

UNIVERSIDAD DEL AZUAY

FACULTAD DE ADMINISTRACION DE EMPRESAS

ESCUELA DE INGENIERÍA DE SISTEMAS

"Modelo Digital de Lógica Difusa, aplicada a un sistema inteligente"

Tesis previa a la obtención del
Título de Ingeniero de sistemas

Director:

Ing. Fernando Balarezo

Autores:

*Patricia Cabrera C.
Loly Calle O.*

Cuenca-Ecuador
2004

Todos los conceptos y enunciados
Vertidos en este trabajo son de
Responsabilidad de los autores.

Patricia Cabrera C.

Loly Calle O.

AGRADECIMIENTO

A Dios por darnos la fortaleza y la capacidad de llevar a cabo este tan anhelado trabajo, por habernos permitido avanzar un peldaño más en nuestro camino profesional y personal.

A nuestros hijos por su sonrisa al llegar luego de una larga jornada de trabajo, por no reclamar las horas de encargo y por ser los seres mas importantes de nuestra vida. A nuestros esposos por su apoyo incondicional.

A nuestros padres, seres forjadores de nuestras primeras enseñanzas en esta vida estudiantil, y ser los que siempre estuvieron motivándonos a salir adelante, a nuestros hermanos y amigos por animarnos a continuar luchando.

Un agradecimiento muy especial al Ing. Oto Parra, catedrático de la Universidad de Cuenca por su ayuda y orientación en el desarrollo de esta monografía, agradecemos también al Ing. Pablo Pintado y al Ing. Fernando Balarezo por sus oportunas revisiones y correcciones de esta monografía.

Patricia Cabrera C.

Loly Calle O.

DEDICATORIA

Este trabajo lo dedico en primer lugar a mis padres, quienes fueron los forjadores de toda mi vida estudiantil, a ellos quien con sus desvelos y esfuerzo supieron siempre darme lo que necesitaba tanto económica como espiritualmente, a ellos va dedicado este trabajo monográfico en el que puedo brindarles un poco de lo mucho que siempre me dieron.

Lo dedico a mi esposo, quien con su corazón generoso, supo orientarme y apoyarme a culminar mi especialidad, a él que durante mucho tiempo velo el sueño de mis hijos para que yo pueda salir adelante.

Y finalmente a mis hijos de quienes sacrifiqué su tiempo de cuidado, a ellos que ahora son la mayor fuerza para salir adelante, a ellos que son la razón de mi existencia, a ellos les dedico este trabajo con todo mi amor.

Patty Cabrera C.

DEDICATORIA

A mis hijos: Nicolás; mi ángel de mirada traviesa, curioso e incansable investigador del por que? Y para que?; Sebastián; el pequeño que con sus patadas y ahora con la inocente sonrisa han sabido reanimarme en mis momentos de flaqueza y desánimo, a ellos mis más grandes maestros de la vida, pues siendo tan tiernos y pequeños me han dado las lecciones más importantes, me han enseñado que los sueños se hacen realidad si los deseas con el corazón y que el amor de los hijos es el regalo más hermoso que Dios concede a las madres.

A mi esposo, Marcos, por su apoyo incondicional, por enseñarme que aunque el mundo gire continuamente y los días pasen siempre queda el sabor de los logros de días pasados, y que la suma de esos logros son los peldaños para seguir avanzando en el largo camino de la vida.

A mis padres, por haber sembrado en mí la semilla de la constancia y la dedicación. A ellos y a mis hermanos que durante muchos días y noches se convirtieron en los protectores de los sueños de mis pequeños para que yo pudiera avanzar con mi trabajo y logre alcanzar esta meta tan anhelada.

A todos ellos mil gracias por confiar en mí y por darme todo su apoyo y amor incondicional, los amo con todo mi corazón.....

Loly Calle O.

INDICE

1	LOGICA DIFUSA	1
1.1	INTRODUCCION	1
1.2	RESEÑA HISTORICA	2
1.3	APLICACIONES.....	4
1.4	CONCEPTOS BASICOS.....	4
1.4.1	¿Qué es la Lógica Difusa?	5
1.4.2	En que se aplica la lógica difusa?	8
1.5	CONJUNTOS DIFUSOS.....	9
1.5.1	Predicados Vagos y Conjuntos Difusos:	9
1.6	FUNCIÓN DE PERTENENCIA(PARTICIÓN):.....	11
1.6.1	Tipos de Pertenencia	12
1.6.2	Notación de conjuntos difusos.....	14
1.7	OPERACIONES LOGICAS SOBRE CONJUNTOS DIFUSOS	15
1.8	OPERADORES COMPOSICIONALES	17
1.8.1	Negaciones:	18
1.8.2	Operadores Binarios	20
1.8.3	Productos cartesianos.	21
1.9	ETIQUETAS LINGUISTICAS:	25
1.10	LOGICAS PROPOSICIONALES DIFUSAS	26
1.10.1	Sintaxis	26
1.10.2	Semánticas basadas en conjunción y negación	29
1.10.2.1	Incertidumbre	30
1.11	DESdifusIFICAR.....	30
1.11.1	Criterios de desdifusificación.....	30
1.11.1.1	Primer Máximo.	30
1.11.1.2	Corte-a.....	31
1.11.1.3	Centroide.....	31
1.12	SISTEMAS BASADOS EN LOGICA DIFUSA	31
1.12.1	FLS	31
1.12.2	Funciones de partición	34
1.12.3	Las Reglas	35
1.12.4	Manejo del conocimiento.....	36
1.12.4.1	Con lógica binaria.	36
1.12.4.2	Con lógica difusa	36
1.13	CONTROL DIFUSO.....	37
2	MATLAB.....	40
2.1	INTRODUCCION	40
2.2	CONCEPTO DE MATLAB	40
2.3	TOOLBOXES DE MATLAB	41
2.4	COMO INICIAR MATLAB	42
2.5	FUNCIONAMIENTO DEL MATLAB	44
2.6	ESPECIFICACION DE VARIABLES	44
2.6.1	Borrado de variables.	45
2.6.2	La variable NaN	45
2.7	USO DE CARACTERES ESPECIALES EN MATLAB	46
2.8	OPERACIONES BASICAS.....	46
2.9	FUNCIONES	47
2.9.1	Formato de una función:	47
2.9.2	Funciones trigonométricas.....	47
2.9.3	Funciones para LOGARITMOS	48
2.9.4	Funciones matemáticas especiales.	48
2.10	OPERACIONES LOGICAS EN MATLAB.....	49
2.11	LOS VECTORES Y MATRICES EN MATLAB	51
2.11.1	Arreglos (Arrays) ó Vectores.....	52
2.11.1.1	Definir arreglos	52
2.11.1.2	Localización de los elementos de un vector	52
2.11.1.3	Vectores (su orientación)	53

2.11.1.4	Modificaciones de los arreglos.....	54
2.11.1.5	Concatenar arreglos.....	56
2.11.1.6	Matemáticas con arreglos.....	56
2.11.2	Matrices.....	58
2.11.2.1	Operaciones con matrices.....	60
2.12	COMANDOS BASICOS PARA GRAFICAR EN MATLAB.....	63
2.12.1	Comandos para gráficas en dos dimensiones.....	63
2.13	COMANDOS DE INFORMACION.....	68
2.14	CREACIÓN DE ARCHIVOS DE PRAGRAMAS EN MATLAB.....	69
2.14.1	M.files.....	69
2.14.1.1	Tipos de M-files: existen dos tipos.....	70
2.15	ESTRUCTURAS DE COMANDOS.....	70
2.15.1	For.....	70
2.15.2	While.....	71
2.15.3	IF ELSE END.....	71
2.15.4	SWITCH.....	72
2.16	SIMULINK.....	73
2.16.1	Elementos básicos.....	73
2.16.1.1	Los Bloques:.....	73
2.16.1.2	Líneas:.....	74
2.17	HERRAMIENTA UTILIZADA PARA REALIZAR INTERFACE.....	77
2.18	FUZZY LOGIC.....	78
3	MODELO DIGITAL DE LOGICA DIFUSA, APLICADA A UN SISTEMA INTELIGENTE.	
	82	
3.1	PLANTEAMIENTO DEL PROBLEMA.....	82
3.2	ESPECIFICACION DE REQUERIMIENTOS.....	83
3.3	DEFINICION DEL MODELO MATEMATICO.....	83
3.4	GENERACION DE RESULTADOS.....	84
3.5	CONTROLADOR PID.....	84
3.5.1	Como trabaja el controlador PID.....	84
3.5.2	Calcular las constantes del PID.....	85
3.6	DISEÑO Y DESARROLLO DE LA APLICACIÓN PRACTICA.....	90
3.6.1	Problema a resolver.....	90
3.6.2	Términos a Emplear.....	91
3.6.3	Modelado matemático de sistemas dinámicos.....	93
3.6.4	Uso del control inteligente de procesos en la monografía.....	95
3.6.5	Modelo de procesos.....	100
3.6.6	Estudio del problema a desarrollar.....	103
3.6.7	MODELO EN MATLAB.....	105
3.7	COMO CARGAR LA APLICACIÓN REALIZADA.....	119
4	LOGICA DIFUSA VS. LOGICA CLASICA.....	121
4.1	DIFERENCIAS ENTRE LÓGICA DIFUSA VS. LÓGICA CLÁSICA.....	121

CAPITULO I

1 LOGICA DIFUSA

1.1 INTRODUCCION

La lógica difusa es una teoría que se ha implantado en el campo científico-técnico y que en definitiva nos resulta realmente útil si nos interesa que un determinado dispositivo (máquina, programa, aplicación) piense tal y como lo haría la mente humana. Esta lógica se basa fundamentalmente en crear una relación matemática entre un elemento y un determinado conjunto difuso con el fin de que una computadora sea capaz de realizar una valoración similar a como lo hacemos nosotros. Para conseguirlo la lógica difusa utiliza una función de pertenencia $[0,1]$ entre un elemento y un determinado conjunto que a priori¹ será confuso para el computador.

La lógica difusa no se vende ni se puede conseguir en cualquier tienda o comercio, de forma concisa tal y como compramos un antivirus o contratamos un servicio en Internet, es decir, una empresa no ofrece nunca dicha tecnología a su cliente de forma que pueda ser comprada como habitualmente los hacemos con programas utilitarios. Debemos saber y comprender que el mercado de esta aplicación no está ni mucho menos definido aunque ya existe un sinfín de máquinas, programas, computadoras, etc. que usan esta tecnología, por ejemplo en los equipos PALM, que actualmente se encuentran muy comercializadas, utilizan la lógica difusa para interpretar los caracteres que el usuario dibuja, en las lavadoras, para definir los ciclos de lavado.

Se puede decir que muchas de las aplicaciones tecnológicas basadas en lógica difusa son en un porcentaje muy elevado creaciones propias realizadas por los ingenieros de la materia en cada compañía, como es el caso de esta monografía en la que se utilizará esta lógica de la forma más concisa para así interpretar y alcanzar el objetivo planteado, en una aplicación definida, la misma que se indicará posteriormente.

¹ A priori: Antes de examinar un asunto concreto

Este estudio está en vías expansivas, es integrado y eficaz.

El Objetivo de la Lógica Difusa es imitar el razonamiento humano, si nos basamos en estudios anteriores que ya están contrastados podemos ratificar que su fiabilidad es muy alta, así es que el índice de fracaso en principio es muy elevado debido a que un computador necesita tener predefinidos todos y cada uno de los posibles inconvenientes que pueden surgir, pero la aplicación debe depurarse de modo tal que se alcance los objetivos planteados.

Se puede ratificar que se trata de una aplicación con un presente legible y un futuro inmediato sin límites. El objetivo es elevar las funciones y capacidades de las máquinas a niveles comparables al del ser humano y toda aplicación electrónica o programable que lo consiga mediante esta teoría.

En consecuencia, es importante mencionar que la aplicación de la lógica difusa en la empresa se verá recompensada en una considerable disminución de la competencia ya que a priori no todas las compañías utilizan dicha tecnología y gracias a ello se pueden conseguir numerosos beneficios.

Existen ya algunos sistemas expertos que permiten utilizar la inteligencia artificial para desarrollar aplicaciones de este tipo, en nuestro caso utilizaremos MATLAB, como utilitario para desarrollar nuestra aplicación.

1.2 RESEÑA HISTORICA ²

Desde la época de los grandes filósofos griegos se ha venido cuestionando la efectividad de la dicotomía³ cierto-falso y posteriores pensadores también han formulado sus conceptos hasta cuando Zadeh habla de los conjuntos difusos y moldea luego la teoría de la lógica difusa como se observa a continuación:

² http://tesla.cuao.edu.co/automatica/mauricio/documentos/introduccion_logica_difusa.pdf

³ Dicotomía: Dicotómico/Método de clasificación

- En el 380 A.C., Aristóteles propone la existencia de grados de verdad o falsedad.
- En el siglo XVIII, en Inglaterra el filósofo David Hume habla de la lógica del sentido común (razonamiento basado en la experiencia que la gente comúnmente adquiere de sus vivencias por el mundo). El filósofo norteamericano Charles Sander Pierce, fue el primero en considerar la vaguedad⁴ en vez de la dicotomía cierto-falso, como una forma de enmarcar cómo el mundo y las personas funcionan. También en este siglo es inventada la teoría original de conjuntos clásicos de unos y ceros por el matemático alemán George Kantor.
- En 1920 el filósofo polaco Jan Lukasiewicz propone la primera lógica de vaguedad. Desarrolló conjuntos con posibles valores de pertenecía 0, $\frac{1}{2}$ y 1 (lógica trivaluada). Posteriormente los extendió hacia un número infinito de valores entre 0 y 1 (lógica multievaluada).
- En 1962 Lotfi Zadeh cuestiona la efectividad de las matemáticas tradicionales, las cuales resultaban intolerantes ante la imprecisión y ante verdades parciales.
- En 1964 Aparece por primera vez la noción de conjuntos difusos en un memorándum debido al mismo Zadeh en la Universidad de California en Berkeley. Dicho memorándum es publicado un año más tarde bajo el título: "Fuzzy Sets" (Conjuntos difusos).
- En 1965, la revista "Information and Control" publica el memorándum anterior, en donde aparece el artículo de Zadeh, "Fuzzy Sets".
- En 1971, Zadeh publica el artículo, "Quantitative Fuzzy Semantics", en donde Introduce los elementos formales que acabarían componiendo el cuerpo de la doctrina de la lógica difusa y sus aplicaciones tal como se conocen en la actualidad.
- En 1974, el Británico Ebrahim Mandani, demuestra la aplicabilidad de la lógica difusa en el campo del control. Desarrolla el primer sistema de control Fuzzy práctico, la regulación de un motor de vapor.
- A finales de los 70's, Los ingenieros daneses Lauritz Peter Holmbland y Jens-Jurgen Ostergaard desarrollan el primer sistema de control difuso comercial, destinado a una planta de cemento. Los japoneses empiezan a explotar la lógica

⁴ Vaguedad: Imprecisión, falta de exactitud

difusa de forma masiva. Los occidentales asumieron una actitud reacia principalmente por dos razones: la primera era porque la palabra "Fuzzy" sugería algo confuso y sin forma, y la segunda porque no había forma de probar analíticamente que la teoría funcionaba correctamente, ya que el control fuzzy no estaba basado en modelos matemáticos. Aparecen toda una serie de investigadores japoneses en el campo de la lógica difusa tales como Sugeno, Togai, Bart Kosko (el fuzzensei) entre otros.

- En 1986, Yamakawa, publica el artículo, "Fuzzy Controller hardward system". Desarrolla controladores fuzzy en circuitos integrados.
- En 1987, se inaugura en Japón el subterráneo de Sendai, uno de los más espectaculares sistemas de control difuso creados por el hombre. Desde entonces el controlador inteligente ha mantenido los trenes rodando eficientemente.
- En 1987, "FUZZY BOOM", se comercializan multitud de productos basados en la lógica difusa (sobre todo en Japón).

1.3 APLICACIONES

La lógica difusa tiene un amplio abanico de aplicaciones, como por ejemplo:

- Aplicaciones de Control: control de aviones (Rockwell Corporation), control del metro de Sendai (Hitachi, etc..)
- Catalogación y optimización: análisis del stock en el mercado (Yamaichi Securities), ascensor inteligente (Hitachi, Fujitech, Mitsubishi)
- Analisis de señales: ajustes del color de TV (Sony), reconocimiento de caligrafía (Sony Palm Top), autoenfoco de videocámaras (Sanyo/Fisher, Canon), etc...

1.4 CONCEPTOS BASICOS

Hablar de conceptos básicos es hablar de términos que nos ayudarán a tener una mejor interpretación y aprendizaje a lo largo de toda esta monografía, cabe indicar que es necesario el aprendizaje de términos difusos por parte de los técnicos ya que estamos hablando de una tecnología innovadora, es por esto que el equipo técnico especializado debe ser capaz de resolver problemas con fluidez.

1.4.1 ¿Qué es la Lógica Difusa?

La **Lógica Difusa**, que hoy en día se encuentra en constante evolución, nació en los años 60 como la lógica del razonamiento aproximado, y en ese sentido podía considerarse una extensión de la Lógica Multivaluada. La Lógica Difusa actualmente está relacionada y fundamentada en la teoría de los Conjuntos Difusos. Según esta teoría, el grado de pertenencia de un elemento a un conjunto va a venir determinado por una función de pertenencia, que puede tomar todos los valores reales comprendidos en el intervalo $[0,1]$. La representación de la función de pertenencia de un elemento a un Conjunto Difuso se representa según la figura 1.1.⁵

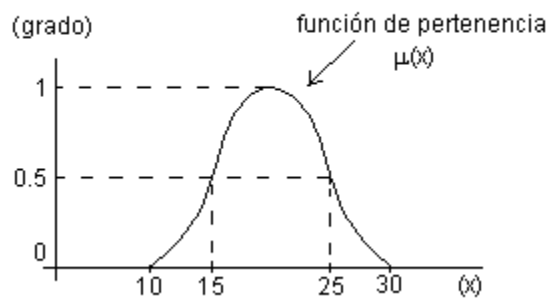


Figura 1.1. Ejemplo de una función de pertenencia a un Conjunto Difuso.

Los **operadores lógicos** que se utilizarán en Lógica Difusa (AND, OR, etc.) se definen también usando tablas de verdad, pero mediante un "**principio de extensión**" por el cual gran parte del aparato matemático clásico, se puede utilizar y adaptar para la correcta manipulación de los Conjuntos Difusos y, por tanto, a la de las **variables lingüísticas**, cuyo significado lo indicaremos más adelante.

Implicación

La operación más importante para el desarrollo y creación de Reglas Lógicas es la implicación, simbolizada por " \rightarrow " que representa el "Entonces" de las reglas heurísticas: Si (...) Entonces (\rightarrow) (...).

Así, en la Lógica Difusa hay muchas maneras de definir la implicación. Se puede elegir una "función (matemática) de implicación" distinta en cada caso para representar a la implicación.

Otra característica de los sistemas lógicos es el *procedimiento de razonamiento*, que permite **inferir**⁶ resultados lógicos a partir de una serie de antecedentes. De aquí podemos decir que generalmente, el razonamiento lógico se basa en silogismos⁷, en los que los antecedentes son por un lado las proposiciones condicionales (nuestras reglas), y las observaciones presentes por otro (serán las premisas⁸ de cada regla).

Los esquemas de razonamiento utilizados son:

- Esquema de Razonamiento Aproximado
- Representación material

""esquemas de razonamiento aproximado", intentan reproducir los esquemas mentales del cerebro humano en el proceso de razonamiento. Estos esquemas consistirán en una generalización de los esquemas básicos de inferencia en Lógica Binaria (silogismo clásico)".⁹

Tan importante será la selección de un esquema de razonamiento como su **"representación material**, ya que el objetivo final es poder desarrollar un procedimiento analítico concreto para el diseño de controladores difusos y la toma de decisiones en general".¹⁰

Una vez que dispongamos de representaciones analíticas de cada uno de los elementos lógicos que acabamos de enumerar, estaremos en capacidad de desarrollar formalmente un controlador "heurístico"¹¹ que nos permita inferir el control adecuado

⁵ Tomado de la Dirección Electrónica: <http://www.keithley.cl/SERVICEnSUPPORT/literature=152.html>

⁶ inferir: Conducir a un resultado, implicar

⁷ Silogismo: es una formula lógica con premisas seguidas de una conclusión.

⁸ Premisas: Primera preposición de un silogismo

⁹ Tomado libro Schweizer, B. and Sklar, A., "Associative functions and abstract semi-groups", pag.34

¹⁰ Tomado de la dirección electrónica <http://www.keithley.cl/SERVICEnSUPPORT/literature=152.html>

¹¹ Heurístico: Algoritmo que consta de utilizar pruebas, exámenes o aproximaciones para llegar a dar con una solución. De esta forma, sin conocer unos datos base exactos, podemos llegar a un resultado final. Suele aplicarse mucho en la detección de virus.

de un determinado proceso en función de un conjunto de reglas "lingüísticas", definidas de antemano tras la observación de la salida y normas de funcionamiento de éste.

Todos estos conceptos se tomarán en la aplicación a desarrollar en la que se explicará claramente El Esquema de Razonamiento y la Representación Material, en esta parte creo que cabe dar a conocer que lo que se espera representar es el Control de la Temperatura en un Horno Eléctrico, para lo cual necesitaremos Preposiciones condicionales es decir nuestras reglas de inferencia y, posteriormente en base a las observaciones presentes podemos inferir en un resultado apropiado.

Variable Lingüística:

Es aquella noción o concepto que vamos a calificar de forma difusa. Por ejemplo: la altura, la edad, el error, la variación del error. Le aplicamos el adjetivo "lingüística" porque definiremos sus características mediante el lenguaje hablado.

Universo de discurso :

Es el rango de valores que pueden tomar los elementos del conjunto que poseen la propiedad expresada por la variable lingüística.

Valor lingüístico:

Llamamos a las diferentes clasificaciones que efectuamos sobre la variable lingüística: Por ejemplo si hablamos de la variable lingüística temperatura, podríamos dividir el universo de discurso en los diferentes valores lingüísticos: 'baja','mediana' y 'alta'. Como veremos, cada valor lingüístico tendrá un conjunto difuso asociado, de forma que hablaremos de los conjuntos difusos 'bajo','alto', asociados a la variable lingüística 'temperatura'.

1.4.2 En que se aplica la lógica difusa?

La Lógica Difusa ha sido probada para ser particularmente útil en sistemas expertos y otras aplicaciones de inteligencia artificial. Es también utilizada en algunos correctores de voz para sugerir una lista de probables palabras a reemplazar en una mal dicha.

Desde su aparición en la década de los 60's hasta nuestros días, las aplicaciones de la Lógica Difusa se han ido consolidando, paulatinamente al comienzo, y con un desbordado crecimiento en los últimos años. Se encuentran en soluciones a problemas de control industrial, en predicción de series de tiempo, como metodologías de archivo y búsqueda de Bases de Datos, en Investigación Operacional, en estrategias de mantenimiento predictivo y en otros campos más.

Las principales razones para tal proliferación de aplicaciones quizás sean la sencillez conceptual de los Sistemas basados en Lógica Difusa, su facilidad para adaptarse a casos particulares con pocas variaciones de parámetros, su habilidad para combinar en forma unificada expresiones lingüísticas con datos numéricos, y el no requerir de algoritmos muy sofisticados para su implementación.

El uso de la tecnología de la Lógica Difusa se ha extendido rápidamente en el diseño de productos en donde requiera satisfacer los siguientes requerimientos:

- Desarrollo de sistemas de control con características no lineales y controladores de sistemas que realizan decisiones
- Desarrollo de sistemas que involucren sensores y procesamiento de gran cantidad de información
- Para reducir el tiempo de desarrollo
- Para reducir costos asociados con la incorporación de la tecnología en el producto.

La tecnología difusa puede satisfacer estos requerimientos por las siguientes razones:

Las características no lineales son realizadas en lógica difusa por la partición de la regla del espacio, por las reglas de peso y por funciones de membresía no lineales. En el razonamiento difuso, los límites de estas partes se traslapan¹², y los resultados locales son combinados por su peso apropiadamente. Es por eso que la salida de un sistema difuso es una función (smooth) no linear.

La lógica difusa es muy utilizada en aplicaciones que incluyen productos como cámaras fotográficas y de vídeo, lavadoras, aparatos de aire acondicionado, refrigeradores, alarmas, electrodomésticos, etc., así como una gran variedad de controladores industriales, dispositivos médicos, sistemas de seguridad en reactores nucleares, robots y otros sistemas relativamente complejos.

Dada la facilidad de la lógica difusa para representar conocimientos, se ha empleado también en la solución de problemas sociológicos, psicológicos, políticos, administrativos, económicos, epidemiológicos y de otras disciplinas. Existen paquetes como el Fuzzy Decision Maker, que ayudan a las personas a tomar decisiones de todo tipo, como por ejemplo solucionar un problema familiar.

1.5 Conjuntos Difusos

Un conjunto difuso es un aquel que puede contener elementos con grados parciales de pertenencia, a diferencia de los Conjuntos Clásicos (Crisp Sets) en los que los elementos pueden solamente "pertenecer" ó "No Pertenecer" a dichos conjuntos.

1.5.1 Predicados Vagos y Conjuntos Difusos:

Los conjuntos clásicos se definen mediante un predicado que da lugar a una clara división del Universo de Discurso X en los valores "Verdadero" y "Falso", lo que implica simples operaciones lógicas. Sin embargo, el razonamiento humano utiliza frecuentemente predicados que no se pueden reducir a este tipo de división: son los

¹² Traslapan = superposición, es decir cubrir total o parcialmente algo con otra cosa.

denominados *predicados vagos*, llamados así porque no se reducen a una simple división de Verdadero(1) y Falso (0).

Por ejemplo, tomando el Universo de Discurso formado por todas las posibles temperaturas ambientales en la ciudad de Cuenca, se puede definir en dicho universo el conjunto A como aquél formado por las temperaturas "cálidas".

Por supuesto, es imposible dar a A una definición clásica, ya que su correspondiente predicado no divide el universo X en dos partes claramente diferenciadas. No podemos afirmar con exactitud que una temperatura es "cálida" o no lo es. El problema podría resolverse en parte considerando que una temperatura es "cálida" cuando su valor supera cierto umbral fijado de antemano. Se dice que el problema tan sólo se resuelve en parte, y de manera no muy convincente, por dos motivos: de una parte el umbral¹³ mencionado se establece de una manera arbitraria¹⁴ (subjettiva en mucha de las veces), y por otro lado podría darse el caso de que dos temperaturas con valores muy diferentes fuesen consideradas ambas como "cálidas". Evidentemente, el concepto "calor" así definido nos daría una información muy pobre sobre la temperatura ambiental.

Viene aquí entonces la necesidad de utilizar una manera más apropiada de dar solución a este problema. Entonces es necesario considerar en este punto que la pertenencia o no pertenencia de un elemento x al conjunto A no es absoluta sino gradual.

En definitiva, definiremos A como un **Conjunto Difuso**. Su función de pertenencia ya no adoptará valores en el conjunto discreto {0,1} (lógica booleana), sino en el intervalo cerrado [0,1]. En conclusión podemos observar que los Conjuntos Difusos son una generalización de los conjuntos clásicos.

Mediante notación matemática se define un Conjunto Difuso B como:

$$B = \{ (x, \mu_B(x)) / x \in X \}$$

¹³ Umbral: Valor mínimo de una magnitud a partir del cual se produce un efecto determinado.

¹⁴ Arbitraria: adoptar soluciones.

Donde

B es el conjunto difuso

X es el elemento a evaluar

$\mu_B(x)$ Función de pertenencia de x respecto al conjunto B

x/x cumple cierta condición

$$\mu_B: X \rightarrow [0,1]$$

Por ejemplo consideremos el conjunto de todos los números primos P en el dominio de los números naturales N. Entonces podremos definir P como:

$$P = \{x \in \mathbb{N} \mid x \text{ es primo}\}$$

En notación binaria diríamos $\mu(x)=1$ si x es primo, o $\mu(x)=0$ en otro caso

1.6 Función de Pertenencia(Partición):

Definimos **función de pertenencia** como aquella aplicación que asocia a cada elemento de un conjunto difuso al grado con que pertenece al valor lingüístico asociado.

Los conjuntos difusos son caracterizados por sus funciones de pertenencia, a veces resulta difícil comprender este concepto pero si proseguimos la lectura veremos un ejemplo que interprete este concepto de forma clara.

Diremos que un conjunto es difuso cuando el concepto al que representa tiene una función de pertenencia difusa asociada a él.

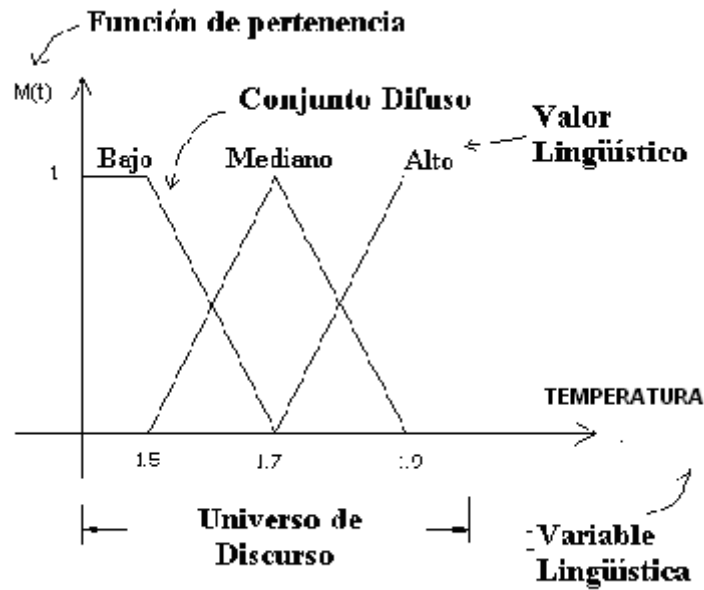


Figura 1.2. Conjuntos Difusos

En esta ilustración hemos dibujado 3 conjuntos difusos sobre la variable lingüística temperatura, cuyos valores lingüísticos asociados son 'baja', 'mediana' y 'alta' respectivamente. Las funciones de pertenencia son de tipo 'L' para el bajo, 'Lambda' o 'Triángulo' para el mediano y 'Gamma' para el alto.

En el siguiente punto aclararemos porqué usamos estos nombres, que únicamente determinan qué forma tendrán las funciones de pertenencia.

Entonces tenemos que si hacemos una relación entre la lógica booleana y la lógica difusa, se nota que los valores de la función de pertenencia de la primera son 0 o 1, la lógica difusa se mueve en todo el intervalo $[0,1]$.

Se suele normalizar el grado de pertenencia máximo a 1.

Así expresamos que mientras un elemento puede estar dentro de un determinado conjunto, puede no cumplir las especificaciones al cien por cien.

1.6.1 Tipos de Pertenencia

Aunque en principio cualquier función sería válida para definir conjuntos difusos, en la práctica hay ciertas funciones típicas que siempre se suelen usar, tanto por

la facilidad de computación que su uso conlleva como por su estructura lógica para definir su valor lingüístico asociado.

