

UNIVERSIDAD AUTÓNOMA DE BAJA CALIFORNIA
FACULTAD DE CIENCIAS QUÍMICAS E INGENIERÍA
MAESTRÍA Y DOCTORADO EN CIENCIAS E INGENIERÍA



MODELACIÓN NEURO-GENÉTICO-DIFUSA TIPO-2:
ARQUITECTURAS, ANÁLISIS Y APLICACIONES

TESIS

QUE PARA OBTENER EL GRADO DE
DOCTOR EN CIENCIAS

P R E S E N T A:

JUAN RAMÓN CASTRO RODRÍGUEZ

DIRECTOR DE TESIS

DR. OSCAR CASTILLO LÓPEZ

CO-DIRECTOR DE TESIS

DR. ANTONIO RODRÍGUEZ DÍAZ

TIJUANA, BAJA CALIFORNIA

FEBRERO 2009

Universidad Autónoma de Baja California
FACULTAD DE CIENCIAS QUÍMICAS E INGENIERÍA
COORDINACIÓN DE POSGRADO E INVESTIGACIÓN

FOLIO No. 016

Tijuana, B. C., a 26 de enero de 2009

C. JUAN RAMÓN CASTRO RODRÍGUEZ
Pasante de: Doctor en Ciencias
Presente

El tema de trabajo y/o tesis para su examen profesional, en la
Opción TESIS

Es propuesto, por los C. Dres. Oscar Castillo López y Antonio Rodríguez Díaz
quienes serán los responsables de la calidad del trabajo que usted presente,
referido al tema MODELACIÓN NEURO-GENÉTICO-DIFUSA TIPO-2: ARQUITECTURAS,
ANÁLISIS Y APLICACIONES.

el cual deberá usted desarrollar, de acuerdo con el siguiente orden:

- I.- INTRODUCCIÓN
- II.- MODELACIÓN NEURO-GENÉTICO-DIFUSA TIPO-2
- III.- EXPERIMENTOS REALIZADOS
- IV.- DISCUSIÓN Y RESULTADOS
- V.- CONCLUSIONES
- VI.- REFERENCIAS

UNIVERSIDAD AUTÓNOMA
DE BAJA CALIFORNIA



FACULTAD DE CIENCIAS
QUÍMICAS E INGENIERÍA

Dr. Oscar Castillo López

Asesor

Dr. Antonio Rodríguez Díaz

Asesor

MC. Rubén Guillermo Sepúlveda Marqués

Sub-Director Secretario

MC. Eduardo Raymundo de la Trinidad Reyes Rodríguez

Director

© Copyright *Juan Ramón Castro Rodríguez*, Febrero 2009

Todos los derechos reservados

ÍNDICE

Página

AGRADECIMIENTOS	IV
DEDICATORIA	V
RESUMEN	VI
ABSTRACT	VIII
LISTA DE FIGURAS	X
LISTA DE TABLAS	XII
CAPÍTULO 1. INTRODUCCIÓN.....	1
1.1 CONTEXTO GENERAL.....	1
1.2 DESCRIPCIÓN DEL PROBLEMA.....	2
1.3 SOLUCIÓN PROPUESTA.....	8
1.4 APORTACIONES DEL TRABAJO	10
CAPÍTULO 2. MODELACIÓN NEURO-GENÉTICO-DIFUSA TIPO-2.....	12
2.1 DEFINICIÓN DEL MODELO HGA-IT2FNN.....	12
2.2 RED NEURO-DIFUSA TIPO-2 POR INTERVALOS (IT2FNN)	18
2.2.1 Arquitecturas IT2FNN.....	21
2.2.2 Algoritmo de aprendizaje híbrido heurístico numérico.....	45
2.2.3 Algoritmo de aprendizaje híbrido con HGA.....	53
2.2.4 Algoritmo de aprendizaje híbrido con PSO.....	55
2.3 APROXIMADOR UNIVERSAL.....	62
CAPÍTULO 3. EXPERIMENTOS REALIZADOS.....	67
3.1 CONDICIONES GENERALES DE LOS EXPERIMENTOS	67
3.2 EXPERIMENTOS DE IT2FNN.....	69
3.3 EXPERIMENTOS DE HGA-IT2FNN.....	78
3.4 EXPERIMENTOS DE PSO-IT2FNN.....	82
CAPÍTULO 4. DISCUSIÓN DE RESULTADOS	85
4.1 ANÁLISIS DE RESULTADOS OBTENIDOS CON IT2FNN.....	87
4.2 ANÁLISIS DE RESULTADOS OBTENIDOS CON HGA-IT2FNN.....	95
4.3 ANÁLISIS DE RESULTADOS OBTENIDOS CON PSO-IT2FNN.....	98
CAPÍTULO 5. CONCLUSIONES.....	101
REFERENCIAS.....	103
APÉNDICE	112

AGRADECIMIENTOS

Especialmente agradezco a los Doctores Oscar Castillo López y Elba Patricia Melín Olmeda por su ayuda incondicional, asesoramiento científico y estímulo emprendedor para seguir creciendo intelectualmente durante toda mi formación en el Posgrado.

Agradezco al Doctor Antonio Rodríguez Díaz por su apoyo, paciencia y orientación en las actividades de investigación de esta Tesis.

En especial agradezco a mis Co-Directores de Tesis, Dr. Oscar Castillo López y Antonio Rodríguez Díaz por compartir sus conocimientos y sus consejos para el desarrollo de este trabajo.

Mi agradecimiento y respeto a los Doctores Elba Patricia Melín Olmeda, Guillermo Licea Sandoval y Roberto Sepúlveda Cruz quienes formaron parte del comité revisor, por sus conocimientos y consejos.

A mis Maestros por compartir sus conocimientos y enseñanzas.

A mis amigos y compañeros en especial a: Guillermo Rodríguez Ventura, Héctor Salgado Herrera, Gabriela Lozano Olvera, María Evangelina Herrán Paz, Carmen Jauregui Romo, Olivia Mendoza Duarte, Luis Guillermo Martínez Méndez, Juan Emilio Moreno Pérez, Miguel Ángel López Ramírez, Salvador Valera Lamas, Miguel Alberto Leyva Montijo.

Al personal de la Coordinación de Posgrado e Investigación de la FCQI, UABC por todas las facilidades otorgadas en la estancia de mis estudios de Doctorado.

Al Consejo Nacional de Ciencia y Tecnología (CONACYT) por la Beca 203230 otorgada por la realización de mis estudios de Doctorado en Ciencias.

A la Universidad Autónoma de Baja California y a la División de Estudios de Posgrado e Investigación del Instituto Tecnológico de Tijuana, por el apoyo otorgado para concluir los estudios de Doctorado en Ciencias.

DEDICATORIA

A mi abuelita, Antonía Vargas Rentería[†], por su amor y cariño durante toda mi infancia.

A mis padres José Castro Vargas[†] y Catalina Rodríguez de la Cruz, por enseñarme el valor de la dedicación al estudio a lo largo de toda mi vida; por haber hecho de mí lo que soy, por quererme de la forma que lo hacen y mucho más de lo que hubiera podido desear en la vida.

Gracias por tenerme paciencia, por tener fe en mí y por hacerme saber que no importan los obstáculos, que siempre estarán a mi lado. Una vez más lo hemos logrado juntos y si de algo puedo estar seguro es que este triunfo es de Ustedes como mío.

A mis hijos por ser una de mis inspiraciones para iniciar, desarrollar y finalizar mis estudios de posgrado, por su apoyo y comprensión en todo momento de mis triunfos y tropiezos. Los amo.

A la mujer que esta a mi lado, por darme su amor, depositar en mí toda su confianza y tener la paciencia que has tenido.

A Dios, por darme energía, la sabiduría y el entendimiento necesario para lograr tan difícil meta.

RESUMEN

Se presenta una novedosa estrategia de integración homogénea de un *sistema de inferencia difuso tipo-2 por intervalos* (IT2FIS) con razonamiento de Takagi-Sugeno-Kang (IT2FIS:TSK), en una red neuronal (NN) adaptativa con aprendizaje híbrido heurístico numérico (IT2FNN:BP+RLS) y metaheurístico poblacional (HGA-IT2FNN+RLS y PSO-IT2FNN+RLS) para la obtención sistemática de un *sistema de lógica difusa tipo-2 por intervalos* (IT2FLS:TKS). En la primera aproximación, el IT2FIS:TSK y la NN, trabajan juntos en una arquitectura homogénea *neuro-difuso tipo-2 por intervalos* (IT2FNN), que puede ser interpretada como una NN adaptativa utilizando aprendizaje híbrido heurístico numérico. Los parámetros de los consecuentes se actualizan por el algoritmo de mínimos cuadrados recursivo (RLS); los antecedentes por el algoritmo de retro-propagación (BP). En la segunda aproximación se propone un modelo híbrido-cooperativo *neuro-genético-difuso tipo-2 por intervalos* (HGA-IT2FNN+RLS) que integra los métodos de razonamiento aproximado de los IT2FIS con las capacidades de aprendizaje de las NN y robustecido cooperativamente por un proceso de aprendizaje basado en *algoritmos genéticos jerárquicos* (HGAs), los cuales presentan la capacidad de explorar y explotar espacios de búsqueda complejos; además, la codificación genética que emplean les permite incorporar conocimiento a priori de una forma muy sencilla y aprovecharlo para guiar búsquedas globales. En la tercera aproximación se propone un modelo híbrido-cooperativo *neuro-swarm-difuso tipo-2 por intervalos* (PSO-IT2FNN+RLS) integrando los métodos IT2FIS con NN y robustecido cooperativamente por un proceso de aprendizaje basado en PSO. La estructura IT2FNN, HGA-IT2FNN y PSO-IT2FNN permite el uso de métodos cualitativos y cuantitativos en la construcción de modelos, en la etapa de aprendizaje,

funcionamiento y realimentación de conocimiento al humano, permitiendo así, todo el conocimiento disponible. Se presentan resultados de las aplicaciones de identificación de funciones no-lineales y pronóstico de la serie Caótica de Mackey-Glass ($\tau=13, 17, 30, 60, 100$) en presencia de diferentes valores de ruido SNR (razón señal a ruido). Se demuestra utilizando una base experimental y una serie de experimentos comparativos con análisis de validación cruzada, Criterio de Información de Akaike (*AIC*) y Prueba *F* que las arquitecturas de los modelos IT2FNN, HGA-IT2FNN y PSO-IT2FNN son robustas para manejar situaciones de incertidumbre, adaptabilidad, capacidad de aprendizaje y generalización, inherentes a modelar sistemas complejos reales.

ABSTRACT

This thesis develops a novel strategy for homogeneous integration of an Interval Type-2 Fuzzy Inference System (IT2FIS) with Takagi-Sugeno-Kang reasoning (TSK:IT2FIS), in an adaptive neural network (NN) with hybrid heuristic algorithm learning (IT2FNN:BP+RLS) and hybrid metaheuristic algorithm (HGA-IT2FNN+RLS and PSO-IT2FNN+RLS) for automatically generate an Interval Type-2 Fuzzy Logic System (TSK:IT2FLS). In a first approximation, TSK:IT2FIS and NN work together in a homogeneous Interval Type-2 Fuzzy Neural Network (IT2FNN) architecture, that can be interpreted as an adaptive NN using hybrid heuristic algorithm learning. Consequent parameters are updated by recursive least square (RLS) algorithm and antecedents by back-propagation (BP) algorithm. In a second approximation an Interval Type-2 Fuzzy Neural Network-Genetic (HGA-IT2FNN+RLS) cooperative hybrid model is proposed, that integrates IT2FIS approximate reasoning methods with NN's learning capabilities and, for robustness, collaboratively improved by a learning process based on Hierarchical Genetic Algorithms (HGAs), which provide searching and exploiting capabilities for complex domain spaces; also, genetic coding used by HGA enables them to add *a priori* knowledge for guiding global search in a very simple manner. In a third approximation a hybrid-cooperative Interval Type-2 Fuzzy Neural Network-SWARM model is proposed (PSO-IT2FNN+RLS) integrating IT2FIS and NN methods and collaborative improved by a learning process based on PSO. IT2FNN, HGA-IT2FNN and PSO-IT2FNN structures allow the use of qualitative and quantitative methods in model building, learning and human knowledge processing. Results from non-linear function identification and Time Series Chaotic Mackey-

Glass ($\tau=13, 17, 30, 60, 100$) with SNR noise cases are discussed. With these results, a comparison using cross-validation analysis, Akaike Information Criteria (AIC) and F test, it is shown that IT2FNN, HGA-IT2FNN and PSO-IT2FNN are robust for handling uncertainty, adaptability, learning and generalization, inherent for complex systems modelling.

LISTA DE FIGURAS

	Página
FIGURA 2.1 INTERVAL TYPE-1 FUZZY NEURON (IT1FN).....	20
FIGURA 2.2 INTERVAL TYPE-2 FUZZY NEURON (IT2FN)	21
FIGURA 2.3 INTERVAL TYPE-1 FUZZY NEURAL NETWORK (IT1FNN).....	21
FIGURA 2.4 ARQUITECTURA IT2FNN-1:A2C0.....	24
FIGURA 2.5 ARQUITECTURA IT2FNN-2:A2C0.....	28
FIGURA 2.6 ARQUITECTURA RBIT2FNN-2:A2C0.....	30
FIGURA 2.7 ARQUITECTURA IT2FNN-1:A2C1.....	32
FIGURA 2.8 ARQUITECTURA IT2FNN-2:A2C1.....	35
FIGURA 2.9 ARQUITECTURA IT2FNN-2A:A2C1	36
FIGURA 2.10 ARQUITECTURA RBIT2FNN-2:A2C1	38
FIGURA 2.11 ARQUITECTURA RBIT2FNN-2A:A2C1	39
FIGURA 2.12 ARQUITECTURA IT2FNN-3:A2C1	41
FIGURA 2.13 ARQUITECTURA IT2FNN-3A:A2C1	42
FIGURA 2.14 ARQUITECTURA RBIT2FNN-3:A2C1	44
FIGURA 2.15 ESTRUCTURA JERÁRQUICA DE UN CROMOSOMA PARA CODIFICAR UN HGA:IT2FLS	54
FIGURA 3.1 VALORES RMSE DE LOS MODELOS ANFIS Y IT2FNN EVALUADOS CON VALIDACIÓN CRUZADA PARA LA IDENTIFICACIÓN NO LINEAL DEL EXPERIMENTO 1.	70
FIGURA 3.2 VALORES RMSE DE LOS MODELOS ANFIS Y IT2FNN EVALUADOS CON VALIDACIÓN CRUZADA PARA LA IDENTIFICACIÓN NO LINEAL DEL EXPERIMENTO 2.	71
FIGURA 3.3 VALORES RMSE DETERMINADOS EN LOS MODELOS ANFIS Y IT2FNN PARA LA IDENTIFICACIÓN NO-LINEAL DEL EXPERIMENTO 3, CON VALIDACIÓN CRUZADA.	72
FIGURA 3.4 VALORES RMSE (TRN) DETERMINADOS EN LOS MODELOS ANFIS Y IT2FNN CON VALIDACIÓN CRUZADA PARA EL PRONÓSTICO DE LA SERIE DE TIEMPO CAÓTICA DE MACKEY-GLASS CON $\tau=13$	73
FIGURA 3.5 VALORES RMSE (CHK) DETERMINADOS EN LOS MODELOS ANFIS Y IT2FNN CON VALIDACIÓN CRUZADA PARA EL PRONÓSTICO DE LA SERIE DE TIEMPO CAÓTICA DE MACKEY-GLASS CON $\tau=13$	74
FIGURA 3.6 VALORES RMSE (TRN) DETERMINADOS EN LOS MODELOS ANFIS Y IT2FNN CON VALIDACIÓN CRUZADA PARA EL PRONÓSTICO DE LA SERIE DE TIEMPO CAÓTICA DE MACKEY-GLASS CON $\tau=17$	74
FIGURA 3.7 VALORES RMSE (CHK) DETERMINADOS EN LOS MODELOS ANFIS Y IT2FNN CON VALIDACIÓN CRUZADA PARA EL PRONÓSTICO DE LA SERIE DE TIEMPO CAÓTICA DE MACKEY-GLASS CON $\tau=17$	75

FIGURA 3.8	VALORES RMSE (TRN) DETERMINADOS EN LOS MODELOS ANFIS Y IT2FNN CON VALIDACIÓN CRUZADA PARA EL PRONÓSTICO DE LA SERIE DE TIEMPO CAÓTICA DE MACKEY-GLASS CON $\tau=30$.	75
FIGURA 3.9	VALORES RMSE (CHK) DETERMINADOS EN LOS MODELOS ANFIS Y IT2FNN CON VALIDACIÓN CRUZADA PARA EL PRONÓSTICO DE LA SERIE DE TIEMPO CAÓTICA DE MACKEY-GLASS CON $\tau=30$.	76
FIGURA 3.10	VALORES RMSE (TRN) DETERMINADOS EN LOS MODELOS ANFIS Y IT2FNN CON VALIDACIÓN CRUZADA PARA EL PRONÓSTICO DE LA SERIE DE TIEMPO CAÓTICA DE MACKEY-GLASS CON $\tau=60$.	76
FIGURA 3.11	VALORES RMSE (CHK) DETERMINADOS EN LOS MODELOS ANFIS Y IT2FNN CON VALIDACIÓN CRUZADA PARA EL PRONÓSTICO DE LA SERIE DE TIEMPO CAÓTICA DE MACKEY-GLASS CON $\tau=60$.	77
FIGURA 3.12	VALORES RMSE (TRN) DETERMINADOS EN LOS MODELOS ANFIS Y IT2FNN CON VALIDACIÓN CRUZADA PARA EL PRONÓSTICO DE LA SERIE DE TIEMPO CAÓTICA DE MACKEY-GLASS CON $\tau=100$.	77
FIGURA 3.13	VALORES RMSE (CHK) DETERMINADOS EN LOS MODELOS ANFIS Y IT2FNN CON VALIDACIÓN CRUZADA PARA EL PRONÓSTICO DE LA SERIE DE TIEMPO CAÓTICA DE MACKEY-GLASS CON $\tau=100$.	78
FIGURA 3.14	VALORES RMSE DE LOS MODELOS HGA-ANFIS Y HGA-IT2FNN EVALUADOS CON VALIDACIÓN CRUZADA PARA LA IDENTIFICACIÓN NO LINEAL DEL EXPERIMENTO 1.	79
FIGURA 3.15	VALORES RMSE DE LOS MODELOS HGA-ANFIS Y HGA-IT2FNN EVALUADOS CON VALIDACIÓN CRUZADA PARA LA IDENTIFICACIÓN NO LINEAL DEL EXPERIMENTO 2.	80
FIGURA 3.16	VALORES RMSE DE LOS MODELOS HGA-ANFIS Y HGA-IT2FNN EVALUADOS CON VALIDACIÓN CRUZADA PARA LA IDENTIFICACIÓN NO LINEAL DEL EXPERIMENTO 3.	81
FIGURA 3.17	VALORES RMSE DE LOS MODELOS PSO-ANFIS Y PSO-IT2FNN EVALUADOS CON VALIDACIÓN CRUZADA PARA LA IDENTIFICACIÓN NO LINEAL DEL EXPERIMENTO 1.	82
FIGURA 3.18	VALORES RMSE DE LOS MODELOS PSO-ANFIS Y PSO-IT2FNN EVALUADOS CON VALIDACIÓN CRUZADA PARA LA IDENTIFICACIÓN NO LINEAL DEL EXPERIMENTO 2.	83
FIGURA 3.19	VALORES RMSE DE LOS MODELOS PSO-ANFIS Y PSO-IT2FNN EVALUADOS CON VALIDACIÓN CRUZADA PARA LA IDENTIFICACIÓN NO LINEAL DEL EXPERIMENTO 3.	84

