

7.4 Viviendas en el área de Boston

En esta sección se utiliza el modelo ANFIS en el marco de un problema de regresión no lineal denominado **housing** [11], y se realiza un preprocesamiento denominado **selección de variables** destinado a encontrar un subconjunto de variables representativas, con poder de predicción sobre la salida, con el objetivo de reducir la dimensión del conjunto de entrada y el costo computacional que su tamaño implica. El conjunto de entrenamiento, disponible en [30], muestra el valor medio, calculado en miles de dólares, de 506 viviendas modelo dentro del área de Boston junto con otras 12 características numéricas y una etiquetada. Para simplificar la presentación del ejemplo, se ha omitido la variable de entrada etiquetada del conjunto original, utilizando únicamente el conjunto de variables numéricas.

A continuación se lista el conjunto de variables de entrada considerado en el problema de housing:

1) Tasa de crimen per cápita de la ciudad.
2) Proporción de tierras residenciales por encima de 2300 m^2.
3) Proporción de acres de negocios al por mayor en la ciudad.
4) Concentración de óxido nítrico (partes cada 10 millones).
5) Número promedio de habitaciones por vivienda.
6) Proporción de unidades ocupadas construidas antes de 1940.
7) Distancia ponderada a 5 centros de empleo de Boston.
8) Indice de accesibilidad a autopistas radiales.
9) Tasa total de impuestos a la propiedad por cada 10000 dólares.
10) Proporción de estudiantes-profesores de la ciudad.
11) Indice: $1000(B-0.63)^2$, con B=Proporción de población negra.
12) Porcentaje de población de bajos recursos.

Existen dos problemas particulares asociados con este problema y se explican a continuación:

- Escasez de datos: Para un problema de ajuste de complejidad media con una variable de entrada, usualmente se necesitan 10 puntos de entrenamiento para llegar a un modelo aceptable. De la misma forma, para dos variables de entrada se necesitan aproximadamente 10^2=100 patrones de entrenamiento. El problema de housing tiene 12 variables de entrada, lo que hace necesario disponer de 10^{12} puntos de entrenamiento, pero se dispone de solamente 506. Esto equivaldría a tener $506^{1/12}$=1.68 puntos para el caso univariado. En estos casos, generalmente se divide el conjunto de datos en un conjunto de entrenamiento y un conjunto de chequeo donde el primero se utiliza para construir el modelo y el segundo para validarlo.

- Partición del espacio de entrada: aunque la partición en grilla es muy popular, para 12 entradas el mecanismo produce al menos 2^{12}=4096 reglas de inferencia que resultan en 4096*(12+1)=53248 coeficientes lineales. Este número de coeficientes ajustables es demasiado elevado, y esto resulta en un modelo que no generaliza de forma correcta. Para solucionar este problema se pueden aplicar técnicas de clustering para reducir el número de reglas, o seleccionar ciertas entradas con más poder de predicción en lugar de utilizar todas las variables (este último enfoque desarrollado a continuación).

Teniendo en cuenta las consideraciones anteriores, se dividen los 506 datos en un conjunto de entrenamiento y un conjunto de validación, cada uno con 253 patrones elegidos de forma aleatoria. Gracias al estimador de cuadrados mínimos utilizado en el modelo ANFIS, luego de unas pocas épocas, el rendimiento del modelo es representativo de su comportamiento final luego de completar el entrenamiento. En la figura 7.7 se muestran, de todas las posibilidades de elegir 2 variables entre las 12 originales, los 20 pares que obtienen el mejor rendimiento luego de completar 5 épocas. Se puede ver que el par de variables que tiene más poder de predicción sobre el precio de la vivienda es (5,12) que corresponde a **(Habitaciones por vivienda, población de bajos recursos)**.

En la figura 7.8 se muestran el error de entrenamiento y el error de validación durante las 200 épocas de entrenamiento realizadas sobre este par de variables de entrada.