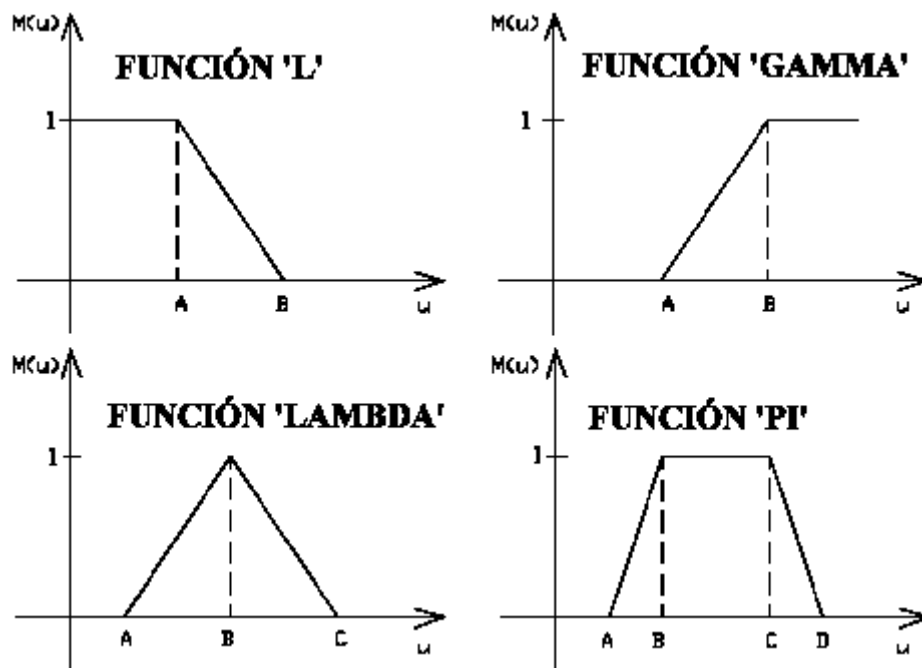


Figura 1.3. Funciones de Pertenencia

Las funciones "**L**" y "**GAMMA**" se usan para calificar valores lingüísticos extremos, tales como "Bajo" o "Alto" respectivamente. Su ventaja es que la función se extiende al infinito. Las funciones "**PI**" y "**LAMBDA**" se usan para describir valores intermedios. Su diferencia reside en que la función "PI" implica un margen de tolerancia alrededor del valor que se toma como más representativo del valor lingüístico asociado al conjunto difuso.

Un conjunto de la lógica tradicional también es expresable con una función de pertenencia. De hecho, como ya se dijo anteriormente la lógica difusa es una extensión de la lógica tradicional y por tanto la incluye. Todas las reglas y propiedades de la lógica difusa son aplicables a la tradicional. En cierta manera, la lógica tradicional es una particularización de la difusa, aunque nosotros la hemos conocido en primera instancia resulta ser esta una particularización de la Difusa.

Así pues, un conjunto de la lógica tradicional se puede expresar mediante un conjunto difuso cuya función de pertenencia sería:

$$\mu_F(x) = \begin{cases} 0 & \text{si } x \in F \\ 1 & \text{si } x \notin F \end{cases}$$

Es decir, una función escalón centrada en el valor umbral de decisión.

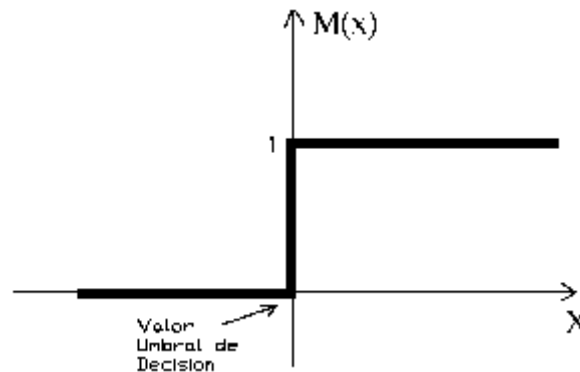


Figura 1.4. Representación de un conjunto de lógica Booleana

1.6.2 Notación de conjuntos difusos

Sea F un conjunto difuso definido sobre el universo U .

$$F = \{ (u, \mu_F(u)) \mid u \in U \}$$

- Indica que F está formado por todos los pares ordenados " u " y el resultado de la función de pertenencia para todo elemento u dentro del universo de discurso U .

La notación que eligió Zadeh para describir los conjuntos difusos es la siguiente:

si el universo es **discreto**:

$$F = \sum_u \frac{\mu_F(u)}{u}$$

si el universo es **continuo**.

$$F = \int \frac{\mu_F(u)}{u}$$

Donde:

“u” es el elemento

μ_F Quiere decir función de pertenencia.

Nota: Es importante entender en esta notación que el sumatorio o la integral pierden su significado habitual. En lógica difusa quieren simbolizar una mera enumeración de tuplas¹⁵.

Una tupla es un par ordenado como el ya visto (u, MF(u)). La fracción tampoco indica quebrado, simplemente separa los dos elementos de la tupla. Así, el conjunto difuso discreto, "Temperatura alta del horno" se define por:

$$F = \{ 0/1 + 0/2 + 0.3/3 + 0.6/4 + 0.9/5 + 1/6 \}$$

La parte derecha de la tupla indica el elemento y la parte izquierda el grado de pertenencia.

1.7 OPERACIONES LOGICAS SOBRE CONJUNTOS DIFUSOS

Las operaciones lógicas que se pueden establecer entre conjuntos difusos son:

- la intersección,
- la unión y
- el complemento,

De la misma manera que la usamos en lógica booleana. Mientras que el resultado de operar dos conjuntos 'abruptos'¹⁶ es un nuevo conjunto 'abrupto', las mismas operaciones con conjuntos difusos nos darán como resultado otros conjuntos también difusos.

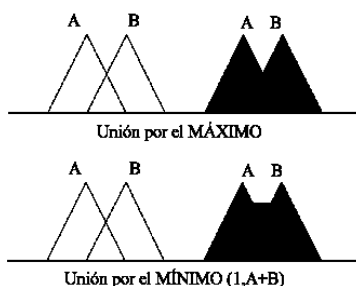
Cabe la pena indicar que las operaciones que se indican en este documento para la lógica difusa a más de sustentar la monografía en mención, también son aplicables a la lógica Bivaluada obteniendo idénticos resultados.

Hay muchas formas de definir estas operaciones utilizando Lógica Difusa, indicaremos entonces lo siguiente:

- Cualquier operación que cumpla las restricciones de una T-Norma puede ser usada para intersectar. Las T-Normas especifican un conjunto de condiciones que deben reunir aquellas operaciones que deseen ser usadas para intersectar conjuntos.
- Cualquier S-Norma puede ser usada para unir conjuntos difusos, las S-Normas hacen lo propio para las uniones.

Las operaciones lógicas más usuales son la unión, la intersección y el complemento.

Unión

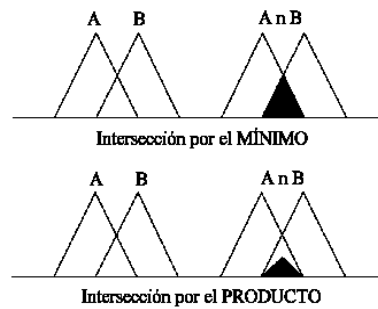


¹⁵ Tupla: par ordenado, es decir s

¹⁶ Abrupto: Difícil, desigual

ción.

Intersección



Algebraicamente, las dos formas de expresar la intersección difusa se expresan:

$$\mu_{A \cap B} = \min(\mu_A(u), \mu_B(u))$$

$$\mu_{A \cap B} = \mu_A(u) \cdot \mu_B(u)$$

Análogamente, para la unión tenemos:

$$\mu_{A \cup B} = \max(\mu_A(u), \mu_B(u))$$

$$\mu_{A \cup B} = \max(1, \mu_A(u) + \mu_B(u))$$

y para el complemento,

$$\mu_{\neg A} = 1 - \mu_A(u)$$

Donde μ_A , quiere decir función de pertenencia del conjunto A

$\mu_{A \cap B}$ quiere decir Función de pertenencia del conjunto A en intersección con el conjunto B,

$\mu_{A \cup B}$ quiere decir Función de pertenencia del conjunto A en unión con el conjunto B

1.8 OPERADORES COMPOSICIONALES

Es bien sabido que los conjuntos usuales pueden ser operados para formar otros nuevos. Existen algunas maneras de extender las operaciones conjuntistas convencionales a conjuntos difusos.

1.8.1 Negaciones:

Veamos cómo extender a la operación de "complemento". En los conjuntos usuales, un punto está en el complemento de un conjunto si y solo si no está en el conjunto. Si vemos al conjunto como su propia función característica, tenemos que la operación complemento "voltea" los valores de pertenencia: A los puntos donde se tuviese un valor de pertenencia 1 el complemento les asignará el valor 0 y viceversa.

Un operador de *negación* es pues una función N que a cada valor t en el intervalo $[0,1]$ le asocia un valor $N(t)$ en el mismo intervalo $[0,1]$, de manera tal que $N(0)=1$, $N(1)=0$ y que además es una función no-creciente, es decir, si $t \leq s$ entonces $N(t) \geq N(s)$

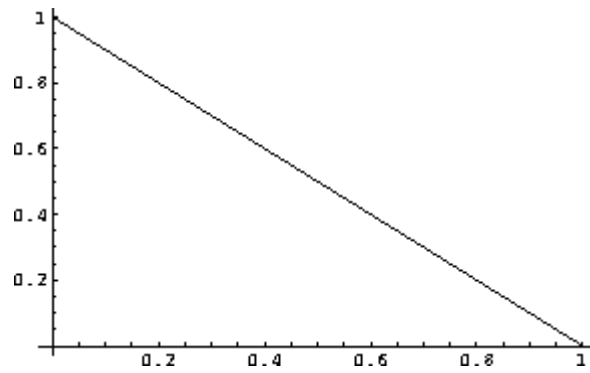
EJEMPLOS DE NEGACIONES:

Lineal

La función

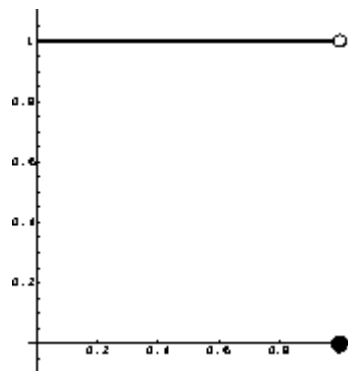
$$N_1 : x \mapsto 1 - x$$

es una negación. Su gráfica es:



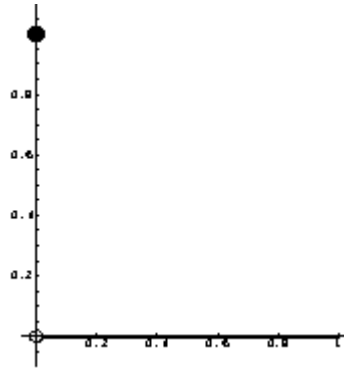
Negación de la verdad:

Sea $N_0 : [0, 1] \rightarrow [0, 1]$, la función $N_0 : x \mapsto \begin{cases} 1 & \text{si } x < 1, \\ 0 & \text{si } x = 1. \end{cases}$, su gráfica es:



Negación de la falsedad

Sea $N_\infty : [0, 1] \rightarrow [0, 1]$, la función $N_\infty : x \mapsto \begin{cases} 1 & \text{si } x = 0, \\ 0 & \text{si } x > 0. \end{cases}$
Su gráfica es:



1.8.2 Operadores Binarios

Una colección de operadores en conjuntos se dice ser completa si cualquier otro operador conjuntista se expresa en términos de los operadores en esa colección.

Se tiene que:

- 1.- $\{\overline{}, \cap\}$ Es completo, donde $\overline{}$ denota la operación complemento
- 2.- $\{\overline{}, \cup\}$ es completo
- 3.- $\{\overline{}, .imp.\}$ es completo, donde $A.imp.B = \overline{A} \cup B$
- 4.- $\{\uparrow\}$ es completo donde $A \uparrow B = \overline{A \cap B}$ es el complemento de la intersección
- 5.- $\{\downarrow\}$ es completo donde $A \downarrow B = \overline{A \cup B}$ es el complemento de la unión

En efecto, utilizando propiedades de álgebra booleana, de entre ellas a las leyes de De Morgan de manera principal,

- 1.- en términos de $\{\overline{}, \cap\}$ se tiene

$$\begin{aligned} A \cup B &= \overline{\overline{A \cap B}} \\ A.imp.B &= \overline{A \cap B} \end{aligned}$$

2.- en términos de $\{\neg, \cup\}$ se tiene $A \cap B = \overline{\overline{A \cup B}}$ y, por el punto anterior, esto

basta para tener lo aseverado.

3.- En términos de $\{\neg, .imp.\}$ se tiene $A \cup B = \overline{A}.imp.B$, por el punto anterior, esto basta para tener lo aseverado,

4.- En términos de $\{\uparrow\}$ se tiene

$$\begin{aligned}\overline{A} &= A \uparrow A \\ A \cap B &= (A \uparrow B) \uparrow (A \uparrow B)\end{aligned}$$

y como $\{\neg, \cap\}$ es completo se tiene lo aseverado.

5.- En términos de $\{\downarrow\}$ se tiene

$$\begin{aligned}\overline{A} &= A \downarrow A \\ A \cup B &= (A \downarrow B) \downarrow (A \downarrow B)\end{aligned}$$

y como $\{\neg, \cup\}$ es completo se tiene lo aseverado

Con cada clase de conectivos¹⁷, los conjuntos difusos forman una estructura algebraica que, sin ser una álgebra booleana, posee varias de las propiedades características de estas últimas.

1.8.3 Productos cartesianos.

¹⁷ conectivo: que une o liga cada parte

Recordemos que para dos conjuntos usuales A, B su producto cartesiano consta de todas las parejas ordenadas de la forma (a,b) donde $a \in A$. Así pues, si \diamond es un operador conjuntor y A y B son conjuntos difusos en sendos universos X e Y , su producto cartesiano es el conjunto difuso

$$A \times B : X \times Y \rightarrow [0, 1], (x, y) \mapsto A(x) \diamond B(y).$$

Una relación, en el sentido usual, entre dos conjuntos es un subconjunto de su producto cartesiano. Por tanto, se puede considerar a una relación difusa entre dos universos como un conjunto en su producto cartesiano.

$$\approx: (x, y) \mapsto \frac{1}{1+(x-y)^2}$$

EJEMPLO: ¹⁸

Consideremos un conjunto de 10 chicos,

Chicos = {Abel, Beto, Carlos, Damián, Ernesto, Felipe, Guillermo, Héctor, Ignacio, Juan}

Y otro de 10 chicas

Chicas: = { Queta, Rosa, Sofia, Teresa, Ursula, Virginia, Wanda, Ximena, Yolanda, Zenaida}

¹⁸ Sugeno, M., "Industrial applications of fuzzy control". Elsevier Science Pub. Co. Pag. 88

Una relación difusa, llamemos la *Prefiere _A*, del conjunto de Chicos sobre el conjunto de Chicas se muestra así

<i>Prefiere_A</i>	<i>Q</i>	<i>R</i>	<i>S</i>	<i>T</i>	<i>U</i>	<i>V</i>	<i>W</i>	<i>X</i>	<i>Y</i>	<i>Z</i>
<i>A</i>	$\frac{4}{5}$	$\frac{4}{5}$	0	$\frac{1}{10}$	$\frac{4}{5}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{10}$	$\frac{1}{10}$	0
<i>B</i>	$\frac{1}{2}$	$\frac{1}{5}$	0	$\frac{3}{5}$	$\frac{1}{5}$	$\frac{1}{2}$	$\frac{7}{10}$	$\frac{9}{10}$	$\frac{3}{10}$	0
<i>C</i>	$\frac{1}{5}$	$\frac{1}{5}$	$\frac{3}{5}$	$\frac{4}{5}$	$\frac{4}{5}$	$\frac{1}{2}$	$\frac{9}{10}$	$\frac{1}{10}$	$\frac{3}{10}$	$\frac{3}{10}$
<i>D</i>	$\frac{1}{2}$	$\frac{9}{10}$	$\frac{1}{5}$	$\frac{1}{5}$	$\frac{3}{10}$	$\frac{2}{5}$	$\frac{4}{5}$	0	$\frac{3}{10}$	$\frac{4}{5}$
<i>E</i>	$\frac{1}{10}$	0	$\frac{2}{5}$	0	$\frac{1}{5}$	$\frac{1}{5}$	$\frac{7}{10}$	$\frac{2}{5}$	$\frac{7}{10}$	$\frac{1}{10}$
<i>F</i>	$\frac{1}{2}$	$\frac{3}{5}$	0	$\frac{1}{10}$	$\frac{7}{10}$	$\frac{7}{10}$	$\frac{7}{10}$	$\frac{1}{5}$	$\frac{4}{5}$	$\frac{1}{2}$
<i>G</i>	$\frac{9}{10}$	$\frac{3}{10}$	$\frac{1}{10}$	$\frac{9}{10}$	$\frac{4}{5}$	$\frac{1}{5}$	$\frac{1}{10}$	$\frac{3}{5}$	$\frac{4}{5}$	$\frac{3}{10}$
<i>H</i>	$\frac{3}{5}$	$\frac{3}{5}$	$\frac{3}{5}$	$\frac{1}{2}$	$\frac{7}{10}$	$\frac{1}{10}$	$\frac{9}{10}$	$\frac{7}{10}$	$\frac{3}{10}$	$\frac{3}{5}$
<i>I</i>	$\frac{2}{5}$	$\frac{3}{10}$	0	$\frac{1}{5}$	$\frac{1}{10}$	$\frac{1}{5}$	$\frac{1}{2}$	0	$\frac{3}{10}$	$\frac{9}{10}$
<i>J</i>	$\frac{7}{10}$	$\frac{9}{10}$	$\frac{7}{10}$	$\frac{4}{5}$	$\frac{9}{10}$	0	$\frac{9}{10}$	$\frac{4}{5}$	$\frac{4}{5}$	$\frac{4}{5}$

Figura 1.6. Relación *Prefiere _A* de Chicos a Chicas

En esa relación, un chico preferirá mas a una chica si el correspondiente valor de esa pareja es más cercano a 1. Si es 0, el chico definitivamente no prefiere a la chica. Juan, por ejemplo, a ninguna prefiere mas que a Rosa, a Ursula y a Wanda, pero a ellas tres las prefiere por igual (aunque el de hecho esta manifestando cualquier preferencia con mucho ímpetu). Sofía es acaso de las menos preferidas, y Abel, Beto, Felipe e Ignacio para nada la prefieren.

Pero, obviamente, las chicas también tienen su preferencia. Consideremos la relación *Elige _A* de chicas en chicos que se muestran en la siguiente figura

<i>Elige_A</i>	<i>A</i>	<i>B</i>	<i>C</i>	<i>D</i>	<i>E</i>	<i>F</i>	<i>G</i>	<i>H</i>	<i>I</i>	<i>J</i>
<i>Q</i>	$\frac{2}{5}$	$\frac{4}{5}$	$\frac{1}{10}$	$\frac{9}{10}$	$\frac{2}{5}$	$\frac{1}{2}$	$\frac{4}{5}$	$\frac{1}{2}$	$\frac{3}{10}$	$\frac{1}{5}$
<i>R</i>	$\frac{7}{10}$	$\frac{3}{10}$	$\frac{3}{10}$	$\frac{1}{5}$	$\frac{7}{10}$	$\frac{9}{10}$	$\frac{3}{10}$	$\frac{1}{2}$	$\frac{3}{10}$	$\frac{1}{10}$
<i>S</i>	$\frac{3}{5}$	$\frac{9}{10}$	$\frac{1}{10}$	$\frac{3}{10}$	$\frac{3}{5}$	0	$\frac{1}{2}$	$\frac{7}{10}$	$\frac{1}{2}$	$\frac{3}{5}$
<i>T</i>	$\frac{1}{5}$	$\frac{3}{10}$	$\frac{4}{5}$	$\frac{1}{10}$	$\frac{7}{10}$	$\frac{2}{5}$	$\frac{2}{5}$	$\frac{1}{10}$	0	0
<i>U</i>	$\frac{3}{5}$	$\frac{1}{10}$	$\frac{2}{5}$	$\frac{9}{10}$	0	$\frac{3}{10}$	$\frac{9}{10}$	$\frac{1}{10}$	$\frac{1}{2}$	0
<i>V</i>	$\frac{1}{2}$	0	$\frac{1}{5}$	$\frac{2}{5}$	$\frac{9}{10}$	$\frac{1}{10}$	$\frac{1}{2}$	$\frac{3}{10}$	$\frac{9}{10}$	$\frac{9}{10}$
<i>W</i>	$\frac{3}{10}$	$\frac{1}{10}$	$\frac{3}{10}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{2}{5}$	$\frac{2}{5}$	$\frac{7}{10}$	$\frac{7}{10}$	$\frac{3}{10}$
<i>X</i>	$\frac{9}{10}$	$\frac{1}{5}$	$\frac{1}{5}$	$\frac{3}{5}$	$\frac{9}{10}$	$\frac{9}{10}$	$\frac{3}{10}$	$\frac{9}{10}$	$\frac{4}{5}$	$\frac{1}{10}$
<i>Y</i>	$\frac{7}{10}$	$\frac{4}{5}$	$\frac{3}{5}$	$\frac{1}{5}$	$\frac{4}{5}$	$\frac{1}{5}$	$\frac{1}{5}$	$\frac{3}{5}$	$\frac{3}{5}$	$\frac{2}{5}$
<i>Z</i>	$\frac{3}{5}$	$\frac{1}{10}$	$\frac{7}{10}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{3}{5}$	$\frac{1}{5}$	$\frac{7}{10}$	$\frac{4}{5}$	$\frac{4}{5}$

Figura 1.7 Relación Elige _A de Chicas a Chicos

En las relaciones definidas, se tiene, por ejemplo, que Beto prefiere mas a Ximena pero ella elige mas a Abel, Ernesto, Felipe y Hector.

La composición (Elige_A) o (Prefiere_A), llamémosla Sientese_Rival_De (SRD), es una relación de chicos en chicos: Fulano Sientese_Rival_De Zultano si la chica que más prefiere Fulano elige mas a Zultano. Usando como conjuntor a la operación Min se tiene la relación mostrada en este cuadro:

SRD	A	B	C	D	E	F	G	H	I	J
A	$\frac{7}{10}$	$\frac{4}{5}$	$\frac{2}{5}$	$\frac{4}{5}$	$\frac{7}{10}$	$\frac{4}{5}$	$\frac{4}{5}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$
B	$\frac{9}{10}$	$\frac{1}{2}$	$\frac{3}{5}$	$\frac{3}{5}$	$\frac{9}{10}$	$\frac{9}{10}$	$\frac{1}{2}$	$\frac{9}{10}$	$\frac{4}{5}$	$\frac{1}{2}$
C	$\frac{3}{5}$	$\frac{3}{5}$	$\frac{4}{5}$	$\frac{4}{5}$	$\frac{7}{10}$	$\frac{2}{5}$	$\frac{4}{5}$	$\frac{7}{10}$	$\frac{7}{10}$	$\frac{3}{5}$
D	$\frac{7}{10}$	$\frac{1}{2}$	$\frac{7}{10}$	$\frac{1}{2}$	$\frac{7}{10}$	$\frac{9}{10}$	$\frac{1}{2}$	$\frac{7}{10}$	$\frac{4}{5}$	$\frac{4}{5}$
E	$\frac{7}{10}$	$\frac{2}{5}$	$\frac{3}{5}$	$\frac{1}{2}$	$\frac{7}{10}$	$\frac{1}{2}$	$\frac{2}{5}$	$\frac{7}{10}$	$\frac{5}{7}$	$\frac{5}{7}$
F	$\frac{7}{10}$	$\frac{4}{5}$	$\frac{3}{5}$	$\frac{2}{5}$	$\frac{4}{5}$	$\frac{3}{5}$	$\frac{5}{7}$	$\frac{7}{10}$	$\frac{7}{10}$	$\frac{5}{7}$
G	$\frac{7}{10}$	$\frac{5}{5}$	$\frac{5}{5}$	$\frac{9}{10}$	$\frac{4}{5}$	$\frac{3}{5}$	$\frac{4}{5}$	$\frac{3}{5}$	$\frac{3}{5}$	$\frac{2}{5}$
H	$\frac{7}{10}$	$\frac{3}{5}$	$\frac{3}{5}$	$\frac{7}{10}$	$\frac{7}{10}$	$\frac{7}{10}$	$\frac{7}{10}$	$\frac{7}{10}$	$\frac{7}{10}$	$\frac{3}{5}$
I	$\frac{3}{5}$	$\frac{5}{5}$	$\frac{7}{10}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{3}{5}$	$\frac{2}{5}$	$\frac{7}{10}$	$\frac{4}{5}$	$\frac{4}{5}$
J	$\frac{4}{5}$	$\frac{4}{5}$	$\frac{4}{5}$	$\frac{9}{10}$	$\frac{4}{5}$	$\frac{9}{10}$	$\frac{9}{10}$	$\frac{4}{5}$	$\frac{4}{5}$	$\frac{4}{5}$

Figura 1.8 Relación SRD = (Elige _A) o (Prefiere _A) de chicos a Chicos

La terminología es desafortunada pues, por ejemplo, Carlos SRD de sí mismo (lo que es muy bueno). El prefiere mas que a nadie a Wanda, pero ella no se muestra muy afecta a los chicos propuestos. En cambio, Carlos y Teresa se atraen recíprocamente con 0.8. Por otro lado Juan quiere tanto a todas las chicas menos a Virginia, que SRD de cualquier otro chico.

Finalmente, la composición (Prefiere_A) o (Elige_A), llamémosla Celosa_De (CD), es una relación de Chicas en Chicas. Se tiene la relación mostrada en el siguiente cuadro.

CD	Q	R	S	T	U	V	W	X	Y	Z
Q	$\frac{4}{5}$	$\frac{9}{10}$	$\frac{1}{2}$	$\frac{4}{5}$	$\frac{4}{5}$	$\frac{1}{2}$	$\frac{4}{5}$	$\frac{4}{5}$	$\frac{4}{5}$	$\frac{4}{5}$
R	$\frac{7}{10}$	$\frac{7}{10}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{7}{10}$	$\frac{7}{10}$	$\frac{7}{10}$	$\frac{1}{2}$	$\frac{4}{5}$	$\frac{1}{2}$
S	$\frac{3}{5}$	$\frac{3}{5}$	$\frac{3}{5}$	$\frac{3}{5}$	$\frac{7}{10}$	$\frac{1}{2}$	$\frac{7}{10}$	$\frac{9}{10}$	$\frac{3}{5}$	$\frac{3}{5}$
T	$\frac{2}{5}$	$\frac{2}{5}$	$\frac{3}{5}$	$\frac{4}{5}$	$\frac{4}{5}$	$\frac{1}{2}$	$\frac{4}{5}$	$\frac{2}{5}$	$\frac{7}{10}$	$\frac{2}{5}$
U	$\frac{9}{10}$	$\frac{9}{10}$	$\frac{2}{5}$	$\frac{9}{10}$	$\frac{4}{5}$	$\frac{1}{2}$	$\frac{4}{5}$	$\frac{3}{5}$	$\frac{4}{5}$	$\frac{4}{5}$
V	$\frac{7}{10}$	$\frac{9}{10}$	$\frac{7}{10}$	$\frac{4}{5}$	$\frac{9}{10}$	$\frac{1}{2}$	$\frac{9}{10}$	$\frac{4}{5}$	$\frac{4}{5}$	$\frac{9}{10}$
W	$\frac{3}{5}$	$\frac{3}{5}$	$\frac{3}{5}$	$\frac{1}{2}$	$\frac{7}{10}$	$\frac{2}{5}$	$\frac{7}{10}$	$\frac{7}{10}$	$\frac{1}{2}$	$\frac{7}{10}$
X	$\frac{4}{5}$	$\frac{4}{5}$	$\frac{3}{5}$	$\frac{1}{2}$	$\frac{4}{5}$	$\frac{7}{10}$	$\frac{9}{10}$	$\frac{7}{10}$	$\frac{4}{5}$	$\frac{4}{5}$
Y	$\frac{7}{10}$	$\frac{7}{10}$	$\frac{3}{5}$	$\frac{3}{5}$	$\frac{7}{10}$	$\frac{1}{2}$	$\frac{7}{10}$	$\frac{4}{5}$	$\frac{7}{10}$	$\frac{3}{5}$
Z	$\frac{7}{10}$	$\frac{4}{5}$	$\frac{7}{10}$	$\frac{4}{5}$	$\frac{4}{5}$	$\frac{3}{5}$	$\frac{4}{5}$	$\frac{4}{5}$	$\frac{4}{5}$	$\frac{4}{5}$

Figura 1.9 Relación $CD=(Prefiere_A)$ o $(Elige_A)$ de chicas a Chicas

Virginia esta muy descontenta consigo misma, $CD(V,V) = 1/2$, porque los tres chicos que elegiría mas la prefieren a ella menos que a las demás.

Si R es una relación difusa en $X \times Y$, entonces todo conjunto difuso B en Y determina un conjunto difuso $A = B \circ R$ en X como

$$A = B \circ R : x \mapsto \text{Max}\{R(x, y) \diamond B(y) | y \in Y\}$$

A se dice ser la composición de R con B .

1.9 ETIQUETAS LINGUISTICAS:

Una etiqueta lingüística es el nombre que se le dá a un conjunto difuso. Es decir, es una por ejemplo (Nombre, A, X) , donde Nombre es el nombre asociado al conjunto difuso A en el universo X . Es convencional confundir a la etiqueta lingüística con su propio nombre.

Por ejemplo si $X=[0,80]$ es el conjunto de Temperaturas posibles de un horno, medida en grados centígrados, la etiqueta lingüística Baja puede corresponder al nombre de un conjunto difuso.

1.10 LOGICAS PROPOSICIONALES DIFUSAS

1.10.1 Sintaxis

Debemos tener presente que un cálculo de proposiciones (Cprop) se construye a partir de un conjunto finito de variables proposicionales $P = \{p_1, \dots, p_n\}$, de los valores constantes 0, 1 a los que se identifica como Falso y Verdadero, respectivamente, y de algunos conectivos, entre los cuales están:

$$\neg, \vee, \wedge, \rightarrow, \leftrightarrow$$

llamados negación, disyunción, conjunción, implicación y equivalencia, respectivamente. Las formas proposicionales son las así llamadas fórmulas bien formadas. Para precisar el concepto de fórmula bien formada asignemos primeramente prioridades a los conectivos:

$$\begin{array}{ll} \neg & \text{Tiene prioridad 1} \\ \wedge, \vee & \text{Tiene prioridad 2} \\ \rightarrow, \leftrightarrow & \text{Tiene prioridad 2} \end{array}$$

En el manejo de prioridades, la convención es usual: "Menores valores numéricos corresponden a prioridades mayores y, con prioridades iguales, se aplican primero los conectivos mas a la izquierda."¹⁹ El conjunto FP de formas proposicionales se define inductivamente, y al mismo tiempo se define la noción de conectivo principal de FP's. A continuación presentamos estas definiciones precisas:

1.- Las variables proposicionales son formas proposicionales con conectivo principal nulo:

$$x \in P \cup \{0, 1\} \Rightarrow (x \in FP) \& (\text{ConPrin}(x) = nil), (\text{Prior}(nil) = 0)$$

2.- Las negaciones de formas proposicionales son formas proposicionales

$$\phi \in FP \Rightarrow \begin{cases} \text{Prior}(\text{ConPrin}(\phi)) \leq 1 \Rightarrow \neg\phi \in FP \ \& \ \\ \text{ConPrin}(\neg\phi) = \neg \\ \text{Prior}(\text{ConPrin}(\phi)) > 1 \Rightarrow \neg(\phi) \in FP \ \& \ \\ \text{ConPrin}(\neg(\phi)) = \neg \end{cases}$$

3.- Las conjunciones y disyunciones, así como las implicaciones y equivalencias, de formas proposicionales son formas proposicionales.

$$\square \in \{\vee, \wedge\}, \text{Prior}(\text{ConPrin}(\square)) = k, \phi_1, \phi_2 \in FP \Rightarrow \begin{cases} \left. \begin{array}{l} \text{Prior}(\text{ConPrin}(\phi_1)) \leq k \ \& \ \\ \text{Prior}(\text{ConPrin}(\phi_2)) \leq k - 1 \end{array} \right\} \Rightarrow \left\{ \begin{array}{l} \phi_1 \square \phi_2 \in FP \ \& \ \\ \text{ConPrin}(\phi_1 \square \phi_2) = \square \end{array} \right. \\ \left. \begin{array}{l} \text{Prior}(\text{ConPrin}(\phi_1)) \leq k \ \& \ \\ \text{Prior}(\text{ConPrin}(\phi_2)) > k - 1 \end{array} \right\} \Rightarrow \left\{ \begin{array}{l} \phi_1 \square (\phi_2) \in FP \ \& \ \\ \text{ConPrin}(\phi_1 \square (\phi_2)) = \square \end{array} \right. \\ \left. \begin{array}{l} \text{Prior}(\text{ConPrin}(\phi_1)) > k \ \& \ \\ \text{Prior}(\text{ConPrin}(\phi_2)) \leq k - 1 \end{array} \right\} \Rightarrow \left\{ \begin{array}{l} (\phi_1) \square \phi_2 \in FP \ \& \ \\ \text{ConPrin}((\phi_1) \square \phi_2) = \square \end{array} \right. \\ \left. \begin{array}{l} \text{Prior}(\text{ConPrin}(\phi_1)) > k \ \& \ \\ \text{Prior}(\text{ConPrin}(\phi_2)) > k - 1 \end{array} \right\} \Rightarrow \left\{ \begin{array}{l} (\phi_1) \square (\phi_2) \in FP \ \& \ \\ \text{ConPrin}((\phi_1) \square (\phi_2)) = \square \end{array} \right.$$

EJEMPLO: ²⁰

Por ejemplo, consideremos el acertijo siguiente:

Ha ocurrido un cuantioso robo en una tienda. Los asaltantes transportaron su botín en una camioneta. Posteriormente se atrapa a tres maleantes sospechosos A, B y C. Las pesquisas muestran evidencias de que A siempre se acompaña de B o de C para sus fechorías, C por su lado nunca actuaría solo, pero también A no se acompaña de C en un atraco. El atraco solo pudo haber sido cometido por A, B o C y al menos uno de ellos es culpable. Hay que decidir la culpabilidad de ellos.