LISTA DE TABLAS

		Página
TABLA 2.1	TÉCNICAS DE OPTIMIZACIÓN HEURÍSTICA NUMÉRICA.....	46
TABLA 2.2	PROCEDIMIENTO DE APRENDIZAJE DE LA RED ADAPTATIVA IT2FNN.....	47
TABLA 3.1	VALORES RMSE DE LOS MODELOS ANFIS Y IT2FNN CON VALIDACIÓN CRUZADA PARA LA IDENTIFICACIÓN NO-LINEAL DEL EXPERIMENTO 1.....	70
TABLA 3.2	VALORES RMSE DE LOS MODELOS ANFIS Y IT2FNN CON VALIDACIÓN CRUZADA PARA LA IDENTIFICACIÓN NO-LINEAL DEL EXPERIMENTO 2.....	71
TABLA 3.3	VALORES RMSE DETERMINADOS EN LOS MODELOS ANFIS Y IT2FNN PARA LA IDENTIFICACIÓN NO-LINEAL DEL EXPERIMENTO 3, CON VALIDACIÓN CRUZADA.....	72
TABLA 3.4	VALORES RMSE (TRN/CHK) Y ζ DETERMINADOS EN LOS MODELOS ANFIS Y IT2FNN CON VALIDACIÓN CRUZADA PARA EL PRONÓSTICO DE LA SERIE DE TIEMPO CAÓTICA DE MACKEY-GLASS CON $\tau=13$	73
TABLA 3.5	VALORES RMSE (TRN/CHK) Y ζ DETERMINADOS EN LOS MODELOS ANFIS Y IT2FNN CON VALIDACIÓN CRUZADA PARA EL PRONÓSTICO DE LA SERIE DE TIEMPO CAÓTICA DE MACKEY-GLASS CON $\tau=17$	74
TABLA 3.6	VALORES RMSE (TRN/CHK) Y ζ DETERMINADOS EN LOS MODELOS ANFIS Y IT2FNN CON VALIDACIÓN CRUZADA PARA EL PRONÓSTICO DE LA SERIE DE TIEMPO CAÓTICA DE MACKEY-GLASS CON $\tau=30$	75
TABLA 3.7	VALORES RMSE (TRN/CHK) Y ζ DETERMINADOS EN LOS MODELOS ANFIS Y IT2FNN CON VALIDACIÓN CRUZADA PARA EL PRONÓSTICO DE LA SERIE DE TIEMPO CAÓTICA DE MACKEY-GLASS CON $\tau=60$	76
TABLA 3.8	VALORES RMSE (TRN/CHK) Y ζ DETERMINADOS EN LOS MODELOS ANFIS Y IT2FNN CON VALIDACIÓN CRUZADA PARA EL PRONÓSTICO DE LA SERIE DE TIEMPO CAÓTICA DE MACKEY-GLASS CON $\tau=100$	77
TABLA 3.9	VALORES RMSE DE LOS MODELOS HGA-ANFIS Y HGA-IT2FNN CON VALIDACIÓN CRUZADA PARA LA IDENTIFICACIÓN NO-LINEAL DEL EXPERIMENTO 1.....	79
TABLA 3.10	VALORES RMSE DE LOS MODELOS HGA-ANFIS Y HGA-IT2FNN CON VALIDACIÓN CRUZADA PARA LA IDENTIFICACIÓN NO-LINEAL DEL EXPERIMENTO 2.....	80
TABLA 3.11	VALORES RMSE DE LOS MODELOS HGA-ANFIS Y HGA-IT2FNN CON VALIDACIÓN CRUZADA PARA LA IDENTIFICACIÓN NO-LINEAL DEL EXPERIMENTO 3.....	81
TABLA 3.12	VALORES RMSE DE LOS MODELOS PSO-ANFIS Y PSO-IT2FNN CON VALIDACIÓN CRUZADA PARA LA IDENTIFICACIÓN NO-LINEAL DEL EXPERIMENTO 1.....	82
TABLA 3.13	VALORES RMSE DE LOS MODELOS PSO-ANFIS Y PSO-IT2FNN CON VALIDACIÓN CRUZADA PARA LA IDENTIFICACIÓN NO-LINEAL DEL EXPERIMENTO 2.....	83
TABLA 3.14	VALORES RMSE DE LOS MODELOS PSO-ANFIS Y PSO-IT2FNN CON VALIDACIÓN CRUZADA PARA LA IDENTIFICACIÓN NO-LINEAL DEL EXPERIMENTO 3.....	84

TABLA 4.1	VALORES DEL CRITERIO DE INFORMACIÓN DE AKAIKE (AIC) PARA LOS MODELOS ANFIS Y IT2FNN DEL EXPERIMENTO 1.	87
TABLA 4.2	VALORES ESTADÍSTICOS F PARA LOS MODELOS ANFIS/IT2FNN DEL EXPERIMENTO 1.	87
TABLA 4.3	VALORES DEL CRITERIO DE INFORMACIÓN DE AKAIKE (AIC) PARA LOS MODELOS ANFIS Y IT2FNN DEL EXPERIMENTO 2.	88
TABLA 4.4	VALORES ESTADÍSTICOS F PARA LOS MODELOS ANFIS/IT2FNN DEL EXPERIMENTO 2.	88
TABLA 4.5	VALORES DEL CRITERIO DE INFORMACIÓN DE AKAIKE (AIC) PARA LOS MODELOS ANFIS Y IT2FNN DEL EXPERIMENTO 3.	89
TABLA 4.6	VALORES ESTADÍSTICOS F PARA LOS MODELOS ANFIS/IT2FNN DEL EXPERIMENTO 3.	89
TABLA 4.7	VALORES DEL CRITERIO DE INFORMACIÓN DE AKAIKE (AIC) DE TRN/CHK Y ζ DETERMINADOS EN LOS MODELOS ANFIS Y IT2FNN CON VALIDACIÓN CRUZADA PARA EL PRONÓSTICO DE LA SERIE DE TIEMPO CAÓTICA DE MACKEY-GLASS CON $\tau=13$	90
TABLA 4.8	VALORES ESTADÍSTICOS F (TRN/CHK) Y ζ DETERMINADOS EN LOS MODELOS ANFIS Y IT2FNN CON VALIDACIÓN CRUZADA PARA EL PRONÓSTICO DE LA SERIE DE TIEMPO CAÓTICA DE MACKEY-GLASS CON $\tau=13$	90
TABLA 4.9	VALORES DEL CRITERIO DE INFORMACIÓN DE AKAIKE (AIC) DE TRN/CHK Y ζ DETERMINADOS EN LOS MODELOS ANFIS Y IT2FNN CON VALIDACIÓN CRUZADA PARA EL PRONÓSTICO DE LA SERIE DE TIEMPO CAÓTICA DE MACKEY-GLASS CON $\tau=17$	91
TABLA 4.10	VALORES ESTADÍSTICOS F (TRN/CHK) Y ζ DETERMINADOS EN LOS MODELOS ANFIS Y IT2FNN CON VALIDACIÓN CRUZADA PARA EL PRONÓSTICO DE LA SERIE DE TIEMPO CAÓTICA DE MACKEY-GLASS CON $\tau=17$	91
TABLA 4.11	VALORES DEL CRITERIO DE INFORMACIÓN DE AKAIKE (AIC) DE TRN/CHK Y ζ DETERMINADOS EN LOS MODELOS ANFIS Y IT2FNN CON VALIDACIÓN CRUZADA PARA EL PRONÓSTICO DE LA SERIE DE TIEMPO CAÓTICA DE MACKEY-GLASS CON $\tau=30$	92
TABLA 4.12	VALORES ESTADÍSTICOS F (TRN/CHK) Y ζ DETERMINADOS EN LOS MODELOS ANFIS Y IT2FNN CON VALIDACIÓN CRUZADA PARA EL PRONÓSTICO DE LA SERIE DE TIEMPO CAÓTICA DE MACKEY-GLASS CON $\tau=30$	92
TABLA 4.13	VALORES DEL CRITERIO DE INFORMACIÓN DE AKAIKE (AIC) DE TRN/CHK Y ζ DETERMINADOS EN LOS MODELOS ANFIS Y IT2FNN CON VALIDACIÓN CRUZADA PARA EL PRONÓSTICO DE LA SERIE DE TIEMPO CAÓTICA DE MACKEY-GLASS CON $\tau=60$	93
TABLA 4.14	VALORES ESTADÍSTICOS F (TRN/CHK) Y ζ DETERMINADOS EN LOS MODELOS ANFIS Y IT2FNN CON VALIDACIÓN CRUZADA PARA EL PRONÓSTICO DE LA SERIE DE TIEMPO CAÓTICA DE MACKEY-GLASS CON $\tau=60$	93
TABLA 4.15	VALORES DEL CRITERIO DE INFORMACIÓN DE AKAIKE (AIC) DE TRN/CHK Y ζ DETERMINADOS EN LOS MODELOS ANFIS Y IT2FNN CON VALIDACIÓN CRUZADA PARA EL PRONÓSTICO DE LA SERIE DE TIEMPO CAÓTICA DE MACKEY-GLASS CON $\tau=100$	94
TABLA 4.16	VALORES ESTADÍSTICOS F (TRN/CHK) Y ζ DETERMINADOS EN LOS MODELOS ANFIS Y IT2FNN CON VALIDACIÓN CRUZADA PARA EL PRONÓSTICO DE LA SERIE DE TIEMPO CAÓTICA DE MACKEY-GLASS CON $\tau=100$	94
TABLA 4.17	VALORES DEL CRITERIO DE INFORMACIÓN DE AKAIKE (AIC) PARA LOS MODELOS HGA-ANFIS Y HGA-IT2FNN DEL EXPERIMENTO 1.	95
TABLA 4.18	VALORES ESTADÍSTICOS F PARA LOS MODELOS HGA:ANFIS/IT2FNN DEL EXPERIMENTO 1.	95
TABLA 4.19	VALORES DEL CRITERIO DE INFORMACIÓN DE AKAIKE (AIC) PARA LOS MODELOS HGA-ANFIS Y HGA-IT2FNN DEL EXPERIMENTO 2.	96

TABLA 4.20	VALORES ESTADÍSTICOS F PARA LOS MODELOS HGA:ANFIS/IT2FNN-0 DEL EXPERIMENTO 2.....	96
TABLA 4.21	VALORES DEL CRITERIO DE INFORMACIÓN DE AKAIKE (AIC) PARA LOS MODELOS HGA-ANFIS Y HGA-IT2FNN DEL EXPERIMENTO 3.....	97
TABLA 4.22	VALORES ESTADÍSTICOS F PARA LOS MODELOS HGA:ANFIS/IT2FNN DEL EXPERIMENTO 3.....	97
TABLA 4.23	VALORES DEL CRITERIO DE INFORMACIÓN DE AKAIKE (AIC) PARA LOS MODELOS PSO-ANFIS Y PSO-IT2FNN DEL EXPERIMENTO 1.....	98
TABLA 4.24	VALORES ESTADÍSTICOS F PARA LOS MODELOS PSO:ANFIS/IT2FNN DEL EXPERIMENTO 1.....	98
TABLA 4.25	VALORES DEL CRITERIO DE INFORMACIÓN DE AKAIKE (AIC) PARA LOS MODELOS PSO-ANFIS Y PSO-IT2FNN DEL EXPERIMENTO 2.....	99
TABLA 4.26	VALORES ESTADÍSTICOS F PARA LOS MODELOS PSO:ANFIS/IT2FNN-0 DEL EXPERIMENTO 2.....	99
TABLA 4.27	VALORES DEL CRITERIO DE INFORMACIÓN DE AKAIKE (AIC) PARA LOS MODELOS PSO-ANFIS Y PSO-IT2FNN DEL EXPERIMENTO 3.....	100
TABLA 4.28	VALORES ESTADÍSTICOS F PARA LOS MODELOS PSO:ANFIS/IT2FNN DEL EXPERIMENTO 3.....	100

CAPÍTULO 1. INTRODUCCIÓN

1.1 Contexto general

En este trabajo de Tesis se aborda la problemática de cómo tratar la información imperfecta utilizando modelación híbrida neuro-genética difusa tipo-2 por intervalos, como metodología de computación suave (*Soft Computing*), haciendo de este modo que las debilidades de las redes neuronales adaptativas, algoritmos genéticos y sistemas de lógica difusa tipo-2 por intervalos, se compensen con las bondades sobresalientes de cada modelo individual. El trabajo se divide en cinco capítulos y un apéndice, a continuación se resumen los contenidos de cada uno.

El capítulo uno corresponde a esta introducción, presentaremos los conceptos básicos sobre los que se fundamentan el trabajo de investigación realizado. Comenzaremos presentando el entorno de la problemática que se pretende resolver, seguido de las alternativas de solución que se proponen en esta investigación.

En el capítulo dos se presentan las principales aportaciones de la tesis. En primer lugar se describen las arquitecturas híbridas neurodifusas tipo-2 por intervalos (IT2FNN), una nueva manera de integrar un sistema de inferencia difusa tipo-2 por intervalos (IT2FIS) en una red neuronal adaptativa, la cual hace uso de información imprecisa afectada con incertidumbre. En segundo lugar se describen los métodos de aprendizaje híbrido basado en gradientes, algoritmos genéticos jerárquicos (HGA) y optimización con enjambre de partículas (PSO).

El capítulo tres describe dos aplicaciones de modelación de sistemas, que hacen uso de las arquitecturas de los modelos IT2FNN, HGA-IT2FNN y PSO-IT2FNN expuestos en el capítulo dos; en primer lugar se describen las condiciones generales de los experimentos para la

identificación de funciones no-lineales, y en segundo lugar se muestran los resultados de los modelos para la predicción de la serie de Mackey-Glass afectada con niveles de ruido.

En el capítulo cuatro se hace un análisis de los resultados con validación cruzada, Criterio de Información de Akaike, prueba F para validar los modelos IT2FNN, HGA-IT2FNN y PSO-IT2FNN. Los resultados del análisis se organizan en tablas comparativas entre arquitecturas y algoritmos de aprendizaje.

En el capítulo cinco, se presentan las conclusiones y aportaciones del presente trabajo. Se propondrán también algunas líneas de trabajo futuras.

En el apéndice A se documentan algunos desarrollos auxiliares del capítulo dos.

1.2 Descripción del problema

En general la modelación de sistemas es, uno de los problemas de mayor interés en numerosas ramas de la ciencia. Los modelos son abstracciones de la realidad que pueden ser aplicados para mejorar nuestra comprensión de fenómenos en el mundo real. En diversas áreas de aplicación, como la economía, ingeniería, biología, sociología y medicina, surge el problema de establecer una relación funcional entre las diferentes variables que intervienen en el fenómeno que se está estudiando. Particularmente la aproximación de una función desconocida a partir de un conjunto de muestras experimentales ha sido y sigue siendo, una parte fundamental en un gran número de disciplinas científicas e industriales. Un problema típico de modelado de funciones de gran importancia en diversas áreas, es la predicción de series de tiempo. En este caso se dispone de una secuencia de muestras en el tiempo y se pretende predecir el comportamiento de dicha serie de muestras a corto o largo plazo. Este problema se suele tratar como un problema de modelado donde se pretenden obtener muestras futuras (salida) en función de muestras pasadas (entradas).

Cuando el ser humano trata de describir los fenómenos naturales o hacer inferencias a partir de su estudio, se ve enfrentado a diversos tipos de incertidumbre originadas por la información imperfecta, por los modelos usados para hacer inferencias o por la interpretación dada a esos modelos. Las técnicas e instrumentos utilizados en la recolección de información no son perfectas de ahí que se tengan datos imperfectos. Se pueden tener datos imperfectos porque son tomados con equipos que tienen una limitada gradación en las unidades de medición. Así mismo, la imperfección puede asociarse con lecturas deficientes hechas por los encargados de las mediciones. Otras veces la naturaleza del fenómeno estudiado genera lecturas fluctuantes o variables, originando datos inciertos. También las características o la mala calibración de los equipos y los errores humanos pueden generar datos inexactos o errados [1,2,6].

La imperfección en los datos es debida a la incertidumbre, la imprecisión, la vaguedad y a la inconsistencia de los datos. La incertidumbre se debe a la falta de información acerca del verdadero estado de la naturaleza, lo que impide determinar si ciertas proposiciones son ciertas o falsas. La imprecisión es la causa de la granularidad del lenguaje. La vaguedad como una nueva categoría de información imperfecta. La inconsistencia es originada por la redundancia de los datos que generan conflictos o contradicciones y no se podría estar seguro sobre cual de los datos que se encuentran en conflicto es cierto y cuales no. Normalmente el nivel de incertidumbre varía dependiendo del contexto y problema a tratar. Esta imprecisión o incertidumbre puede venir dada por la inclusión de ruido en los datos [3,4,107,108].

Numerosos problemas de la vida cotidiana se pueden plantear en términos de cantidades numéricas precisas, o con imprecisión cuantificable de forma natural; para tales problemas, asimismo, se espera una respuesta igualmente en términos numéricos. No obstante, existen otros problemas en los que, si bien se puede proporcionar una descripción numérica, incluso con

determinado grado de incertidumbre, resulta mas natural el empleo de términos lingüísticos. Las descripciones lingüísticas están constituidas por palabras, y estas, tal y como las percibimos las personas, tienen connotaciones difusas; en muchas ocasiones no esta claro donde empieza y donde termina su significado. Incluso en palabras que a primera vista parecen unívocamente definidas puede aparecer cierta dosis de ambigüedad. La aproximación lingüística, aun siendo menos precisa que la numérica, presenta algunas ventajas: La descripción lingüística es fácilmente entendida por los seres humanos, incluso cuando los conceptos son abstractos o el contexto es cambiante. Mas aun, disminuye el efecto del ruido, ya que cuanto mas refinada es la escala con la que trabajemos, mas sensible es esta al ruido [1,6,37,103,104,106].

Los sistemas de lógica difusa tipo-1 (T1FLS) y tipo-2 por intervalos (IT2FLS), combinados con técnicas como las redes neuronales (NN) y los algoritmos genéticos (GA) han demostrado su habilidad para resolver diferentes tipos de problemas tales como control [22-24,35,36,40,64,76,96,98], predicción [24, 41,45,57], procesamiento de señales e imágenes [24, 26,34,42,66,68,75], procesos industriales [49,54,55], etc. En muchos de los casos, la clave del éxito era la incorporación del conocimiento experto humano. En la última década, ha existido un gran interés en incluir capacidad de aprendizaje a los IT2FLS [50-54, 56, 58-63, 67]. Esto se ha logrado por medio del desarrollo de técnicas híbridas que incluyen los IT2FLS junto con técnicas complementarias como lo son las redes neuronales, los algoritmos evolutivos o los métodos probabilísticos [39, 50-53, 55, 56, 60-62, 68, 72, 74, 75, 84, 91,95].

Existe consenso entre los investigadores en que se pueden obtener sistemas más inteligentes por medio de la hibridación de metodologías de computación suave [72, 93, 94, 101], haciendo de este modo que las debilidades de unos sistemas se compensen con las bondades de otros. Los *sistemas neurodifusos* tipo-1 (T1FNN) [10, 13, 14, 15-21, 92, 93] y tipo-2 (IT2FNN) [50-56, 60-

63]; los *sistemas evolutivos difusos* tipo-1 [71, 77, 79, 81, 85] y tipo-2 [72-75, 76, 84] constituyen la más notoria representatividad de sistemas híbridos dentro de computación suave. Estos sistemas mezclan los métodos de razonamiento generalizado de los T1FLS [69, 93, 96, 98, 100-104] y IT2FLS [27-33, 43-49] con las capacidades de aprendizaje de las *redes neuronales* [93, 97, 105, 109] y los *algoritmos evolutivos* [78, 80, 82, 83, 91, 97], respectivamente.