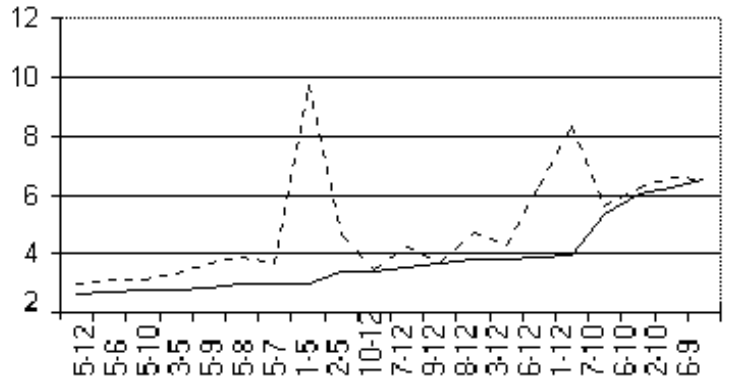

Figura 7.7: Diferentes variables de entrada para el problema de housing.

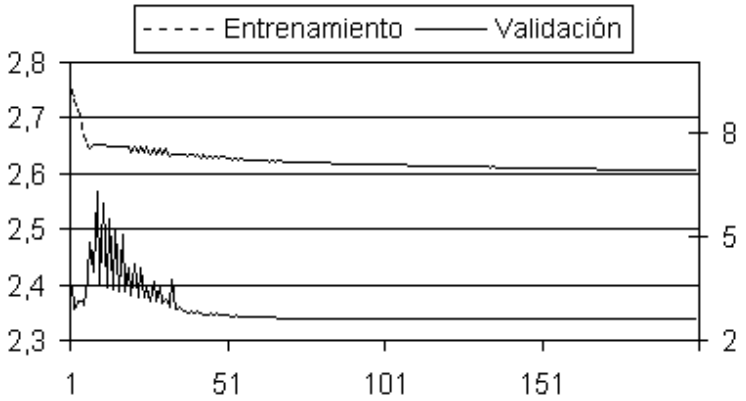

Figura 7.8: Entrenamiento para el par de variables (5,12).

Bibliografía

[1] J.C. Bezdek. *Pattern Recognition with Fuzzy Objetive Function Algorithms.* Plenum Press, New York, 1981.

[2] G. E. P. Box, G.M. Jenkins. *Time series analysis.* Forecasting and Control. pp.532-533, Holden Day, San Francisco, 1970.

[3] A.E. Bryson,Y.C. Ho. *Applied optimal control.* Blaisdell, New York, 1969.

[4] S.L. Chiu. *Fuzzy model identification based on cluster estimation.* Journal of Intelligent and Fuzzy Systems, 2(3), 1994.

[5] R. N. Dave. *Robust Fuzzy Clustering Algorithms.* Second IEEE International Conference on Fuzzy Systems. pp.1281-1286, California, 1993.x

[6] A. DeLuca , S. Termini. *A definition of a non-probabilistic entropy in the setting of fuzzy sets.* Information and Control 20, pp.301-312, 1972.

[7] D. Dubois, H. Prade. *A review of Fuzzy Set Aggregation conectives.* Information Sciences 36, pp.85-121, 1985.

[8] D. Dubois, H. Prade. *Fuzzy numbers: An overview.* Analysis of Fuzzy Information. Vol 1. 3-39, 1987.

[9] D. Filev, R. Yager. *A generalized defuzzification method under BADD distributions.* International Journal of Intelligent Systems 6, pp.687-697, 1991.

[10] D. Goldberg. *Genetic algorithms in search, optimization and machine learning.* Addison-Wesley, Reading, MA, 1989.

[11] D. Harrison, D.L. Rubinfeld. *Hedonic prices and the demand for clean air.* J. Environ. Economics and Management, vol.5, 81-102, 1978.

[12] J.H. Holland. *Adaptation in natural and artificial systems.* University of Michigan Press, Michigan, 1975.

[13] J.S. Jang. *Fuzzy modeling using generalized neural networks and Kalman filter algorithm.* Proceedings of the Ninth National Conference on Artificial Intelligence (AAAI-91), pp.762-767, 1991.