Consideremos tres variables proposicionales para codificar correspondientes hipótesis:

$$\begin{aligned} p_A &: A \text{ es culpable,} \\ p_B &: B \text{ es culpable,} \\ p_C &: C \text{ es culpable.} \end{aligned}$$

¹⁹ Tomado de: LogicaHoy/GMorales990426.ps+%22culpabilidades+de+A,+B+y+C%22&hl=es

Los "hechos" siguientes pueden representarse por correspondientes formatos proposicionales:

- 1.- Si A fuese culpable y B inocente, entonces C ha de ser culpable: $p_A \wedge \neg p_B \rightarrow p_C$
- 2.- C nunca actuaría solo $p_C \rightarrow p_A \vee p_B$
- 3.- A nunca actuaría con C: $\neg(p_A \wedge p_C)$
- 4.- Nadie mas que A, B o C pudieron haber actuado y al menos uno de ellos es culpable: $p_A \vee p_B \vee p_C$

de acuerdo con el acertijo, si los cuatro hechos anteriores fuesen verdaderos, que podría decirse acerca de las culpabilidades de A, B y C? Y si acaso se tuviese una evidencia de que cada uno de esos hechos es verdadero con una cierta probabilidad, que podría decirse acerca de las probabilidades de A, B y C sean culpables?

Para resolver este acertijo podemos utilizar varios métodos:

Interpretación Min-Max: para las proposiciones del ejemplo tenemos que:

$$a = v(p_A), b = v(p_B) \text{ y } c = v(p_C)$$

Entonces en la interpretación Min-Max tenemos:

$$\begin{aligned} v(p_A \wedge \neg p_B \rightarrow p_C) &= \text{Max}[1 - \text{Min}[a, 1 - b], c] = \text{Max}[1 - a, b, c] \\ v(p_C \rightarrow p_A \vee p_B) &= \text{Max}[a, b, 1 - c] \\ v(\neg(p_A \wedge p_C)) &= 1 - \text{Min}[a, c] = \text{Max}[1 - a, 1 - c] \\ v(p_A \vee p_B \vee p_C) &= \text{Max}[a, b, c] \end{aligned}$$

Dicho en palabras: para cualquier valor $k \in [0, 1]$ se tiene que cada una de estas proposiciones tiene un valor mayor o igual a k si y solo si $k \leq \text{Min}[(1-a), b]$, o bien $k \leq \text{Min}[(1-c), b]$, es decir los cuatro hechos serán tanto más verdaderos cuanto a

²⁰ Sugeno, M., "Industrial applications of fuzzy control". Elsevier Science Pub. Co. Pag. 56

la vez A es inocente y B es culpable, o C es inocente y B es culpable (en cualquiera de los dos casos la culpabilidad o inocencia del tercer implicado es irrelevante).

Interpretación Producto-D. Se tiene:

$$\begin{aligned}
 v(p_A \wedge \neg p_b \rightarrow p_C) &= 1 - a + ab + ac - abc \\
 &= 1 - a(1 - b)(1 - c) \\
 v(p_C \rightarrow p_A \vee p_B) &= 1 - c + ac + bc - abc \\
 &= 1 - (1 - a)(1 - b)c \\
 v(\neg(p_A \wedge p_C)) &= 1 - ac \\
 v(p_A \vee p_B \vee p_C) &= a + b - ab + c - ac - bc + abc \\
 &= 1 - (1 - a)(1 - b)(1 - c)
 \end{aligned}$$

Así pues para cualquier $k \in [0, 1]$ cada una de estas proposiciones tiene un valor mayor o igual a k si y solo si se cumplen las siguientes cuatro desigualdades:

$$\begin{aligned}
 a(1 - b)(1 - c) &\leq 1 - k \\
 (1 - a)(1 - b)c &\leq 1 - k \\
 ac &\leq 1 - k \\
 (1 - a)(1 - b)(1 - c) &\leq 1 - k
 \end{aligned}$$

Por ejemplo se a y $(1 - b)$ son pequeñas, (independientemente del valor de c), A es inocente y B es culpable, entonces k tiende a ser 1, es decir, los hechos tienden a ser verdaderos, y el recíproco también vale. Lo mismo pasa si c y $(1 - b)$ son pequeños.

1.10.2 Semánticas basadas en conjunción y negación

A toda forma proposicional se le puede asociar un valor de verdad, el cual ha de estar en función de los valores de verdad de las variables proposicionales que aparezcan en la forma proposicional y de la estructura de esa forma.

$$v : P \rightarrow [0, 1]$$

Una asignación es una función $v(p) \in [0, 1]$ que a cada variable proposicional $p \in P$

le asocia un valor de verdad. Si $v(p)=1$ decimos que p es verdadera, en tanto que si $v(p) = 0$ decimos que p es falsa. A toda asignación v la podemos extender a las constantes 0, 1 haciendo $p(0)=0$ y $p(1) = 1$.

1.10.2.1 Incertidumbre²¹

La propagación de incertidumbres hacia adelante se restringe, en el caso del cálculo proposicional clásico a la evaluación de tablas de verdad.

La propagación de incertidumbres hacia atrás, suele decirse, también, *de diagnóstico*: Si se tiene síntomas con una cierta intensidad, y éstos pueden depender de alguna manera de causas “atómicas” distinguidas, entonces ¿en qué medida están presentes esas causas para ocasionar tales síntomas?

1.11 DESDIFUSIFICAR

La operación de desdifusificar²², u operación-DF para abreviar, consiste en seleccionar un elemento representativo de un conjunto difuso. Con esta operación "se suprime lo difuso" porque habiendo estimado propiedades de un conjunto difuso, se elige a un objeto "concreto" que lo representa. Para esto existen diversos criterios.²³

1.11.1 Criterios de desdifusificación

1.11.1.1 Primer Máximo.

Tómese como representante de un conjunto difuso al primer elemento x_A en el universo X , de acuerdo con un orden dado, tal que $A(x_A) = \text{Max}\{A(x)|x \in X\}$

²¹ Incertidumbre: Falta de credibilidad.

²² Desdifusificar.- llamada en inglés de manera más que simple Defuzzify.

²³ Tomado de la dirección electrónica <http://delta.cs.cinvestav.mx/gmorales/df/node4.html>

Este criterio conlleva la dificultad de calcular un valor máximo de una función real, precisamente A , definida sobre X .

1.11.1.2 Corte-a

Dado un conjunto difuso A en un universo X , sea $a \in]0, 1[$ un número real positivo, pero estrictamente menor que 1. Diremos que el número a es el umbral de corte. Elíjase un elemento $x_0 \in A_a$ en el corte- a de A .

1.11.1.3 Centroide.

Dado un conjunto difuso A en un universo X , sea $m_1(A) = \sum_{x \in X} A(x)p_A(x)$ su centroide. Elíjase al elemento $x_0 \in X$ tal que

$$|A(x_0) - m_1(A)| = \text{Min}\{|A(x) - m_1(A)| \mid x \in X\}$$

Es decir, x_0 es uno de los elementos en el universo X cuyo grado de pertenencia a A es el más cercano al valor esperado de los valores de A .

1.12 SISTEMAS BASADOS EN LOGICA DIFUSA

1.12.1 FLS

Un sistema basado en lógica difusa (fuzzy logic system) FLS, es el único tipo de sistema capaz de tratar simultáneamente con variables numéricas y con variables lingüísticas de modo formal. Las variables numéricas, que son las más habituales, pueden caracterizarse por un valor numérico, por ejemplo, la temperatura del ambiente

es de 20 grados centígrados. Las variables lingüísticas, sin embargo, se caracterizan por un adjetivo que las califica, por ejemplo, la temperatura del ambiente es alta.

Un FLS se basa en un mapeo no lineal de un vector de entrada en una salida escalar. La Teoría de Conjuntos Difusos y la Lógica difusa establecen las especificaciones de este mapeo no lineal. Un FLS puede expresarse matemáticamente como una combinación lineal de funciones base difusas, y es un aproximador universal no lineal de funciones; de modo que el desarrollo en funciones base difusas es muy potente porque éstas pueden obtenerse a partir de datos numéricos o bien conocimiento lingüístico; en ambos casos, se puede presentar en la forma de reglas If-Then

Existen dos tipos de conocimientos sobre un determinado problema:

Conocimiento objetivo: cuantificado habitualmente mediante modelos matemáticos. Ejemplos: ecuaciones de movimiento de un robot, que describen un canal de comunicaciones, etc.

Conocimiento subjetivo: contiene información lingüística que no es posible cuantificar mediante modelos matemáticos tradicionales. Un ejemplo de este tipo sería la siguiente regla que puede ser válida para seguir la pista de un objeto de grandes dimensiones moviéndose lentamente: Si el objeto está localizado en un lugar en un instante temporal, en el siguiente instante no estará demasiado lejos de ese lugar.

Aunque el conocimiento subjetivo suele ser ignorado a la hora de enfrentarnos a un problema, sí se toma en cuenta para evaluar la solución que se le da. Esto sugiere la posibilidad de utilizar ambos tipos de conocimiento para resolver problemas reales, que es lo que hace la Lógica difusa o Fuzzy logic.

Se pueden observar dos formas de abordar un problema:

Basada en modelos: La información objetiva se representa por modelos matemáticos, y la información subjetiva por afirmaciones lingüísticas convertidas en reglas, que son cuantificadas usando Lógica Difusa.

Libre de Modelos: Las reglas se obtienen de los datos numéricos y se combinan con información lingüística (ofrecida por el experto) usando Lógica difusa.

La potencia de la lógica Difusa es que hay muchas posibilidades distintas que conducen a mapeos diferentes.

En cuanto a las razones para la utilización de la Lógica Difusa, hay que destacar que las clases imprecisamente definidas representan un importante papel en el pensamiento humano, particularmente en lo que concierne a reconocimiento de patrones, comunicación de información y capacidad de abstracción. Además ha de tomarse en cuenta que conforme aumente la complejidad de un sistema, la capacidad para realizar afirmaciones precisas sobre su comportamiento disminuye hasta un umbral.

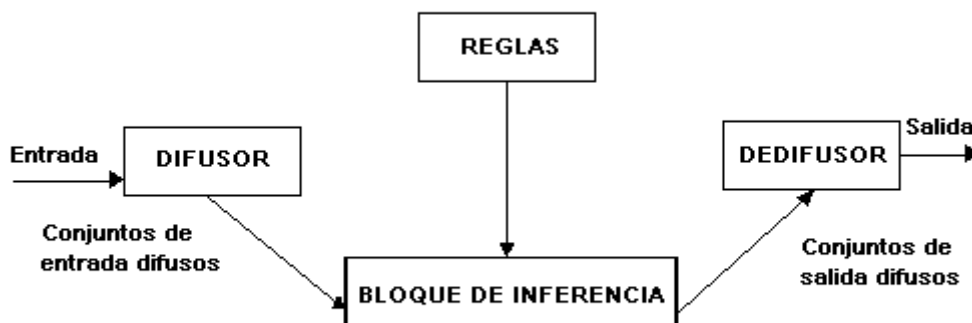
En la siguiente figura Representa un Sistema Difuso utilizando controladores y aplicaciones de Procesado de Señales. Un Sistema difuso mapea entradas convencionales (no difusas) en salidas convencionales. Podemos observar 4 componenetes:

Reglas

Difusor

Bloque de Inferencia

Dedifusor



Difusor: Mapea los números convencionales en conjuntos difusos, lo cual es necesario para activar reglas que están en términos lingüísticos y que tienen conjuntos difusos asociados a ellas.

Bloque de inferencia: Mapea conjuntos difusos en conjuntos difusos y se encarga de combinar las reglas.

El Dedifusor mapea conjuntos de salida en números, debido a que en muchas aplicaciones, a la salida del Sistema de Lógica Difusa se desea obtener un número.

Reglas: pueden ser facilitadas por expertos o bien ser deducidas a partir de los datos numéricos, son un colección de sentencias IF-THEN, por ejemplo “Si a está caliente y b está bajo, entonces gira c a la derecha”. Esta única regla revela que necesitamos tener conocimiento de:

- Correspondencia entre variables lingüísticas y valores numéricos de una variable, por ejemplo caliente puede corresponder a 20 grados.
- Cuantificación de variables lingüísticas (por ejemplo, a puede tener asociado un número finito de términos lingüísticos asociados entre extremadamente caliente y extremadamente frío).
- Conexiones lógicas para las variables lingüísticas.
- Implicaciones

Además, será necesario conocer cómo combinar más de una regla, según las probabilidades que se puedan obtener.

1.12.2 Funciones de partición

Las funciones de partición (membership functions, MF) están, en su mayor parte, asociadas con términos que aparecen en las reglas, ya sea en la parte antecedente o en la consecuente. La parte antecedente de una regla es la formada por las distintas premisas de ésta, mientras que la parte consecuente es la constituida por las consecuencias. Por ejemplo.

Si la temperatura es de 35oC. Encender el ventilador

La parte antecedente estaría formada por una única premisa, que el ambiente está muy cálido, la parte consecuente sería la orden de encender un ventilador.

Las formas mayormente utilizadas para las Funciones son la triangular, trapezoidal, lineal a trozos, gaussiana o con forma de campana (algo más cuadrada que la gaussiana y la más utilizada en redes neuro-difusas). Aunque lo habitual es elegir unas u otras basándonos en la propia experiencia, existen algunos procedimientos que permiten optimizar esta decisión.

Cuanto mayor es el número de funciones mayor posibilidad de alcanzar una mayor resolución aunque, evidentemente, acarreará un mayor coste computacional, de tal manera que en cada caso se ha de llegar a la solución óptima por compromiso.

Aunque no es necesario que las MFs se solapen²⁴, gran parte de la potencia de la Lógica Difusa reside en esta posibilidad. En este sentido existe la capacidad de distribuir las decisiones sobre más de una clase de entrada, lo que permite implementar FLS más robustos.

A pesar de que las funciones no tienen por qué estar escaladas entre cero y la unidad, se suele hacer así. Siempre se puede normalizar un conjunto difuso dividiendo $u(x)$ por su valor máximo o bien se puede seguir otras estrategias de normalización.

1.12.3 Las Reglas

Uno de los componentes básicos de los Sistemas difusos son las reglas, las reglas se expresarán como implicaciones lógicas en las formas de sentencias IF-THEN. Una regla representa un tipo especial de relación entre A Y B, cuya MF se denota por:

²⁴ Solapen: Cubrise entre ellas.

$$\mu_A(x) \rightarrow B(x,y)$$

La extensión de la lógica ordinaria a la lógica Difusa se realiza sustituyendo las funciones bivalentes²⁵ por funciones difusas. Démonos cuenta que la Función de partición (MF) da idea del grado en qué es verdad la relación de implicación entre x e y.

1.12.4 Manejo del conocimiento.

1.12.4.1 Con lógica binaria.

Con el advenimiento de las computadoras digitales, se creó toda una ciencia para el manejo de información mediante secuencias de ceros y unos. Esta forma de representación utiliza la lógica binaria como base para el almacenamiento y recuperación de información.

Para el manejo del conocimiento se utiliza un lenguaje simbólico. Estos símbolos se combinan mediante reglas y a partir de éstas pueden modelarse sistemas complejos. Por ejemplo, los sistemas expertos en inteligencia artificial. La principal desventaja de esta representación del conocimiento es que se necesitan demasiadas reglas para poder representar fielmente un sistema del mundo real. Básicamente porque hay una relación de uno a uno. Es decir, para un caso particular del sistema del mundo debe existir una regla en el sistema experto.

1.12.4.2 Con lógica difusa

La representación del conocimiento mediante lógica difusa permite disminuir drásticamente el número de reglas que se necesitan para modelar un sistema. Mediante este enfoque, una regla difusa cubre varios casos dentro del sistema del mundo real.

²⁵ Bivalentes: Que tiene dos valencias.

1.13 CONTROL DIFUSO

El modelo desarrollado de un horno eléctrico empleando el ToolBox de Lógica Difusa, incluido en Matlab, y Simulink permite disponer de una herramienta que facilite las tareas de simulación mediante el uso de reglas las cuales son obtenidas a través de la experiencia de un experto.

La lógica difusa tiene la ventaja de manejar información abundante e imprecisa que caracteriza al mundo real. Esta información, como se mencionó antes, es expresada mediante reglas en el formato de IF (CONDICION) THEN.

De acuerdo a la literatura sobre lógica difusa, “una lógica basada en dos valores TRUE y FALSE es algunas veces inadecuado cuando se describe el razonamiento humano. La lógica difusa usa el intervalo completo entre 0 (FALSE) y 1 (TRUE) para describir el razonamiento humano. Como un resultado, la lógica difusa está siendo aplicada en controladores automáticos basados en reglas”.

En cualquier controlador difuso, (Ver controlador PID, Pag. 82) la relación entre objetos juega un papel importante. Algunas relaciones tienen que ver con elementos dentro del mismo universo: una medición es mayor que otra, un evento ocurre antes que otro, un elemento se asemeja a otro, etc. Otras relaciones tienen que ver con universos disjuntos: la medición es mayor y su razón de cambio es positiva, la coordenada X es mayor y la coordenada Y es pequeña, por ejemplo. Estos ejemplos son relaciones entre dos objetos, pero en principio se pueden tener relaciones que empleen cualquier número de objetos.

El diseño de un controlador difuso requiere más decisiones de diseño que lo usual, mediante el uso de reglas, de una máquina de inferencia, del proceso de desfusificación y fusificación, y del pre y post procesamiento de datos.

Mientras que el diseño de un controlador con PID es relativamente fácil, la inclusión de reglas difusas crea muchos problemas extras y, aunque muchos libros sobre Lógica Difusa explican el control con lógica difusa, existen muy pocas directivas para establecer los parámetros de un controlador simple con lógica difusa.

El método tradicional de diseño consiste en tres pasos:

1. Iniciar con un controlador PID
2. Insertar un controlador difuso lineal equivalente
3. Hacerlo gradualmente no lineal.

El control difuso es un método de control basado en lógica difusa. La lógica difusa se considera como un método de cálculo basado en palabras antes que en números y, el control difuso se describe como un método de control mediante sentencias en vez de ecuaciones.

Un controlador difuso puede incluir reglas empíricas y es especialmente útil en plantas controladas por operadores.

En un controlador basado en reglas, la estrategia de control es almacenada en más o menos un lenguaje natural. La estrategia de control está aislada en una base de reglas opuesta a una descripción basada en ecuaciones. Un controlador basado en reglas es fácil de comprender y fácil de mantener para un usuario final no especialista en el área.

Las reglas pueden emplear varias variables tanto en la condición como en la conclusión de las reglas.

Los controladores puede, por lo tanto, ser aplicados tanto a problemas de múltiple entrada y múltiple salida (MIMO: MULTIPLE INPUT – MULTIPLE OUTPUT) como en problemas de simple entrada y simple salida (SISO: SINGLE INPUT – SINGLE OUTPUT). El problema típico SISO es regular una señal de control basado en una señal de error. El controlador puede realmente necesitar tanto el error, el cambio del error, y el error acumulado como entradas.

El diseñador se enfrenta al problema de cómo construir el conjunto de términos. Existen dos preguntas específicas a considerar: (i) ¿Como se determina la forma de los conjuntos?, (ii) ¿Cuántos conjuntos son suficientes y necesarios?

Para responder estas preguntas se tienen algunas reglas a considerar:

- (i) Un conjunto de términos debería ser lo suficientemente amplio como para permitir el ruido en la medición.

- (ii) Una cierta cantidad de superposición (traslape) es deseable; caso contrario, el controlador puede ejecutar en estados pobremente definidos dando lugar a que no resulte una salida bien definida.

En cuanto a las formas de los conjuntos se recomienda lo siguiente:

- Empezar con conjuntos triangulares. Todas las funciones miembros para una entrada o salida particular deberían ser triángulos simétricos del mismo ancho.
- El traslape debería ser al menos del 50%. Los anchos deberían inicialmente ser escogidos de modo que cada valor del universo es un miembro de al menos dos conjuntos, excepto posiblemente para los elementos de los extremos. Si, por otro lado, existe un espacio entre dos conjuntos la función del controlador no está definida.

CAPITULO II

2 MATLAB

2.1 INTRODUCCION

En este capítulo realizaremos un enfoque meramente práctico de las funciones y aplicaciones básicas del MATLAB, que será desde aquí para adelante el sistema experto donde desarrollaremos la aplicación propuesta en nuestra monografía.

Se mostrará al Matlab como una herramienta efectiva para la realización de los cálculos matemáticos que necesitaremos realizar en la aplicación que le daremos posteriormente un nombre, cabe indicar esta referencia bibliográfica será claramente sustentada con ejemplos de modo que cualquier usuario interesado en conocer a Matlab en su forma integra y sus bondades podrá utilizar, aunque es muy difícil plasmar la cantidad de comandos y bondades que tiene MATLAB, nos serviremos de los ejemplos realizados por nosotros en las prácticas previas a esta investigación.

Ponemos entonces a su disposición esta ilustración que le será de ayuda y guía práctica a lo largo de esta monografía sí como en el uso del MATLAB, en otros campos a ser aplicados.

2.2 CONCEPTO DE MATLAB

MATLAB = 'MATrix LABoratory' (LABORATORIO DE MATRICES), MATLAB es un medio computacional técnico, con un gran desempeño para el cálculo numérico computacional y de visualización, es un sistema experto en todo el ámbito de la palabra, manipula muchísimas herramientas que lo hacen poderoso para el trabajo en distintos ámbitos.

MATLAB integra análisis numérico, matrices, procesamiento de señales y gráficas, todo esto en un ambiente donde los problemas y soluciones son expresados tal como se escriben matemáticamente. Cabe indicar que si usted ha trabajado con

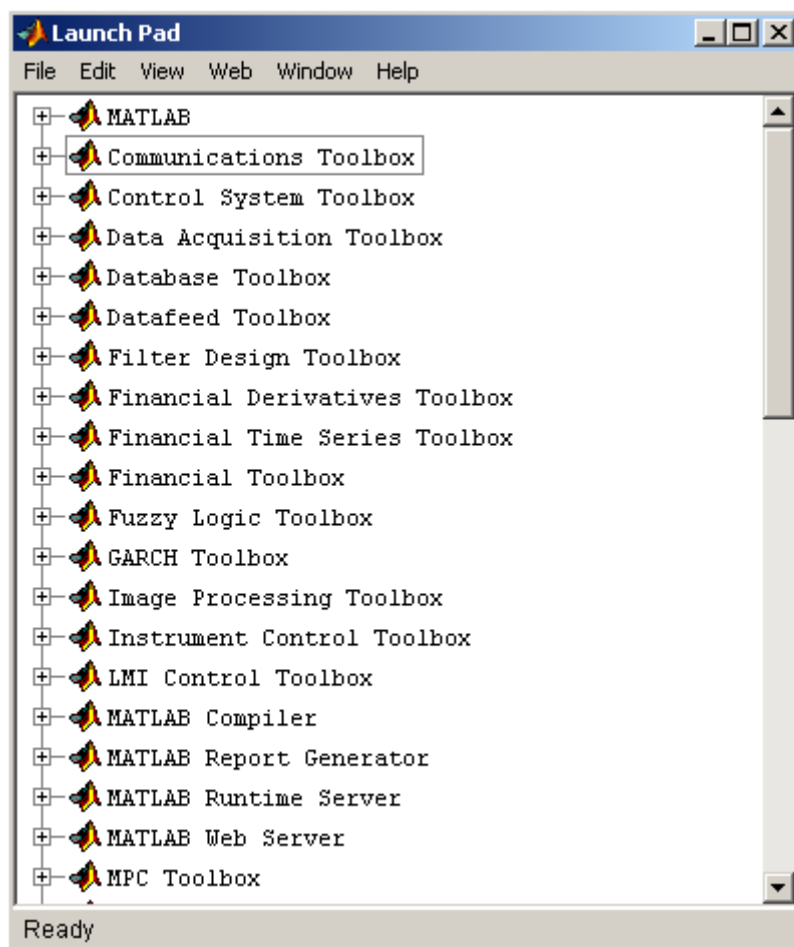
funciones en una hoja electrónica estará más interrelacionado con esta herramienta. MATLAB fue escrito originalmente en fortran, actualmente está escrito en lenguaje C.

MATLAB es un lenguaje de programación amigable al usuario con características más avanzadas y mucho más fáciles de usar que los lenguajes de programación como basic, pascal o C.

Haciendo hincapié en nuestra monografía, utilizaremos MATLAB con sus herramientas el FUZZY LOGIC, y el SIMULINK, para realizar un diagrama de bloques., no así cabe indicar que MATLAB contiene muchas otras herramientas que pueden ser utilizados por el experto de acuerdo a sus necesidades.

2.3 TOOLBOXES DE MATLAB

Cuando se carga MATLAB se presentan los toolboxes de forma muy visible listos a ser utilizados.



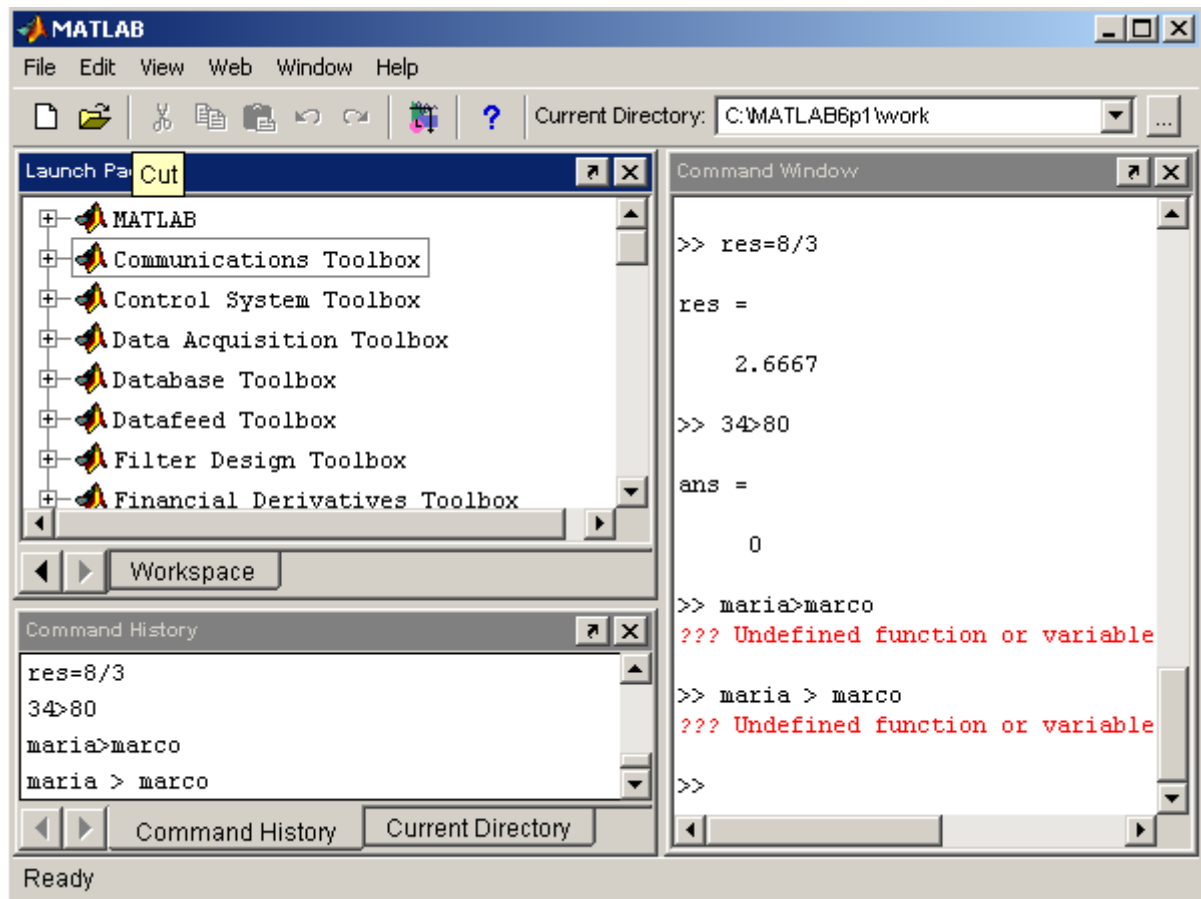
- Control system Toolbox, Robust Control Toolbox
- Frequency Domain System Identification Toolbox
- **Fuzzy Logic Toolbox**
- Higher Order Spectral Analysis Toolbox
- Image Processing Toolbox
- Model Predictive Control Toolbox
- Mu Analysis and Synthesis Toolbox
- NAG Foundation Toolbox
- Neural Network Toolbox
- Nonlinear Control Design Toolbox
- Optimization Toolbox
- Quantitative Feedback Theory Toolbox
- Signal Processing Toolbox
- **SIMULINK, SIMULINK Real Time Workshop**
- Spline Toolbox
- Statistics Toolbox
- Symbolic Math Toolbox
- System Identification Toolbox.

2.4 COMO INICIAR MATLAB

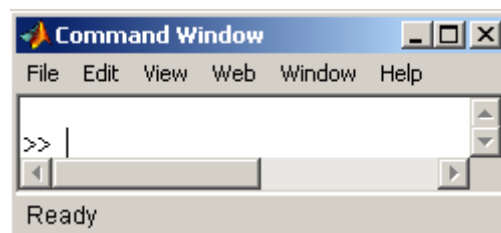
MATLAB se inicia directamente desde Windows, en nuestro caso tenemos el siguiente icono en el escritorio, que indica claramente la versión de MATLAB usada en esta monografía.



Al invocarse MATLAB aparecerá una pantalla como la siguiente:



En la pantalla de comandos, el símbolo `>>`, el cual indica la entrada de instrucciones para ser evaluadas, tenemos la ventana de los TOOLBOXES, anteriormente mencionada y una ventana de “Command History”, que es la ventana de la Historia de los comandos usados últimamente.



`>>` Comando o instrucción a evaluar

`< enter >`

Para hacer la suma de dos números, escribimos :

`>> 20 + 5` `< enter >`

Presionamos la tecla entrar.

ans =

25

El resultado es desplegado y se guarda en la variable ans (answer).