Los sistemas IT2FNN combinan los sistemas de inferencia difusos tipo-2 por intervalos (IT2FIS) y las redes neuronales, aprovechando las características sobresalientes de cada modelo. Los IT2FLS proveen un mecanismo intuitivo y de alto nivel para representar el conocimiento mediante la utilización de reglas IF-THEN. Son una gran herramienta para manejar situaciones de incertidumbre inherentes a sistemas complejos. Las redes neuronales poseen un alto grado de adaptabilidad y capacidad de aprendizaje y generalización. La construcción de herramientas que se nutren de estas dos áreas ha demostrado ser un mecanismo eficiente a la hora de modelar sistemas reales.

Un sistema evolutivo difuso tipo-2 por intervalos [72-75] es básicamente un IT2FLS robustecido por un proceso de aprendizaje basado en un algoritmo evolutivo [91], en particular los *algoritmos genéticos* [80-83, 91], los cuales están considerados actualmente como la técnica de búsqueda global más conocida y empleada. Este tipo de algoritmos presentan la capacidad de explorar y explotar espacios de búsqueda complejos, lo que les permite obtener soluciones muy próximas a la óptima en dichos espacios. Además, la codificación genética que emplean les permite incorporar conocimiento a priori de una forma muy sencilla y aprovecharlo para guiar la búsqueda.

Recientemente, ha aparecido una nueva familia de algoritmos bioinspirados, para resolver una amplia gama de problemas de optimización [91]. La optimización con enjambre de partículas,

más conocida en la literatura científica como Particle Swarm Optimization (PSO) [88-91], nace, al igual que otras técnicas estocásticas de cálculo evolutivo, en un intento por imitar y mimetizar el comportamiento de procesos naturales. El PSO junto con la optimización con una colonia de hormigas o Ant Colony Optimization (ACO) [86, 87, 91] son los dos métodos más populares y utilizados en el área de la inteligencia computacional. Ambos tratan de imitar los comportamientos sociales de un colectivo a partir de la interacción de los individuos entre sí y con el entorno.

En forma genérica, existen dos maneras de abordar el problema de modelación. La primera es de carácter deductivo. Básicamente, el método consiste en la división del sistema en subsistemas que son modelados a partir de leyes físicas y relaciones previamente aceptadas como hipótesis, usando como herramienta fundamental las ecuaciones diferenciales. La segunda aproximación es de naturaleza inductiva, ya que realiza la estimación de los modelos a partir del análisis y medida de los datos. Este proceso de estimación se denomina comúnmente aprendizaje. En este contexto, aprendizaje significa obtener una representación de los datos que pueda ser usada para diferentes objetivos [3, 10, 44, 45, 57, 68, 69, 93].

Una tarea importante en el área de modelación y diseño de IT2FLS, es proveer una metodología para el desarrollo de modelos difusos tipo-2 por intervalos, es decir, la obtención sistemática de un modelo difuso tipo-2 por intervalos a partir del conocimiento del sistema real que se quiere modelar. Los primeros intentos utilizaban el denominado enfoque directo, en el cual se utiliza fuertemente el conocimiento de expertos humanos, extrayendo de ellos las reglas lingüísticas y funciones de membresía necesarias para construir los sistemas de inferencia difuso tipo-2 por intervalos [5, 22, 25, 26, 35, 41, 44]. Desafortunadamente, este esquema presenta varias dificultades, ya que el conocimiento de expertos humanos es frecuentemente incompleto,

subjetivo y no sistemático. Es por ello que últimamente se han desarrollado mecanismos tendientes a automatizar la construcción del modelo difuso tipo-2 por intervalos. En general no se conoce el funcionamiento interno del sistema real, y la información que se posee es un conjunto de pares de entrada-salida observados, denominado conjunto de entrenamiento. Cada elemento de este conjunto se denomina patrón de entrenamiento. A partir, entonces, del conjunto de entrenamiento, la construcción del modelo difuso tipo-2 por intervalos se puede dividir en dos etapas. En la primera se determina su estructura, encontrando un conjunto de reglas y una partición del espacio de entrada que se adecue al sistema real. En la segunda etapa se identifican los parámetros (antecedentes, reducción de tipo, consecuentes) que describen mas precisamente al sistema modelado aplicando para ello variadas técnicas de optimización numérica [6, 44, 50-57, 60-68], evolutivas [72-76, 84, 91] y bioinspiradas [91].

En este trabajo de tesis se presentan tres arquitecturas híbridas neurodifusas tipo-2 por intervalos (IT2FNN) o bien tres estrategias de integración homogénea de un sistema de inferencia difuso tipo-2 por intervalos (IT2FIS) con razonamiento de Takagi-Sugeno-Kang (IT2FIS:TSK), en una NN adaptativa. En la primera aproximación, el IT2FIS:TSK y la NN, trabajan juntos en una arquitectura homogénea que puede ser interpretada como una NN adaptativa con parámetros difusos o como un IT2FIS:TSK con parámetros o funcionamiento distribuidos, utilizando aprendizaje híbrido heurístico numérico (IT2FNN:BP+RLS). Los parámetros de los consecuentes se actualizan por el algoritmo de mínimos cuadrados recursivo (RLS); los antecedentes por el algoritmo de retro-propagación (BP). En la segunda aproximación se propone un modelo cooperativo IT2FNN con algoritmos de aprendizaje metaheurísticos poblacionales (HGA-IT2FNN+RLS y PSO-IT2FNN+RLS); para la obtención sistemática de un IT2FIS:TSK. En el caso particular de HGA-IT2FNN+RLS, se usa un algoritmo genético

jerárquico (HGA) para determinar la estructura, conjunto de reglas, partición del espacio de entrada, parámetros de los antecedentes, consecuentes y reducción de tipo del IT2FIS:TSK. Similarmente para el PSO-IT2FNN+RLS se usa un PSO para determinar los antecedentes y un RLS para los consecuentes a partir del conocimiento del sistema real que se quiere modelar.

1.3 Solución propuesta

En este trabajo, se proponen tres arquitecturas híbridas neurodifusas tipo-2 por intervalos (IT2FNN-1, IT2FNN-2 y IT2FNN-3) con aprendizaje híbrido heurístico numérico (IT2FNN:BP+RLS) y aprendizaje híbrido metaheurístico poblacional (HGA-IT2FNN+RLS y PSO-IT2FNN+RLS), para la obtención sistemática de un modelo IT2FLS:TSK. El algoritmo de aprendizaje híbrido heurístico numérico (IT2FNN:BP+RLS) actualiza los parámetros de los antecedentes en base a técnicas de optimización numérica empleadas para el entrenamiento de NN [93, 105, 109]. Los parámetros de los consecuentes se actualizan con el algoritmo RLS [92, 93].

El algoritmo de aprendizaje híbrido metaheurístico poblacional (HGA-IT2FNN+RLS), auto-adapta los parámetros (consecuentes y reducción de tipo) y las reglas usando un HGA [70, 91]. Los parámetros de los consecuentes se actualizan en base a la técnica RLS. En el algoritmo de aprendizaje híbrido PSO-IT2FNN+RLS auto-adapta los parámetros (consecuentes y reducción de tipo) en base a un algoritmo PSO [91]. En el caso que la tolerancia no se alcance se activa el algoritmo híbrido heurístico numérico IT2FNN:BP+RLS.

La primera arquitectura (IT2FNN-1) es una red neuronal adaptativa que contiene dos tipos de nodos: adaptativos y no-adaptativos. Los nodos adaptativos, y sus salidas dependen, no solo de sus entradas, sino también de parámetros modificables internos al nodo; por el contrario, los nodos no adaptativos son función únicamente de las entradas. El proceso de reducción de tipo usa el

algoritmo de KM [30, 31, 38]. La segunda arquitectura (IT2FNN-2) es una red neuronal adaptativa con neuronas difusas tipo-1 adaptativas (salidas difusas, entradas y pesos sinápticos precisos), neuronas difusas tipo-1 no adaptativas (dependen únicamente de entradas difusas) y neuronas adaptativas lineales (consecuentes de las reglas). En esta arquitectura los elementos de procesamiento (neuronas difusas tipo-1) de la capa-1, se interconectan en pares para formar neuronas difusas tipo-2 por intervalos (IT2FN) en la capa-2. Asimismo en la capa de los consecuentes cada par de neuronas adaptativas lineales integran una neurona difusa tipo-1 por intervalos. El proceso de reducción de tipo es similar a IT2FNN-1. La tercera arquitectura (IT2FNN-3) es similar a la segunda, excepto que el proceso de reducción de tipo, se integra en las capas de la red adaptativa. Las neuronas difusas tipo-2 por intervalos y tipo-1 por intervalos son continuas y diferenciables. Las arquitecturas IT2FNN-2 y IT2FNN-3 son reducibles a una arquitectura de red neuronal con funciones de base radial. Se realiza un análisis para probar que las redes neuronales adaptativas neurodifusas tipo-2 por intervalos (IT2FNN) son aproximadores universales.

Para verificar la validez de la solución propuesta, se realizaron experimentos comparativos con redes adaptativas neurodifusas tipo-1 (ANFIS) [92, 93] y sistemas de lógica difusa tipo-2 por intervalos optimizados por diversas técnicas [44, 45, 57, 60, 72-74] y análisis de validación cruzada [105, 113, 114].

El alcance del presente trabajo se limitó al diseño y análisis de arquitecturas híbridas neurodifusas tipo-2 por intervalos con razonamiento de Takagi-Sugeno-Kang, entradas singleton, aprendizaje heurístico numérico y metaheurístico poblacional (HGA y PSO), aplicados al pronóstico de series de tiempo e identificación de sistemas no-lineales.

1.4 Aportaciones del trabajo

En general esta investigación ha proporcionado varias contribuciones de interés para la comunidad científica internacional. A continuación se presentan las publicaciones que se han derivado del presente trabajo:

Revistas Internacionales:

1. **Hybrid Learning Algorithm for a New Class Interval Type-2 Fuzzy Neural Networks.** Castro, J. R., Castillo, O., Melin, P., Rodríguez-Díaz, A.; *Journal of Information Sciences*, Accepted 4 October 2008.
2. **Identification of waste packaging profiles using fuzzy logic.** Gabriela Lozano-Olvera, Sara Ojeda-Benítez, Juan Ramón Castro-Rodríguez, Miguel Bravo-Zanoguera, Antonio Rodríguez-Díaz; *Journal Resources, Conservation and Recycling* 52(8):1022-1030. July 2008. ISSN: 0921-3449
3. **Interval type-2 fuzzy logic toolbox.** Castro, J. R., Castillo, O., Martínez, L. G.; *Journal of Engineering Letters* 15(1):89-98, August 2007. ISSN: 1816-0948 (online version); 1816-093X (print version)
4. **Data mining for extraction of fuzzy if-then rules using Mamdani and Takagi-Sugeno-Kang FIS.** Juan E. Moreno, Oscar Castillo, Juan R. Castro, Luis G. Martínez, Patricia Melin; *Journal of Engineering Letters* 15(1):75-88, August 2007. ISSN: 1816-0948 (online version); 1816-093X (print version)

Capítulos de libro:

1. **A Hybrid Learning Algorithm for Interval Type-2 Fuzzy Neural Networks: The Case of Time Series Prediction.** Castro, J. R., Castillo, O., Melin, P., Rodríguez-Díaz, A.; In *Soft Computing for Hybrid Intelligent Systems*. Book Chapter, Volume 154/2008: 363-386, September 2008. Springer Berlin / Heidelberg. ISBN: 978-3-540-70811-7, ISSN: 1860-949X (Print) 1860-9503 (Online)
2. **Building fuzzy inference systems with a new interval type-2 fuzzy logic toolbox.** Castro, J. R., Castillo, O., Melin, P., Rodríguez-Díaz, A.; In *Transactions on Computational Science*. Book Chapter, Volume 4750: 104-114, June 2008. Springer-Verlag. ISBN: 978-3-540-79298-7, ISSN: 0302-9743 (Print) 1611-3349 (Online)

3. **Building fuzzy inference systems with interval type-2 fuzzy logic toolbox.** Castro, J. R., Castillo, O., Melin, P., Martinez, L.G., Escobar, S., Camacho, I.; Analysis and Design of Intelligent Systems using Soft Computing Techniques, Number 1 in Studies in Fuzziness and Soft Computing, 6, Book Chapter, pages 53-62, Springer-Verlag, Germany, 1 edition, June 2007. ISBN: 978-3-540-72431-5, ISSN : 1615-3871 (Print) 1860-0794 (Online)

Congresos Internacionales:

1. **Computational Intelligence Software for Interval Type-2 Fuzzy Logic.** Oscar Castillo, Patricia Melin, Juan R. Castro.; In Workshop on Building Computational Intelligence and Machine Learning Virtual Organizations, George Mason University Fairfax, VA, USA, October 24, 2008.
2. **Intelligent control using an interval type-2 fuzzy neural network with a hybrid learning algorithm.** Castro, J., Castillo, O., Melin, P., Rodríguez-Díaz, A., Martinez, L.; In Proceedings FUZZ-IEEE 2008, Hong Kong, June 2008. ISBN: 978-1-4244-1819-0, ISSN: 1098-7584
3. **A hybrid learning algorithm for interval type-2 fuzzy neural networks in time series prediction for the case of air pollution.** Castro, J. R., Castillo, O., Melin, P., Rodríguez-Díaz, A.; In Proceedings NAFIPS 2008, New York, May 2008. ISBN: 978-1-4244-2352-1
4. **Hybrid learning algorithm for interval type-2 fuzzy neural networks.** Castro, J.R., Castillo, O., Melin, P., Rodríguez-Díaz, A.; In Proceedings of Granular Computing 2007, pages 157-162, Silicon Valley, Ca, USA, November 2007. ISBN: 0-7695-3032-X
5. **An interval type-2 fuzzy logic toolbox for control applications.** Castro, J. R., Castillo, O., Melin, P.; In Proceedings FUZZ-IEEE 2007, pages 61 - 67, IEEE, London, July 2007. ISBN: 1-4244-1210-2, ISSN: 1098-7584
6. **Interval type-2 fuzzy logic for intelligent control applications.** Castro, J. R., Castillo, O.; In Proceedings NAFIPS 2007, Number 1, pages CD Rom, San Diego, USA, June 2007. ISBN: 1-4244-1214-5, Library of Congress: 2007924575

Desarrollo de Software:

1. **Interval Type-2 Fuzzy Logic Toolbox For Use with MATLAB® version 1.0 and 2.0**
2. **Interval Type-2 Fuzzy Logic Toolbox User's Guide**

CAPÍTULO 2. MODELACIÓN NEURO-GENÉTICO-DIFUSA TIPO-2

2.1 Definición del modelo HGA-IT2FNN.

Un sistema *neuro-genético-difuso tipo-2 por intervalos* (HGA-IT2FNN) es un modelo híbrido-cooperativo que integra los métodos de razonamiento aproximado de los *sistemas de lógica difusa tipo-2 por intervalos* (IT2FLS) con las capacidades de aprendizaje de las *redes neuronales* (NNs) y robustecido cooperativamente por un proceso de aprendizaje basado en *algoritmos genéticos jerárquicos* (HGAs), los cuales presentan la capacidad de explorar y explotar espacios de búsqueda complejos; además, la codificación genética que emplean les permite incorporar conocimiento a priori de una forma muy sencilla y aprovecharlo para guiar búsquedas globales. Los tres métodos son complementarios y como consecuencia, el enfoque híbrido proporciona un marco de razonamiento aproximado poderoso que tiene capacidades de aprendizaje y generalización.

Otro sistema híbrido cooperativo es un *neuro-swarm-difuso tipo-2 por intervalos* (PSO-IT2FNN) integra los métodos de razonamiento aproximado de los *sistemas de lógica difusa tipo-2 por intervalos* (IT2FLS) con las capacidades de aprendizaje de las *redes neuronales* (NNs) y robustecido cooperativamente por un proceso de aprendizaje basado en PSO.

La estructura HGA-IT2FNN y PSO-IT2FNN permite el uso de métodos cualitativos y cuantitativos en la construcción de modelos, en la etapa de aprendizaje, funcionamiento y realimentación de conocimiento al humano, permitiendo así, todo el conocimiento disponible. Los métodos cuantitativos permiten realizar un aprendizaje a partir de datos disponibles del sistema y de la forma matemática de funcionamiento, mientras los métodos cualitativos reflejan

el sentido lingüístico que tiene la base de conocimiento de los IT2FLS y que genera una interacción sistema-humano, permitiendo la introducción de conocimiento previo al sistema y la realimentación del estado final del modelo en forma de reglas expresadas lingüísticamente desde el sistema al humano.

La selección de reglas difusas encuentra una cantidad considerable de observación heurística para expresar estrategias apropiadas del conocimiento. Es difícil que los expertos humanos examinen todos los datos de entrada-salida de los sistemas complejos para encontrar las reglas apropiadas para un sistema difuso. Se proponen en la literatura varios acercamientos para generar reglas difusas tipo-1 [15-21, 71, 77, 79, 81, 85, 92, 93] y tipo-2 [50-56, 60-63, 72-76, 84] de datos numéricos. Estos planteamientos consisten de dos fases de aprendizaje: el *aprendizaje de la estructura* que implica encontrar las variables de entrada principales de todas las posibles, especificando las funciones de membresía, la partición del espacio de entrada y determinando el número de reglas difusas; y el *aprendizaje de los parámetros* que implica la determinación de los parámetros no conocidos y la optimización de los ya existentes en el modelo, usando métodos de optimización híbridos (heurísticos numéricos + metaheurísticos poblacionales) basado en la información lingüística obtenida del humano experto y de los datos numéricos obtenidos del sistema actual que se modelará.

La razón más importante para combinar IT2FIS con NN es la capacidad de aprendizaje pues tales combinaciones son capaces de aprender reglas lingüísticas y/o funciones de membresía tipo-2 por intervalos (IT2MF), o de optimizar las ya existentes. Aprender, en este caso significa crear una base de reglas o IT2MFs, basadas en el entrenamiento de datos presentados como un problema de aprendizaje. Para crear una base de reglas por lo menos deben tenerse las IT2MF iniciales. Considérense las siguientes aproximaciones:

1. El sistema comienza sin reglas, y crea nuevas reglas hasta que el problema de aprendizaje se resuelve. La creación de una nueva regla se obtiene por un patrón de entrenamiento que no es cubierto lo suficiente por la base de reglas actual. Esta aproximación puede generar grandes bases de reglas si las IT2MFs no se escogen adecuadamente (si no se traslapan lo suficiente). Esto es comparable a una mala generalización de una NN. Dependiendo del problema de aprendizaje es posible obtener una base de reglas inconsistente. El sistema deberá borrar algunas reglas inconsistentes después de la generación de la base de reglas, usando el procedimiento SVD-QR de Wang y Mendel [44, 99].
2. El sistema comienza con todas las reglas creadas con algoritmos heurísticos para la partición de las variables [92, 93] y borra las reglas insuficientes de la base de reglas usando el procedimiento SVD-QR. Para este procedimiento se necesita una evaluación del funcionamiento de las reglas individuales. Estos planteamientos son complejos cuando se aplican a problemas con gran número de variables. No se permiten bases de reglas inconsistentes porque parte de la evaluación es una prueba de consistencia. Así, mediante este procedimiento es posible obtener bases de reglas con muy pocas reglas.
3. El sistema comienza con una base de reglas elegidas aleatoriamente, con un cierto número de reglas que son reemplazadas durante el aprendizaje, mientras se revisa la consistencia de la base de reglas a cada paso por el procedimiento SVD-QR. La desventaja de este planteamiento es el número fijo de reglas y el hecho de que debe implementarse un esquema de evaluación para el borrado de reglas y un procedimiento de análisis de datos para la adquisición de nuevas reglas; de no hacerse, el aprendizaje equivale a un proceso estocástico. En el caso de degradación del funcionamiento, los reemplazos pueden cancelarse.