[14] J.S. Jang. *ANFIS: Adaptive-network-based fuzzy inference systems.* IEEE Trans. on Systems, Man, and Cybernetics, 23(03) pp.665-685, 1993.

[15] J.S. Jang. *Structure determination in fuzzy modeling: a fuzzy CART approach.* Proc. of IEEE international conference on Fuzzy Systems, 1994.

[16] J.S. Jang, C.T. Sun, E. Mizutani. *Neuro-fuzzy and Soft Computing.* MathLab Curriculum Series. Prentice Hall. 1997.

[17] S. Kirkpatrick, C.D. Gelatt, M.P. Vecchi. *Optimization by simulated annealing.* Science, 220(4598), pp.671-680, 1983.

[18] G.J. Klir, T.A. Folger. *Fuzzy Sets and Systems: Theory and Applications.* Prentice-Hall: Englewood Cliffs, NJ, 1988.

[19] R. Krishnapuram, J. Keller. *A Possibilistic Approach to Clustering.* IEEE Transactions on Fuzzy Systems, 1(2), pp.98-110, 1993.

[20] E.H. Mamdani, S. Assilian. *An experiment in linguistic synthesis with a fuzzy logic controller.* International Journal of Man-Machine Studies 7, pp.1-13, 1975.

[21] J. Matyas. *Random optimization.* Automation and Remote Control, 26, pp.246-253, 1954.

[22] R.H.J.M. Otten, L.P.P.P van Ginneken. *The annealing algorithm.* Kluwer Academic, 1989.

[23] D.E. Rumelhart, G.E. Hinton, R.J. Williams. *Learning internal representations by error propagation.* Paralel distributed processing: explorations in the microstructure of cognition, 1(8), pp.318-362, MIT Press, Cambridge, MA, 1986.

[24] S. Shah, F. Palmieri. *MEKA – a fast, local algorithm for training feed forward neural networks.* Proceedings of the International Joint Conference on Neural Networks pp.41-46, 1990.

[25] S. Shah, F. Palmieri. *Optimal filtering algorithms for fast learning in feed-forward neural networks.* Neural Networks (5) pp.779-787, 1992.

[26] M. Sugeno, T. Takagi. *Multidimensional fuzzy reasoning.* Fuzzy Sets and Systems 9, 1983.

[27] M. Sugeno, G.T. Kang. *A fuzzy logic based approach to qualitative modeling.* IEEE Trans. on Fuzzy Systems, 1(1) pp.7-31, 1993.

[28] C.T. Sun. *Rulebase structure identificacion in an adaptive network based fuzzy inference system.* IEEE Trans. on Fuzzy Systems, 2(1) pp.64-73, 1994.

[29] T. Takagi, M. Sugeno. *Fuzzy identification of systems and its application to modeling and control.* IEEE Trans. Systems, Man Cybernet. 15, pp.116-132. 1985.

[30] University of California at Irvine. *Repository of Machine Learning Databases and Domain Theories.*
`ftp://ics.uci.edu/pub/machine-learning-databases/`

[31] P. Werbos. *Beyond regression: New tools for prediction and analysis in the behavioral sciences.* PhD thesis, Harvard University, 1974.

[32] R. Yager. *On ordered weighted averaging aggregation operators in multi-criteria decision making.* IEEE Transactions on Systems, Man and Cybernetics 18, pp.183-190, 1988.

[33] R. Yager, D. Filev. *SLIDE: A simple adaptive defuzzification method.* IEEE Transactions on Fuzzy Systems 1, pp.69-78, 1993.

[34] R. Yager, D. Filev. *Essentials of Fuzzy Modeling and Control.* John Wiley & Sons, Inc.

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

[36] Y. Yoshinari, W. Pedrycz, K. Hirota. *Construction of fuzzy models through clustering techniques.* Fuzzy Sets and Systems, 54, pp. 157-165, 1993.

[37] L.A. Zadeh. *Fuzzy Sets.* Information and Control 8, pp.338-353, 1965.