2.5 FUNCIONAMIENTO DEL MATLAB

MATLAB puede almacenar información en variables tales como :

```
a = 100
```

"<ENTER> para evaluar la celda "

Cada vez que capturamos información en MATLAB y presionamos <ENTER> ésta es desplegada inmediatamente, pero si ponemos un punto y coma al final de la instrucción MATLAB omite el desplegado de información.

Por ejemplo :

```
b = 50 ;
```

Si se quiere saber el valor de alguna variable capturada sólo se tiene que poner el nombre de la variable y <ENTER> y MATLAB lo despliega. Estas variables residen en el espacio de trabajo de MATLAB.

2.6 ESPECIFICACION DE VARIABLES

- Las variables son sensibles a las mayúsculas, por lo que las siguientes variables son diferentes :

```
Suma = 1
suma = 1
```

- Las variables pueden contener hasta 19 caracteres. Éstas deben empezar con una letra, seguida por cualquier número de letras, dígitos o guiones de subrayado.
- Los caracteres de puntuación no son permitidos en las variables.

Cuando se trabaja con muchas variables estas son difíciles de recordar.

El comando **who** muestra un desplegado de todas aquellas variables que se han estado utilizando.

```
who
```

```
whos
```

Muestra las variables con información adicional.

En general si usted desea operar variables y constantes lo puede realizar con total tranquilidad. Ejm:

A=1
 B=2
 Suma = A+B
 Total = Suma+15

Total nos dará 18

2.6.1 Borrado de variables.

Para borrar el valor de una variable simplemente ponemos

`clear a` Borra la variable " a "

`a`
`clear a b c` Borra las variables " a ", " b " y " c "

" **CLEAR** " Borra todas las variables y no se pueden recuperar.

2.6.2 La variable NaN

Cuando encontramos una situación en la que se da como resultado un error pues por ejemplo al dividir 0/0 en MATLAB el programa no se detiene como suele ocurrir en otros lenguajes de programación, aquí tenemos únicamente como un Warning o mensaje de que se ha producido este inconveniente, el resultado se da en una variable interna NaN (Not a Number),

Ejemplo: defina

a=[10 12 0] y

b=[10 12 0] ahora pida la división elemento a elemento (comando "./")

`a ./ b`

observemos como se realizaría esta práctica y el uso de la variable NaN.

```

Command Window
>> a=[10 12 0]

a =

    10    12     0

>> b = [10 12 0]

b =

    10    12     0

>> a./b
Warning: Divide by zero.

ans =

     1     1   NaN

>> |

```

2.7 USO DE CARACTERES ESPECIALES EN MATLAB

[]	Son usados para formar vectores y matrices	[1 2 3 ; 4 5 6]
()	Usados para expresiones matemáticas.	sqrt(2)
=	Usado para hacer asignaciones.	x = 5
'	Transpuesta de una matriz	A'
'	Usado para separar texto	'texto'
.	Punto decimal	3.1415
...	Al final de una línea indican que continua en el siguiente renglón.	2,3,4,5,6 7,8,9,10]
,	Para separar elementos	[1,2,3,4]
;	Para separar filas en las matrices.	[1 2; 3 4]
;	Para evitar que se despliegue la información capturada.	[3] ;
%	Para hacer comentarios	% este programa, etc.
!	Para ejecutar un comando del Ms-dos	!dir

2.8 OPERACIONES BASICAS

En MatLab las operaciones se realizan con total facilidad pues recordemos es un experto en las matemáticas.

Operación	Ejemplo
Suma	C = a + b
Resta	d = a - b
Multipliación	e = a * b
División	F = a / b
Potencia	a ^ 2

2.9 FUNCIONES

Como se dijo anteriormente, MATLAB, trae incorporado una serie de funciones, para los casos vale la pena recordar, lo que es una función y su formato:

FUNCION: Es una operación matemáticas, estadística, trigonométrica, de fecha, etc.... que se encuentra incorporada y lista para ser utilizada, que tiene como misión fundamental minimizar el grado de error a 0 y realizar la resolución inmediata de cualquiera de las operaciones escogidas. Para poder realizar todo esto se necesita de un “Argumento”

ARGUMENTO: Valor o valores que se utilizarán para realizar la resolución de una función, se coloca dentro de los paréntesis.

2.9.1 Formato de una función:

Nombre función(<Argumento>)

Ejemplo:

`sin (0.5)`

Seno de (0.5)

2.9.2 Funciones trigonométricas

Así mismo tenemos las siguientes funciones trigonométricas

cos (x)	tan (x)		
asin (x)	acos (x)	atan (x)	inversa
sinh (x)	cosh (x)	tanh (x)	hiperbólica
asinh (x)	acosh (x)	atanh (x)	inversa- hiperbólica

2.9.3 Funciones para LOGARITMOS

<code>log (0.5)</code>	Logaritmo natural
<code>log₁₀ (x)</code>	Logaritmo decimal.

2.9.4 Funciones matemáticas especiales.

<code>abs (-3)</code>	Valor absoluto o magnitud de un número complejo
<code>ceil (128.123123)</code>	Redondea hacia más infinito, el resultado es 129
<code>floor (x)</code>	Redondea hacia menos infinito
<code>fix (x)</code>	Redondea hacia cero
<code>round (x)</code>	Redondea hacia el entero más próximo
<code>imag ()</code>	Parte imaginaria de un número complejo

Por ejemplo: `imag (4-30z)`, el resultado es 30

<code>real (x)</code>	Parte real de un número complejo
<code>angle (x)</code>	Angulo de un número complejo
<code>conj (x)</code>	Complejo conjugado
<code>sign (-5)</code>	Función signo : Devuelve el signo del argumento (1 si es positivo, -1 si es negativo)
<code>exp (1)</code>	Exponencial : $e (x)$
<code>rem (x,y)</code>	Resto después de la división (x / y)
<code>sqrt ()</code>	Raíz cuadrada

2.10 OPERACIONES LOGICAS EN MATLAB

Las operaciones lógicas en matlab se las trabaja de una manera muy sencilla, se dispone pues de un conjunto muy completo de operadores lógicos, relacionales y matemáticos como ya hemos visto, se mostrarán la lista de estos operadores y posteriormente realizaremos algunos ejemplos para dejar plasmado su utilización, dado que en nuestra monografía tiene un valor importante con su uso común como si lo trabajáramos en cualquier otro lenguaje de programación.

>	Mayor que
<	Menor que
>=	Mayor que o igual
<=	Menor que o igual
= =	Igual a
~=	No es igual a

Ejemplos:

Si nosotros ponemos

34 < 2 34 menor que 2

Como 34 es mayor que 2, la respuesta es falsa por lo que obtenemos un 0.

Las únicas respuestas posibles con las operaciones lógicas son:

Verdadero = 1 y Falso = 0.

Operadores lógicos:

AND	&
OR	
NOT	~

Para que la operación AND sea verdadera las dos relaciones deben ser verdaderas.

Recordemos una tabla de verdad con el siguiente ejemplo: Se obtiene graduación si la Nota de la Monografía escrita es mayor a 18 y la nota de sustentación mayor a 16. Evaluemos:

Monografía Escrita	Sustentación	Resultado
19	18	Si graduación
1	1	1

La tabla de verdad correspondiente a continuación

0	0	0	Falso
0	1	0	Falso
1	0	0	Falso
1	1	1	Verdadero

(10 < 20) & (20 < 35) Verdadero.

(10 < 20) & (25 < 12) Falso.

Para la operación **OR** :

0	0	0
0	1	1
1	0	1
1	1	1

(15 < 20) | (2 < 1) Verdadero.

Para la operación **NOT** :

~ 0 | 1

~ 1 | 0

~ (25 < 18) Verdadero.

Solución de ecuaciones de segundo grado.

Aunque este ejemplo es tomado de un libro de MATLAB, nos pareció importante ponerlo en forma textual en esta parte de la monografía pues aclara y da amplitud a nuestra forma de razonar en cuanto a los alcances que uno puede tener al saber manipular MATLAB como esa herramienta poderosa, y precisa para la resolución de problemas matemáticos, como podemos observar en el ejemplo siguiente resulta muy

simple resolver esta ecuación, inclusive sin conocer mucho de la parte matemática en cuestión.

²⁶MATLAB se puede resolver fácilmente ecuaciones del tipo $\mathbf{ax}^2 + \mathbf{bx} + \mathbf{c} = \mathbf{0}$, haciéndolo como si fuera una sola instrucción. La fórmula para resolver una ecuación de segundo grado de este tipo es :

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Si tenemos los siguientes valores :

$$a = 1, b = 2, c = 3$$

Escribimos la formula para x1 :

$$x1 = (-b + \text{sqrt} (b^2 - 4 * a * c)) / 2 * a$$

Para x2 :

$$x2 = (-b - \text{sqrt} (b^2 - 4 * a * c)) / 2 * a$$

Podemos hacer la comprobación para x1.

$$a * x1^2 + b * x1 + c$$

Comprobación x1

2.11 LOS VECTORES Y MATRICES EN MATLAB

Como ya hemos mencionado anteriormente Matlab es un experto en el tratamiento matemático a base de Matrices de ahí que es de vital importancia comprender todas las bondades que ofrece para el tratamiento de estos arreglos, aunque nuestra monografía no utiliza mucho estas formas matemáticas es importante conocer su funcionamiento y su aplicación.

²⁶ Tomado Tutorial de MATLAB Datta, B.N, Linear Álgebra in Signals, Systems and Control, Society for industrial & applied Maths, USA. 1988, pag.23

2.11.1 Arreglos (Arrays) ó Vectores.

Para definir un arreglo lo hacemos con [], aunque como se verá posteriormente existe un sinnúmero de formas de manipular y definir arreglos y matrices.

2.11.1.1 Definir arreglos

```
x = [ 0, 0.50, 1, 1.50, 2 ]
```

Define un arreglo con valores desde 0 que van aumentando de 0.5 hasta llegar a 2.

Se pueden omitir las comas cuando se capturan los números.

Con los números capturados, se puede obtener cualquier función, seno, coseno, tangente.

```
sin (x)
cos(x)
tan(x)
```

Si necesitaríamos capturar valores en arreglos mucho más grandes resultaría tedioso y con datos no precisos.

Para evitar capturarlos a mano, MATLAB nos permite crear un vector de la siguiente manera:

Variable = (**Valor inicial : Con incrementos de : Valor final**)

El ejemplo anterior lo podríamos haber hecho de la siguiente manera.

```
x = ( 0 : 0.5 : 2 )
```

sen (x) Ahora se puede obtener el coseno de la variable R.

Otra forma de crear arreglos es :

```
x1 = 1 : 5
```

Arreglo de 1 a 5, con incremento de 1

```
x2 = 10 : 5 : 100
```

Arreglo de 10 a 100, con incrementos de 5.

2.11.1.2 Localización de los elementos de un vector

Hagamos el siguiente vector :

```
x = ( 0 : 5 : 20 )
```

Si queremos saber cual es el tercer elemento del vector ponemos:

```
elem (3)
```

el resultado sería 15.

Si nos interesan los elementos 1 al 3 :

```
y( 1 : 3 )
```

Entonces aquí utilizamos los: para implicar rango de valores.

Otras opciones son:

```
y( 1 : 2 : 9 )
```

 Toma los elementos del 1 al 9 con incrementos de 2

```
y([ 1, 3, 7,10])
```

 Toma los elementos 1, 3, 7 y 10 del array

2.11.1.3 Vectores (su orientación)

Si separamos cada elemento del arreglo con punto y coma tenemos un arreglo de una sola columna:

```
a = [ 1; 2; 3; 4; 5; 6 ]
```

Es necesario usar los corchetes, porque si no los usamos obtenemos el último valor que capturamos:

Para crear una columna con 10 elementos hacemos lo siguiente:

```
d = ( 2 : 2 : 20 )
```

y trasponemos el renglón a columna, es decir buscamos la transpuesta. (')

```
c = d'
```

Ahora nuevamente tengamos la inversa de c.

```
c'
```

A continuación tenemos el ejemplo en Matlab.

```
>> d=(2:2:20)
```

```
d =
```

Columns 1 through 5

```

    2   4   6   8  10
Columns 6 through 10
    12  14  16  18  20

>> c=d'

c =

    2
    4
    6
    8
   10
   12
   14
   16
   18
   20

>> c'

ans =

Columns 1 through 5
    2   4   6   8  10
Columns 6 through 10
    12  14  16  18  20

```

2.11.1.4 Modificaciones de los arreglos

Si el noveno elemento del array debió ser el número 5 en vez de 3, corregimos de la siguiente manera :

```
y(9) = 5
```

Otra forma de hacer arreglos, es con **linspace** :

Linspace (Valor inicial , Valor final , Número de elementos)

Note el uso de comas (#, #, #)

```
z = linspace(0 , 10, 101)
```

Linspace describe una relación lineal de espaciado entre sus elementos.

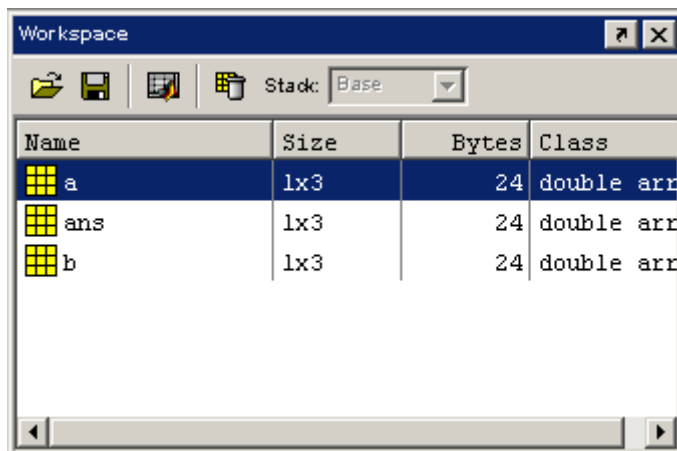
Logspace describe una relación de espaciado "logarítmica".

Logspace (Primer exponente , Último exponente , Cantidad de valores)

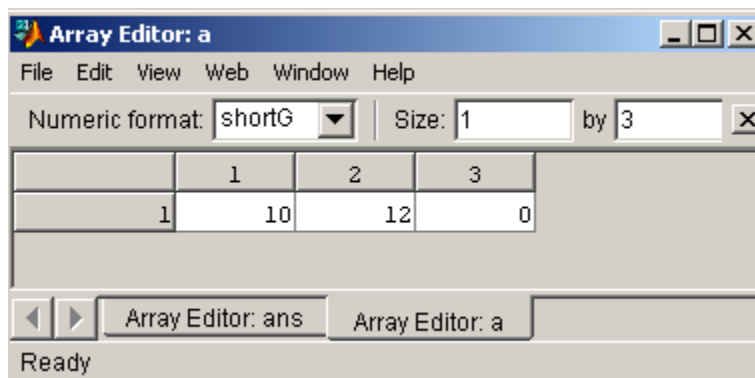
Logspace (0 , 2 , 10)

Hemos creamos un arreglo que comienza en 10^0 y termina en 10^2 , conteniendo 10 valores.

Cuando creamos arreglos en Matlab automáticamente se va generando en el Workspace todas las especificaciones de estos como nos podemos percatar en el siguiente ejemplo:



Si damos por ejemplo doble clic en el arreglo a vizualizaremos lo que se denomina el editor de arreglos:

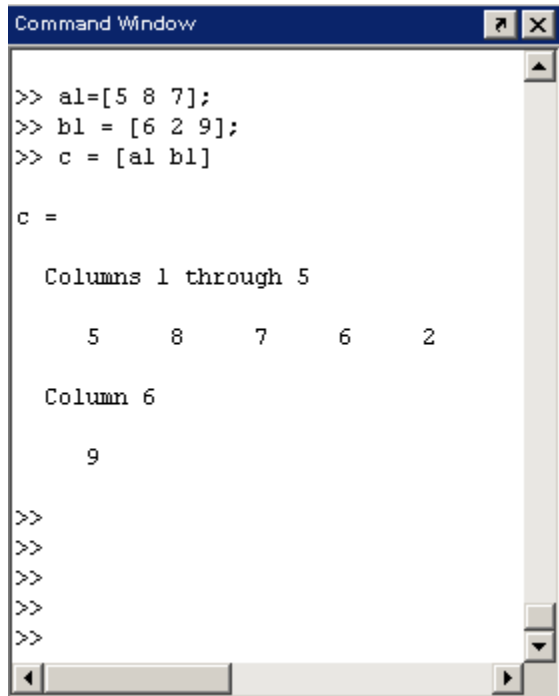


en este editor podemos manipular el tipo de datos que conforman el arreglo, los datos mismos, y hasta sus dimensiones, dando de esta manera un trabajo mucho más interactivo y cómodo para el ingeniero.

2.11.1.5 Concatenar arreglos

Si se quiere concatenar los arreglos basta con ponerlos uno a continuación de otro.

Por ejemplo:



```

Command Window
>> a1=[5 8 7];
>> b1 = [6 2 9];
>> c = [a1 b1]

c =

Columns 1 through 5
    5     8     7     6     2

Column 6
     9

>>
>>
>>
>>
>>

```

2.11.1.6 Matemáticas con arreglos.

a = 1 : 5 Define un vector de 5 elementos con incrementos de 1, pues cuando no se especifica el incremento, por omisión es 1.

a =

1 2 3 4 5

b = 1 : 2 : 10 Vector de 5 elementos con incremento de 2

b =

1 3 5 7 9

Se le puede sumar o multiplicar un número a todo el arreglo, por ejemplo

a + 8 Suma de un escalar con un arreglo

ans=
9 10 11 12 13

a * 8 Multiplicación de un escalar con un arreglo

ans =
8 16 24 32 40

Para hacer la suma de los arreglos a y b, solamente escribimos :

a + b La respuesta se guarda en ans :
ans =
2 5 8 11 14

Se pueden hacer operaciones como :

z = 100 - 2 * a + b
z =
99 99 99 99 99

Importancia del punto (antes del operador)

La multiplicación de arreglos se hace con (**. ***), ya que cuando se utiliza el asterisco sin punto indica multiplicación matricial, y además provoca un error.

z = a .* b

La división también lleva un punto antes del signo, caso contrario se utiliza el punto nos referimos a la división matricial que es muy diferente.

z = a ./ b

La siguiente operación obtiene el cuadrado del arreglo "a".

z = a .^ 2

2.11.2 Matrices

Muchos de los conceptos utilizados para crear vectores son aplicados a las matrices y sus operaciones, no obstante se enfocará todo eso a base de ejemplos.

Para hacer una matriz se utiliza el punto y coma (;)

Para formar la matriz

1	4	6
8	10	12
14	16	18

Escribimos :

```
>> a=[2 4 6;8 10 12;14 16 18]
```

```
a =
```

2	4	6
8	10	12
14	16	18

A más de definir las matrices como ya se especificó podemos obtener matrices especiales con los siguientes comandos.

```
ones(3)
```

Hace una matriz de unos, de 3 x 3.

```
zeros(3,4)
```

Hace una matriz de ceros, de 3 x 4.

```
rand(2)
```

Hace una matriz de 2 x 2, con datos al azar.

```
eye(4)
```

Hace una matriz identidad de 4 x 4.

Ejemplos:

```
>> ones(3)
```

```
ans =
```

1	1	1
1	1	1
1	1	1

```
>>
```

```
>> zeros(3,4)
```

```
ans =
```

```
0 0 0 0
0 0 0 0
0 0 0 0
```

```
>> rand(2)
```

```
ans =
```

```
0.9501 0.6068
0.2311 0.4860
```

```
>> eye(4)
```

```
ans =
```

```
1 0 0 0
0 1 0 0
0 0 1 0
0 0 0 1
```

Como acceder a los datos de una matriz

Si tenemos la matriz:

```
a = [ 1 5 3; 8 5 7; 7 6 9 ]
```

Si deseamos cambiar el número 5 de la segunda fila por 0 debemos hacer lo siguiente

```
a(2,2)= 0
```

Variable (renglón, columna)= nuevo valor

Si tenemos la matriz identidad de 3 x 3 :

```
eye(3)
```

```
ans =
```

```
1 0 0
0 1 0
0 0 1
a = [ 1 0 0 ; 0 1 0; 0 0 1 ]
```

pero por algún error la matriz identidad debió de haber sido de 4 x 4.

Lo que podemos hacer es lo siguiente

```
a(4,4) = 1
```

Agregamos un 1 en el renglón 4 columna 4, y como este no existían previamente, las columnas y renglones se completan agregando ceros.

Si deseamos una matriz identidad de 3 x 3 y tenemos capturada una de 4 x 4.

Podemos utilizar :

Matriz ("Renglón" **inicio : Fin** , "Columna" **inicio : Fin**)

```
B = a ( 1 : 3, 1: 3)
```

Ahora si queremos que la matriz identidad sea :

```
0 0 1
0 1 0
1 0 0
```

```
C = B ( 3 : -1 : 1 , 1 : 3 )
```

Poner dos puntos (:) indica que se deben tomar todas las columnas (1 : 5). Esto es valido también para los renglones.

```
C = A ( : , [ 1 3 5 ] )
```

Toma **todos los renglones**, pero **sólo** toma las **columnas 1, 3 y 5**.

2.11.2.1 Operaciones con matrices

Concatenación de matrices

Si creamos las siguientes matrices A y B :

```
>> A=[5 6 8; 2 3 1]
```

```
A =
```

```
5    6    8
2    3    1
```

```
>> B=[4 2 1; 6 5 4]
```

```
B =
```

```
4    2    1
6    5    4
```

```
>> C=[A B]
```

C =

Columns 1 through 5

5	6	8	4	2
2	3	1	6	5

Column 6

1
4

A partir de la matriz A queremos tomar las columnas 1, 2 y de la matriz B queremos tomar las columnas 1 y 3, para formar una matriz D.

`D = [A(:, [1 2]) B(:, [1 3])]`

`D = [A(:, [1 2]) B(:, [1 3])]`

D =

5	6	4	1
2	3	6	4

`D(:,1)=[]`

Elimina la columna número uno.

`D(:,1)=[]`

D =

6	4	1
3	6	4

Ejemplo de Resolución de Ecuaciones:

Aunque este ejemplo es tomado de un ejemplo de un libro, nos pareció muy oportuno poner este ejemplo en esta parte puesto que muestra de una manera muy práctica el uso de las matrices para la resolución de problemas matemáticos con incógnitas.

En esta parte lo que se tiene que conocer es en realidad la forma de cargar matrices y como operar con ellas..

Por ejemplo para resolver el siguiente sistema de ecuaciones.²⁷

$$2x + 0y + 5z = 100$$

$$3x + 5y + 9z = 251$$

$$1x + 5y + 7z = 301$$

1. Capturamos los valores de x, y, z ; formando una matriz.

$$A = [\begin{matrix} 2 & 0 & 5 \\ 3 & 5 & 9 \\ 1 & 5 & 7 \end{matrix}]$$

Después capturamos el valor al cual están igualadas las ecuaciones en otra matriz.

$$b = [\begin{matrix} 100 \\ 251 \\ 301 \end{matrix}]$$

Una forma de solucionar las ecuaciones es obteniendo el inverso de la matriz, es decir : A^{-1} (menos uno)

El asterisco indica multiplicación matricial.

$$c = \text{inv} (A) * b$$

Otra forma de resolverlo, es utilizando la división matricial.

$$c = A \setminus b$$

Es también posible obtener la determinante de una matriz.

$$\det (A)$$

Definamos las siguientes matrices 'g' y 'h'.

$$g = [\begin{matrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{matrix}]$$

g =

$$\begin{matrix} 1 & 2 & 3 \end{matrix}$$

$$\begin{matrix} 4 & 5 & 6 \end{matrix}$$

$$\begin{matrix} 7 & 8 & 9 \end{matrix}$$

$$h = [\begin{matrix} 1 & 0 & 2 \\ 11 & 2 & 3 \\ 3 & 5 & 12 \end{matrix}]$$

h =

```

1   0   2
11  2   3
3   5  12

```

La suma de las matrices g y h se muestra enseguida :

```
k = g + h
```

```
k =
```

```

2   2   5
15   7   9
10  13  21

```

```
k = g * h
```

Multiplicación de dos matrices.

2.12 COMANDOS BASICOS PARA GRAFICAR EN MATLAB

En MATLAB se pueden crear gráficas de una manera muy rápida y eficaz, Matlab posee una interfaz gráfica muy amigable, en nuestra monografía esta interfaz lo estamos tomando de un Toolbox (Simulink) y lo utilizamos para graficar en forma sólida las reglas de inferencia, de ahí que nos pareció importante mencionar la interpretación gráfica básica en esta parte.

Mostraremos a base de ejemplos sencillos la representación gráfica.

2.12.1 Comandos para gráficas en dos dimensiones

Plot

El comando que se utilizará es el comando plot().

```
A=[4 5 6 7 3 2 8]
```

```
A =
```

```
Columns 1 through 5
```

²⁷ ZADEH, L. Fuzzy sets, Information & Control., 8, 1965. Pag. 45

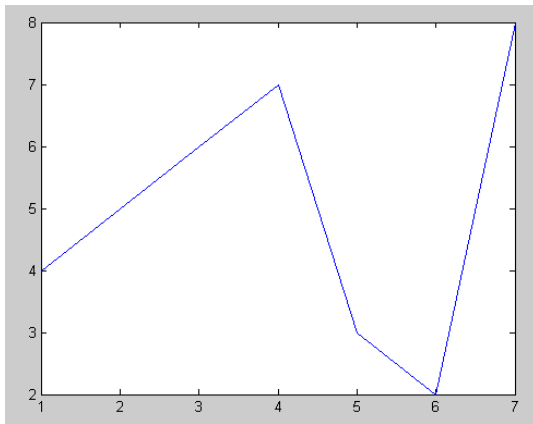
```
4 5 6 7 3
```

```
Columns 6 through 7
```

```
2 8
```

```
>> plot(A)
```

nos dará como resultado una gráfica sencilla pero completamente legible, donde los valores se grafican en el eje Y, a partir del eje X.



Como se vio en el ejemplo que es posible graficar una serie de puntos y MATLAB automáticamente ajusta los ejes donde se gráfica.

Por ejemplo, para graficar la función seno se pueden crear un rango de valores

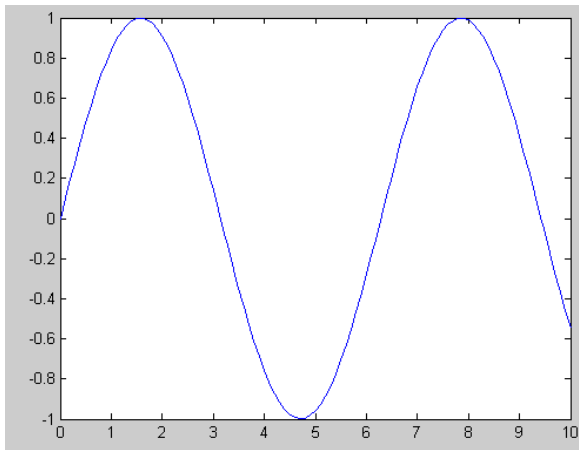
```
x = 0 : 0.1 : 10;
```

x = vector de cero a 10 con incrementos de 0.1

Seno del vector (x)

```
plot (x,y)
```

Gráfica del seno

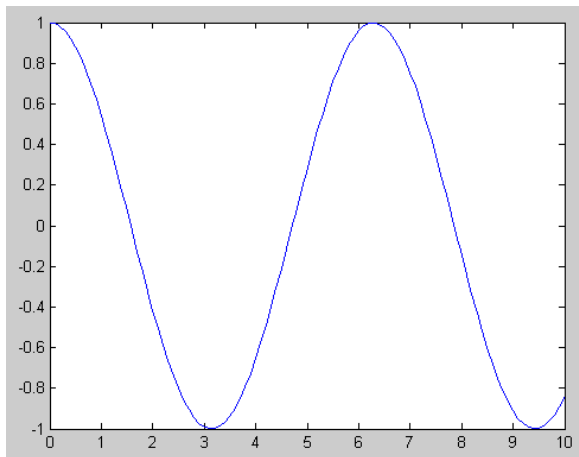


```
z = cos(x);
```

Coseno del vector anterior

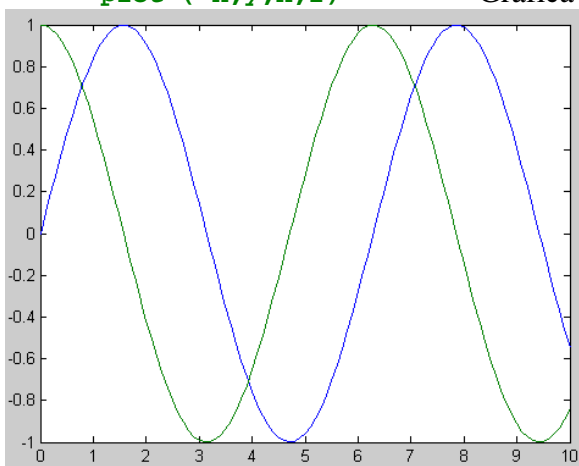
```
plot (x,z)
```

Gráfica del coseno de x.



```
plot ( x,y,x,z)
```

Gráfica del seno y coseno en la misma pantalla



```
plot (x,z,'*')
```

Gráfica del coseno con los signos '*'

Hace la gráfica en azul, y los signos '+', intercambiando los ejes.

```
plot ( z, x, 'b+')
```

Como se ve es posible graficar en Matlab con símbolos y además escoger el color, tal como se muestra en la tabla inferior.

Símbolo	Color	Símbolo	Estilo de línea
y	Amarillo	.	punto
m	Magenta	o	circulo
c	Cían	x	equis
r	Rojo	+	más
g	Verde	*	asterisco
b	Azul	-	menos
w	Blanco	:	dos puntos
k	Negro	- .	menos punto
		- -	menos menos

Es posible agregar un cuadriculado a la gráfica, para tener más precisión, con el comando. **grid**

Se pueden agregar títulos a las gráficas y etiquetas en los ejes con los comandos siguientes.

```
title(' Gráfica del coseno de x')
```

Para ponerle etiquetas a los ejes se puede utilizar los comandos

```
ylabel ('etiqueta')
```

```
xlabel('etiqueta')
```

axis off

Desaparece los ejes.

Subplot

El comando subplot nos permite desplegar en pantalla varias gráficas.

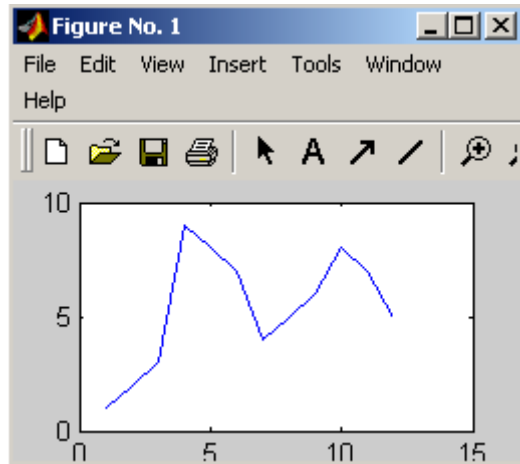
```
subplot(m,n,a)
```

'm' y 'n' son una matriz que representa las cantidades de gráficas que se van desplegar; 'a' indicaría el lugar que ocuparía la gráfica en el subplot.

Hagamos la gráfica de los siguientes puntos. La desplegaremos en cuatro puntos diferentes en pantalla para ver las características de subplot.