Optimizar una base de reglas corresponde parcialmente a aprenderla. Para algunas situaciones existen ya reglas, algunas faltantes deben ser todavía descubiertas y algunas reglas superfluas deben borrarse. En este sentido la tercera aproximación mencionada es un procesamiento de optimización. Aprender u optimizar funciones de membresía es menos complejo que adaptar una base de reglas. Las funciones de membresía pueden describirse fácilmente por parámetros, que pueden ser optimizados con respecto a la medida del error global. Por medio de restricciones adecuadas, ciertas condiciones pueden ser resueltas (las IT2MFs deben traslaparse). La adaptación de parámetros es una tarea estándar para las NN, y hay muchas combinaciones con IT2FIS para aprender IT2MF. Podemos distinguir las siguientes alternativas para aprender IT2MF en una IT2FNN:

1. Las IT2MFs se describen por nodos adaptativos que se optimizan en el proceso de aprendizaje.
2. Una neurona difusa tipo-2 por intervalos (IT2FN) aprende a producir valores de IT2MF para el conjunto de entrenamiento deseado.
3. Una NN aprende a producir valores de IT2MF para valores de entrada dadas por el conjunto de entrenamiento.

La tercera alternativa tiene el inconveniente que las IT2MF no son explícitamente conocidas y entonces, por lo general se escoge la primera y segunda alternativa.

Mientras que la capacidad de aprendizaje es una ventaja desde el punto de vista de un IT2FLS, desde el punto de vista de una NN hay ventajas adicionales para sistemas combinados. Debido a que los sistemas IT2FNN se basan en reglas ligüísticas podemos integrar fácilmente al sistema conocimiento previo. Las reglas establecidas y las IT2MF pueden inicializar al sistema, y esto sustancialmente acorta el proceso de aprendizaje. La adaptación da lugar a una

modificación conveniente de la base de reglas y/o de las IT2MFs, y esto significa que el resultado de aprendizaje se interpreta como un IT2FLS. De este modo, el aspecto de caja negra de una NN se evita, y es posible obtener nuevos conocimientos del sistema.

La arquitectura de un modelo IT2FNN es normalmente determinada por las reglas, reducción de tipo y los conjuntos difusos tipo-2 por intervalos usados por el problema en cuestión, así que no es necesario especificar ningún parámetro de la red (número de neuronas ocultas por capa). Las desventajas que no pueden evitarse, aún por una combinación, son problemas con el proceso de aprendizaje pues solo se garantiza un resultado cuasi-óptimo del entrenamiento.

Un camino común para aplicar el algoritmo de aprendizaje de un IT2FLS es representarlo en una arquitectura NN adaptativa. Entonces un algoritmo de aprendizaje de retro-propagación, evolutivo o bioinspirado es usado para entrenar al sistema IT2FNN. Sin embargo, hay algunos problemas: los algoritmos de aprendizaje son normalmente métodos basados en gradientes y no pueden aplicarse directamente a un IT2FIS porque las funciones usadas para el proceso de inferencia por lo general no son diferenciables. Hay dos soluciones a este problema:

1. Sustituir las funciones usadas en el IT2FIS (min y max) por funciones diferenciables (producto y suma)
2. Usar algoritmos de aprendizaje de optimización heurística numérica.

Los sistemas neurodifusos modernos por lo general se representan como redes adaptativas multicapas. Hay diferentes planteamientos que tienen mucho en común, pero difieren en aspectos de implementación [15-21]. Para enfatizar con las características en común de todos estos planteamientos, y darles un significado adecuado al término de sistemas IT2FNN, debemos restringirlos a sistemas que tengan las siguientes propiedades:

1. Un sistema IT2FNN es entrenado por un algoritmo de aprendizaje derivado generalmente de la teoría de NN adaptativas. El procedimiento de aprendizaje heurístico opera con información local y sólo causa modificaciones locales en el IT2FIS.
2. Un IT2FLS es interpretado como un sistema de reglas difusas tipo-2 por intervalos. Es posible crear el sistema sin datos de entrenamiento desde el principio e inicializarlo con conocimientos previos de las reglas difusas tipo-2 por intervalos.
3. El procedimiento de aprendizaje de un sistema IT2FNN adquiere sus propiedades semánticas del IT2FLS.
4. Un sistema IT2FNN aproxima una función n-dimensional generalmente desconocida con intervalo de incertidumbre, que es parcialmente dada por los datos de entrenamiento. Las reglas difusas tipo-2 por intervalos codificadas dentro del sistema representan el conocimiento lingüístico de los datos de entrenamiento.
5. Un sistema IT2FNN puede ser visto como una especie de NN adaptativa unidireccional (feedforward). Las neuronas difusas tipo-2 por intervalos (IT2FN) adaptan los valores de las funciones de membresía inferior-superior (valores lingüísticos), las T1FN no-adaptativas usan t-normas, t-conormas para adaptar los valores de las fuerzas de disparo inferior-superior y las IT1FN adaptan los consecuentes de las reglas. Los conjuntos difusos tipo-2 por intervalos son codificados como pesos difusos tipo-2 por intervalos de conexión.

El Algoritmo de aprendizaje trabaja modificando su estructura y/o parámetros, es decir por inclusión o exclusión de neuronas y por adaptación de pesos. Estos cambios causados por el proceso de aprendizaje pueden interpretarse desde el punto de vista tanto de NN como de IT2FLS. Un proceso de aprendizaje exitoso es idéntico al incremento del conocimiento explícito,

representado en la forma de una base de reglas difusas tipo-2 por intervalos. Un sistema IT2FNN híbrido aprende en modo supervisado con aprendizaje off-line y on-line. Sin embargo, un apropiado sistema IT2FNN híbrido tiene la ventaja de llevar todas las tareas en una arquitectura unificada, también es posible usar aprendizaje previo que pueda acelerar al proceso de aprendizaje.

2.2 Red neuro-difusa tipo-2 por intervalos (IT2FNN)

Una IT2FNN es una red adaptativa neuro-difusa tipo-2 por intervalos que combina un sistema de inferencia difuso tipo-2 por intervalos (IT2FIS) de Takagi-Sugeno-Kang (TSK), en una red neuronal adaptativa, aprovechando las características sobresalientes de cada modelo. Aunque generalmente IT2FNN es usada utilizando el modelo de TSK, es posible construir una red adaptativa equivalente para los modelos de Mamdani o Tsukamoto. Sin embargo estos modelos son mucho más complejos, además se debe considerar el costo computacional de la reducción de tipo, por lo que generalmente es utilizado el modelo de TSK. Las redes adaptativas proveen un marco teórico que unifica casi todos los tipos de redes neuronales con capacidades de aprendizaje, y sus propiedades fundamentales son un elemento clave para comprender los demás paradigmas.

Una IT2FNN se caracteriza por un grafo dirigido $G = (V, E)$, donde existen nodos con fan-in cero (entradas) y nodos con fan-out cero (salidas). Cada nodo $n \in V$ representa una unidad de procesamiento y tiene asociada una función estática denotada f_n , y cada arco $e \in E$ indica la relación causal entre los nodos conectados. El conjunto de nodos se puede dividir en dos subconjuntos, $V = (A \cup N)$. Los nodos $n \in A$ se denominan adaptativos, y sus salidas dependen, no solo de sus entradas, sino también de parámetros modificables, denominados $\{\xi_1^n, \xi_2^n, \dots\}$ internos al nodo. Por el contrario, los nodos $n \in N$, cuyas funciones dependen únicamente de las

entradas, se denominan no adaptativos. En general, al representar gráficamente un IT2FNN, se utilizan rectángulos para los nodos adaptativos y círculos para los nodos no adaptativos. Los valores de salida de los nodos pares (color verde) e impares (color azul) representan el intervalo de incertidumbre (Figuras 2.4-2.14).

En este tipo de redes adaptativas neurodifusas tipo-2 por intervalos, los nodos representan las unidades de procesamiento llamadas neuronas, estas se pueden clasificar en neuronas precisas y difusas. En este modelo IT2FNN, se extiende el modelo de una neurona difusa tipo-1 (T1FN) propuesto por Hirota y Pedrycz [8, 9], a neurona difusa tipo-1 por intervalos (IT1FN) y neurona difusa tipo-2 por intervalos (IT2FN).

Una T1FN adaptativa [7, 11, 12], es una neurona adaptativa no lineal con entradas y pesos precisos con función de transferencia (gaussiana, campana generalizada o logística) y salidas difusas. Asimismo una T1FN no adaptativa, es una neurona no adaptativa con entradas-salidas difusas, sin parámetros modificables y función de transferencia t-norm o s-norm.

El modelo de una IT1FN adaptativa (Figura 2.1), es una red adaptativa lineal de tres capas con n entradas en la capa 0, dos neuronas adaptativas lineales en la capa 1 y dos neuronas no adaptativas en la capa 2. La neurona adaptativa lineal con entradas $\mathbf{x}$, pesos $\mathbf{c}$ y la neurona no adaptativa con j -ésima salida ${}_l y_k^j$, representa el valor izquierdo del intervalo del k -ésimo consecuente de las reglas del IT2FIS de TSK. La neurona adaptativa lineal con entradas $|\mathbf{x}|$, pesos $\mathbf{s}$ y la neurona no adaptativa con j -ésima salida ${}_r y_k^j$, representa el valor derecho del intervalo del k -ésimo consecuente de las reglas del IT2FIS de TSK. Los valores $\tilde{y}_k^j \in [{}_l y_k^j, {}_r y_k^j]$ se calculan con la siguiente ecuación.

$$\begin{aligned} {}_l y_k^j &= \sum_{i=1}^n c_{k,i}^j x_i + c_{k,0}^j - \sum_{i=1}^n s_{k,i}^j |x_i| - s_{k,0}^j \\ {}_r y_k^j &= \sum_{i=1}^n c_{k,i}^j x_i + c_{k,0}^j + \sum_{i=1}^n s_{k,i}^j |x_i| + s_{k,0}^j \end{aligned} \quad (2.1)$$

Las neuronas IT1FN se interconectan para generar una red adaptativa lineal IT1FNN (Figura 2.3) que contiene todas las reglas de los consecuentes, con $2r$ (r es el número de reglas) nodos de salida.

El modelo de una IT2FN adaptativa (Figura 2.2), es una red adaptativa no lineal de tres capas con conexión parcial, una variable de entrada en la capa 0 por dos T1FN adaptativas en la capa 1 con función de transferencia μ (gaussiana, campana generalizada, logística, etc.) y dos T1FN no adaptativas en la capa 2 para generar un valor lingüístico (IT2MF) con funciones de membresía inferior (nodo T) y superior (nodo S). El nodo T y S en la capa de salida tienen función de transferencia t-norm (producto algebraico) y s-norm (suma algebraica) respectivamente. Las funciones de membresía inferior $\underline{\mu}$ y superior $\bar{\mu}$ se calculan usando las ecuaciones 2.2 y 2.3.

$$\begin{aligned} net_1 &= w_{1,i}x + b_1 \quad ; \mu_1 = \mu(net_1) \\ net_2 &= w_{2,i}x + b_1 \quad ; \mu_2 = \mu(net_2) \end{aligned} \quad (2.2)$$

$$\begin{aligned} \underline{\mu}(x) &= \mu(net_1) \cdot \mu(net_2) \\ \bar{\mu}(x) &= \mu(net_1) + \mu(net_1) - \underline{\mu}(x) \end{aligned} \quad (2.3)$$

Los pesos $\mathbf{w}$ representan la desviación y $\mathbf{b}$ la media con incertidumbre de una IT2MF. Las neuronas IT2FN se interconectan para generar los valores lingüísticos de los antecedentes de las reglas.

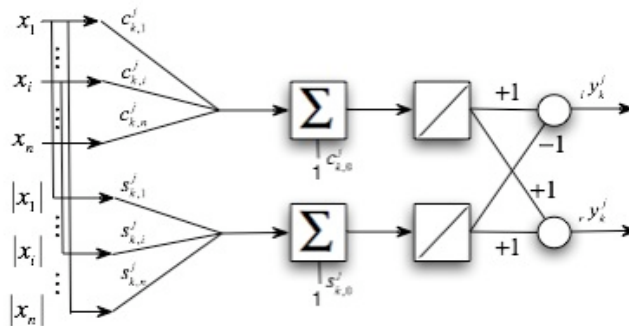


Figura 2.1 Interval Type-1 Fuzzy Neuron (IT1FN)

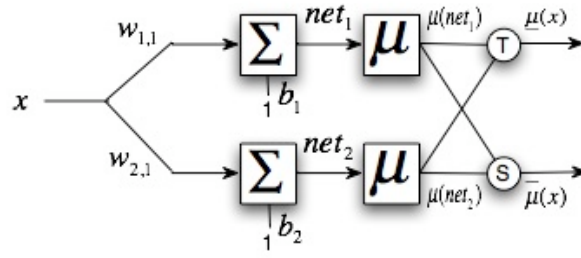


Figura 2.2 Interval Type-2 Fuzzy Neuron (IT2FN)

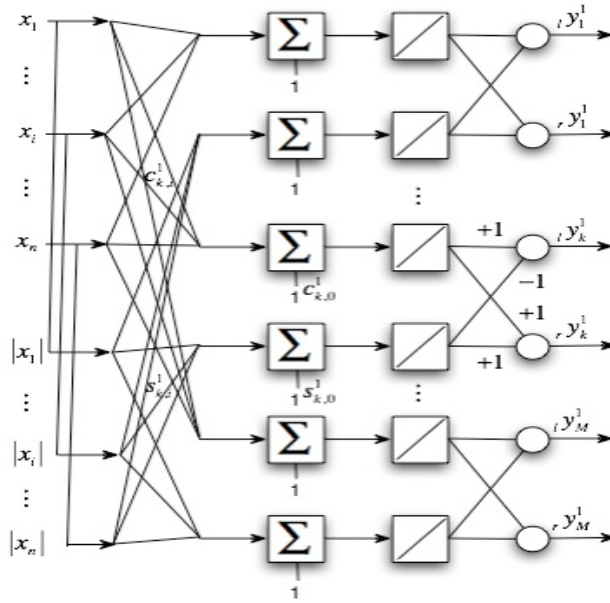


Figura 2.3 Interval Type-1 Fuzzy Neural Network (IT1FNN)

2.2.1 Arquitecturas IT2FNN.

En este trabajo se proponen tres arquitecturas principales (IT2FNN-1, IT2FNN-2 y IT2FNN-3) para integrar un IT2FIS de TSK primer orden, con consecuentes reales (A2C0) y consecuentes por intervalos (A2C1). En cada arquitectura se analizan estrategias de integración de los elementos de proceso de un IT2FIS de TSK (fuzzificación, base de conocimiento, reducción de tipo y defuzzificación).

La arquitectura IT2FNN-1, consta de 5 capas, utiliza nodos adaptativos con función equivalente a la membresía inferior-superior en la capa de fuzzificación (capa 1). Los nodos no adaptativos en la capa de reglas (capa 2) se conectan con la capa de fuzzificación (capa 1) para generar los antecedentes de las reglas del IT2FIS de TSK. Los nodos adaptativos en la capa de los consecuentes (capa 3) tienen conexión con la capa de entrada (capa 0) para generar los consecuentes de las reglas. Los nodos no adaptativos en la capa de reducción de tipo (capa 4), evalúan los valores izquierdo-derecho con el algoritmo de Karnik y Mendel (KM). El nodo no adaptativo en la capa de defuzzificación (capa 5) promedia los valores izquierdo-derecho.

En el caso particular de la arquitectura IT2FNN-2, tiene 6 capas, utiliza IT2FN para fuzzificar las entradas (capas 1-2). Los nodos no adaptativos en la capa de reglas (capa 3), se conectan con la capa de valores lingüísticos inferior-superior (capa 2) para generar los antecedentes de las reglas del IT2FIS de TSK. Los nodos adaptativos en la capa de los consecuentes (capa 4) tienen conexión con la capa de entrada (capa 0) para generar los consecuentes de las reglas. Los nodos no adaptativos en la capa de reducción de tipo (capa 5), evalúan los valores izquierdo-derecho con el algoritmo de Karnik y Mendel (KM). El nodo no adaptativo en la capa de defuzzificación (capa 6) promedia los valores izquierdo-derecho.

La arquitectura IT2FNN-3, tiene 8 capas, utiliza IT2FN para fuzzificar las entradas (capas 1-2). Los nodos no adaptativos en la capa de reglas (capa 3), se conectan con la capa de valores lingüísticos inferior-superior (capa 2) para generar los antecedentes de las reglas del IT2FIS de TSK. Los nodos adaptativos en la capa 4, adaptan las fuerzas de disparo izquierda-derecha, ponderando las fuerzas de disparo inferior-superior de las reglas con los pesos sinápticos entre las capas 3-4. Los nodos no adaptativos de la capa 5, normalizan las fuerzas de disparo inferior-superior de las reglas. Los nodos adaptativos en la capa de los consecuentes (capa 6) tienen

conexión con la capa de entrada (capa 0) para generar los consecuentes de las reglas. Los nodos no adaptativos en la capa de reducción de tipo (capa 7), evalúan los valores izquierdo-derecho, sumando del producto de las fuerzas de disparo inferior-superior normalizadas por los valores izquierdo-derecho de los consecuentes de las reglas respectivamente. El nodo en la capa de defuzzificación es adaptativo y su salida, $\hat{y}$, se define como un promedio ponderado de los valores izquierdo-derecho y el parámetro γ . El parámetro γ (valor por defecto 0.5) ajusta el intervalo de incertidumbre definido por los valores izquierdo-derecho $[\hat{y}_l, \hat{y}_r]$.

A continuación se hace una descripción operativa de cada arquitectura IT2FNN, con las estrategias de integración de los elementos de un IT2FIS de TSK en una red adaptativa.

2.2.1.1 Arquitectura IT2FNN-1:A2C0

Una IT2FNN-1:A2C0 (Figura 2.4), es una red adaptativa neuro-difusa tipo-2 por intervalos con 6 capas, integra un modelo de inferencia difuso tipo-2 por intervalos de TSK de primer orden IT2FIS:A2C0 (Antecedentes difusos tipo-2 por intervalos y consecuentes reales), en una red neuronal adaptativa.