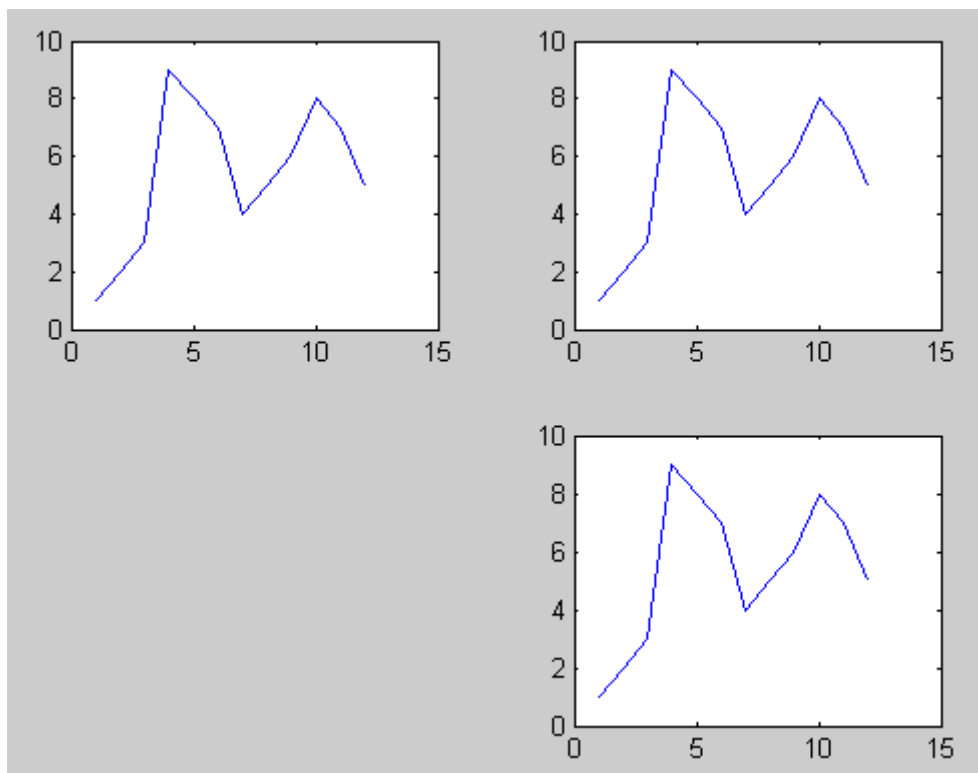
```
a=[ 1 ,2 ,3 9 ,8 ,7 ,4, 5, 6, 8, 7, 5];
```

```
plot (a)
```



Vamos hacer una matriz de 2 x 2 para graficar, cuatro posibles ventanas o gráficas. Y queremos que la primera gráfica ocupe el lugar (1,1) de la matriz. entonces escribimos.

```
subplot(2,2,1)    ,plot(a)
subplot(2,2,2)    , plot(a)
subplot(2,2,4), plot(a)
```



como nos podemos dar cuenta el dibujo de la posición 3 no se presenta pues no se pidió ser graficado.

CLF borra todos los objetos de la gráfica.

CLF RESET Borra todo lo que hay en la gráfica y resetea todas las propiedades de la figura.

```
clf
```

Cabe indicar que el comando plot se puede extender a 3 dimensiones con el comando plot3 .

zlabel ('etiqueta')

Se utiliza para dar etiquetas al eje z, en las gráficas en tres dimensiones.

Es posible cambiar el sentido de orientación de las gráficas con el comando view(x,y)

```
view(0,0)
```

```
view(90,0)
```

gtext(' texto ')

Se utiliza para colocar texto en una gráfica, con la ayuda del mouse. Simplemente se ejecuta el comando y con el mouse se selecciona la coordenada deseada y se presiona el botón derecho del mouse, quedando fijo el texto en la pantalla.

```
cplxroot(3,10)
```

Superficie de una raíz cubica.

2.13 COMANDOS DE INFORMACION

Son aquellos comandos que como la ayuda típica siempre nos están informando a cerca de lo que solicitamos.

Comando Help, visualiza la ayuda de un comando específico, para ello se utiliza el formato help <Comando>, por ejemplo

```
>> help or
```

Logical OR.

$A \mid B$ is a matrix whose elements are 1's where either A or B has a non-zero element, and 0's where both have zero elements. A and B must have the same dimensions unless one is a scalar.

$C = \text{OR}(A,B)$ is called for the syntax ' $A \mid B$ ' when A or B is an object.

See also XOR.

- **What :** Muestra de todos los archivos *.m en el directorio actual
- **dir :** Muestra todos los archivos en el directorio actual
- **type nombre_archivo :** Lista el programa, (Programas con terminación *.M).
- **Which nombre_archivo :** Muestra el path en el cual esta el archivo.

2.14 CREACIÓN DE ARCHIVOS DE PRAGRAMAS EN MATLAB

Para crea programas en Matlab lo podemos realizar de la siguiente manera:

1. Cargue el Editor

- Vaya a la ventana options, escogiendo editor preference, y cargando el editor que más sea comodo para utilizar.

2. Abra un archivo M.file

- Vaya a la opción File y escoga la opción New M.file, de esta manera creará la ventana para poder empezar a escribir el código.

3. Grabe el archivo utilizando la terminación .M.

- Escriba el código y guarde el archivo utilizando la terminación archivo.M.

2.14.1 M.files

Los M-files son ficheros de texto con una serie de comandos MATLAB.

Estos archivos evitan teclear los comandos uno a uno, sobre todo cuando las tareas son repetitivas. Su extensión debe ser .m. Desde MATLAB.

Se invocan tecleando su nombre.

En estos archivos se pueden incluir comentarios que documenten cada línea del programa o lo que el programador considere mas importante, para esto se usa %.

Utilizamos como ayuda (help nombre_fichero).

El orden de búsqueda es: variables, funciones MATLAB, directorios.

2.14.1.1 Tipos de M-files: existen dos tipos

Scripts: realizan una tarea siempre igual

Funciones: realizan una tarea con datos variables que constituyen los parámetros.

Sintaxis de una función:

Function[datos_salida] = nombre_funcion(parámetros)

Diferencias entre scripts y funciones:

- Los scripts pueden ser llamados solo desde prompt o desde otro script, mientras que las funciones son llamadas desde cualquier parte.
- Las variables que generan los scripts son globales, y las que generan las funciones son locales a la misma.

2.15 Estructuras de comandos

Como ya lo sabemos la estructura de comandos dentro de cualquier lenguaje de programación resultan ser muy importantes y necesarias, aquí en Matlab también podemos utilizarlas, esta son:

For
While
If.

2.15.1 For

El formato de la Estructura de comando es la siguiente.

For x = Número inicial : número final

Instrucción(es)

End.

Ejemplo:

```
for x = 1 : 10
```

```
x = x + 1
```

```
end
```

También se pueden hacer operaciones como la siguiente :

```
matriz = [ 1 2 3 4; 1 2 3 4; 1 2 3 4; 1 2 3 4]
```

```
for x = matriz
```

```
    x = n(1)*n(2)*n(3)*n(4)
```

```
end
```

2.15.2 While

While permite que ciertas instrucciones sean repetidas un número indefinido de veces mientras se cumpla una determinada condición previamente establecida por el operador.

El formato es el siguiente:

```
while condición, instrucción(es),end
```

```
n
```

Por ejemplo

```
n = 1;
```

```
while n=1,a=a+1;end
```

```
n
```

2.15.3 IF ELSE END

Esta estructura la utilizamos en nuestra monografía en gran porcentaje dado que mediante ella hemos podido establecer todas las reglas de inferencia, para de esta manera poder realizar las correspondientes comprobaciones de temperatura.

El formato de la estructura condicional, es la siguiente

```
If expresión (verdadero)
    acción(es)
End.
```

```
If expresión (verdadero)
    Acción(es) 1
else                                (Falso)
    Acción(es) 2
End.
```

```
If expresión            (verdadero)
    acción(es) 1
elseif expresión      (verdadero)
    acción(es) 2
...
else                                (Falso)
    acción "n"
End
```

2.15.4 SWITCH

Esta estructura la utilizamos para analizar estructura de casos, es decir podemos analizar alternativas que puede tener una variable o una operación, pueden ser más de dos condiciones.

El formato de la estructura es la siguiente:

```
switch expresión
    case condición1
        instrucción(es)
    case condición2
        instrucción(es)
```

```

        case condición3
            instrucción(es)
        .....
    otherwise
        instrucción(es)
    end

```

2.16 SIMULINK

El simulink es una herramienta que consideramos una extensión gráfica de MATLAB para el modelado y la simulación de sistemas.

El simulink está basado en bloques y líneas.

Las ventanas que tenemos disponibles en el SIMULINK son:

- SIMULINK Library Browser
- Venta de modelo (nuevo o desde fichero .mdl)

2.16.1 Elementos básicos

2.16.1.1 Los Bloques:

Generan, modifican, combinan y muestran señales, estos son modificables, es decir tienen parámetros que pueden ser cambiados.

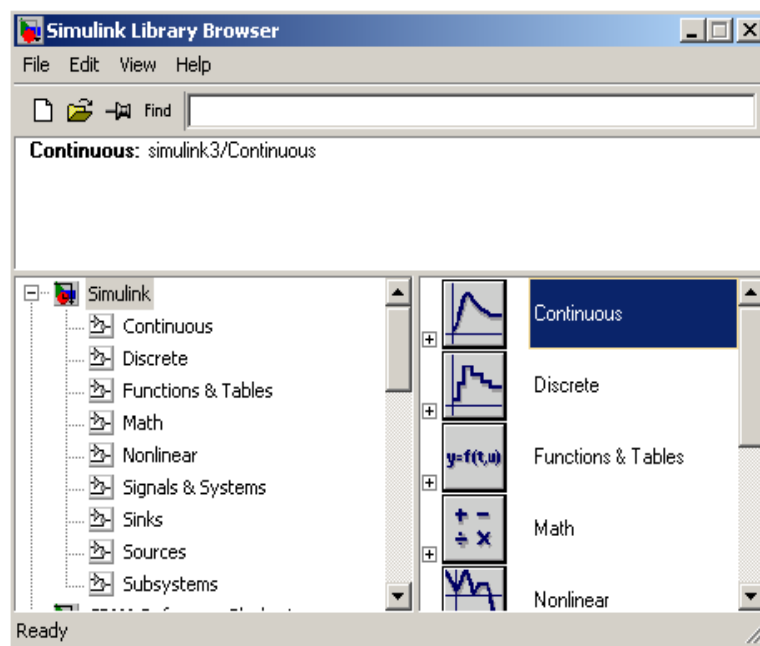
La mayor parte de los bloques que utilizaremos se encuentran en la librería Simulink:

- Continuous
- Discrete
- Functions and tables
- Non linear
- Signals and systems
- Sinks
- Sources

2.16.1.2 Líneas:

- Transfieren señales de un bloque a otro
- Van de la salida de un bloque a la entrada de otro
- Pueden tener bifurcaciones
- No pueden tener uniones

Se puede observar claramente la biblioteca de elementos que se cargan cuando se invoca al Simulink desde la ventana de comandos de Matlab a continuación.



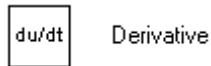
En nuestra monografía utilizamos el simulink para generar un diagrama de bloques en base al modelo matemático diseñado, todo esto se explicará con mayor detalle en la documentación del sistema desarrollado.

Los elementos de Simulink utilizados en nuestro sistema son los siguientes:

Elementos utilizados dentro de cada combo de Simulink en el Sistema que compete a nuestra monografía

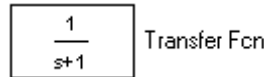
CONTINUOUS.

Dentro de esta herramienta escogemos el combo **Derivativo**, esto nos sirvió para modificar la señal de alimentación.



Derivative

Transfer sn Trasferencia, Especifica el tipo de señal de transferencia en base al modelo matemático.



Transfer Fcn

Transfer Delay, especifica el Retardo, cabe indicar que toda función de transferencia debe tener un Delay o retardo que permite la alimentación de señales.



Transport Delay

MATH

Sumador: Se necesita el sumador para poder hacer el cambio del signo y la correcta entrada de los datos al multiplexor.



Sum

NONLINEAR

Switch, realiza el cambio automático necesario después de realizar la fusificación de los datos con el controlador PID.



Switch

SIGNALS & SYSTEMS

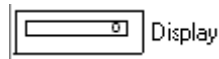
Mux: Es el multiplexor que permite que ingresen varias señales y devuelve una sola.



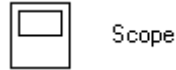
Mux

SINKS

Display: Para visualizar la salida

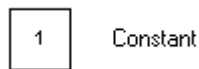


Scope: Funciona como un osciloscopio y nos permite visualizar las ondas de entrada y de salida.

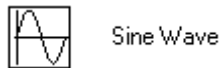


SOURCES.

Constant. Se utiliza para pasar una constante, en nuestro caso para pasar dato de temperatura constante.

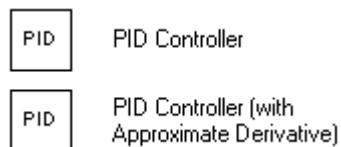


Sine Wave: Forma de la onda Sinusoidal, se explica en el documento de describe el sistema el porqué la forma de esa onda.



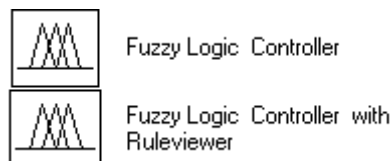
SIMULINK EXTRAS

Additional Linear, aquí se escoge el tipo de controlador de la planta, en nuestro caso escogimos el controlador PID.



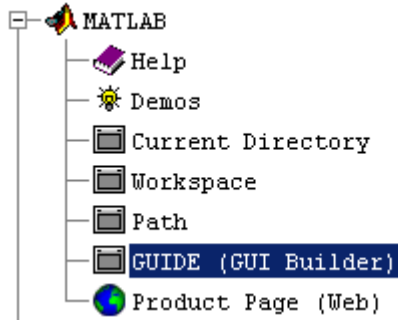
FUZZY LOGIC TOOLBOX

Member Ship Function: Es aquí donde nosotros escogemos en el combo del controlador Difuso para poder analizar las señales en forma difusa.

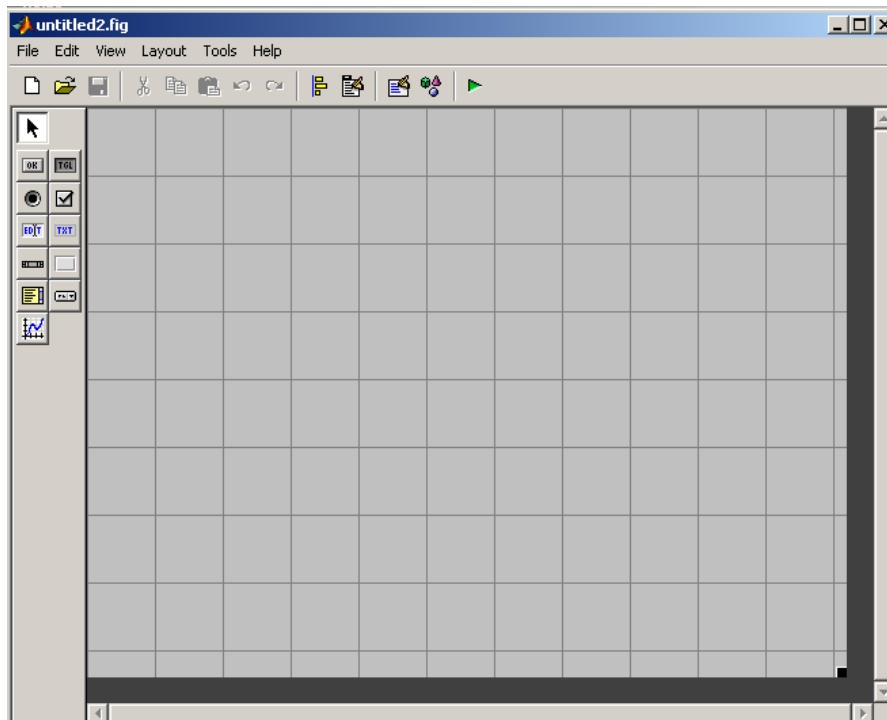


2.17 Herramienta utilizada para realizar interface

En Matlab también es posible trabajar con un ambiente de amigable e interactivo y generar el código fuente utilizando lo que en muchos de los casos denominamos en formularios, para ello es necesario Utilizar la Opción del Menú Math/GUI Builder que se encuentra dentro de sus herramientas como se ve a continuación:

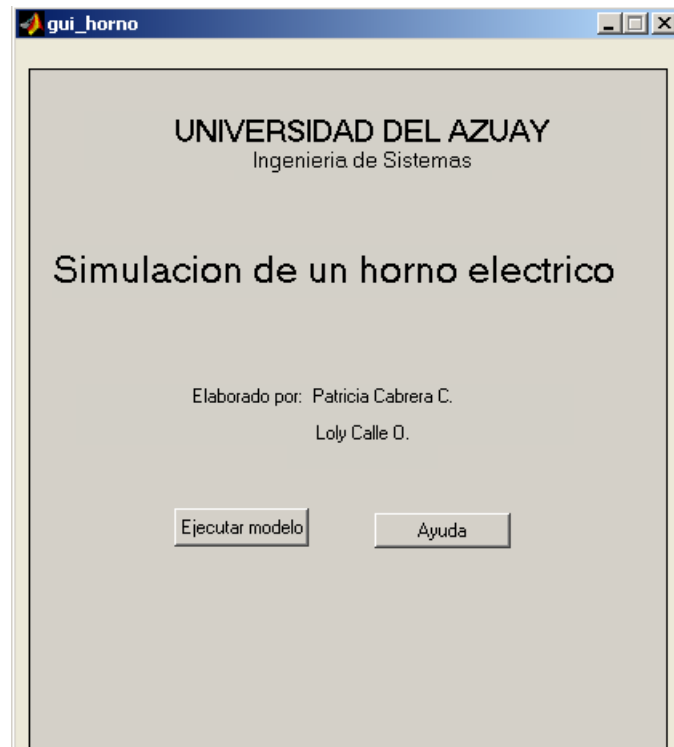


se visualiza una hoja de formulario lista a ser utilizada:



Escoja pues las herramientas necesarias para crear su formulario.

Por Ejm:



2.18 FUZZY LOGIC

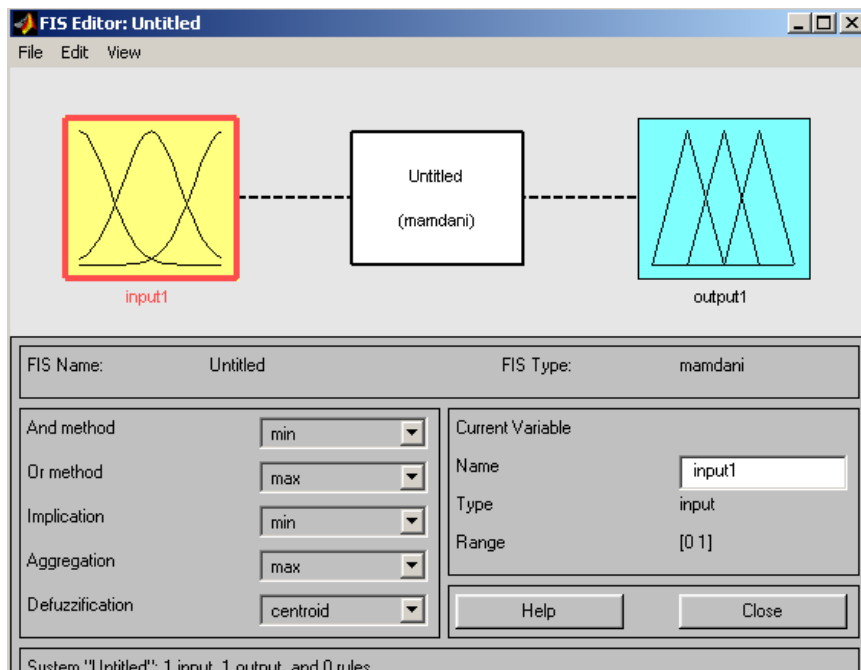
Dentro de Matlab como se dijo anteriormente tenemos una serie de ToolBoxes, pero el que nos sirve en nuestro sistema es el FUZZY LOGIC, pues nuestro sistema realizará la simulación de una planta (horno) con sus señales lingüísticas de entrada y las interpretará a base de este controlador, observaremos de forma rápida la forma de iniciar una aplicación utilizando el Fuzzy Logic en Matlab.

En la ventana de comandos digitamos la palabra

```
>> fuzzy
```

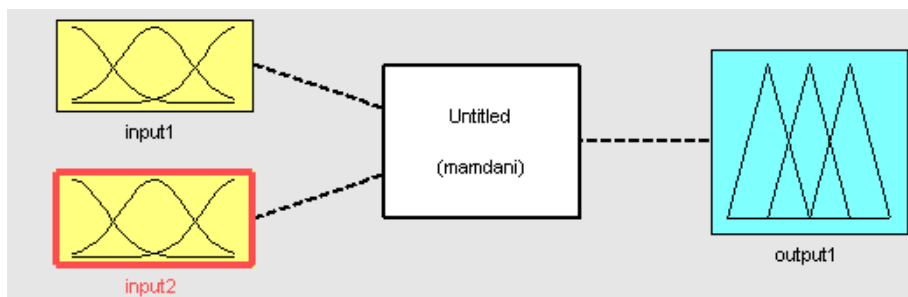
se carga el módulo de lógica difusa con la ventana por omisión. Cabe indicar que por default se carga una señal de entrada y una de salida. Usted puede también ya generar el editor FIS con un nombre, por ejemplo:

```
>> fuzzy horno
```

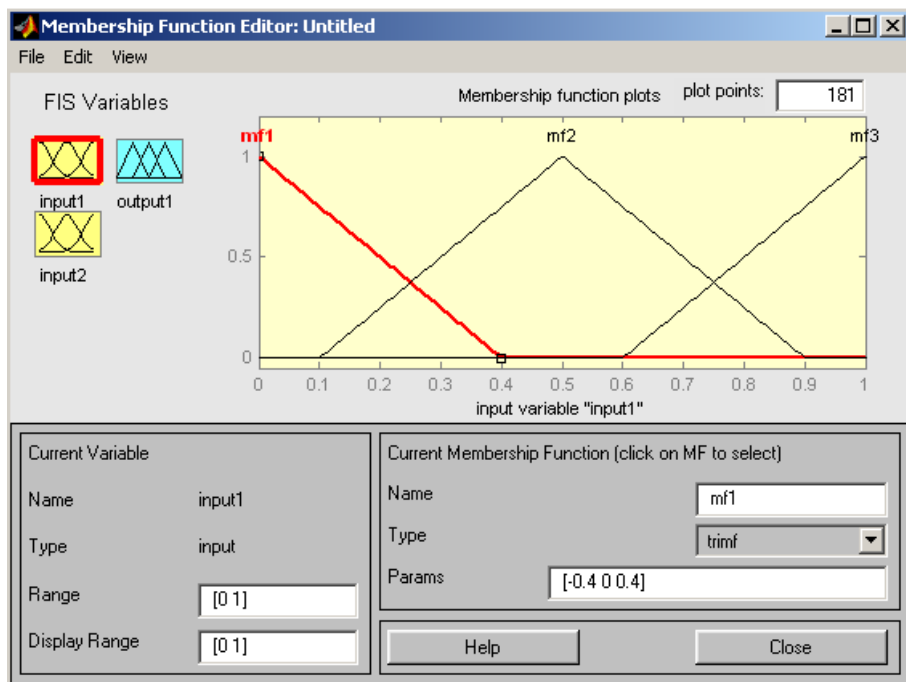


De un click en la opción File/New FIS, si desea crear otra aplicación a partir a más de la que se carga por default.

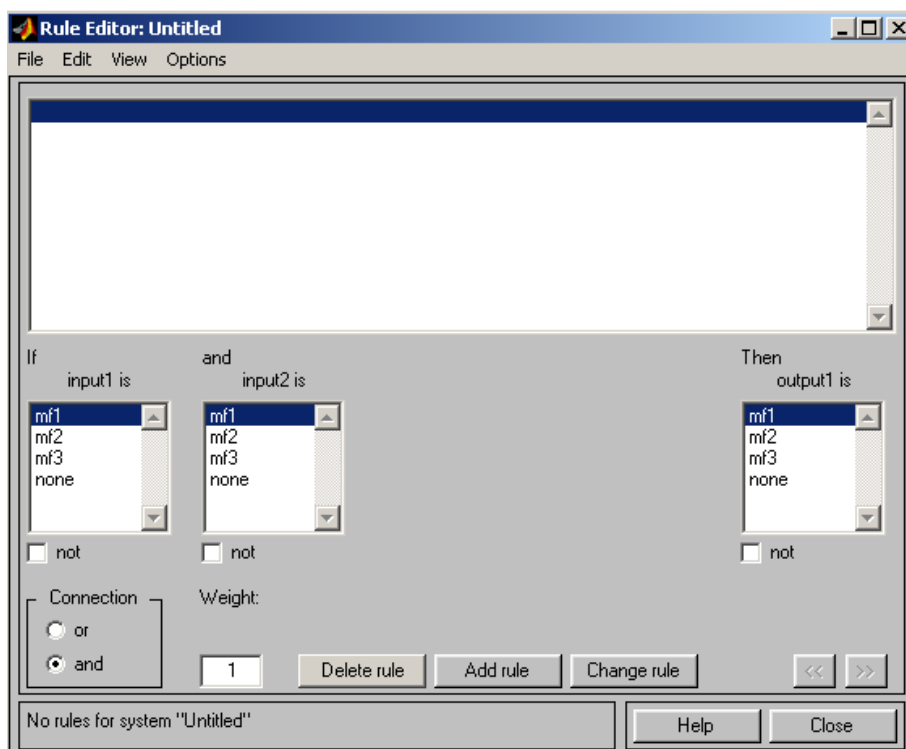
Si desea cargar otra variable vaya a la opción de Edit/Add Variable, obtendrá como resultado:



Al dar doble clic en cada variable por defecto podrá cambiar las especificaciones de las variables tanto como nombre, parámetros de rango, ect.



Para ingresar en el editor de reglas que constituye en el elemento central generado por fuzzy logic, podrá visualizar la siguiente pantalla:



Es aquí donde puede incluir las reglas de inferencia para que el sistema funcione de acuerdo a las necesidades.

A continuación en el siguiente capítulo se documentará a detalla la aplicación escogida que interpreta todos los conceptos antes mencionados en lo que se refiere a Lógica Difusa y Matlab.

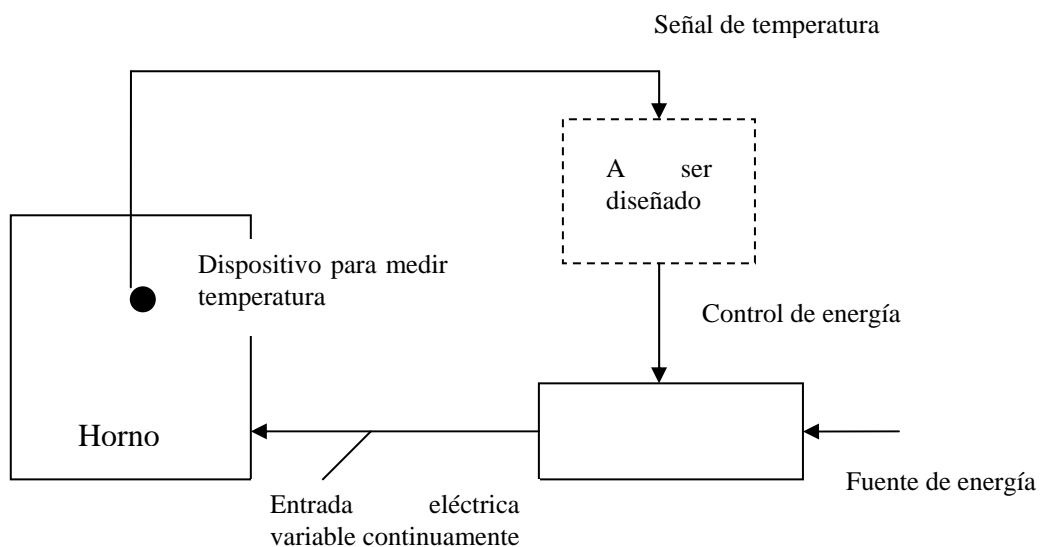
CAPITULO III

3 MODELO DIGITAL DE LOGICA DIFUSA, APLICADO A UN SISTEMA INTELIGENTE.

3.1 PLANTEAMIENTO DEL PROBLEMA

Se plantea el desarrollo de un sistema para el control de temperatura de un horno eléctrico provisto de un sistema de medición de temperatura que tiene una termocupla y con una entrada de energía continuamente variable, controlada remotamente.

El esquema del sistema en consideración es el siguiente:



Se trata de plantear las ecuaciones necesarias que permitan resolver el problema, mediante el uso de técnicas de lógica difusa, a través de una herramienta como MatLab.

3.2 ESPECIFICACION DE REQUERIMIENTOS

Para el desarrollo del proceso es necesario considerar los siguientes requerimientos del problema:

- a. Muestreo de la señal de medición de temperatura en un intervalo apropiado
- b. Transferencia de la señal de medición dentro del computador, seguido por la conversión y validación.
- c. Comparación de las temperaturas medidas con un valor de temperatura deseada, con la finalidad de formar una señal de error.
- d. Operación, sobre la señal de error, por un algoritmo apropiado para formar una señal de salida.
- e. Ajuste del nivel de la señal de salida y transferencia a través del interfaz hacia la unidad de control de potencia. La potencia del horno se mantendrá constante sobre cada intervalo de muestreo.

3.3 DEFINICION DEL MODELO MATEMATICO

Con la finalidad de resolver el problema se empleará la Teoría de Control, específicamente el método de Ziegler – Nichols, así como la Transformada de Laplace para obtener el modelo matemático que permita plantear el problema desde el punto de vista matemático y resolverlo mediante el uso de Matlab a través de dos Toolkit: el de sistemas y señales y el de lógica difusa; para que pueda resolverse este paso es necesario consultar el libro de: *Digital Control Applied, de Leigh, Jr. Pg. 219-245*, donde se encuentran definidos los modelos matemáticos que se mostraran en esta aplicación puesto que al ser una investigación “de ingeniería de sistemas” no definiremos a detalle los coeficientes obtenidos tanto para la función de transferencia como para el controlador PID; no obstante daremos definiciones muy claras de cada uno de ellos. Refiérase al numeral 3.5 de este capítulo (Controlador PID).

3.4 GENERACION DE RESULTADOS

Mediante el MatLab se generarán resultados que permitan mostrar la forma de actuar del sistema de control diseñado.

Estos resultados podrán ser obtenidos a partir de la señal de transferencia obtenida para el horno eléctrico en consideración.

3.5 CONTROLADOR PID

Los controladores PID (proporcionales, integrales y derivativos) son los más utilizados y comunes dentro de la industria, se utilizan para controlar un sinnúmero de procesos, tales como plantas químicas, control de temperatura, hornos de calisa, etc... y algunas aplicaciones automotrices. De ahí que en nuestra monografía se ve necesario la utilización de estos controladores por el comportamiento que tiene la función de transferencia.

El objetivo de un controlador es suprimir los efectos de interrupciones en variables del proceso y también debe forzar a una variable específica a seguir un punto de referencia deseado.

3.5.1 Como trabaja el controlador PID

El controlador PID mide la salida del proceso y calcula la diferencia (error) entre lo que se ha medido y el punto de referencia. Si existe un error, el controlador ajusta su salida para alterar el proceso con el fin de acercarlo al punto de referencia, disminuyendo así el error. Cada vez que se calcule un error, el controlador debe decidir cuánto alterará el proceso. Si el controlador es demasiado agresivo (bajo amortiguación), puede provocar que el proceso se vuelva inestable y oscile. Si por otro

lado no es lo suficientemente agresivo, el sistema puede necesitar demasiado tiempo para recuperarse.²⁸

La agresividad del controlador es determinada por las constantes(coeficientes) PID que proporciona el controlador. Las constantes o coeficientes Proporcionales, Integrales y Derivativas se utilizan para calcular cuál debería ser la salida en relación al error medido.

La **constante proporcional** representa el área en la cual el controlador realmente está controlando el proceso, y determina la banda de operación.

La **parte integral** corrige cualquier desfase entre el punto de consigna y la variable del proceso reiniciando o desplegando de manera automática la banda proporcional.

La **constante derivativa** determina la velocidad a la que el controlador reacciona ante los cambios en la variable del proceso.

3.5.2 Calcular las constantes del PID

Existen muchos métodos, y cada uno de ellos proporcionan valores para cada constante, pero lo que realmente se debe estudiar con detenimiento es el comportamiento matemático que tienen los valores para obtener el mejor resultado

Los efectos de cada uno de los controladores P, I, y D en un sistema a lazo cerrado, es decir un sistema con retroalimentación continua, como es el caso de nuestro sistema se resumen en la siguiente tabla



	CAMBIO	RESULTADO1	RESULTADO2	ERROR
P	Baja	Sube	Poco Cambio	Baja
I	Baja	Sube	Sube	Elimina
D	Poco Cambio	Baja	Baja	Poco Cambio

Este es solamente un cuadro ilustrativo de lo que puede ocurrir con un CONTROLADOR PID, estas correlaciones podrían no ser exactamente seguras, porque

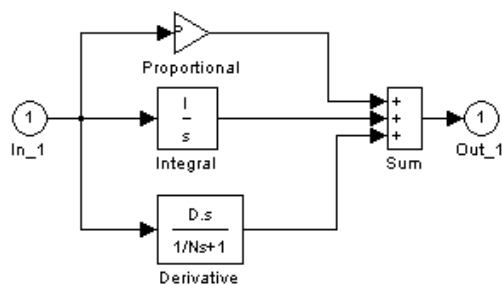
²⁸ Tomado de la dirección electrónica <http://fiobera.unam.edu.ar/Materias/ControlDigital/PID.html>

P, I, y D son dependientes entre sí. De hecho, cambiando una de estas variables se puede variar el efecto de las otras dos. Por esta razón, la tabla deberá usarse únicamente como referencia cuando se determina los valores de P, I y D.

Cualquiera que sea el caso dentro de MATLAB tenemos incorporado este tipo de controladores que nos evitan realizar toda la tarea de la Transformada de Laplace pues se encuentra incorporado y resuelta tal y cual como se llamara a una función. Lo encontramos dentro del SIMULINK.

	PID Controller
	PID Controller (with Approximate Derivative)

Internamente el PID trae incorporado el módulo:



Todos estos conceptos se han aplicado en nuestro proyecto para poder despejar los coeficientes del controlador PID en base a la función de transferencia.

Entonces tenemos el siguiente análisis:

Debido a que el enfriamiento de la temperatura del horno depende de cuanto baje usted el cambio y el calentamiento del horno depende de cuanto suba usted la temperatura, la curva de respuesta para el enfriamiento y el calentamiento difiere significativamente. No obstante la ausencia de procesos lineales nos hace pensar que los pasos de las magnitudes diferentes aplicadas a diferentes inicios de temperatura produce así una diferencia en la curva de respuesta. La tabla a continuación sintetiza los resultados razonables en base a varios experimentos:

Entrada Inicial Voltaje v_i	Entrada Final Voltaje v_i	Salida Inicial Voltaje V_o	Salida Final Voltage v_o	Proceso Ganancia (k)	Contante De tiempo T_2 (seconds)	Tiempo de simulación T_1 (seconds)	Curva de Respuesta
0	0.28	0	0.84	3	4800	300	Curva 1
0.15	0.89	0.73	1.93	1.63	3480	270	Curva 2
0.15	0	0.73	0	4.9	6600	180	Curva 3

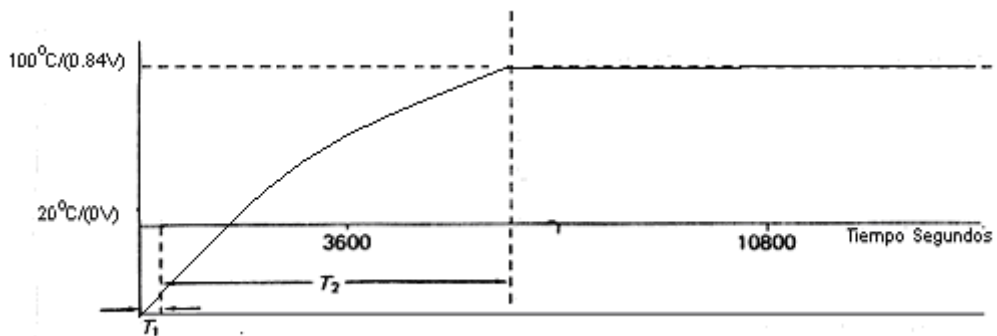


Figura Curva1

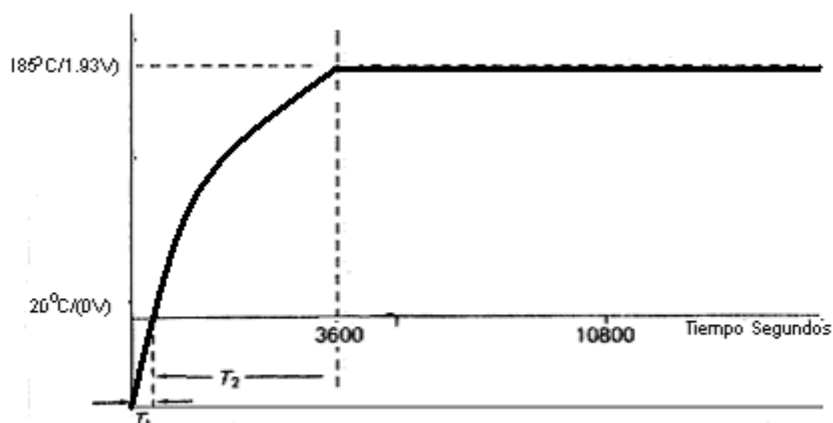


Figura Curva 2

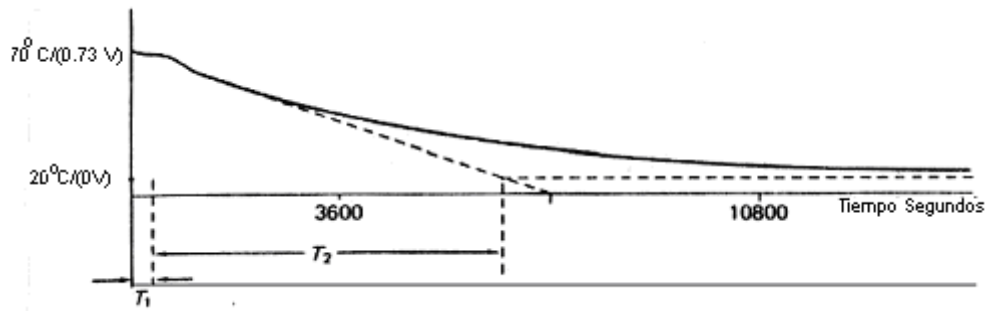


Figura Curva 3

Escogeremos la curva de la figura de la curva 2 como una respuesta razonable a las curvas obtenidas durante el experimento.

Para obtener mayor información del uso y comportamiento del controlador PID puede consultar el libro Ingeniería de Control Moderna, Katsuhiko Ogata, Tercera edición, Pg 130-155.

Al obtener los tres coeficientes del PID., en nuestro sistema los coeficientes son:

Proporcional : 9.5

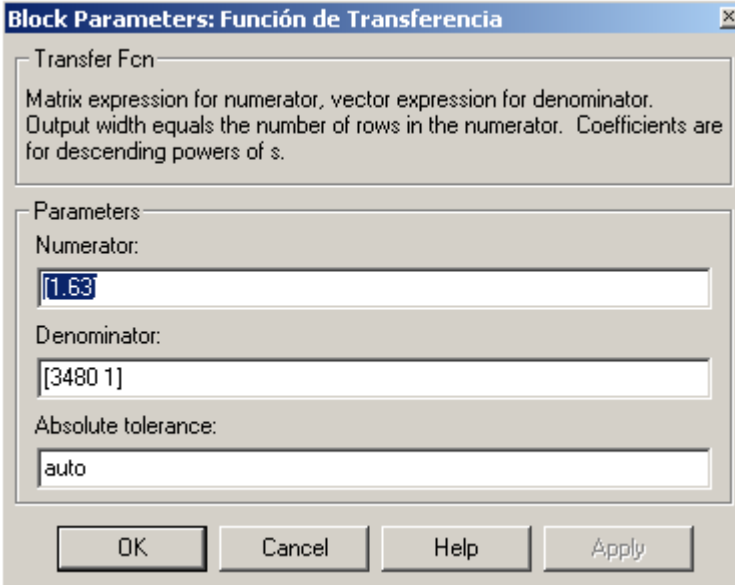
Integral: 0.015575

Derivativo: 12825

Estos valores son ingresados dentro de los módulos del diagrama de bloques de acuerdo a la siguiente correspondencia.

Función de transferencia

$$\frac{1.63}{3480s+1}$$



Block Parameters: Función de Transferencia

Transfer Fcn

Matrix expression for numerator, vector expression for denominator.
Output width equals the number of rows in the numerator. Coefficients are for descending powers of s.

Parameters

Numerator:

1.63

Denominator:

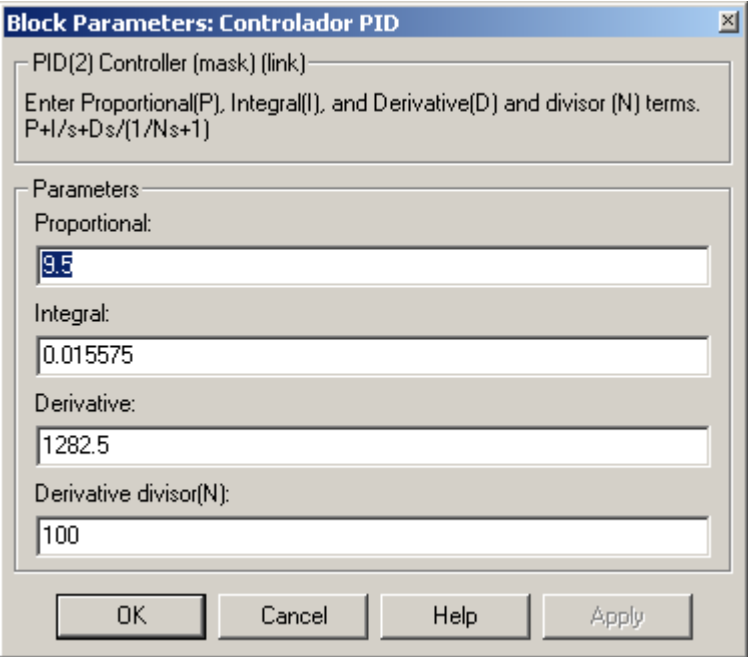
[3480 1]

Absolute tolerance:

auto

OK Cancel Help Apply

Los parámetros del controlador PID lo incorpora de esta forma



Block Parameters: Controlador PID

PID(2) Controller (mask) (link)

Enter Proportional(P), Integral(I), and Derivative(D) and divisor (N) terms.
 $P+I/s+Ds/(1/Ns+1)$

Parameters

Proportional:

9.5

Integral:

0.015575

Derivative:

1282.5

Derivative divisor(N):

100

OK Cancel Help Apply

De ahí la importancia de estudiar el modelo matemáticamente en forma precisa para poder ingresar estos valores y obtener resultados óptimos.

3.6 DISEÑO Y DESARROLLO DE LA APLICACIÓN PRACTICA

3.6.1 Problema a resolver

- En la figura 3.1 se muestra un horno eléctrico provisto con medición de temperatura por medio de termocupla que tiene una entrada de energía continuamente variable y controlada remotamente. Se trata de implementar un módulo, en la posición mostrada con un rectángulo punteado, para proveer un control de temperatura del horno por medio de lógica difusa.

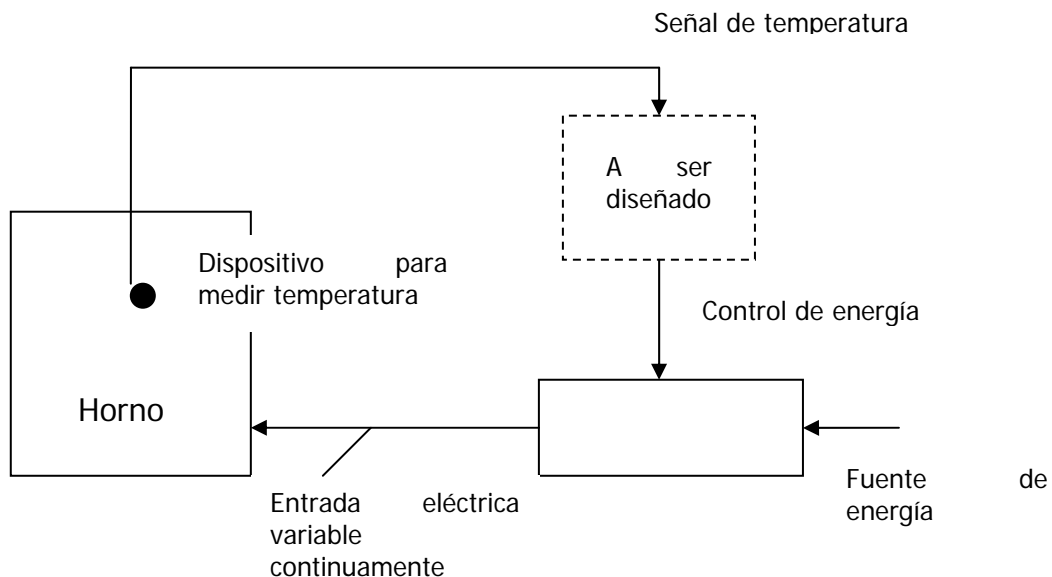


Figura 3.1 Esquema del problema a resolver

Las funciones dentro del lazo de control se puede dividir en:

Muestreo de la señal de medición de temperatura a una razón apropiada.

Transferencia de la señal de medición al computador, seguido por la conversión y validación.

Comparación de la temperatura medida con una temperatura deseada almacenada para formar una señal de error.

Operación sobre la señal de error por un algoritmo apropiado para formar una señal de salida.

Ajuste del nivel de la señal de salida y transferencia a través del interfaz para la unidad de control de energía. La energía para el horno se mantendrá constante sobre cada intervalo de muestreo.

La función de transferencia del horno es:

$$G(s) = \frac{1.63 e^{-270s}}{1 + 3480s}$$

En donde:

G(s) : función de transferencia

e: exponencial para despejar la variable s utilizando Laplace.

s: variable a ser despejada utilizando Laplace

1.63 : coeficiente de retardo en tiempo

3480 Constante de Tiempo

$$Y(s) = X(s) \cdot G(s) \Rightarrow G(s) = \frac{Y(s)}{X(s)}$$

3.6.2 Términos a Emplear

Los términos a emplear en el presente trabajo, son:

Señal de salida: es la variable que se desea controlar (posición, velocidad, presión, temperatura, etc.). También se denomina variable controlada.

Señal de referencia: es el valor que se desea que alcance la señal de salida.

Error: es la diferencia entre la señal de referencia y la señal de salida real.

Señal de control: es la señal que produce el controlador para modificar la variable controlada de tal forma que se disminuya, o elimine, el error.

Señal análoga: es una señal continua en el tiempo.

Señal digital: es una señal que solo toma valores de 1 y 0. El PC solo envía y/o recibe señales digitales.

Conversor análogo/digital: es un dispositivo que convierte una señal analógica en una señal digital (1 y 0).

Conversor digital/análogo: es un dispositivo que convierte una señal digital en una señal analógica (corriente o voltaje).

Planta: es el elemento físico que se desea controlar. Planta puede ser: un motor, un horno, un sistema de disparo, un sistema de navegación, un tanque de combustible, etc.

Proceso: operación que conduce a un resultado determinado.

Sistema: consiste en un conjunto de elementos que actúan coordinadamente para realizar un objetivo determinado.

Perturbación: es una señal que tiende a afectar la salida del sistema, desviándola del valor deseado.

Sensor: es un dispositivo que convierte el valor de una magnitud física (presión, flujo, temperatura, etc.) en una señal eléctrica codificada ya sea en forma analógica o digital. También es llamado transductor. Los sensores, o transductores, analógicos envían, por lo regular, señales normalizadas de 0 a 5 voltios, 0 a 10 voltios o, 4 a 20 mA.

Sistema de control en lazo cerrado: es aquel en el cual continuamente se está monitoreando la señal de salida para compararla con la señal de referencia y calcular la señal de error, la cual a su vez es aplicada al controlador para generar la señal de control y tratar de llevar la señal de salida al valor deseado. También es llamado control realimentado.

Sistema de control en lazo abierto: en estos sistemas de control la señal de salida no es monitoreada para generar una señal de control.

Identificación: es la técnica de construir un modelo a partir de las variables medidas del proceso: entradas o variables de control, salidas o variables controladas y, posiblemente, perturbaciones. En principio y con el objetivo de modelizar se pueden proponer tres formas distintas de utilizar los métodos de identificación:

- Hacer distintas aproximaciones para estructurar el problema: seleccionar las señales de interés, observar la dependencia entre ellas, estudiar el grado de linealidad del proceso.
- Construir un modelo que describa el comportamiento entre las entradas y las salidas, prescindiendo del comportamiento físico. Hay distintas formas de abordar el problema, según se consideren modelos no paramétricos o modelos paramétricos.
- Utilizar los datos para determinar los parámetros no conocidos del modelo físico obtenido a base del estudio de propiedades y leyes físicas del proceso estudiado. En este caso se habla de modelos “hechos a medida” de los cuales se debe estimar solamente los valores de los parámetros no conocidos. Para ello se recurre a ensayos de comportamiento o pruebas físicas y/o a la utilización de técnicas de optimización.

Sistema Estable: Un sistema que, frente a una excitación acotada produce una respuesta acotada es un sistema estable. La excitación puede ser un cambio de referencia o una perturbación. El sistema no debe tender a infinito sino que mantenerse acotado.

Acotada: Variable que se mantiene dentro de límites (escalón unitario, senoide, diente de sierra, pero NO una Rampa). No-Acotado significa, simplemente, que es muy grande porque en ingeniería real no existe infinito (los tanques se rebalsan, las bombas tienen un caudal máximo, los hornos se funden, etc.)

Función de transferencia de un sistema: se define como la relación entre la salida y la entrada del sistema en el dominio de Laplace asumiendo condiciones iniciales nulas.

3.6.3 Modelado matemático de sistemas dinámicos

Un modelo matemático de un sistema dinámico se define como un conjunto de ecuaciones que representan la dinámica del sistema con precisión, o al menos, bastante bien. Un modelo matemático no es único para un sistema determinado. Un sistema puede representarse de muchas formas diferentes, por lo que puede tener muchos modelos matemáticos, dependiendo de cada perspectiva.

La dinámica de muchos sistemas, ya sean mecánicos, eléctricos, térmicos, económicos, biológicos, etc., se describe en términos de ecuaciones diferenciales. Dichas ecuaciones diferenciales se obtienen a partir de leyes físicas que gobiernan un sistema determinado.

Al obtener un modelo matemático se debe establecer un compromiso entre la simplicidad del mismo y la precisión de los resultados del análisis.

En general, cuando se resuelve un problema nuevo, es conveniente desarrollar primero un modelo simplificado para obtener una idea general de la solución. A continuación se desarrolla un modelo matemático más completo y se usa para un análisis con más pormenores.

Un sistema de control puede tener varios componentes. Para mostrar las funciones de cada componente en la ingeniería de control, por lo general se usa una representación denominada diagrama de bloques.

Un diagrama de bloques de un sistema es una representación gráfica de las funciones que lleva a cabo cada componente y el flujo de señales. Un diagrama muestra la relación entre los diversos componentes. A diferencia de una representación matemática puramente abstracta, **un diagrama de bloques tiene la ventaja de indicar de forma más realista el flujo de las señales del sistema real.**

En un diagrama de bloques todas las variables del sistema se enlazan unas con otras mediante bloques funcionales. Este bloque es un símbolo para representar la operación matemática que sobre la señal de entrada hace el bloque para producir la salida. Las funciones de transferencia de los componentes, por lo general, se introducen en los bloques correspondientes, que se conectan mediante flechas para indicar la dirección del flujo de señales.

Las ventajas de la representación mediante diagramas de bloque de un sistema radican en que es fácil formar el diagrama de bloques general de todo el sistema con sólo conectar los bloques de los componentes de acuerdo con el flujo de señales y en

que es posible evaluar la contribución de cada componente al desempeño general del sistema.

Para la formación del diagrama de bloques se emplea Simulink (el cual viene con MATLAB), y se parte del modelo matemático del proceso en el cual se va a identificar los elementos que conformarán dicho diagrama. Una vez adicionados los elementos se ajustan con los parámetros establecidos en el modelo matemático con la finalidad de lograr el comportamiento esperado del sistema.

3.6.4 Uso del control inteligente de procesos en la monografía

El uso de técnicas basadas en la inteligencia artificial, en el campo del control, ha derivado en lo que se conoce como el control inteligente, el cual ha resultado ser uno de los campos con mayor aplicación práctica, debido en parte a su gran capacidad de adaptación y los buenos resultados que se han obtenido.

Aunque el control inteligente se debe en gran parte a la inteligencia artificial, hay que mencionar que el control inteligente ha retroalimentado aportes muy importantes a la propia área de la inteligencia artificial, tanto teóricos como prácticos.

El control inteligente se ha utilizado con mayor énfasis en aquellos sistemas que por alguna situación particular resultan especialmente complejos para la aplicación de otras técnicas de control.

En el área del control inteligente, la **lógica difusa** se ha utilizado para resolver de forma exitosa una gran variedad de problemas de diversa complejidad.

La lógica difusa puede ser definida como un “**proceso de cálculo mediante palabras antes que con números**”.

Los controladores difusos son empleados para controlar productos de consumidor, tales como máquinas lavadoras, cámaras de video, dispositivos de cocina, etc. así como procesos industriales, tales como molinos para cemento, trenes

subterráneos y robots. El control difuso es un método de control basado en lógica difusa.

Un controlador difuso puede incluir reglas empíricas²⁹ y que son especialmente útiles en plantas manejadas por un operador, el cual a través de la experiencia ha obtenido dichas “reglas” que le permiten desarrollar sus tareas.

El conjunto de reglas se denomina REGLAS BASE. Las reglas están en el formato familiar IF THEN ELSE.

Existen al menos cuatro fuentes principales para encontrar las reglas de control:

- a. **Experiencia del experto y el conocimiento de la ingeniería de control.** Un ejemplo clásico es el manual del operador para un molino de cemento. El método más común para establecer la colección de reglas es preguntar al experto o al operador utilizando un cuestionario cuidadosamente organizado.
- b. **Basado en las acciones de control del operador.** Las reglas difusas IF THEN pueden ser deducidas de las observaciones de las acciones de control de un operador o por medio de la revisión de un libro de registro de las tareas. Las reglas expresan las relaciones de entrada – salida.
- c. **Basado en un modelo difuso del proceso.** Una regla lingüística base pueden ser vista como un modelo inverso del proceso controlado. De esta manera, las reglas de control difuso podrían ser obtenidas mediante la inversión de un modelo difuso del proceso. El método está restringido a sistemas de orden relativamente bajos, pero puede proveer una solución explícita asumiendo que se disponen de modelos difusos de sistemas de lazo abierto y cerrado.
- d. **Basado en aprendizaje.** Un controlador auto organizativo es un ejemplo de un controlador que encuentra las reglas por sí mismo. Las redes neuronales es otra posibilidad.

En nuestra monografía utilizamos el método **Basado en las acciones de control del operador**, es decir el uso de las reglas difusas IF THEN que son deducidas de las

²⁹ empírica: práctica, experimental

observaciones de las acciones de control del horno, de ahí que la máquina de inferencia realiza el mejor control y evaluación de estas reglas ajustando el modelo matemático y luego se produce la digitalización del mismo hasta conseguir los resultados esperados, por ejemplo como ya veremos más adelante una de las reglas son:

- *Si el NIVEL DE TENSION es ACEPTABLE y la RAZON DE CAMBIO DE TEMPERATURA es LENTA entonces la TEMPERATURA es ACEPTABLE*

Cabe indicar que aunque estas reglas están definidas en forma un tanto subjetiva es aquí donde juega un papel muy importante la experiencia del Ingeniero respecto a la planta a ser controlada, en nuestro caso el horno.

En general las etapas más importantes utilizadas por un control difuso son las mostradas en la figura 3.2.

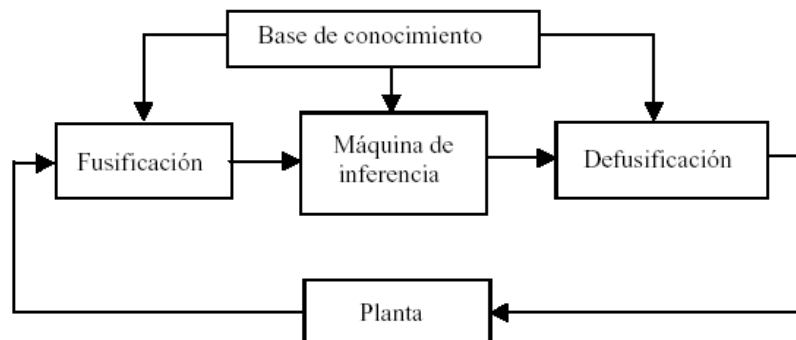


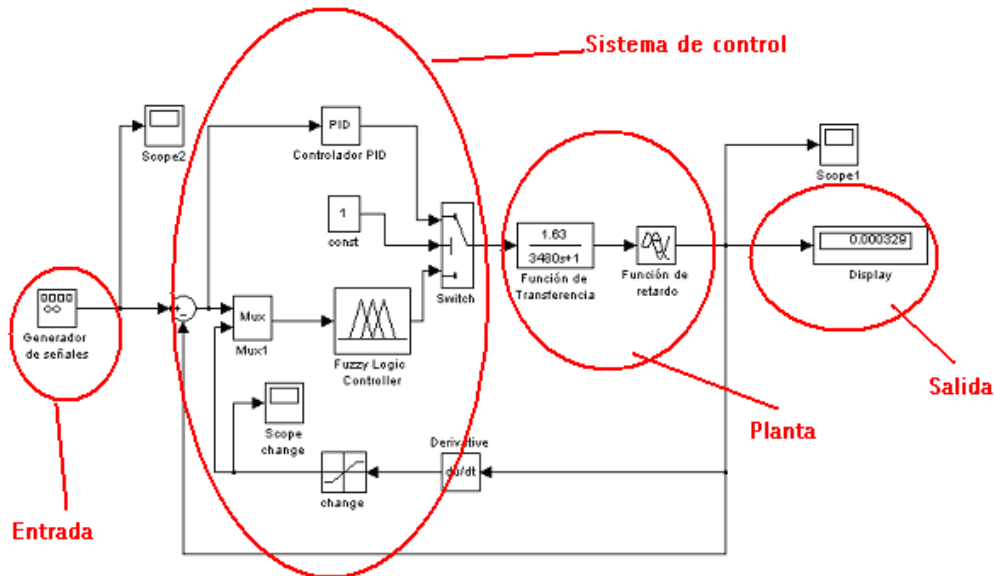
Figura 3.2. Esquema representativo de un control difuso

Las etapas mostradas en la figura 3.2 son:

- a. La fusificación (difusificación) toma valores de la planta y los interpreta como valores lingüísticos.
- b. La máquina de inferencia realiza los planteamientos lógicos necesarios para la toma de decisiones.
- c. La defusificación (desfusificación) consiste en la conversión de datos lingüísticos a datos numéricos, mediante una ponderación y normalización de las sentencias lógicas antecedentes

- d. La base de conocimiento, incluye los parámetros necesarios para la fusificación, inferencia y defusificación, los cuales pueden ser de naturaleza heurística³⁰ u optimizados mediante alguna técnica particular.

En nuestro sistema el control difuso está claramente representado en la siguiente figura que lo conseguimos aplicando el modelo a la herramienta SIMULINK del MATLAB.



La fusificación toma valores de la planta, en la Entrada, es decir con el generador de señales y los interpreta como valores lingüísticos con la ayuda de los multiplexores y las operaciones matemáticas.

La máquina de inferencia realiza los planteamientos lógicos necesarios para la toma de decisiones, es aquí donde utilizamos el controlador PID, para analizar las reglas ya antes analizadas y planteadas, tiene varios parámetros a evaluar, los datos de entrada y el cambio fundamentalmente, aunque también introduce los valores constantes, luego de envía estos resultados a la función de transferencia.

La defusificación (desfusificación) consiste en la conversión de datos lingüísticos a datos numéricos, mediante una ponderación y normalización de las sentencias lógicas antecedentes, es aquí donde se utiliza la función de transferencia con el correspondiente

³⁰ heurística: Detallada

tratamiento matemático (LAPLACE), y la función de retardo que toda función de transferencia involucra, de esta forma se envía la información numérica en la salida.

TODO ESTE PROCESO ES RETROALIMENTADO, por un tiempo X , que el mismo operador es el encargado de marcar para realizar el correspondiente muestreo.

Aunque la lógica difusa surge como un acercamiento a la manera cualitativa de pensar del ser humano y, en muchos casos, es fácil construir reglas y conjuntos difusos con un buen funcionamiento basado solamente en la experiencia, hay ocasiones en que se desea un alto desempeño y es necesario utilizar alguna técnica de optimización o búsqueda.

Existen tres tipos de técnicas tradicionales de optimización y búsqueda que son:

- las basadas en el cálculo,
- las enumerativas,
- y las aleatorias.

Las técnicas basadas en el cálculo son las más ocupadas, pero no es por demás indicar que cuando un problema tiene muchos mínimos puede llegar a tener problemas.

Las técnicas enumerativas prueban solución por solución, y por lo tanto en espacios grandes de búsqueda son excesivamente lentas.

Las técnicas de búsqueda aleatoria son impredecibles. Esto provoca que los métodos de búsqueda no sean robustos.

Los algoritmos genéticos son una alternativa al problema de la robustez, pues su espacio de búsqueda no está limitado por discontinuidades, no linealidades o la inexistencia de derivadas. El algoritmo genético, no tiene limitaciones con respecto al espacio de búsqueda y difícilmente se confunde con mínimos o máximos múltiples, pues su búsqueda es paralela.

En nuestro sistema se utilizan las Técnicas basadas en el cálculo y las enumerativas pues el sistema no ofrece ni muchos mínimos ni se tiene que evaluar en grandes espacios, entonces resulta ser muy óptimo.

3.6.5 Modelo de procesos

Una vez que se han introducido los conceptos relacionados con el área de control de procesos y sobre el uso de la inteligencia artificial y la lógica difusa en el área de control de procesos se procederá a establecer las condiciones previas para el planteamiento del modelo a emplear.

El modelo del proceso normalmente consta de elementos de entrada, salida, comparador, controlador, actuador, medidor, etc. Los elementos mencionados se pueden observar en el diagrama 3.3, que es la representación típica de un sistema de control de procesos:

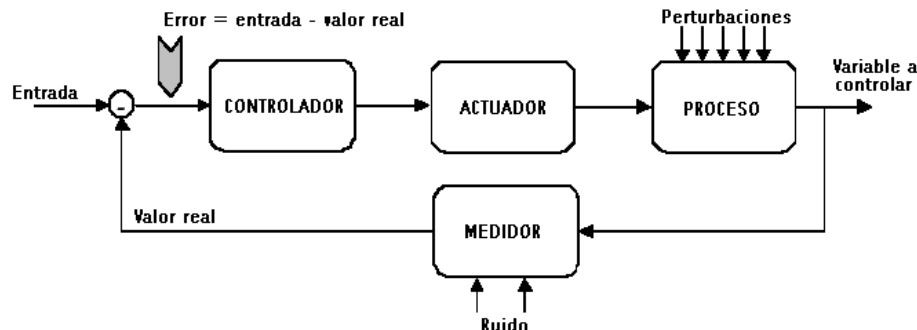


Figura 3.3 Modelo general de un proceso

A continuación se describirán brevemente cada uno de los elementos mostrados en el diagrama 3.3.