El sistema se inicializa con n variables de entrada, $x = (x_1, \dots, x_i, \dots, x_n)$, v_i valores lingüísticos por entrada, $v = (v_1, \dots, v_i, \dots, v_n)$, ϑ funciones de membresía tipo-2 por intervalos (IT2MF),

$\vartheta = \sum_{i=1}^n v_i$, r reglas, y la tabla de índices de las reglas definidas por el usuario o evaluadas

por el siguiente pseudo-código:

function $l_{k,i} = rulelist(v)$

$$r = \prod_{i=1}^n v_i$$

for $k = 1, \dots, r$
 set $\Omega = k - 1$
 for $i = 1, \dots, n$
 set $l_{k,n-i+1} = \left(\Omega - v_{n-i+1} \cdot \left\lfloor \frac{\Omega}{v_{n-i+1}} \right\rfloor \right) + 1$
 set $\Omega = \left\lfloor \frac{\Omega}{v_{n-i+1}} \right\rfloor$
 end
 end

$i_l = s_1 + 1 : 2 : s_{n+1}$
 $i_u = s_1 + 2 : 2 : s_{n+1}$

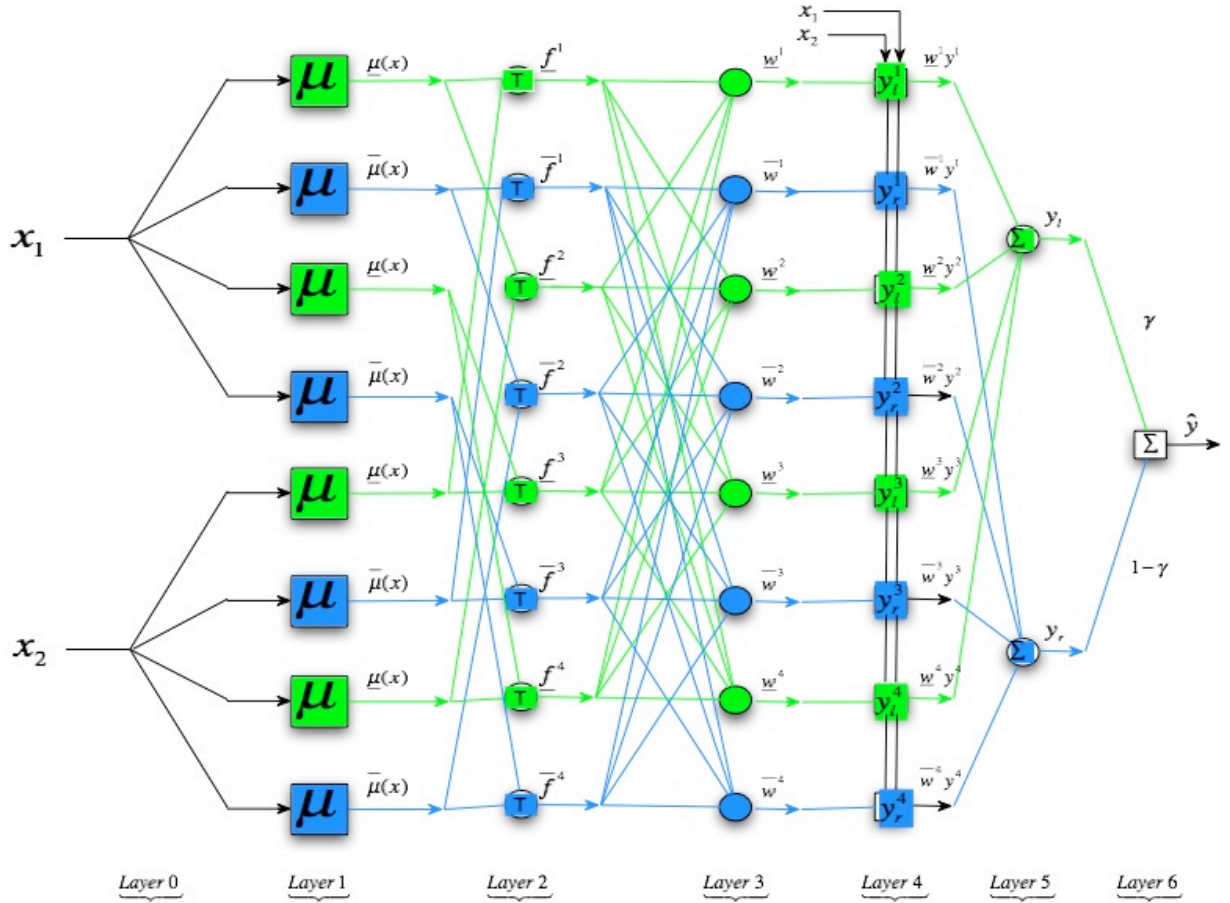


Figura 2.4 Arquitetura IT2FNN-1:A2C0

La arquitectura IT2FNN-1:A2C0 equivalente al mecanismo de razonamiento IT2FIS:A2C0, donde los nodos situados en la misma capa realizan funciones similares, y son explicados a continuación:

Capa 0: Entradas

$$o_i^0 = x_i \quad ; i = 1, \dots, n$$

Capa 1: Fuzzificación de las entradas. Evalúa las IT2MFs, $\tilde{\mu}_{i,j}(x_i) = [\underline{\mu}_{i,j}(x_i), \bar{\mu}_{i,j}(x_i)]$. Los nodos de esta capa son adaptativos, y las funciones, $\underline{\mu}_{i,j}(x_i), \bar{\mu}_{i,j}(x_i)$, puede definirse de varias maneras, teniendo en cuenta que debe ser una IT2MF diferenciable. Una de las formas mas comunes de definir las IT2MFs $\underline{\mu}_{i,j}(x_i), \bar{\mu}_{i,j}(x_i)$ son por ejemplo:

$$\tilde{\mu}_{i,j}(x_i) = [\underline{\mu}_{i,j}(x_i), \bar{\mu}_{i,j}(x_i)] = \text{igaussmtype2}(x_i, [\sigma_{i,j}, {}^1m_{i,j}, {}^2m_{i,j}]) \quad ; i = 1, 2, \dots, n ; j = 1, 2, \dots, v_i$$

donde;

$$\bar{\mu}_{i,j}(x_i) = \begin{cases} {}^1\mu(x_i, [\sigma_{i,j}, {}^1m_{i,j}]) & x_i < {}^1m_{i,j} \\ 1 & {}^1m_{i,j} \leq x_i \leq {}^2m_{i,j} \\ {}^2\mu(x_i, [\sigma_{i,j}, {}^2m_{i,j}]) & x_i > {}^2m_{i,j} \end{cases}$$

$$\underline{\mu}_{i,j}(x_i) = \begin{cases} {}^2\mu(x_i, [\sigma_{i,j}, {}^2m_{i,j}]) & x_i \leq \frac{{}^1m_{i,j} + {}^2m_{i,j}}{2} \\ {}^1\mu(x_i, [\sigma_{i,j}, {}^1m_{i,j}]) & x_i > \frac{{}^1m_{i,j} + {}^2m_{i,j}}{2} \end{cases}$$

$${}^1\mu_{i,j}(x_i, [\sigma_{i,j}, {}^1m_{i,j}]) = \exp\left(-\frac{1}{2}\left(\frac{x_i - {}^1m_{i,j}}{\sigma_{i,j}}\right)^2\right)$$

$${}^2\mu_{i,j}(x_i, [\sigma_{i,j}, {}^2m_{i,j}]) = \exp\left(-\frac{1}{2}\left(\frac{x_i - {}^2m_{i,j}}{\sigma_{i,j}}\right)^2\right)$$

Las $\underline{\mu}_{i,j}(x_i), \bar{\mu}_{i,j}(x_i)$ se agregan a los nodos de esta capa de acuerdo al siguiente pseudo-código:

```

 $s_1 = 0$ 
for  $i=1, \dots, n$ 
     $s_{i+1} = s_i + 2v_i$ 
end
for  $i=1:n$ 
     $k=s_i$ 
    for  $j=1:v_i$ 
         $k=k+1$ 
         $o_{2k-1}^1 = \underline{\mu}_{i,j}(x_i)$ 
         $o_{2k}^1 = \bar{\mu}_{i,j}(x_i)$ 
    end
end
end

```

Capa 2: Fuerzas de disparo inferior-superior de las reglas. Cada nodo de esta capa es no adaptativo y se define la salida como el producto de sus entradas. Cada salida corresponde a la fuerza de disparo inferior-superior de la k-ésima regla, y en general, se puede utilizar otro operador T-norm en lugar del producto.

$$o_{2k-1}^2 = \underline{f}^k = \underset{i=1}{\overset{n}{*}} \left(\underline{\mu}_{\ell_{k,i}} \right); k = 1, \dots, r$$

$$o_{2k}^2 = \bar{f}^k = \underset{i=1}{\overset{n}{*}} \left(\bar{\mu}_{\ell_{k,i}} \right)$$

Capa 3: Normalización de las fuerzas de disparo inferior-superior de las reglas. Los nodos de esta capa son no adaptativos. Se define la salida como la razón entre las fuerzas de disparo inferior-superior de la k-ésima regla y la suma de las fuerzas de disparo inferior-superior de todas las reglas, denominado fuerzas de disparo inferior-superior normalizado:

$$o_{2k-1}^3 = \underline{w}^k = \frac{f^k}{\sum_{\ell=1}^r f^{\ell}} \quad ; \quad k = 1, \dots, r$$

$$o_{2k}^3 = \overline{w}^k = \frac{\overline{f}^k}{\sum_{\ell=1}^r \overline{f}^{\ell}}$$

Capa 4: Consecuentes de las reglas. Cada nodo de esta capa es adaptativo y sus parámetros son $\{c_i^k, c_0^k\}$. La salida del nodo corresponde a la salida parcial de la k-ésima regla, y^k .

$$y^k = \sum_{i=1}^n c_i^k x_i + c_0^k \quad ; \quad k = 1, \dots, r$$

$$o_{2k-1}^4 = y^k$$

$$o_{2k}^4 = y^k$$

Capa 5: Reducción de tipo. Calcula los valores izquierdo-derecho del intervalo $[\hat{y}_l, \hat{y}_r]$. Los nodos de esta capa son no adaptativos y sus salidas, $\hat{y}_l, \hat{y}_r$, se define como la sumatoria del producto de las fuerzas de disparo inferior-superior normalizadas por los consecuentes de las reglas respectivamente:

$$o_1^5 = \hat{y}_l = \sum_{k=1}^r \underline{w}^k \cdot y^k$$

$$o_2^5 = \hat{y}_r = \sum_{k=1}^r \overline{w}^k \cdot y^k$$

Capa 6: Defuzzificación. El nodo de esta capa es adaptativo y su salida, $\hat{y}$, se define como un promedio ponderado de los valores izquierdo-derecho y el parámetro γ . El parámetro γ (valor por defecto 0.5) ajusta el intervalo de incertidumbre definido por los valores izquierdo-derecho $[\hat{y}_l, \hat{y}_r]$.

$$o_1^6 = \hat{y} = \gamma \hat{y}_l + (1 - \gamma) \hat{y}_r$$

2.2.1.2 Arquitectura IT2FNN-2:A2C0

Una IT2FNN-2:A2C0 (Figura 2.5), es una red adaptativa neuro-difusa tipo-2 por intervalos con 7 capas, integra un modelo de inferencia difuso tipo-2 por intervalos de TSK de primer orden IT2FIS:A2C0 (Antecedentes difusos tipo-2 por intervalos y consecuentes reales), en una red neuronal adaptativa. El sistema se inicializa como en la arquitectura IT2FNN-1:A2C0 y los índices de los nodos i_l, i_u que se usaran en la capa 2 de esta arquitectura. A continuación se describen las capas 0-2:

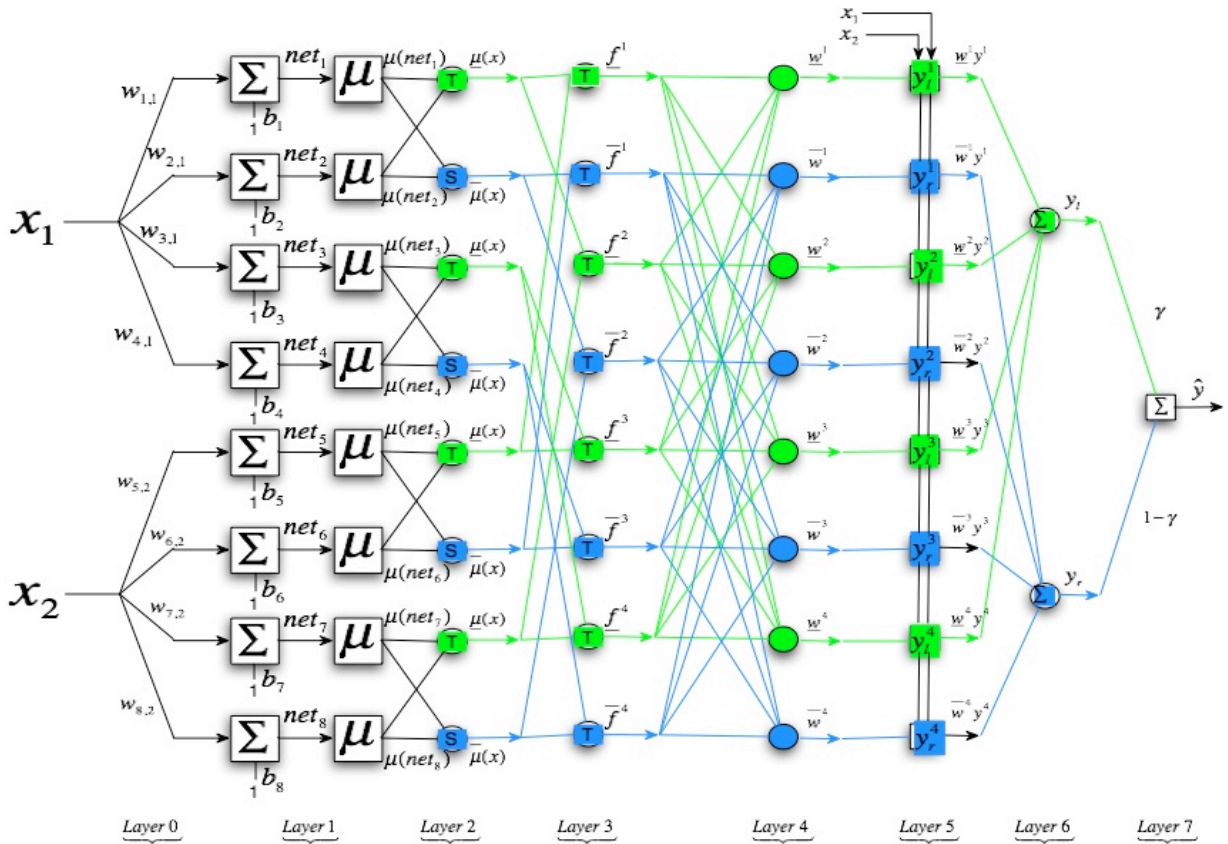


Figura 2.5 Arquitectura IT2FNN-2:A2C0

Capa 0: Entradas

$$o_i^0 = x_i \quad ; i = 1, \dots, n$$

Capa 1: T1FN Adaptativa

for i=1,...,n

for k=s_i+1,...,s_{i+1}

$$net_k^1 = w_{k,i}^{1,0} x_i + b_k^1$$

$$o_k^1 = \mu(net_k^1)$$

end

end

Capa 2: T1FN no adaptativa.

$$o_{2k-1}^2 = o_{2k-1}^1 \cdot o_{2k}^1 \quad \forall k = 1, \dots, \vartheta$$

$$o_{2k}^2 = o_{2k-1}^1 + o_{2k}^1 - o_{2k-1}^2$$

for k=1,...,r

for i=1,...,n

$$\tau = \ell_{k,i} \quad ; \quad \pi = \sum_{j=1}^{i-1} v_j + |\tau|$$

if $\tau > 0$

$$\underline{\mu}_{k,i} = o_{i_l(\pi)}^2 ; \bar{\mu}_{k,i} = o_{i_u(\pi)}^2$$

else

$$\underline{\mu}_{k,i} = null ; \bar{\mu}_{k,i} = null$$

end

end

end

Las capas 3-7 de la arquitectura IT2FNN-2:A2C0 son iguales a las capas 2-6 de la arquitectura IT2FNN-1:C2C0.

2.2.1.3 Arquitectura RBIT2FNN-2:A2C0.

Una RBIT2FNN-2:A2C0 (Figura 2.6), es una red adaptativa neuro-difusa tipo-2 por intervalos de base radial con 6 capas, integra un modelo de inferencia difuso tipo-2 por intervalos de TSK de primer orden IT2FIS:A2C0 (Antecedentes difusos tipo-2 por intervalos y consecuentes reales), en una red neuronal adaptativa.

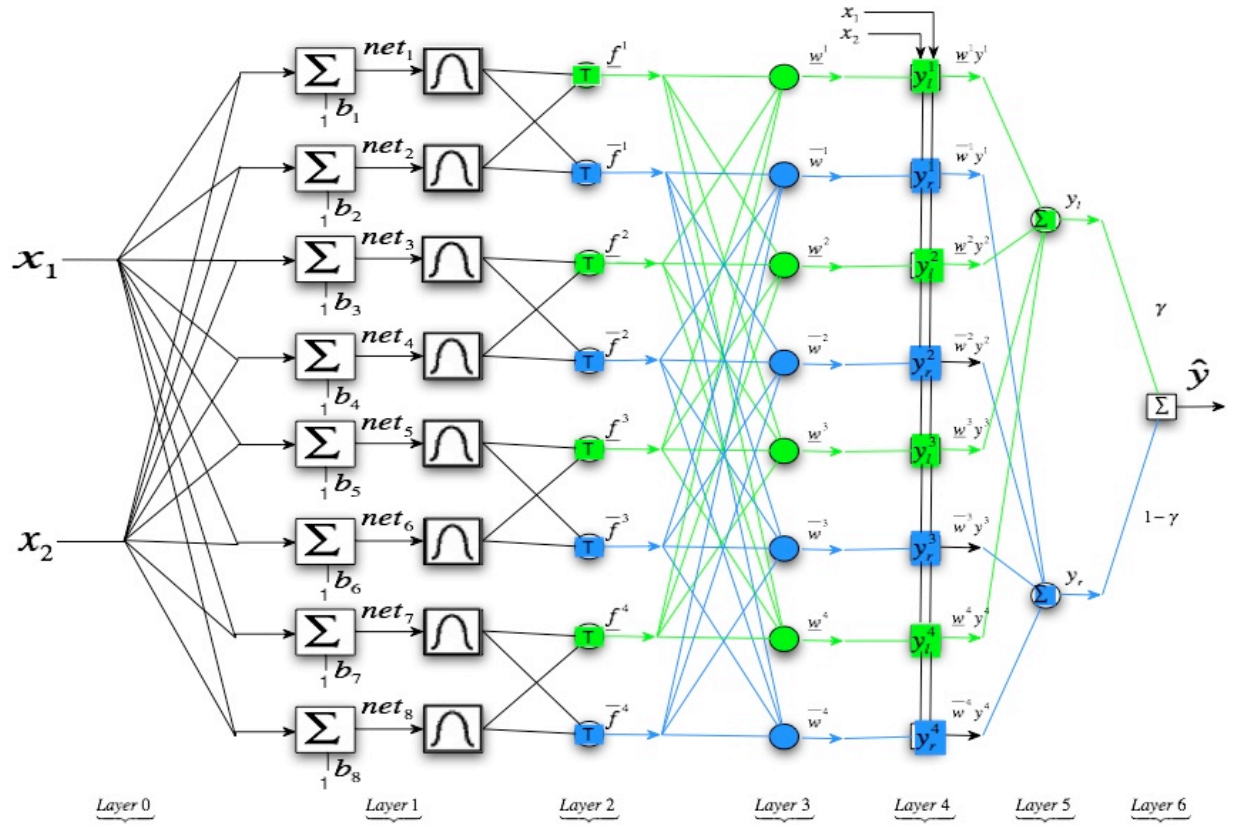


Figura 2.6 Arquitectura RBIT2FNN-2:A2C0

Capa 0: Entradas

$$o_i^0 = x_i \quad ; i = 1, \dots, n$$

Capa 1: T1FN base radial

$$net_k^1 = \|w_k^{1,0} x - b_k^1\| \Leftrightarrow w_k^{1,0} = \frac{1}{\sqrt{2} \sigma_k^1} ; b_k^1 = \frac{c_k^1}{\sqrt{2} \sigma_k^1}$$

$$o_k^1 = \exp\left[-(net_k^1)^2\right] \quad \forall k = 1, \dots, 2r$$

Capa 2: T1FN. Fuerzas de disparo inferior-superior de las reglas

$$o_{2k-1}^2 = \underline{f}^k = o_{2k-1}^1 \cdot o_{2k}^1 \quad \forall k = 1, \dots, r$$

$$o_{2k}^2 = \overline{f}^k = o_{2k-1}^1 + o_{2k}^1 - o_{2k-1}^2$$

Las capas 3-6 de la arquitectura RBIT2FNN-2:A2C0 son iguales a las capas 3-6 de la arquitectura IT2FNN-1:A2C0.

2.2.1.4 Arquitectura IT2FNN-1:A2C1

Una IT2FNN-1:A2C1 (Figura 2.7), es una red adaptativa neuro-difusa tipo-2 por intervalos con 5 capas, integra un modelo de inferencia difuso tipo-2 por intervalos de TSK de primer orden IT2FIS:A2C1 (Antecedentes difusos tipo-2 por intervalos y consecuentes tipo-1 por intervalos), en una red neuronal adaptativa.