Entrada

Las entradas típicas que se utilizan para el diseño, simulación y análisis de sistemas de control son:

- Escalón
- Rampa

· Sinusoidal

En nuestro sistemas tenemos muchas más alternativas, podemos probar en cualquiera de estas, más lo haremos en la sinusoidal.

Salida

La salida se visualiza en dispositivos que se asemejan a osciloscopios de manera que se pueda observar el resultado obtenido en el proceso, referido como una variable del proceso que está siendo controlada.

Para nuestro sistemas simulamos este osciloscopio, utilizamos el elemento “scope” en SIMULINK, para poder ver los efectos claros.

Comparador

En un sistema de control, el comparador calcula la señal de error consistente en restar a la entrada el valor medido por el sensor, por tanto, se trata sólo de una operación aritmética de suma o resta.

Proceso a controlar

Si las condiciones iniciales de un sistema representado por una ecuación diferencial son nulas, cuando se le aplica la transformada de Laplace existe una relación entre la entrada y la salida del sistema (Figura 3.4) que se conoce como función de transferencia.

La función de transferencia ($G(s)$), ver figura 3.5, de un sistema es la relación que existe entre la transformada de Laplace de la salida ($Y(s)$) con respecto a la transformada de Laplace de la entrada ($X(s)$).

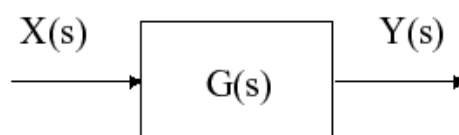


Figura 3.4 Representación de un sistema

$$Y(s) = X(s) \cdot G(s) \Rightarrow G(s) = \frac{Y(s)}{X(s)}$$

Figura 3.5 Función de transferencia de un sistema

Actuador

Los actuadores no dejan de ser sistemas que se modelizan utilizando ecuaciones diferenciales y por tanto descritos por su función de transferencia. Si se observa el diagrama de bloques de un sistema de control, se deduce que los actuadores se encuentran en serie con el proceso y por tanto se pueden combinar las dos funciones de transferencia multiplicándolas y obteniéndose un solo bloque.

Controlador

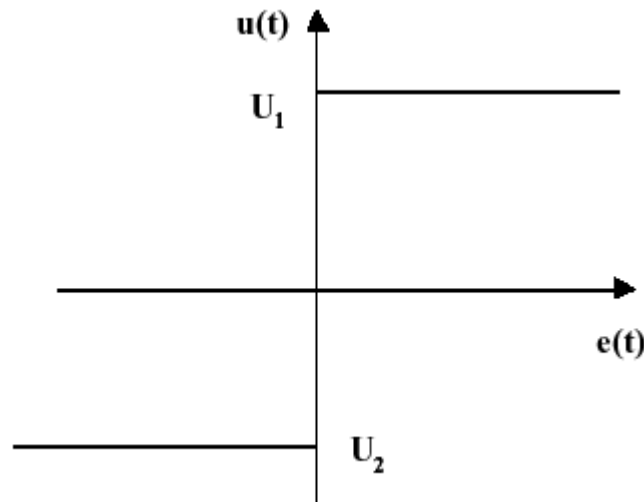
Existen dos tipos de controladores de procesos: continuos y discontinuos. En un controlador discontinuo, el elemento de control sólo proporciona dos niveles:

- Todo
- Nada

El actuador tiene, por tanto, sólo dos posiciones fijas, que en la mayoría de los casos son: conectado y desconectado. Si el error que presenta el controlador es $e(t)$, y la señal de control que proporciona es $u(t)$, el controlador todo o nada se representa por la expresión $u(t)$, mostrado en la figura 3.6.

$$u(t) = \begin{cases} u_1, & \text{para } e(t) > 0 \\ u_2, & \text{para } e(t) < 0 \end{cases}$$

Figura 3.6 Representación de un controlador discontinuo

Figura 3.7 Representación gráfica de $u(t)$

En el caso de un controlador continuo, la salida de este tipo de controladores es continua en todo el rango de variación, es decir, puede tomar cualquier valor entre el 0 y el 100% de la salida. Los principales controladores continuos son: controlador proporcional (P), controlador proporcional integral (PI), controlador proporcional derivativo (PD), controlador proporcional integral derivativo (PID).

3.6.6 Estudio del problema a desarrollar

En este caso, la entrada a emplear será la sinusoidal ya que el horno requiere de una alimentación de corriente alterna de 220 voltios. Ver la figura 3.9 para ver la representación en Matlab.

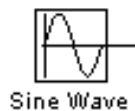


Figura 3.9: Representación en Matlab de la entrada senoidal

Una función de transferencia que tiene la forma de la que se muestra en la figura 3.10 corresponde a la representación de un sistema con retardo.

($G(s)$), de un sistema es la relación que existe entre la transformada de Laplace de la salida ($Y(s)$) con respecto a la transformada de Laplace de la entrada ($X(s)$).

$$Y(s) = X(s) \cdot G(s) \Rightarrow G(s) = \frac{Y(s)}{X(s)}$$

Utilizando Ziegler Nichols y el método trapezoidal se determina la ecuación de la forma.

$$G(s) = \frac{Y(s)}{X(s)} = \frac{k \cdot e^{-s \cdot t_r}}{1 + \tau s}$$

Figura 3.10: Función de transferencia de un sistema con retardo

Como se mencionó, la función de transferencia de la planta a emplear en el presente documento es:

$$G(s) = \frac{1.63 e^{-270s}}{1 + 3480s}$$

En el caso del problema que se está tratando, los valores de los coeficientes respecto de la expresión de la figura 3.10, son: . *Para ver el detalle de la obtención de estos coeficientes puede consultar el Libro: Ingeniería de control moderno, Katsuhiko Ogata, Pg. 134-150*

$$k = 1.63$$

$$t_r = 270$$

$$t = 3480$$

En Matlab se representan a través de dos elementos: un bloque denominado TRANSFER FCN y otro denominado TRANSPORT DELAY. Esto se muestra en la figura 3.11, la cual representa a la función de transferencia del problema en consideración.

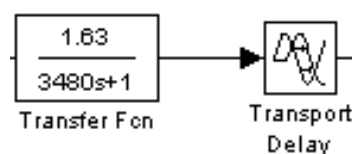


Figura 3.11: Representación en Matlab de la función G(s)

3.6.7 MODELO EN MATLAB

De acuerdo a lo descrito en las páginas anteriores, se procederá a construir el modelo en MATLAB, que represente al problema en consideración para su simulación empleando SIMULINK.

Una simulación de un modelo SIMULINK puede llevarse a cabo mediante la invocación de comandos en modo texto desde la ventana de MATLAB o bien en modo interactivo desde los menús del interfaz gráfico del propio SIMULINK.

La simulación en Simulink generará resultados en base a las reglas establecidas para el modelo y a los parámetros definidos como: tolerancias relativa y absoluta; intervalo de simulación; tamaños de paso inicial, mínimo y máximo.

Para ello, se considera en primera instancia los parámetros de la simulación (ver figura 3.12) que son: tiempo de simulación (simulation time) y el motor de simulación (Solver).

Los datos que se consideran son:

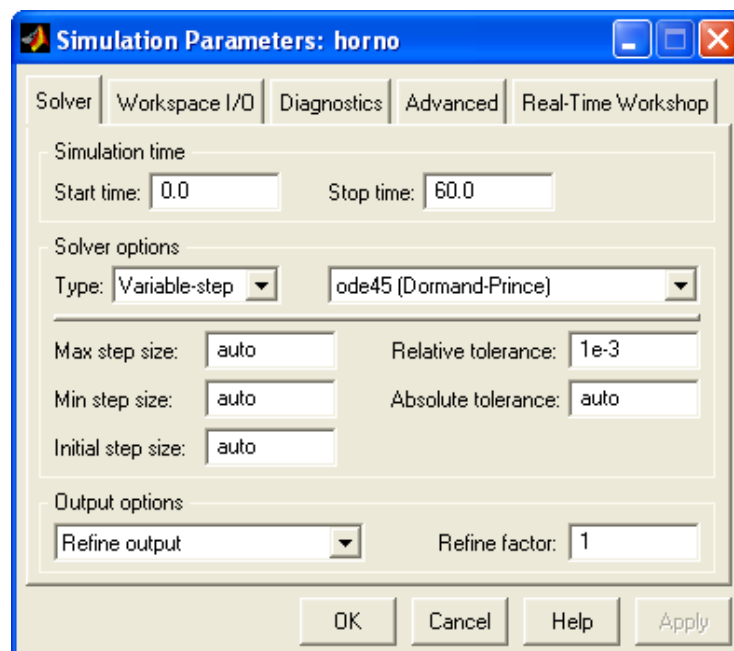


Figura 3.12. Parámetros de la simulación

El tiempo de simulación establece el tiempo de comienzo y fin de la simulación. Tiempo en términos de la propia simulación, que no coincide con el transcurso del tiempo real: si los cálculos implicados en la simulación son sencillos, simular 20 segundos de experimento puede tardar tan sólo 1 segundo. O al revés si los cálculos son muy complejos.

En cuanto a la elección del motor de simulación, ésta dependerá del tipo concreto de modelo con que se esté tratando. Debido a la diversidad de comportamientos de los sistemas dinámicos, SIMULINK permite elegir entre una amplia gama de motores especializados en un tipo de comportamiento particular.

Una primera cuestión a elegir sobre el motor es si será de paso fijo o de paso variable.

Los de paso variable pueden modificar durante la simulación su paso, lo que les permite ciertas ventajas como la posibilidad de detectar los cortes con cero de una señal.

Los de paso fijo no aportan esas posibilidades, pero son los únicos que se pueden elegir si se desea traducir el modelo a un programa ejecutable. Como es el caso de nuestro sistema.

El motor seleccionado es uno genérico denominado ODE45 (ODE son las siglas de Ordinary Differential Equations, es decir, Ecuaciones Diferenciales Ordinarias),. Este motor seleccionado se ha escogido por el comportamiento matemático de nuestro modelo, pero en base a la experiencia del Ingeniero y necesidades de los sistemas se pueden escoger entre tantos otros motores de los que dispone MATLAB.

Como elemento de regulación se empleará un controlador mediante lógica difusa, para lo cual se van a establecer las reglas que permitirán a este elemento realizar su función.

Las variables de entrada y la salida, así como sus valores posibles se indican en la tabla siguiente:

Variables	Valores posibles
NIVEL DE TENSION	Bajo Alto Aceptable
RAZON DE CAMBIO DE TEMPERATURA	Lenta Aceptable Rápida
Salida	Valores posibles
TEMPERATURA	Baja Alta Aceptable

La **base de conocimiento** del controlador difuso tiene las siguientes reglas, que se consideran como válidas para ser aplicadas en el modelo:

- *Si el NIVEL DE TENSION es BAJO entonces la TEMPERATURA es BAJA*
- *Si el NIVEL DE TENSION es ALTO entonces la TEMPERATURA es ALTA*
- *Si el NIVEL DE TENSION es ACEPTABLE entonces la TEMPERATURA es ACEPTABLE*
- *Si el NIVEL DE TENSION es ACEPTABLE y la RAZON DE CAMBIO DE TEMPERATURA es LENTA entonces la TEMPERATURA es ACEPTABLE*
- *Si el NIVEL DE TENSION es BAJO y la RAZON DE CAMBIO DE TEMPERATURA es RAPIDA entonces la TEMPERATURA es ALTA*
- *Si el NIVEL DE TENSION es ALTO y la RAZON DE CAMBIO DE TEMPERATURA es RAPIDA entonces la TEMPERATURA es BAJA*

Estas reglas se muestran en el editor de reglas de lógica difusa de MATLAB (ver figura 3.14).

Para iniciar la ejecución del editor de reglas de lógica difusa se digita en la línea de comandos de MATLAB la palabra “fuzzy”.

En primer lugar, luego de ingresar el comando fuzzy se muestra el FIS EDITOR donde se presente el diagrama con el número de entradas y el de salidas así como la representación de la planta (Figura 3.13).

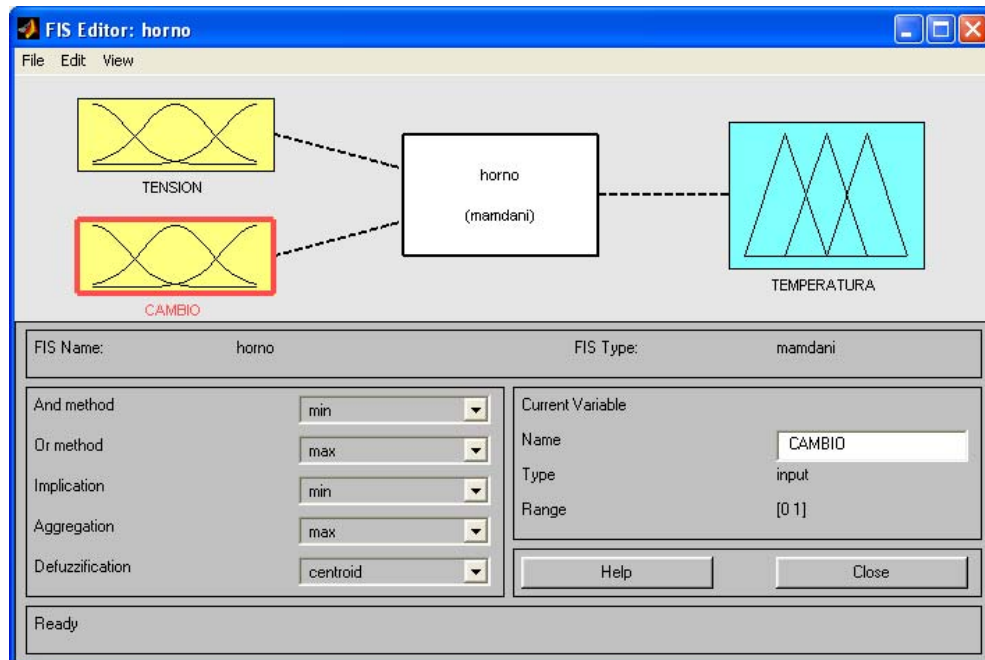


Figura 3.13. Editor FIS, con las entradas y salida del proceso

El siguiente paso es definir los nombres de las entradas y de la salida, en este caso, TENSION y CAMBIO, correspondientes al nivel de tensión que se aplica a la planta y a la razón de cambio de la temperatura que permitirá la variación del calentamiento del horno. La salida del proceso es la TEMPERATURA.

Partiendo de este esquema se establecen las reglas a emplear. Para ello se da doble clic en la planta que se muestra en la pantalla presentada en la figura 7.3, logrando que se muestre el editor de reglas (Figura 3.14).

El proceso de registrar las reglas consiste en seleccionar las variables, que se muestran en la parte inferior de la pantalla, y los valores que se han definido para cada una de ellas de forma tal que se logre conformar la regla en el formato IF THEN (Figura 3.14).

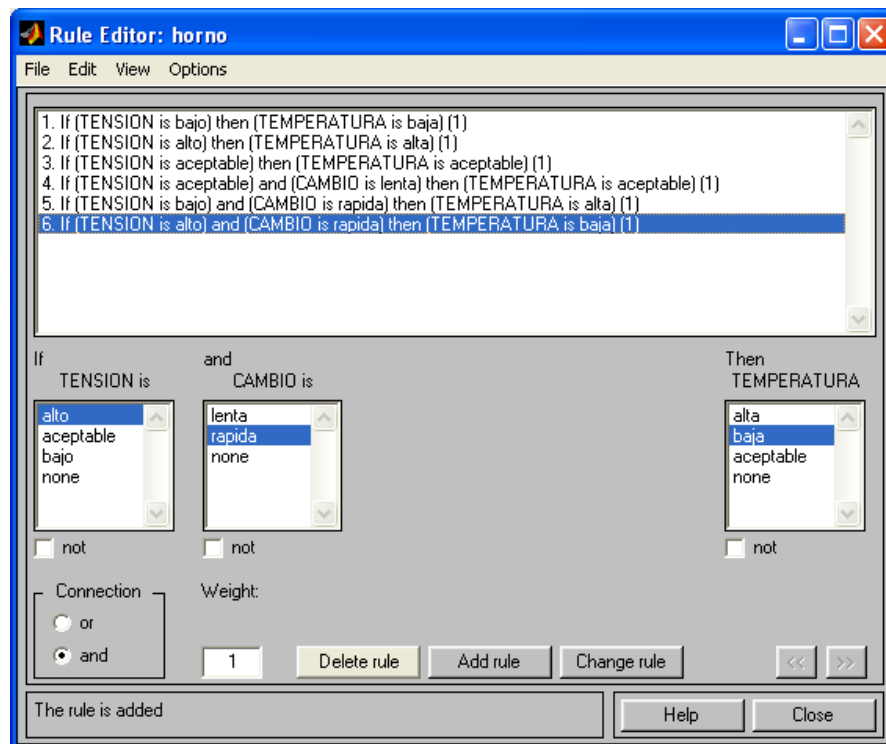


Figura 3.14. Editor de reglas de lógica difusa

En la figura 3.15 se puede observar una forma diferente de presentar las reglas establecidas para el proceso.

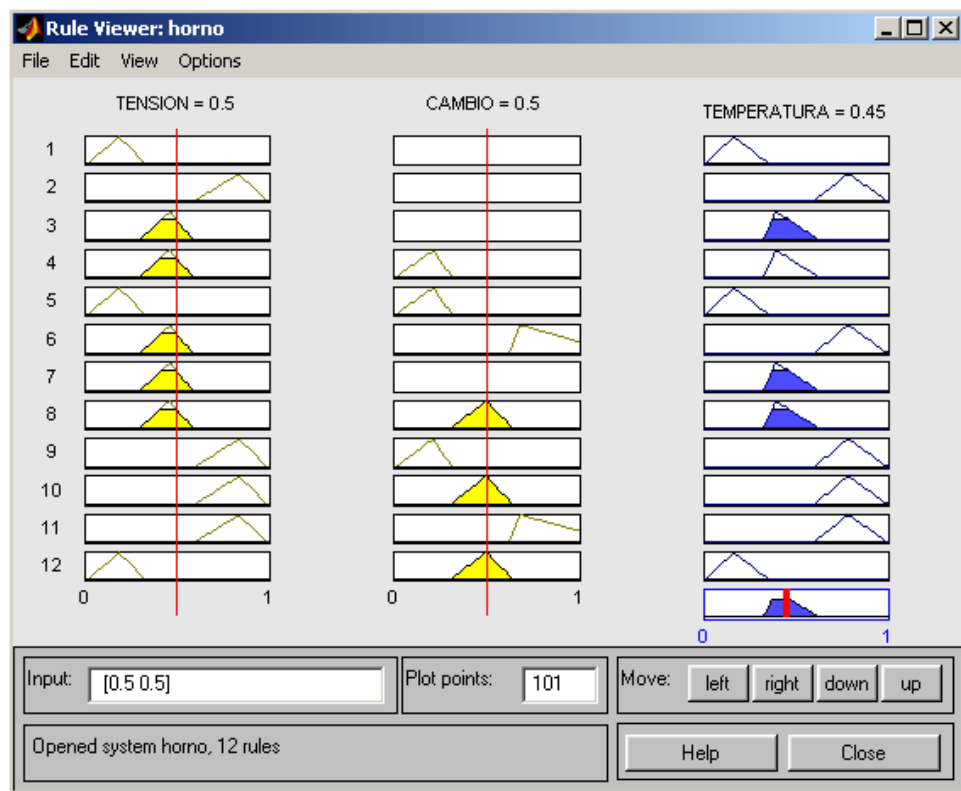


Figura 3.15 Reglas definidas para el proceso

Para ello, se selecciona la opción VIEW/RULES en el menú de MATLAB.

Una forma diferente de presentar las reglas es a través de la imagen que se muestra en la figura 3.16. Esta imagen se denomina superficie de control donde se puede ver, en los ejes del diagrama, las variables de entrada y la salida.

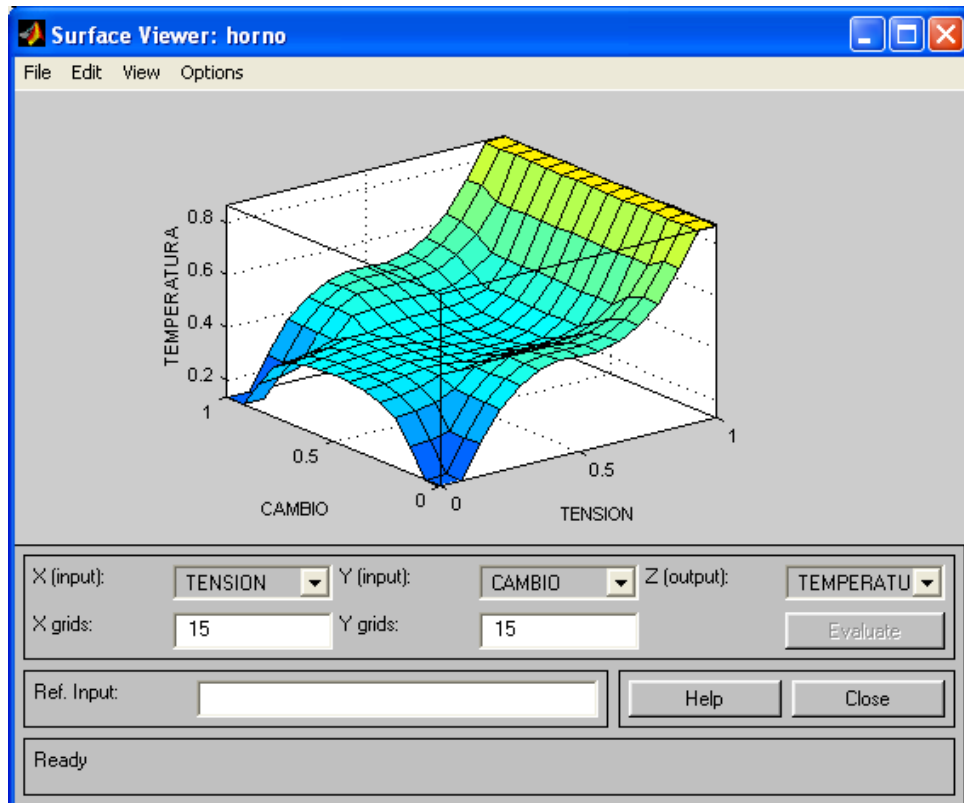


Figura 3.16 Superficie de control

Las reglas definidas en el proceso se almacenan en el archivo denominado HORNO.FIS, y su contenido es el que se muestra a continuación:

```
[System]
Name='horno'
Type='mamdani'
Version=2.0
NumInputs=2
NumOutputs=1
NumRules=6
AndMethod='min'
OrMethod='max'
ImpMethod='min'
AggMethod='max'
DefuzzMethod='centroid'

[Input1]
Name='TENSION'
Range=[0 1]
NumMFs=3
MF1='bajo':'trimf',[-0.4 0 0.4]
MF2='alto':'trimf',[0.1 0.5 0.9]
MF3='aceptable':'trimf',[0.6 1 1.4]

[Input2]
Name='CAMBIO'
```

```

Range=[0 1]
NumMFs=3
MF1='lento': 'trimf',[-0.4 0 0.4]
MF2='rapido': 'trimf',[0.1 0.5 0.9]
MF3='mf3': 'trimf',[0.6 1 1.4]

```

```

[Output1]
Name='TEMPERATURA'
Range=[0 1]
NumMFs=3
MF1='baja': 'trimf',[-0.4 0 0.4]
MF2='alta': 'trimf',[0.1 0.5 0.9]
MF3='aceptable': 'trimf',[0.6 1 1.4]

```

```

[Rules]
1 0, 1 (1) : 1
2 0, 2 (1) : 1
3 0, 3 (1) : 1
3 1, 3 (1) : 1
1 2, 2 (1) : 1
2 2, 1 (1) : 1

```

Considerando lo expuesto hasta el momento, así como las características del horno, el modelo realizado en SIMULINK es el que se muestra en la figura 3.17, donde se especifican los elementos de un diagrama de control.

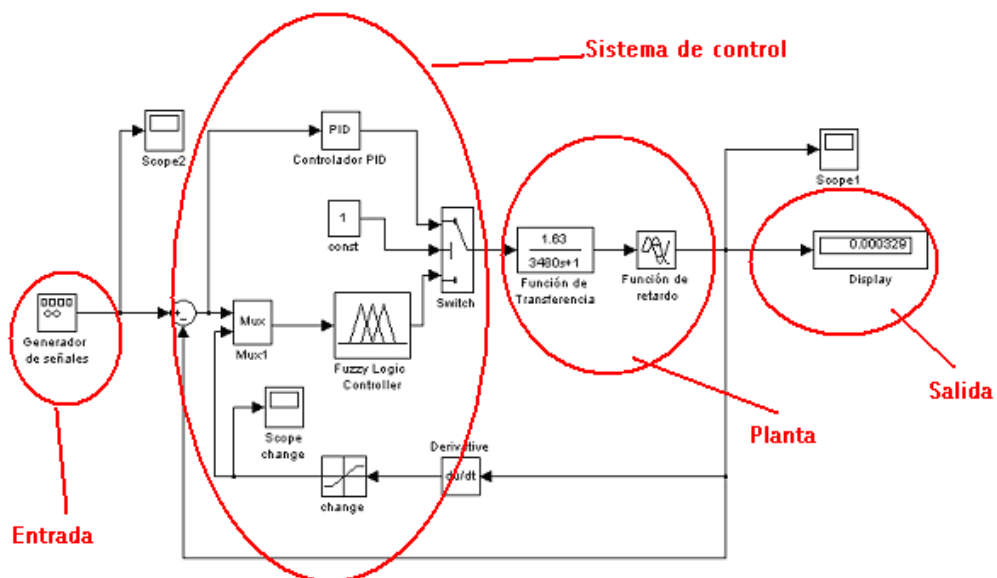


Figura 3.17. Diagrama esquemático del proceso

En este diagrama se puede ver los componentes siguientes:

- La entrada se define a través de un generador de señales, la cual se define con una onda sinusoidal y una amplitud y frecuencia de 1 (Figura 3.18).

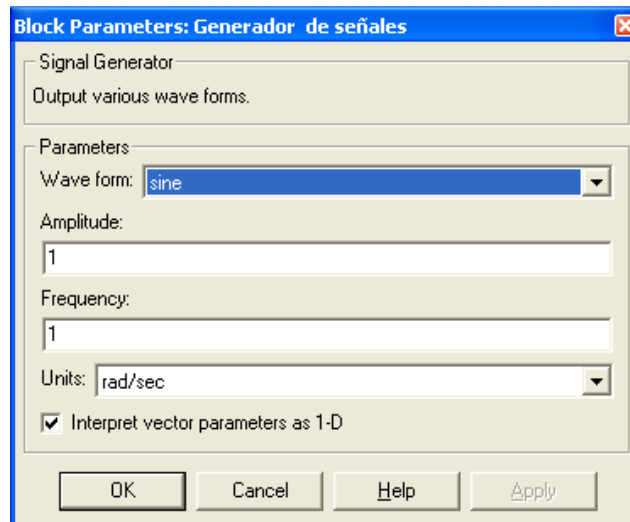


Figura 3.18. Definición de la señal de entrada

La forma de onda sinusoidal de la entrada se puede observar a través de un visor (SCOPE) conectado al generador de señales (Figura 3.19).

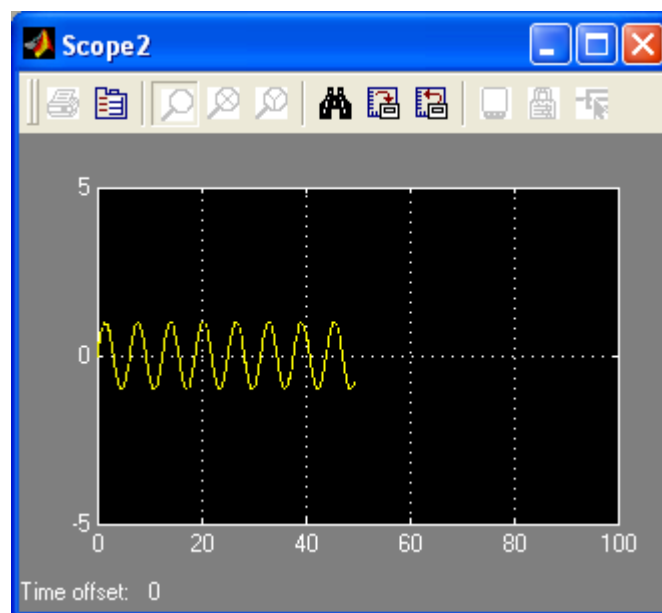


Figura 3.19. Forma de onda de entrada

b. Un controlador PID (Proporcional, Integral, Derivative) con los parámetros que se muestran en la figura 3.20. Estos parámetros se obtuvieron en base al cálculo que se muestra en la descripción de la función de transferencia. *Págs. 75-77*

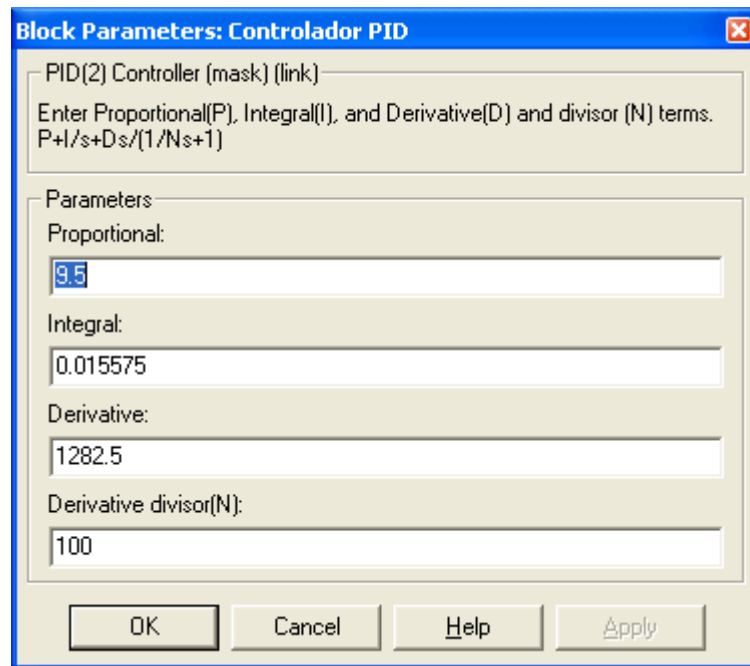


Figura 3.20. Parámetros del controlador PID

- a. El controlador de lógica difusa, cuyos parámetros son las reglas definidas para el proceso de control.
- b. Se emplean elementos adicionales como un multiplexor de señales (MUX), un derivador de señales (DERIVATIVE), un switch. Estos elementos permiten el manejo de la señal así como su variación para la realimentación que permite al proceso lograr su funcionamiento adecuado.

En la figura 3.21, se puede observar con más detalle el esquema descrito.