Capa 0: Entradas

$$o_i^0 = x_i \quad ; i = 1, \dots, n$$

Capa 1: Fuzzificación de las entradas. Evalúa las IT2MFs, $\tilde{\mu}_{i,j}(x_i) = [\underline{\mu}_{i,j}(x_i), \overline{\mu}_{i,j}(x_i)]$. Los nodos de esta capa son adaptativos, y las funciones, $\underline{\mu}_{i,j}(x_i), \overline{\mu}_{i,j}(x_i)$, puede definirse de varias maneras, teniendo en cuenta que debe ser una IT2MF diferenciable. Una de las formas mas comunes de definir las IT2MFs $\underline{\mu}_{i,j}(x_i), \overline{\mu}_{i,j}(x_i)$ son por ejemplo:

$$\tilde{\mu}_{i,j}(x_i) = [\underline{\mu}_{i,j}(x_i), \overline{\mu}_{i,j}(x_i)] = \text{igaussmtype2}(x_i, [\sigma_{i,j}^1, m_{i,j}^1, m_{i,j}^2]) \quad ; i = 1, 2, \dots, n ; j = 1, 2, \dots, v_i$$

donde;

$$\bar{\mu}_{i,j}(x_i) = \begin{cases} {}^1\mu(x_i, [\sigma_{i,j}, {}^1m_{i,j}]) & x_i < {}^1m_{i,j} \\ 1 & {}^1m_{i,j} \leq x_i \leq {}^2m_{i,j} \\ {}^2\mu(x_i, [\sigma_{i,j}, {}^2m_{i,j}]) & x_i > {}^2m_{i,j} \end{cases} \quad \underline{\mu}_{i,j}(x_i) = \begin{cases} {}^2\mu(x_i, [\sigma_{i,j}, {}^2m_{i,j}]) & x_i \leq \frac{{}^1m_{i,j} + {}^2m_{i,j}}{2} \\ {}^1\mu(x_i, [\sigma_{i,j}, {}^1m_{i,j}]) & x_i > \frac{{}^1m_{i,j} + {}^2m_{i,j}}{2} \end{cases}$$

$${}^1\mu_{i,j}(x_i, [\sigma_{i,j}, {}^1m_{i,j}]) = \exp\left(-\frac{1}{2}\left(\frac{x_i - {}^1m_{i,j}}{\sigma_{i,j}}\right)^2\right) \quad {}^2\mu_{i,j}(x_i, [\sigma_{i,j}, {}^2m_{i,j}]) = \exp\left(-\frac{1}{2}\left(\frac{x_i - {}^2m_{i,j}}{\sigma_{i,j}}\right)^2\right)$$

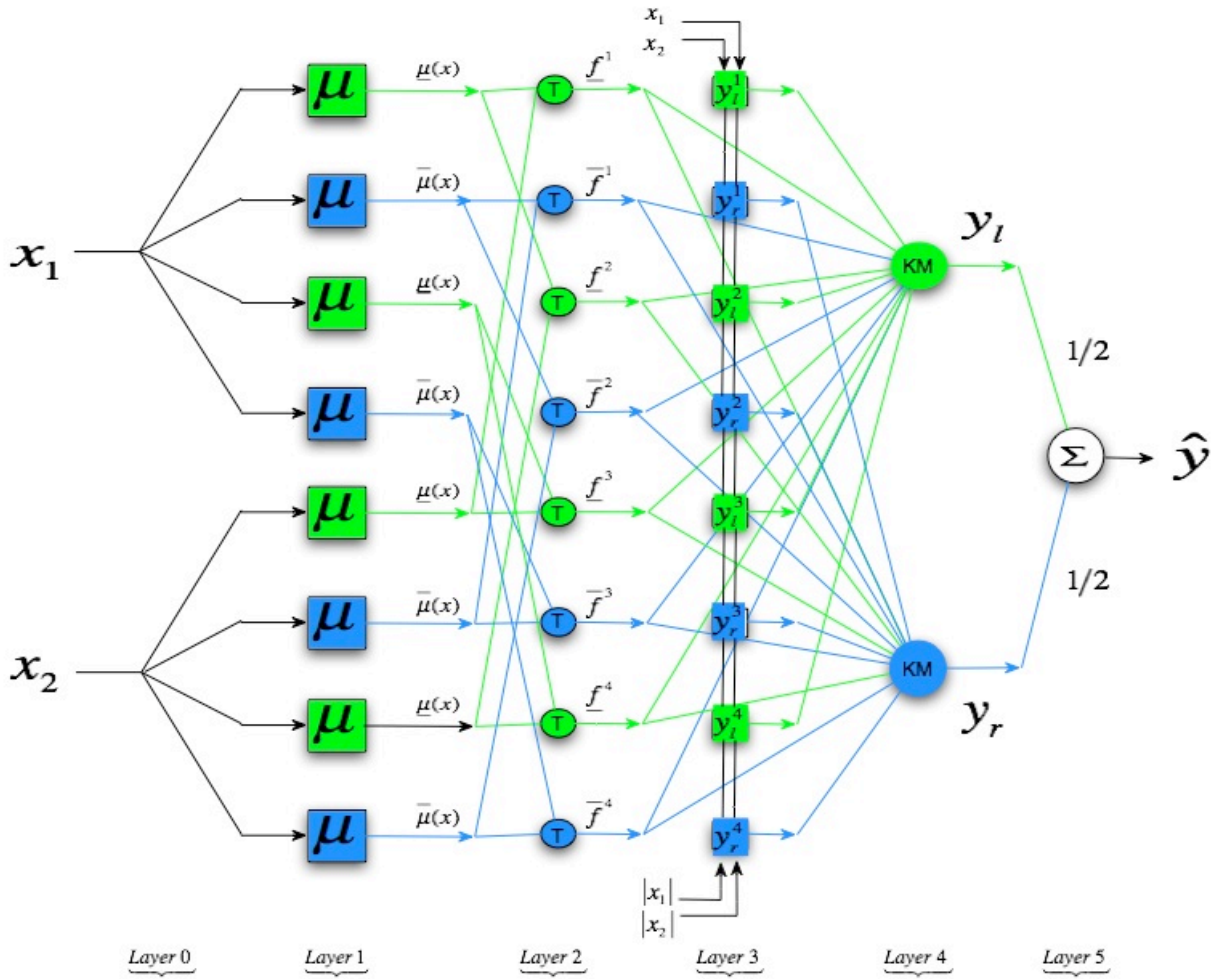


Figura 2.7 Arquitetura IT2FNN-1:A2C1

Las $\underline{\mu}_{i,j}(x_i), \bar{\mu}_{i,j}(x_i)$ se agregan a los nodos de esta capa de acuerdo al siguiente pseudo-código:

$$s_1 = 0$$

for $i=1, \dots, n$

$$s_{i+1} = s_i + 2v_i$$

end

for $i=1:n$

$$k=s_i$$

for $j=1:v_i$

$$k=k+1$$

$$o_{2k-1}^1 = \underline{\mu}_{i,j}(x_i)$$

$$o_{2k}^1 = \bar{\mu}_{i,j}(x_i)$$

end

end

Capa 2: Fuerzas de disparo inferior-superior de las reglas. Cada nodo de esta capa es no adaptativo y se define la salida como el producto de sus entradas. Cada salida corresponde a la fuerza de disparo inferior-superior de la k-ésima regla, y en general, se puede utilizar otro operador T-norm en lugar del producto.

$$o_{2k-1}^2 = \underline{f}^k = \underset{i=1}{\overset{n}{*}} \left(\underline{\mu}_{\ell_{k,i}} \right); k = 1, \dots, r$$

$$o_{2k}^2 = \bar{f}^k = \underset{i=1}{\overset{n}{*}} \left(\bar{\mu}_{\ell_{k,i}} \right)$$

Capa 3: Consecuentes de las reglas. Cada nodo de esta capa es adaptativo y sus parámetros son $\{c_i^k, c_0^k\}$ y $\{s_i^k, s_0^k\}$. La salida del nodo corresponde a la salida parcial de la k-ésima regla, $\tilde{y}^k \in [y_l^k, y_r^k]$.

$$o_{2k-1}^3 = y_l^k = \sum_{i=1}^n c_i^k x_i + c_0^k - \sum_{i=1}^n s_i^k |x_i| - s_0^k \quad ; \quad k = 1, \dots, r$$

$$o_{2k}^3 = y_r^k = \sum_{i=1}^n c_i^k x_i + c_0^k + \sum_{i=1}^n s_i^k |x_i| + s_0^k$$

Capa 4: Reducción de tipo. Calcula los valores izquierdo-derecho del intervalo $[\hat{y}_l, \hat{y}_r]$ y las fuerzas de disparo izquierdo(f_l^k)-derecho(f_r^k) que contribuyen al calculo de los valores $[\hat{y}_l, \hat{y}_r]$ usando el algoritmo KM.

$$f_l^k = \text{leftpoint}(\underline{f}^k, \overline{f}^k, y_l^k)$$

$$f_r^k = \text{rightpoint}(\underline{f}^k, \overline{f}^k, y_r^k)$$

$$o_1^4 = \hat{y}_l = \frac{\sum_{k=1}^r f_l^k y_l^k}{\sum_{k=1}^r f_l^k}$$

$$o_2^4 = \hat{y}_r = \frac{\sum_{k=1}^r f_r^k y_r^k}{\sum_{k=1}^r f_r^k}$$

Capa 5: Defuzzificación. El nodo de esta capa es adaptativo y su salida, $\hat{y}$, se define como un promedio ponderado de los valores izquierdo-derecho y el parámetro γ . El parámetro γ (valor por defecto 0.5) ajusta el intervalo de incertidumbre definido por los valores izquierdo-derecho $[\hat{y}_l, \hat{y}_r]$.

$$o_1^5 = \hat{y} = \gamma \hat{y}_l + (1 - \gamma) \hat{y}_r$$

2.2.1.5 Arquitectura IT2FNN-2:A2C1

Una IT2FNN-2:A2C1 (Figura 2.8), es una red adaptativa neuro-difusa tipo-2 por intervalos con 6 capas, integra un modelo de inferencia difuso tipo-2 por intervalos de TSK de primer orden IT2FIS:A2C1 (Antecedentes difusos tipo-2 por intervalos y consecuentes tipo-1 por intervalos), en una red neuronal adaptativa. La arquitectura IT2FNN-2:A2C1 es equivalente a la arquitectura IT2FNN-2a:A2C1 que se muestra en la Figura 2.9.

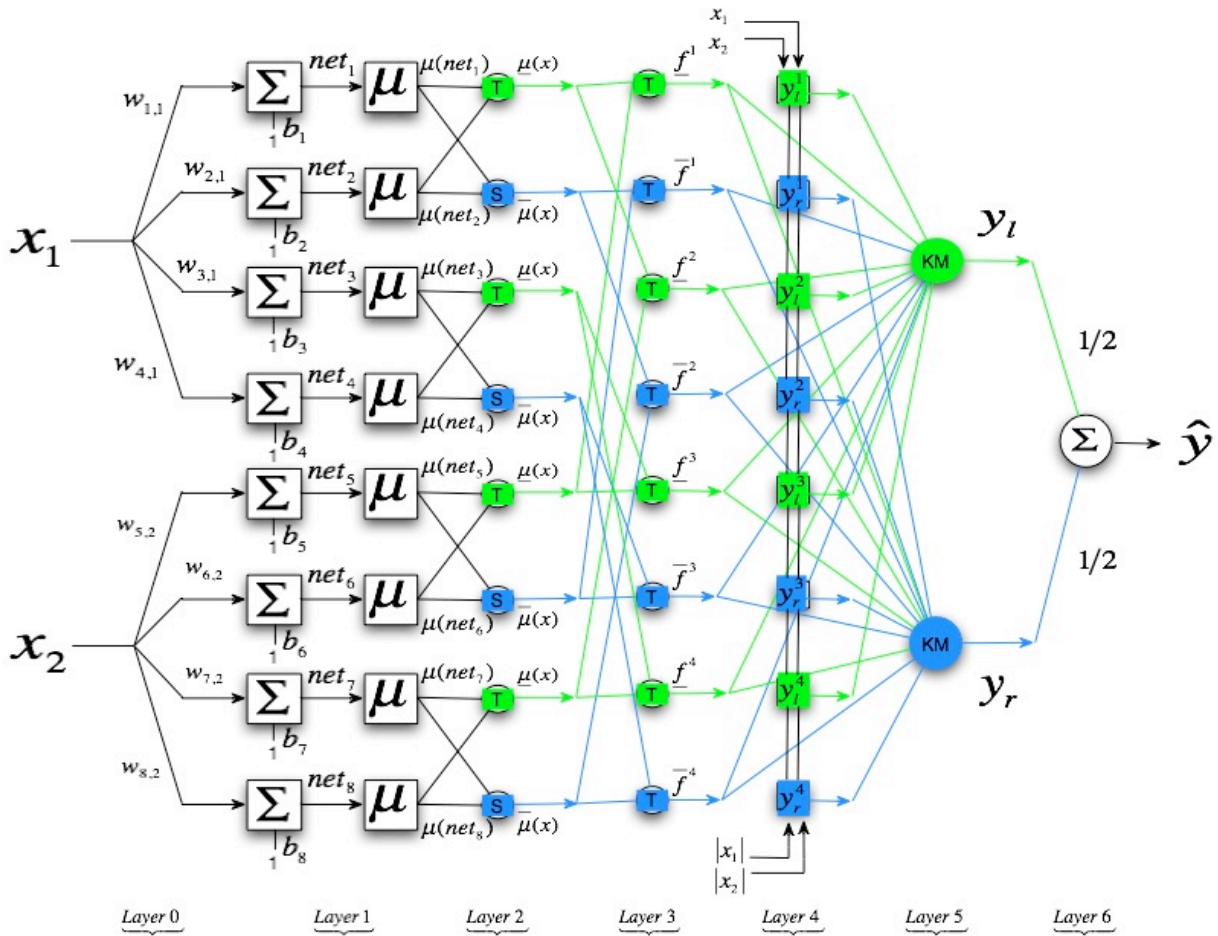


Figura 2.8 Arquitectura IT2FNN-2:A2C1

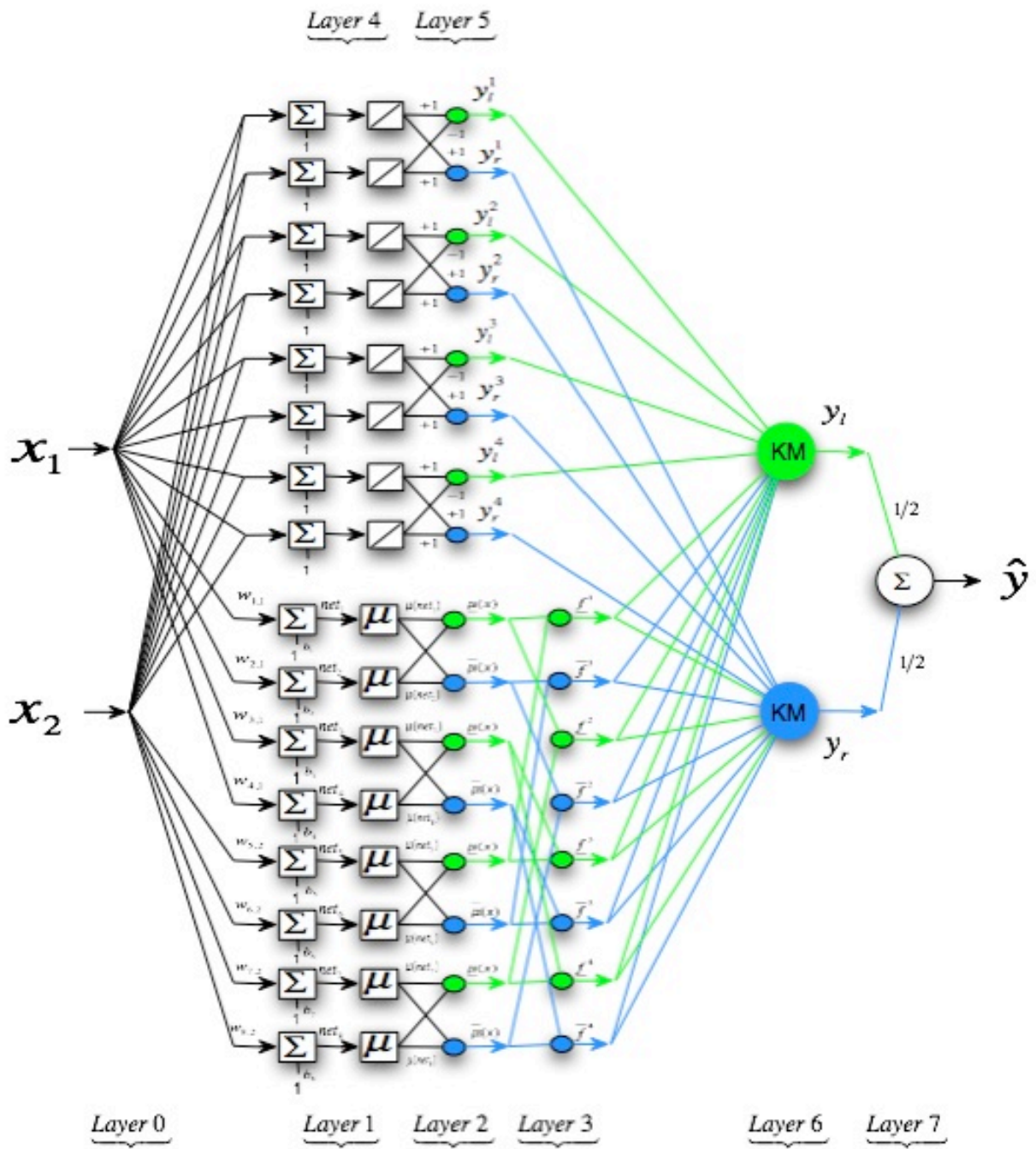


Figura 2.9 Arquitectura IT2FNN-2a:A2C1

Capa 0: Entradas

$$o_i^0 = x_i \quad ; i = 1, \dots, n$$

Capa 1: T1FN Adaptativa

for i=1,...,n

for k=s_i+1,...,s_{i+1}

$$net_k^1 = w_{k,i}^{1,0} x_i + b_k^1$$

$$o_k^1 = \mu(net_k^1)$$

end

end

Capa 2: T1FN no adaptativa.

$$o_{2k-1}^2 = o_{2k-1}^1 \cdot o_{2k}^1 \quad \forall k = 1, \dots, \vartheta$$

$$o_{2k}^2 = o_{2k-1}^1 + o_{2k}^1 - o_{2k-1}^2$$

for k=1,...,r

for i=1,...,n

$$\tau = \ell_{k,i} \quad ; \quad \pi = \sum_{j=1}^{i-1} v_j + |\tau|$$

if $\tau > 0$

$$\underline{\mu}_{k,i} = o_{i_l(\pi)}^2 ; \bar{\mu}_{k,i} = o_{i_u(\pi)}^2$$

else

$$\underline{\mu}_{k,i} = null ; \bar{\mu}_{k,i} = null$$

end

end

end

Las capas 3-6 de la arquitectura IT2FNN-2:A2C1 son iguales a las capas 2-5 de la arquitectura IT2FNN-1:C2C1.

2.2.1.6 Arquitectura RBIT2FNN-2:A2C1

Una RBIT2FNN-2:A2C1 (Figura 2.10), es una red adaptativa neuro-difusa tipo-2 por intervalos de base radial con 5 capas, integra un modelo de inferencia difuso tipo-2 por intervalos de TSK de primer orden IT2FIS:A2C1 (Antecedentes difusos tipo-2 por intervalos y consecuentes difusos tipo-1 por intervalos), en una red neuronal adaptativa. Esta arquitectura es equivalente a la que se muestra en la Figura 2.11.

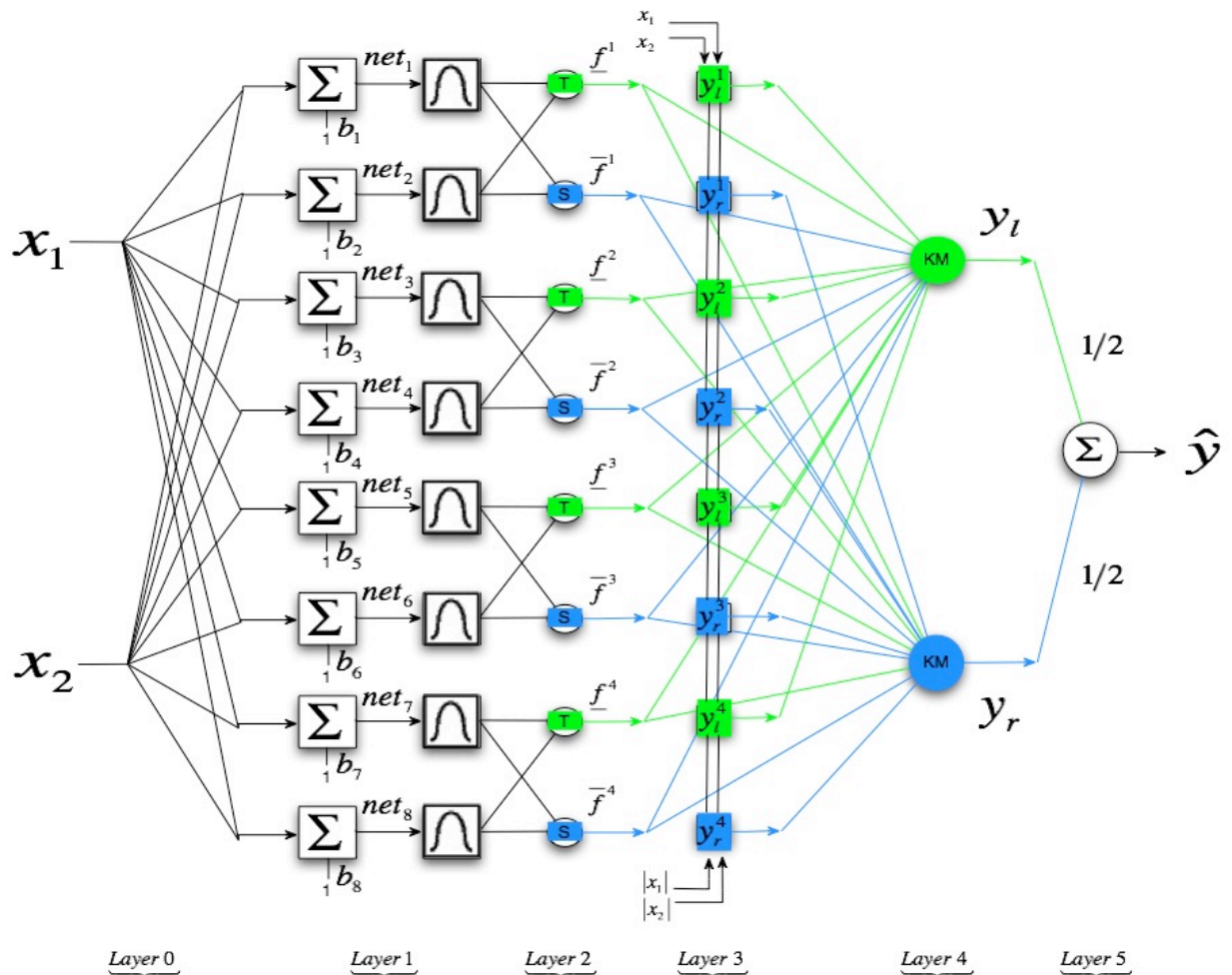


Figura 2.10 Arquitectura RBIT2FNN-2:A2C1

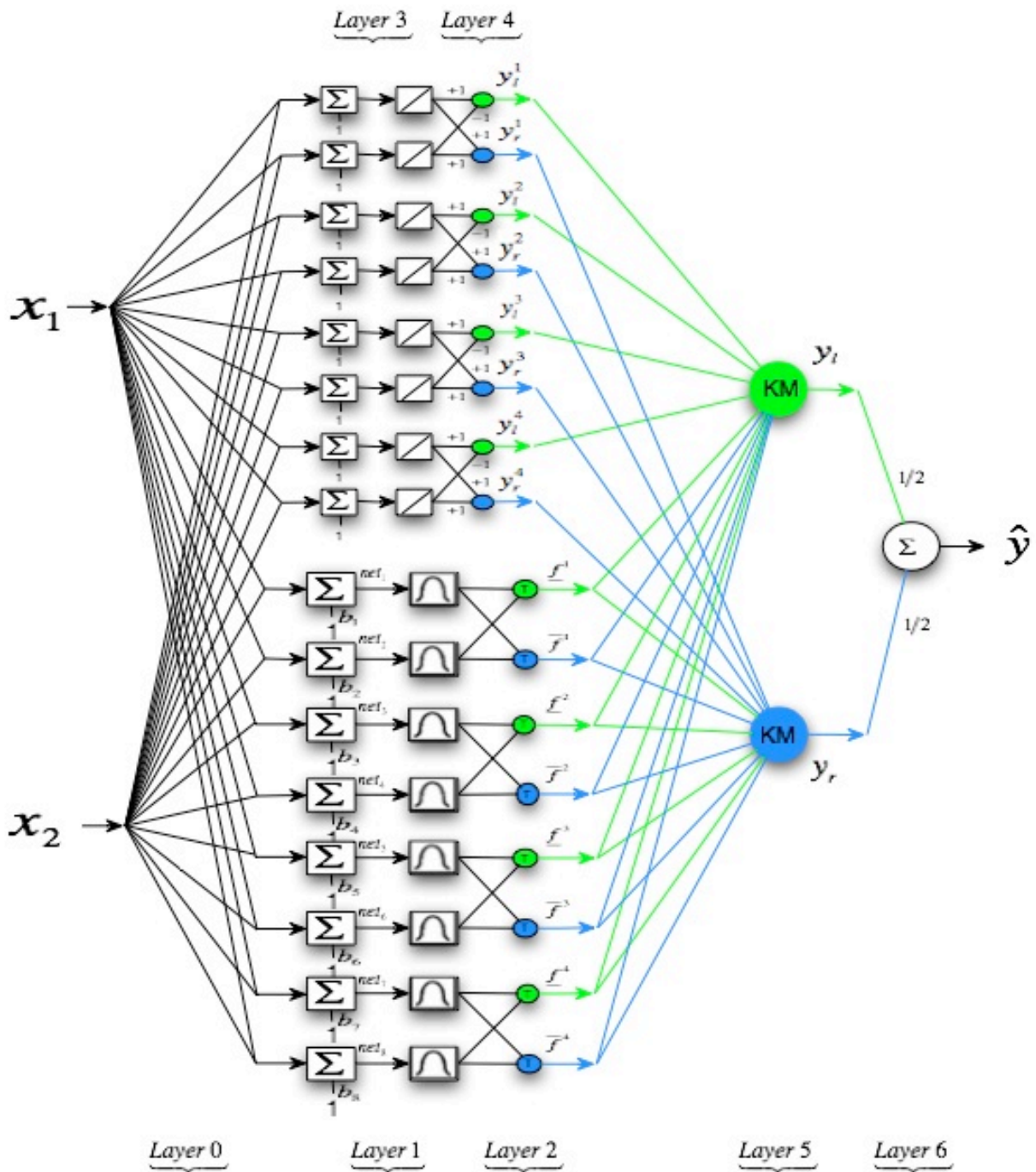


Figura 2.11 Arquitectura RBIT2FNN-2a:A2C1

Capa 0: Entradas

$$o_i^0 = x_i \quad ; i = 1, \dots, n$$

Capa 1: T1FN base radial

$$net_k^1 = \|w_k^{1,0} x - b_k^1\| \Leftrightarrow w_k^{1,0} = \frac{1}{\sqrt{2} \sigma_k^1} \quad ; \quad b_k^1 = \frac{c_k^1}{\sqrt{2} \sigma_k^1}$$

$$o_k^1 = \exp\left[-(net_k^1)^2\right] \quad \forall k = 1, \dots, 2r$$

Capa 2: T1FN. Fuerzas de disparo inferior-superior de las reglas

$$o_{2k-1}^2 = \underline{f}^k = o_{2k-1}^1 \cdot o_{2k}^1 \quad \forall k = 1, \dots, r$$

$$o_{2k}^2 = \overline{f}^k = o_{2k-1}^1 + o_{2k}^1 - o_{2k-1}^2$$

Las capas 3-5 de la arquitectura RBIT2FNN-2:A2C1 son iguales a las capas 4-6 de la arquitectura IT2FNN-2:A2C1.

2.2.1.7 Arquitectura IT2FNN-3:A2C1

Una IT2FNN-3:A2C1 (Figuras 2.12-2.13), es una red adaptativa neuro-difusa tipo-2 por intervalos con 8 capas, integra un modelo de inferencia difuso tipo-2 por intervalos de TSK de primer orden IT2FIS:A2C1 (Antecedentes difusos tipo-2 por intervalos y consecuentes tipo-1 por intervalos), en una red neuronal adaptativa. Las capas 0-3 de la arquitectura IT2FNN-3:A2C1 son iguales a las capas 0-3 de la arquitectura IT2FNN-2:A2C1.

Capa 4: Fuerzas de disparo izquierdo (f_l^k)-derecho (f_r^k) de las reglas. En esta capa podemos sustituirla por diversas estrategias para evaluar las fuerzas de disparo izquierdo-derecho que contribuirán en la reducción de tipo. A continuación proponemos las siguientes:

(i) Los valores de las fuerzas de disparo izquierdo-derecho se evalúan como una suma ponderada

de los pesos $\overline{\omega}_l^k, \underline{\omega}_l^k$ y $\overline{\omega}_r^k, \underline{\omega}_r^k$ y las fuerzas de disparo inferior ($\underline{f}^k$) – superior ($\overline{f}^k$) de

cada regla, estas contribuyen en el calculo de los valores $[\hat{y}_l, \hat{y}_r]$ en la reducci3n de tipo (Figura 2.12). Los pesos ajustan el nivel de incertidumbre de las fuerzas de disparo de las reglas.

$$o_{2k-1}^4 = f_l^k = \frac{\overline{\omega}_l \overline{f}^k + \underline{\omega}_l f^k}{\overline{\omega}_l + \underline{\omega}_l} \quad ; k = 1, \dots, r$$

$$o_{2k}^4 = f_r^k = \frac{\underline{\omega}_r f^k + \overline{\omega}_r \overline{f}^k}{\underline{\omega}_r + \overline{\omega}_r}$$

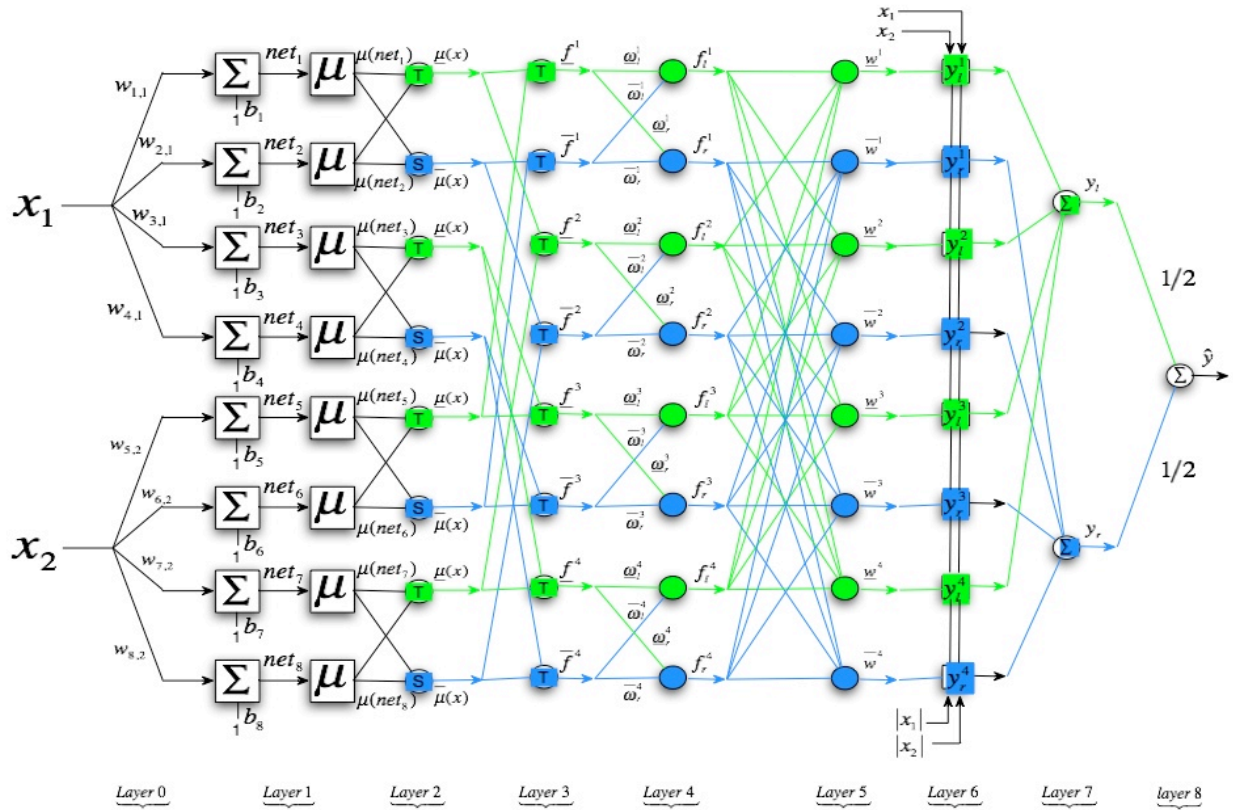


Figura 2.12 Arquitectura IT2FNN-3:A2C1

- (ii) Los valores de las fuerzas de disparo izquierdo-derecho se evalúan como una suma ponderada de los pesos $\omega_{k,j}^{4,3}$ y las fuerzas de disparo inferior (f^k) – superior ($\overline{f}^k$) de cada regla, estas contribuyen en el calculo de los valores $[\hat{y}_l, \hat{y}_r]$ en la reducci3n de tipo

(Figura 2.13). Los pesos ajustan el nivel de incertidumbre de las fuerzas de disparo de las reglas.

$$o_k^4 = \frac{\sum_{j=1}^{2r} \omega_{k,j}^{4,3} o_j^3}{\sum_{j=1}^{2r} \omega_{k,j}^{4,3}} ; k = 1, \dots, 2r \quad \begin{aligned} f_l^k &= o_{2k-1}^4 ; k = 1, \dots, r \\ f_r^k &= o_{2k}^4 \end{aligned}$$

- (iii) Los valores de las fuerzas de disparo izquierdo-derecho se evalúan utilizando una capa competitiva; el peso $\omega_{k,+}^{4,3}$ mas alto en el k-ésimo nodo tendrá la fuerza de disparo u_k que contribuirá en el calculo de los valores $[\hat{y}_l, \hat{y}_r]$ en la reducción de tipo (Figura 2.13).

$$u_k = \text{compet}(\omega_{k,+}^{4,3}) \quad \forall k = 1, \dots, 2r$$

$$o_{2k-1}^4 = f_l^k = o_{u_{2k-1}}^3 \quad \forall k = 1, \dots, r$$

$$o_{2k}^4 = f_r^k = o_{u_{2k}}^3$$

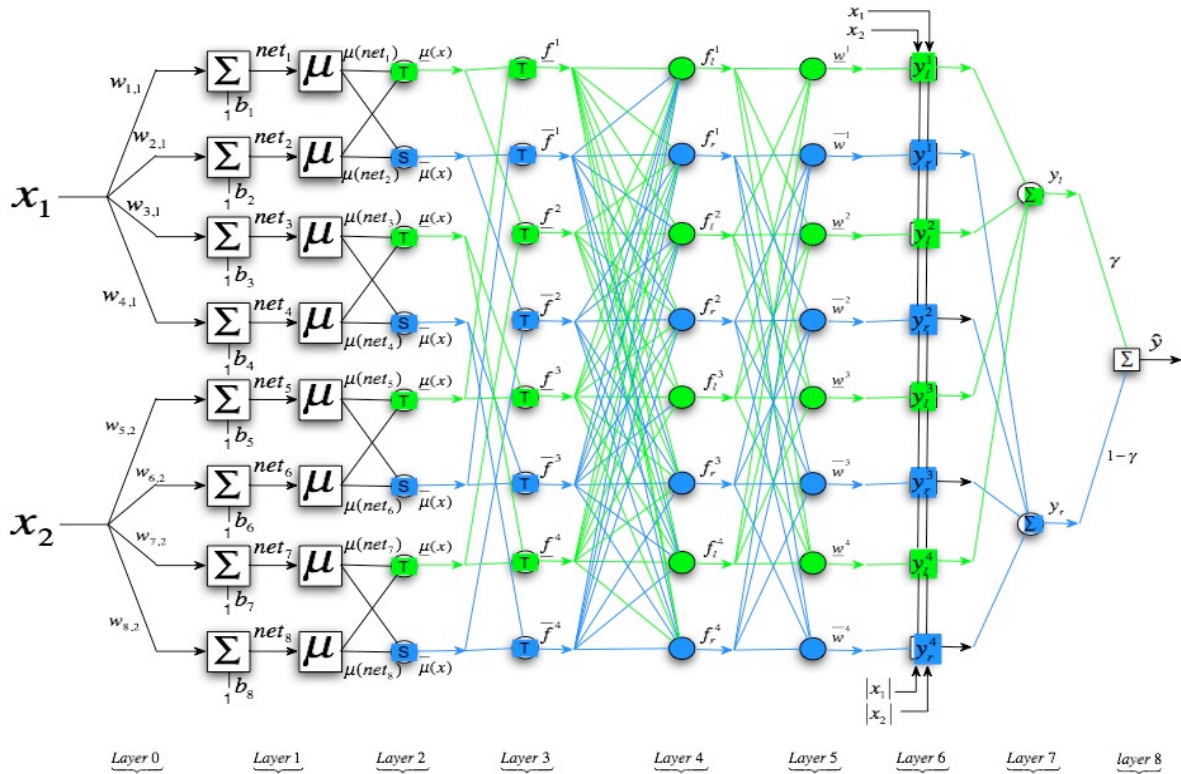


Figura 2.13 Arquitectura IT2FNN-3a:A2C1.