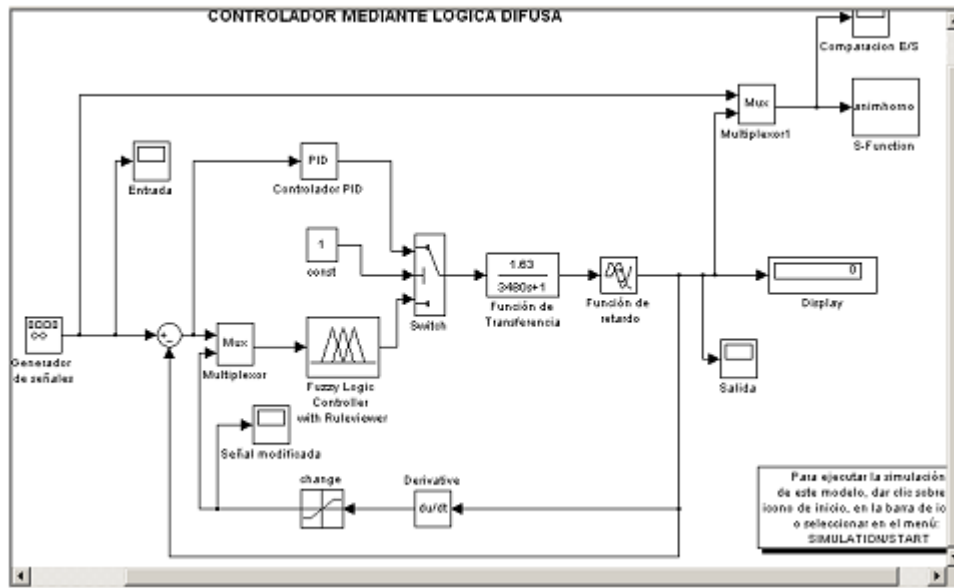


Figura 3.21. Modelo del horno construido en SIMULINK

La forma de los conjuntos (variables de entrada y salida) consideradas para la simulación del proceso es triangular, de acuerdo a lo que se muestra en la figura 3.22:

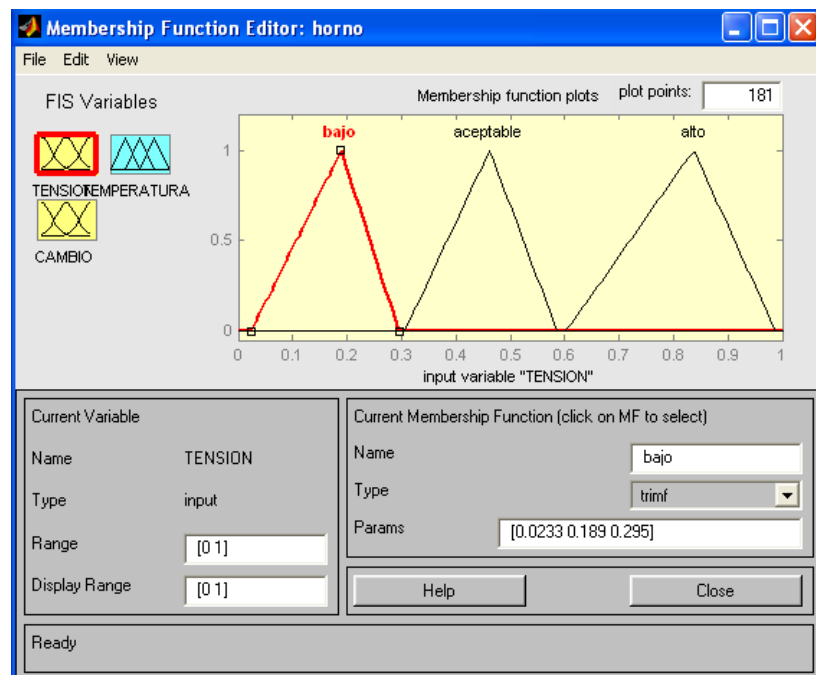


Figura 3.22: Forma de los conjuntos definidos en el modelo

Una simulación del modelo, considerando los parámetros de simulación mostrados y las reglas presentadas, genera el siguiente resultado (figuras 3.23 y 3.24):

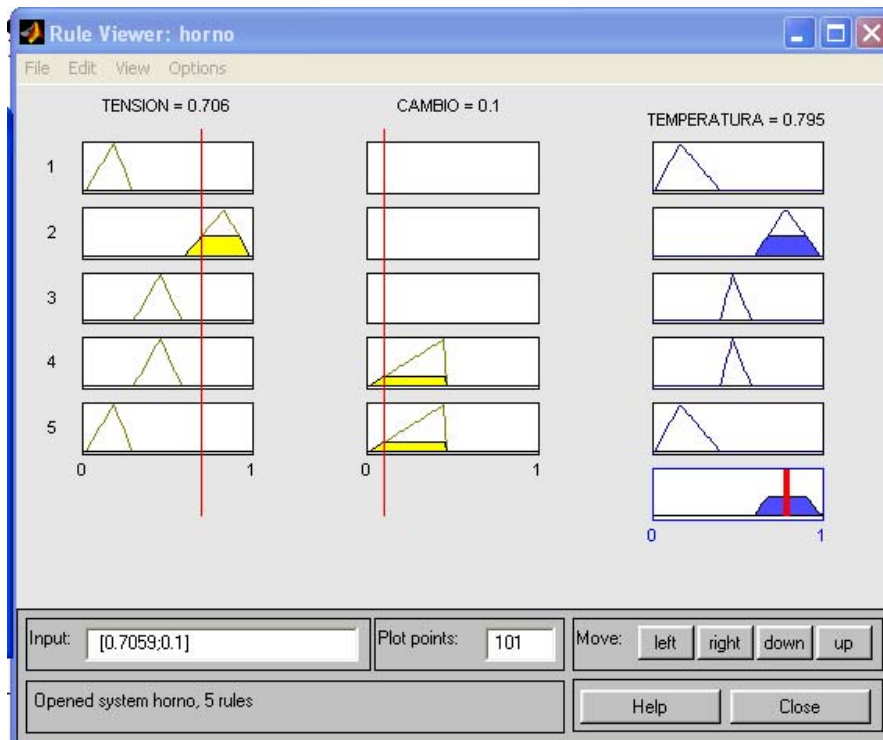


Figura 3.23. Resultado de una simulación

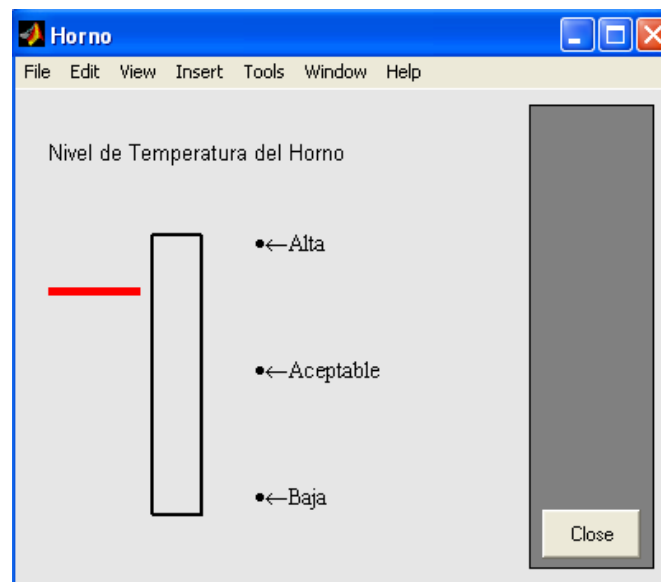


Figura 3.24. Salida del proceso (temperatura alta)

Una modificación en el tiempo de simulación genera otro resultado (figuras 3.25 y 3.26):

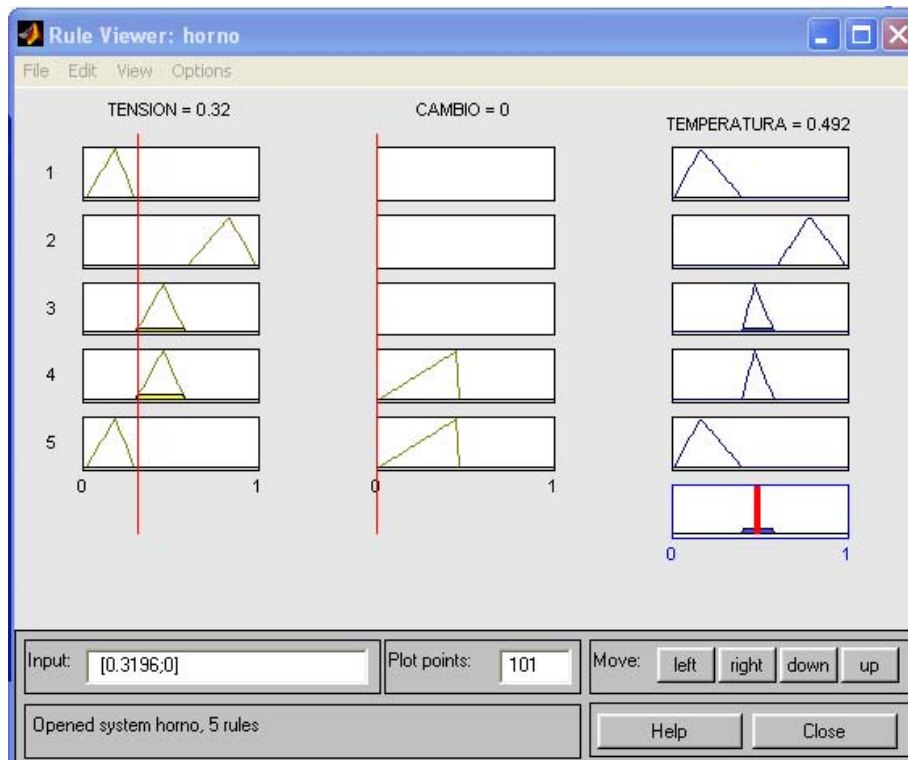


Figura 3.25: Resultado de una simulación

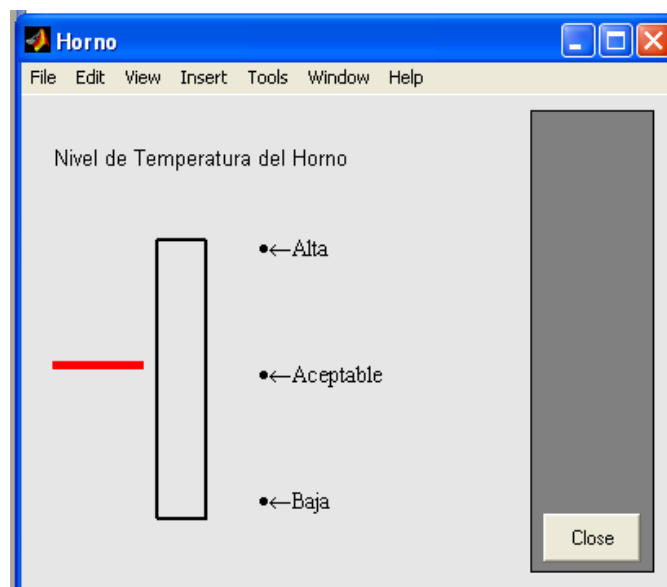


Figura 3.26. Salida del proceso (temperatura aceptable)

Finalmente, se presenta un tercer resultado de la simulación (figuras 3.27 y 3.28):

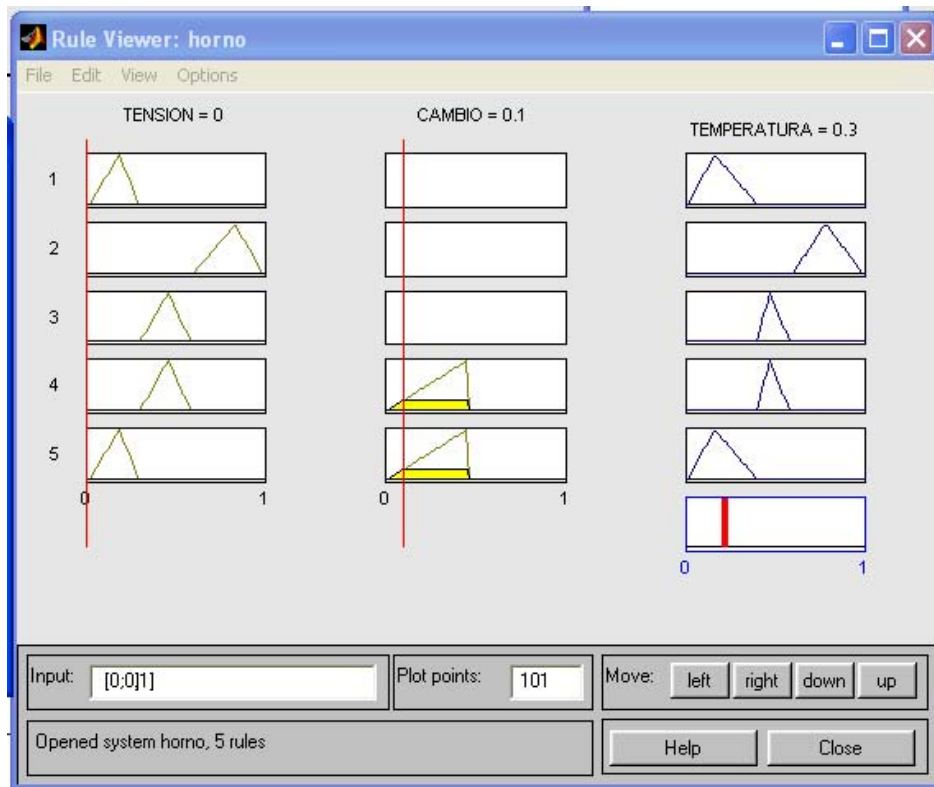


Figura 3.27. Resultado de una simulación

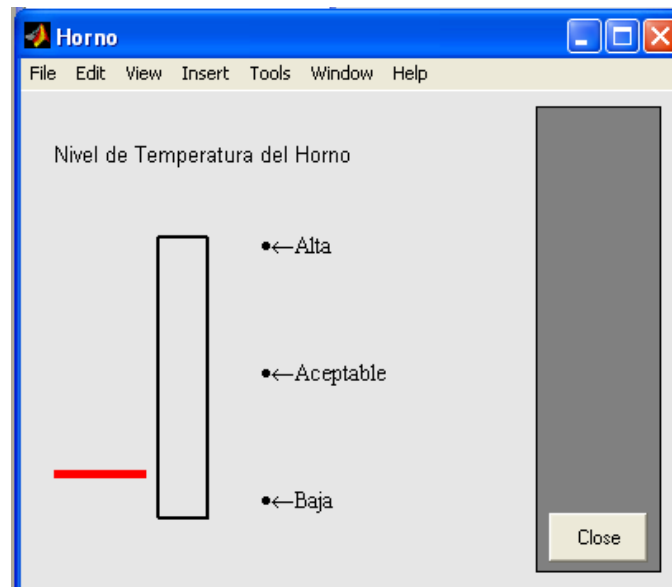
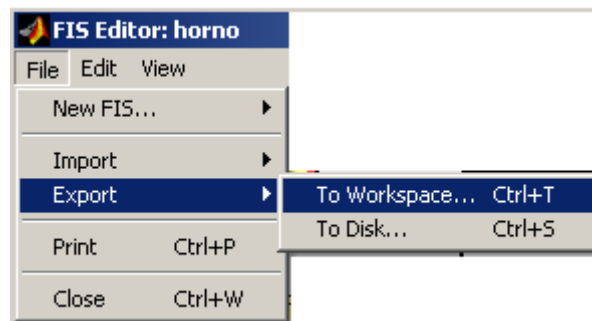


Figura 3.28. Salida del proceso (temperatura baja)

3.7 COMO CARGAR LA APLICACIÓN REALIZADA

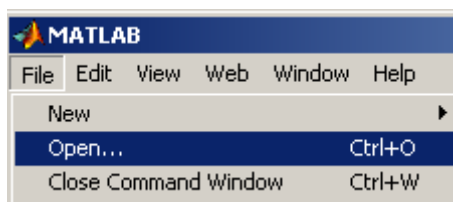


1. Cargue Matlab, dando doble clic en el icono
2. En la ventana de comando digite: `>> fuzzy horno` (carga el módulo de lógica difusa creado y que se llama horno)
3. En la ventana de comando digite: `>> simulink` (carga el simulador del Modelo)
4. En el módulo de Fuzzy Logic horno escoja la opción:

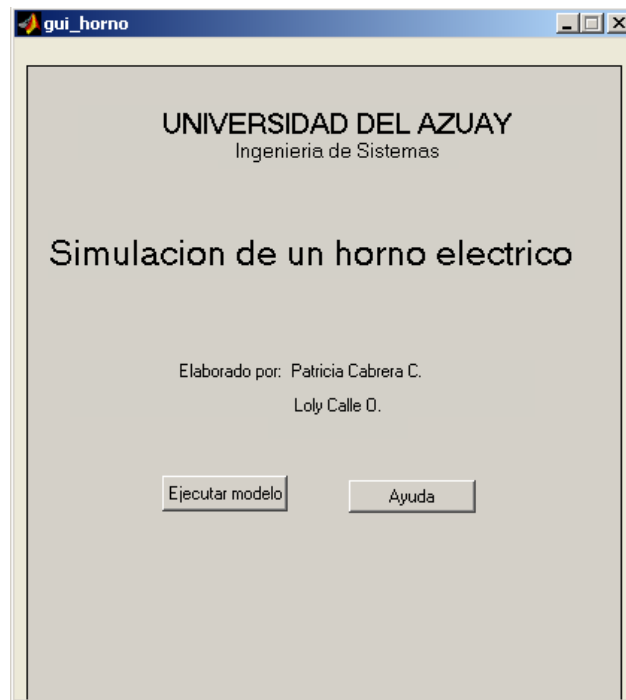


Esto hará que los datos que se encuentran en Fuzzy Logic sean exportados a Simulink para generar el diagrama de bloques.

5. En el módulo de Fuzzy Logic escoja la opción del menú



y escoja el archivo `gui_horno.fig` para cargarlo, esto mostrará la pantalla de presentación:



Desde aquí puede ejecutar el Modelo con el botón ejecutar modelo.

Si desea cambiar los parámetros simplemente deberá ir al módulo de Lógica difusa y dar doble clic sobre la variable a cambiar, ya sea cambio o tensión.

Puede obtener ayuda dando clic en el botón Ayuda que se muestra en la pantalla inicial.

CAPITULO IV

4 LOGICA DIFUSA Vs. LOGICA CLASICA

4.1 Diferencias entre Lógica Difusa Vs. Lógica Clásica.

Contraponer a la Lógica clásica con la lógica difusa conlleva una serie de aspectos a considerar, entre ellos mencionaremos los siguientes:

ASPECTOS GENERALES

- No olvidemos que la lógica DIFUSA, es una generalización de la lógica BOOLEANA, y hablar de generalización conlleva ya en sí una gama mucho más grande de elementos a ser evaluados.
- Al tener una muestra mucho mayor de elementos a ser evaluados tenemos que incorporar muchas técnicas más amplias y precisas para poder analizar los problemas en cuestión.
- Necesitamos pues, no de un simple aprendiz sino de una persona (técnico) que se ocupe de evaluar todos los aspectos necesarios para conseguir los datos analizados.
- La lógica Difusa da respuesta a problemas que con la lógica tradicional resultaban ser borrosos, ahora ya encontramos una solución.
- La lógica difusa facilita la ingeniería de control para varios tipos de problemas que anteriormente fueron difíciles o casi imposibles de plasmarlos a través de un sistema de control, como por ejemplo, hoy en día las máquinas de lavado incluyen en sus procesos lo que denomina “Fuzzy Control”, esto se puede ver diariamente, en el caso de un ama de casa frente a su lavadora automática, si ella desea que el lavado sea sutil o fuerte, simplemente mueve el botón de control,

pero internamente se trata de interpretar lo que ella desea lograr con su lavado, antes no nos hubieramos preguntado que ocurre con la lavadora y su modo de operación, más ahora al desarrollar esta aplicación sabemos que el ciclo de lavado conlleva una serie de factores como son: Un sistema de Retroalimentación a través de una técnica de control que evalúa la función de entrada y Salida de la información.

Entonces podemos enmarcar lo siguiente:

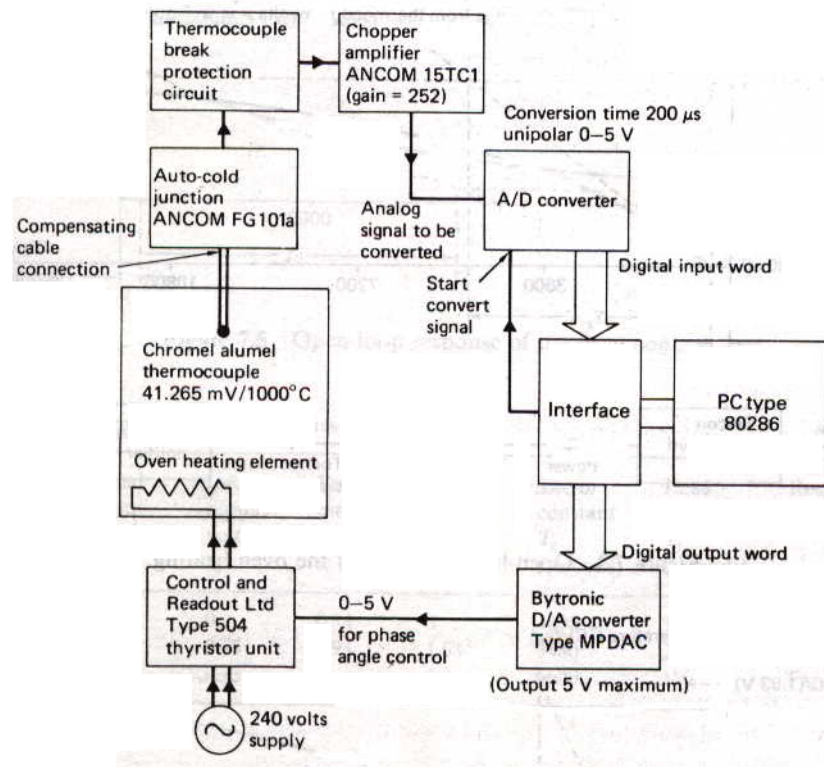
- Como **principal diferencia** entre la lógica tradicional y la difusa tenemos que mientras que los valores de la función de pertenencia de la primera son 0 o 1, la lógica difusa se mueve en todo el intervalo $[0,1]$.

Usted entonces no tendrá un No rotundo ni un Si cerrado, encontramos ya las probabilidades que involucran lo que se denominan “las reglas de inferencia”

Un ejemplo a ser cuestionado es “la temperatura del horno”, entonces nos preguntamos que pasaría si el problema de la temperatura del horno lo queremos desarrollar utilizando la lógica booleana?, la respuesta es que nosotros nos veríamos frente a un problema muy complejo en el que tendríamos que utilizar muchos implementos técnicos para producir la correspondiente codificación de señales, cada n segundos una instrucción inicializa una conversión analógica-digital que toma cerca de 200 milisegundos para ser completada, el programa debe tener un tiempo para esta conversión dándole tiempo también para la lectura en la entrada, entonces se debe producir los algoritmos de operación y luego enviar los resultados y para ser registrados en el convertidor digital-analógico.

A continuación ilustramos un diagrama que representa todos los implementos que serían necesarios para resolver el problema utilizando la lógica booleana, cabe la pena aquí detenernos un instante y pensar un poco más en “términos electrónicos”, que no es nuestro objetivo.

Detalle del Hardware necesario para el control de temperatura en un horno



En el gráfico anterior podemos visualizar una serie de dispositivos de control tales como:

- Termocuplas
- Amplificadores
- Convertidores analógico-digital
- Convertidores digital-analógico

Todos estos elementos (electrónicos) desaparecen al emplear un programa basado en lógica difusa mediante el uso de un simulador (Simulink) con el uso de la herramienta Fuzzy Logic Toolbox para el uso con MatLab, que ha resultado ser una herramienta capaz de resolver este tipo de problemas con una rapidez muy significativa que para el ser humano sería resuelto en un tiempo muy largo.

- Otra gran diferencia es que no podemos esperar que un problema resuelto con la lógica tradicional nos dé respuestas precisas a situaciones que involucren la subjetividad del comportamiento humano. Las interpretaciones subjetivas están incorporadas dentro de la lógica difusa, utilizando técnicas que involucran procesos derivativos e integrales (PID).
- La lógica difusa se basa en una lista de sentencias “if-then”, las mismas que se llaman reglas y que son evaluadas en paralelo sin importar el orden. Antes de construir un sistema que interprete estas reglas debemos definir los términos que planeamos emplear y los adjetivos (variables lingüísticas) que los describan.

En el caso de la lógica tradicional las sentencias “if-then” no presentan dificultad, pues si la premisa es verdadera entonces la conclusión también lo es. Pero si relacionamos la restricción de la lógica binaria e interpretamos el resultado utilizando lógica difusa, ¿cómo nos afectaría la conclusión? Simplemente si el antecedente es verdadero en algunos grados, entonces el resultado es verdadero en el mismo grado, en otras palabras:

En lógica binaria $p \Rightarrow q$ (p y q son verdaderos o los dos son falsos)

En lógica difusa $0.5p \Rightarrow 0.5q$ (Antecedentes parcialmente verdaderos implican el mismo grado de verdad en el resultado).

La lógica difusa es muy empleada en el reconocimiento de patrones, en los procesos con señales de entrada y para simular modelos de aplicaciones.

- Las aplicaciones que desarrollan proyectos utilizando lógica Difusa, también permiten implementar la lógica convencional, entonces resulta ya una gran ventaja de la Difusa frente a la booleana, es por ende que se dice una tecnología en vías de crecimiento, que se pretende estará en el mercado en los próximos años con mucha mayor influencia que en la actualidad, pues presente muchas bondades para el desarrollo de aplicaciones digitales.

CONCLUSIONES

La investigación y el desarrollo de la presente monografía nos ha permitido analizar y evaluar las ventajas y desventajas de la lógica bivaluada y la lógica difusa, para nosotras las diferencias saltaron de inmediato porque hasta hoy habíamos trabajado con la lógica tradicional, pues en nuestros trabajos la usamos diariamente, pero al complementar nuestros estudios de Ingeniería conocimos brevemente la materia de Inteligencia Artificial y nos pareció que el mundo de la tecnología avanza a pasos agigantados y que es muy interesante programar a un PC de modo que pueda interpretar la forma de “pensar” del ser humano.

En nuestro medio el tema de la lógica difusa no es muy conocido como ciencia, aunque en el mercado se comercializan equipos en cuyo funcionamiento se incluyeron procesos de lógica difusa. Luego de realizada nuestra investigación podemos concluir que:

- La lógica difusa no necesariamente reemplaza los métodos convencionales de razonamiento, estos métodos forman parte de ella.
- La lógica difusa resuelve problemas basándose en modelos matemáticos, esta parte matemática resulta fácil incorporarla pues las herramientas que se utilizan en lógica difusa traen incorporadas un sinnúmero de funciones que facilitan la operación para el operador, quien simplemente debe ingresar argumentos y así obtener resultados.
- Tolera datos no precisos, recuerde el pensamiento humano nunca da solamente valores enteros.
- Si nos percatamos y damos un vistazo a la mayor parte de las cosas a nuestro alrededor, estos son imprecisas, la lógica difusa nos permite interpretar esta impresión en términos numéricos.
- La lógica difusa es flexible

- Mediante el uso de herramientas como el Matlab se pueden crear sistemas para que sean coincidentes con datos de entrada y salida, esto es particularmente sencillo pues se puede adaptar técnicas y herramientas tales como Simulink, ANFIS (Adaptive Neuro-Fuzzy Inference Systems).
- Si contrastamos con las redes neuronales, estas toman datos de prueba que generan modelos confusos y opacos, la lógica difusa le permite plasmar una realidad y ser interpretada por gente que nada sabe de la lógica difusa., pero la utiliza. (Control de los ciclos de lavado)
- La lógica difusa necesita técnicos para poder manipular las herramientas.
- El modelo matemático que se analice sirve para una aplicación en particular en la que los datos se mueven de acuerdo al mismo, más resulta no adaptable a otros sistemas.
- Las herramientas que se utilizan en la lógica difusa resultan mucho más caras que las tradicionales pues es una técnica que se encuentra en vías de Expansión.
- La lógica difusa no es para todo, La lógica difusa es conveniente para realizar un muestreo de una señal de entrada y una salida que trata de interpretar ideas borrosas, de lo contrario puede utilizar la lógica booleana, todo depende de los objetivos a obtener.

- La lógica difusa puede realizar muchísimas cosas pero las más importantes es crear y editar sistemas de inferencia. Usted puede crear estos sistemas utilizando herramientas gráficas o comandos y funciones en línea.
- En cuanto a la herramienta utilizada en nuestra monografía, nos referimos a Matlab, esta es una herramienta poderosa, que provee un ambiente para el diseño de sistemas de control, procesos con señales de entrada, modelado, análisis y algoritmos. Es un programa experto en números y también hace que el trabajo sea fácilmente interpretado en forma gráfica usa una interface SIMULINK y gráficos en dos y tres dimensiones. Trae incorporada la herramienta denominada “Real-Time-Workshop”, en la cual usted puede generar programas en código C desde el diagrama de bloques Simulink para prototiparlo e implementarlo en sistemas con tiempos reales. Matlab no solamente resulta ser óptimo para utilizar e implementar aplicaciones utilizando la lógica difusa, también es óptimo para aplicaciones con Redes Neuronales, para ello puede utilizar el módulo “Neuronal Network Toolbox”.

Definitivamente la lógica difusa y todos los programas basados en inteligencia artificial representar el futuro inmediato, de hecho hoy en día, especialmente en la industria, los programas están basados en inteligencia artificial.

RECOMENDACIONES

La persona que realice aplicaciones utilizando lógica difusa debe ser un técnico en la materia, pues involucra conceptos que en muchos de los casos resultan ser abstractos.

Se recomienda que antes de realizar un programa utilizando Lógica Difusa se tenga en cuenta los siguientes puntos.

Qué debe analizarse y cambiarse para poder realizar aplicaciones utilizando FUZZY LOGIC?

1. Analizar los datos para poder ver el comportamiento de los mismos y escoger una función de transferencia.
2. Bloques a utilizar: step, transfer fcn, scope, fuzzy logic, etc. Ver. (Diagrama de bloques Fig. 3.21)
3. Conectar secuencialmente step, transfer fcn y un scope (Reglas de inferencia) Pg. 29-30.
4. Bifurcar la salida de step hacia el otro scope (Retroalimentación)
5. Incorporar las reglas de inferencia (Es lo más importante), (Reglas If then)
6. Modificar los parámetros de la simulación hasta obtener los resultados deseados.

También recomendamos que antes de usar lógica difusa se haga un minucioso análisis del problema y un análisis previo del sistema que se desea elaborar para determinar si se debe o no aplicarla, esto dependiendo básicamente de las necesidades del usuario final.

Es básico que la persona que va a desarrollar el sistema se un experto para que se enmarque bien la problemática a solucionar para que así las reglas de inferencia fluyan

sin dificultad, pues estas son la base, o mas bien el éxito o fracaso del sistema, pues el sistema trabaja y toma decisiones basadas en estas reglas.

BIBLIOGRAFIA

PRADE, Dubois. Fuzzy sets in approximate reasoning II (Logical approaches), Fuzzy sets and systems. pp. 203-244, 1991, .

PRADE, Dubois;, Fuzzy sets and systems: Theory and applications. Editorial Academic Press. 1980.

GODO, Hájek, P; Deductive systems of fuzzy logic, unpublished manuscript, 1997.

KANTROWITZ, M. et al, FAQ. Fuzzy Logic and Fuzzy Expert Systems.
disponible en
<ftp.cs.cmu.edu:/user/ai/pubs/faqs/fuzzy/fuzzy.faq>, (desde 1995).

KAUFMANN, A, Introducción a la teoría de los subconjuntos borrosos. Cía. Editorial Continental, 1982.

ZADEH, L. Fuzzy sets, Information & Control., 8, 1965.

ZADEH, L. Fuzzy logic, IEEE Computer, 1988.

<http://fiobera.unam.edu.ar/Materias/ControlDigital/PID.html>

<http://www.keithley.cl/SERVICEnSUPPORT/literature=152.html>

LEIGH, Jr. Digital Control Applied. Editorial Prentice Hall EEUU. 1985.

KATSUHIKO Ogata. Ingeniería de control moderna. Tercera edición. Editorial Pearson Education. España. 1998.

CONTENIDO DEL CD

- 1.- Un Directorio (work) que contiene los archivos de la aplicación práctica
- 2.- Archivo "Logica difusa" contiene el texto de la monografía
- 3.- Archivo "Usuario" que contiene el detalle de las instrucciones para utilizar la aplicación práctica.

DENUNCIA DE TESIS

Certificado Romel Machado

Solicitud nuestra de aprobacion

Aprobacion de Luis Mario Cabrera

Diseño Tesis