Capa 5: Normalización de las fuerzas de disparo izquierdo-derecho de las reglas. Los nodos de esta capa son no adaptativos. Se define la salida como la razón entre las fuerzas de disparo izquierdo-derecho de la k-ésima regla y la suma de las fuerzas de disparo izquierdo-derecho de todas las reglas, denominado fuerzas de disparo izquierdo-derecho normalizado:

$$o_{2k-1}^5 = \underline{w}^k = \frac{f_l^k}{\sum_{\ell=1}^r f_l^\ell} \quad ; \quad k=1, \dots, r$$

$$o_{2k}^5 = \overline{w}^k = \frac{f_r^k}{\sum_{\ell=1}^r f_r^\ell}$$

Capa 6: Consecuentes de las reglas. Cada nodo de esta capa es adaptativo y sus parámetros son $\{c_i^k, c_0^k\}$ y $\{s_i^k, s_0^k\}$. La salida del nodo corresponde a la salida parcial de la k-ésima regla, $\tilde{y}^k \in [y_l^k, y_r^k]$.

$$o_{2k-1}^6 = y_l^k = \sum_{i=1}^n c_i^k x_i + c_0^k - \sum_{i=1}^n s_i^k |x_i| - s_0^k \quad ; \quad k=1, \dots, r$$

$$o_{2k}^6 = y_r^k = \sum_{i=1}^n c_i^k x_i + c_0^k + \sum_{i=1}^n s_i^k |x_i| + s_0^k$$

Capa 7: Reducción de tipo. Calcula los valores izquierdo-derecho del intervalo $[\hat{y}_l, \hat{y}_r]$. Los nodos de esta capa son no adaptativos y sus salidas, $\hat{y}_l, \hat{y}_r$, se define como la sumatoria del producto de las fuerzas de disparo inferior-superior normalizadas por los consecuentes izquierdo-derecho de las reglas respectivamente:

$$o_1^7 = \hat{y}_l = \sum_{k=1}^r \underline{w}^k y_l^k$$

$$o_2^7 = \hat{y}_r = \sum_{k=1}^r \overline{w}^k y_r^k$$

Capa 8: Defuzzificación. El nodo de esta capa es adaptativo y su salida, $\hat{y}$, se define como un promedio ponderado de los valores izquierdo-derecho y el parámetro γ . El parámetro γ (valor por defecto 0.5) ajusta el intervalo de incertidumbre definido por los valores izquierdo-derecho $[\hat{y}_l, \hat{y}_r]$.

$$o_1^8 = \hat{y} = \gamma \hat{y}_l + (1 - \gamma) \hat{y}_r$$

2.2.1.8 Arquitectura RBIT2FNN-3:A2C1

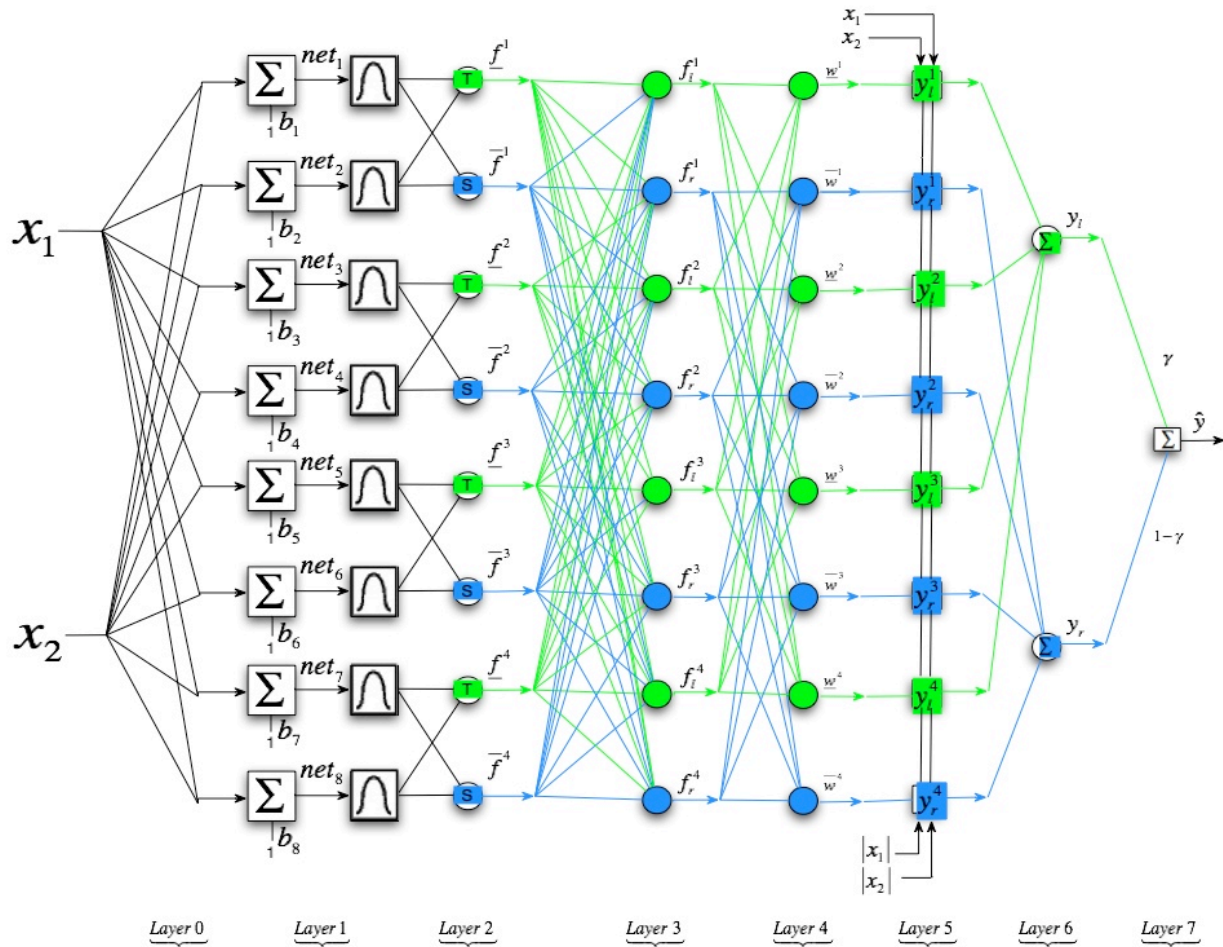


Figura 2.14 Arquitectura RBIT2FNN-3:A2C1

Una RBIT2FNN-3:A2C1 (Figura 2.14), es una red adaptativa neuro-difusa tipo-2 por intervalos de base radial con 7 capas, integra un modelo de inferencia difuso tipo-2 por intervalos de TSK de primer orden IT2FIS:A2C1 (Antecedentes difusos tipo-2 por intervalos y consecuentes difusos tipo-1 por intervalos), en una red neuronal adaptativa.

Las capas 0-2 de la arquitectura RBIT2FNN-3:A2C1 son iguales a las capas 0-2 de la arquitectura RBIT2FNN-2:A2C1. Las capas 3-7 de la arquitectura RBIT2FNN-3:A2C1 son iguales a las capas 4-8 de la arquitectura IT2FNN-3:A2C1.

2.2.2 Algoritmo de aprendizaje híbrido heurístico numérico.

Aunque puede utilizarse el algoritmo de aprendizaje de retro-propagación (BP: backpropagation) con técnicas de optimización heurística numérica [105] (Tabla 2.1) en base a gradientes para el entrenamiento de la red adaptativa IT2FNN para la identificación de parámetros, generalmente es el proceso computacional es muy costoso. Esto se debe a que los algoritmos de optimización, no hacen ninguna suposición sobre el tipo de función objetivo al que pertenecen los parámetros. Sin embargo, recordando que se realiza el aprendizaje de una salida a la vez, se puede observar que algunos de los parámetros de la salida de IT2FNN son lineales y otros no lo son. Hecho por el cual puede dividirse el conjunto total de parámetros S del sistema, en dos conjuntos:

- S : Conjunto total de parámetros del sistema.
- S_1 : Subconjunto de S que contiene los parámetros no-lineales ξ de S .
- S_2 : Subconjunto de S que contiene los parámetros lineales Θ de S o bien el subconjunto resultante de extraer S_1 de S .

Así pues, puede aplicarse un algoritmo al subconjunto de parámetros no-lineales S_1 y otro al

subconjunto de parámetros lineales S_2 , generando así, un algoritmo híbrido de aprendizaje.

Tabla 2.1 Técnicas de optimización heurística numérica.

Algoritmo	Nombre de la función
Levenberg-Marquardt (LM)	trainlm
BFGS Quasi-Newton (BFG)	trainbfg
Resilient Backpropagation (RP)	trainrp
Scaled Conjugate Gradient (SCG)	trainscg
Conjugate Gradient with Powell/Beale Restarts (CGB)	traincgb
Fletcher-Powell Conjugate Gradient (CGF)	traincgf
Polak-Ribière Conjugate Gradient (CGP)	traincgp
One-Step Secant (OSS)	trainoss
Gradient descent (GD)	traingd
Gradient Descent with Momentum (GDM)	traingdm
Gradient descent momentum and an adaptive learning rate (GDX)	traingdx
Gradient descent with adaptive learning rate (GDA)	traingda

En este trabajo se propone que el algoritmo híbrido de la red adaptativa IT2FNN considere los siguientes mecanismos alternativos para acelerar la identificación de los parámetros del sistema:

- **BP:** Todos los parámetros se modifican utilizando el algoritmo BP con técnicas de optimización heurística numérica.
- **BP + una pasada de RLS:** El método de RLS se aplica solamente en la primera época para estimar los valores iniciales de los consecuentes y en las siguientes etapas se modifican todos los parámetros por técnicas de optimización heurística numérica.

- **Algoritmo híbrido BP+RLS:** Se utiliza el método RLS para actualizar los consecuentes y técnicas de optimización heurística numérica para actualizar los parámetros restantes (antecedentes, reducción de tipo y defuzzificación).

En el algoritmo híbrido BP+RLS, cada época esta compuesta por una pasada hacia adelante y una pasada hacia atrás. En la pasada hacia adelante, para cada vector de entrada, se identifican los parámetros de los consecuentes mediante RLS. A continuación, se calculan los errores para cada par del conjunto de entrenamiento y, en la pasada hacia atrás se propagan las señales del error y los parámetros de las premisas son modificados por el mecanismo de backpropagation. En la tabla 2.2 se resume este procedimiento:

Tabla 2.2 Procedimiento de aprendizaje de la red adaptativa IT2FNN.

	Hacia adelante	Hacia atrás
Antecedentes, reducción tipo y defuzzificación	<i>Fijas</i>	<i>Descenso por gradientes</i>
Consecuentes	<i>RLS</i>	<i>Fijos</i>
Señales	<i>Salidas</i>	<i>Errores</i>

Los parámetros de los consecuentes identificados por el método RLS son óptimos bajo la condición que los parámetros de los antecedentes estén fijos. El enfoque híbrido, por lo tanto, converge mas rápidamente que los demás métodos de BP, debido a que se reducen las dimensiones del espacio de búsqueda.

Para demostrar el procedimiento general del algoritmo híbrido BP+RLS utilizaremos la red adaptativa IT2FNN mas sencilla en diversas arquitecturas (IT2FNN-1:A2C0, IT2FNN-2:A2C0 y RBIT2FNN-2:A2C0), mostradas en las Figuras 2.4-2.6 respectivamente. Además se supone

un conjunto de entrenamiento $\{x_{p,1}, \dots, x_{p,i}, \dots, x_{p,n}; y_p\}_{p=1}^q$ compuesto por q patrones y una medida de error $E_p = \frac{1}{2} e_p^2$ para el p-ésimo patrón; donde $e_p = y_p - \hat{y}_p$ y $\hat{y}_p$ es la salida que produce la red para la p-ésima entrada $[x_{p,1}, \dots, x_{p,i}, \dots, x_{p,n}]$ definida por las ecuaciones;

$$\hat{y}_p = \gamma \sum_{k=1}^r \underline{w}_p^k \cdot y_p^k + (1 - \gamma) \sum_{k=1}^r \overline{w}_p^k \cdot y_p^k \quad (2.4)$$

donde

$$\underline{w}_p^k = \frac{f_p^k}{\sum_{\ell=1}^r f_p^\ell} \quad ; \quad k = 1, \dots, r \quad (2.5)$$

$$\overline{w}_p^k = \frac{\overline{f}_p^k}{\sum_{\ell=1}^r \overline{f}_p^\ell}$$

$$y_p^k = \sum_{i=1}^n c_i^k x_{p,i} + c_0^k \quad ; \quad k = 1, \dots, r \quad (2.6)$$

donde $\underline{w}_p^k$, $\overline{w}_p^k$ son las fuerzas de disparo normalizadas para el p-ésimo patrón de entrada y y^k la k-ésima regla de los consecuentes. El objetivo entonces, es minimizar la función error (**sse**), definida como;

$$sse \equiv E = \sum_{p=1}^q E_p = \frac{1}{2} \sum_{p=1}^q (y_p - \hat{y}_p)^2 \quad (2.7)$$

En la fase de propagación hacia adelante, los *parámetros de los antecedentes* se mantienen fijos, la salida puede ser expresada de la misma forma que lo indica el formato de las reglas del modelo de TSK, como una combinación lineal de las variables de entrada y las fuerzas de disparo inferior-superior normalizadas. La salida $\hat{y}$ puede ser reescrita entonces de la siguiente manera:

$$\hat{Y} = [\gamma \underline{A} + (1 - \gamma) \overline{A}] \Theta = A \Theta \quad (2.8)$$

donde A es la matriz que contiene la matriz $\underline{A}$ con los valores de las fuerzas de disparo inferiores normalizadas y los valores de las variables de entrada; la matriz $\bar{A}$ los valores de las fuerzas de disparo superiores y las variables de entrada, además del parámetro γ . Las matrices $\underline{A}$, $\bar{A}$, el vector de parámetros de los consecuentes Θ y el vector de datos deseados Y , se definen como;

$$\underline{A}_{p,:} = [\underline{w}_p^1 \dots \underline{w}_p^r \quad \underline{w}_p^1 x_{p,1} \dots \underline{w}_p^r x_{p,1} \quad \dots \quad \underline{w}_p^1 x_{p,i} \dots \underline{w}_p^r x_{p,i} \quad \dots \quad \underline{w}_p^1 x_{p,n} \dots \underline{w}_p^r x_{p,n}] \quad ; qxr(n+1)$$

$$\bar{A}_{p,:} = [\bar{w}_p^1 \dots \bar{w}_p^r \quad \bar{w}_p^1 x_{p,1} \dots \bar{w}_p^r x_{p,1} \quad \dots \quad \bar{w}_p^1 x_{p,i} \dots \bar{w}_p^r x_{p,i} \quad \dots \quad \bar{w}_p^1 x_{p,n} \dots \bar{w}_p^r x_{p,n}] \quad ; qxr(n+1)$$

$$\Theta = [c_0^1 \dots c_0^r \quad c_1^1 \dots c_1^r \quad \dots \quad c_i^1 \dots c_i^r \quad \dots \quad c_n^1 \dots c_n^r]^T \quad ; r(n+1) \times 1$$

$$Y = [y_1, \dots, y_p, \dots, y_q]^T \quad ; qx1$$

En la mayoría de los casos, el numero de pares de entrenamiento (q) es mayor que el numero de parámetros a estimar $r(n+1)$. En estos casos, el sistema puede no tener solución exacta. Por lo tanto, la ecuación se reemplaza por: $A\theta + e = Y$. Se puede demostrar que esta ecuación se minimiza cuando $\Theta = \Theta^*$, denominado el valor LSE, que satisface la ecuación normal: $(A^T A)\Theta^* = A^T Y$. Si $A^T A$ es no singular, Θ^* es único y esta dado por la ecuación;

$$\Theta^* = (A^T A)^{-1} A^T Y = A^T (AA^T)^{-1} Y \quad (2.9)$$

donde

$$\begin{aligned} A &= \gamma \underline{A} + (1 - \gamma) \bar{A} \\ A^T &= \gamma \underline{A}^T + (1 - \gamma) \bar{A}^T \end{aligned} \quad (2.10)$$

Aunque la ecuación para la solución de Θ se escribe en forma muy concisa, es computacionalmente costosa debido al calculo de la inversa de $A^T A$. Mas aun, la ecuación no

esta definida si $A^T A$ es singular. Para evitar estos problemas, se pueden emplear formulas iterativas que dan lugar a mecanismos mas eficientes para el calculo del LSE denominado **RLS (Recursive Least Square)**.

Si U_i^T es el i-ésima renglón de la matriz A , V_i^T es el i-ésimo elemento de Y ; Θ^* se calcula por medio del método iterativo RLS:

$$\begin{aligned} S_{i+1} &= S_i - \frac{S_i U_{i+1} U_{i+1}^T S_i}{1 + U_{i+1}^T S_i U_{i+1}} \\ \Theta_{i+1} &= \Theta_i + S_{i+1} U_{i+1} (V_{i+1}^T - U_{i+1}^T \Theta_i) \end{aligned} \quad (2.11)$$

donde $\Theta_0 = 0$, $S_0 = \kappa \cdot I$ y κ es un entero positivo grande.

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

En los casos en los que los pares $\{x_{p,1}, \dots, x_{p,i}, \dots, x_{p,n}; y_p\}_{p=1}^q$ son obtenidos on-line, se introduce un factor λ , denominado *factor de olvido*, que otorga mas importancia a los datos mas recientes. En este caso, las ecuaciones del método iterativo RLS son las siguientes:

$$\begin{aligned} S_{i+1} &= \lambda \left(S_i - \frac{S_i U_{i+1} U_{i+1}^T S_i}{\lambda + U_{i+1}^T S_i U_{i+1}} \right) \\ \Theta_{i+1} &= \Theta_i + S_{i+1} U_{i+1} (V_{i+1}^T - U_{i+1}^T \Theta_i) \end{aligned} \quad (2.12)$$

donde λ generalmente pertenece al intervalo $[0.85, 1]$. Mientras menor sea λ , mas rápido decae el efecto de los datos mas viejos. Sin embargo, la elección de un λ demasiado pequeño causa inestabilidad numérica y debe ser evitada.

El la fase de retro-propagación se calculan los errores para p-ésimo conjunto de entrenamiento e_p y se propagan hacia atrás las señales del error $\in_p^\ell$ y los parámetros no lineales ξ (antecedentes ϕ , reducción de tipo y defuzzificación) son modificados por el mecanismo de aprendizaje BP. El mecanismo de aprendizaje especifica como deben ser modificados los

parámetros de los nodos adaptativos de la red para minimizar una medida preestablecida del error (**sse**) entre sus salidas y las deseadas. La regla básica de aprendizaje de las redes adaptativas es la del descenso por el gradiente, donde el vector gradiente $\nabla E = \frac{\partial^+ E}{\partial \xi}$ es calculado mediante la aplicación sucesiva de la regla de la cadena. Para sistematizar la obtención de ∇E se define a continuación la derivada indirecta:

Derivada indirecta

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

$$\frac{\partial^+ n}{\partial m} = \begin{cases} 0 & \text{si } fan_in(n) = 0 \\ 1 & \text{si } n = m \\ \sum_{y \in Y} \frac{\partial f_n}{\partial y} \cdot \frac{\partial^+ y}{\partial m} & \text{de otra manera} \end{cases} \quad (2.13)$$

donde $Y = \{y \in V \mid (y, n) \in E\}$, por simplicidad, se utiliza n para denotar tanto a un nodo genérico como a la variable que representa su salida.

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

En implementaciones reales, donde se necesitan calcular las derivadas parciales de la función error con respecto a los diferentes parámetros $\frac{\partial^+ E}{\partial \xi}$, es importante aprovechar los resultados intermedios para no repetir cálculos y degradar el rendimiento del sistema.

La señal de error $\in_{i_\ell}^\ell$ se define como $\in_{i_\ell}^\ell = \frac{\partial^+ E_p}{o_{i_\ell}^\ell}$. La señal de error en la capa de salida **L** y en

las capas ocultas ℓ se evalúa con la siguiente formula recursiva:

$$\in_{i_\ell}^\ell = \begin{cases} \frac{\partial E_p}{\partial o_{i_\ell}^L} & \text{si } \ell = L \\ \sum_{i_{\ell+1}}^{\ell+1} \in_{i_{\ell+1}}^{\ell+1} \cdot \frac{\partial o_{i_{\ell+1}}^{\ell+1}}{\partial o_{i_\ell}^\ell} & \text{si } \ell \neq L \end{cases} \quad (2.14)$$

Los parámetros genéricos ξ se modifican de acuerdo a una técnica de optimización numérica, por ejemplo el gradiente descendente (GD), donde las ecuaciones de actualización de los parámetros son:

$$\xi_{new} = \xi^{old} - \eta \cdot \frac{\partial^+ E}{\partial \xi} \quad (2.15)$$

donde

$$\frac{\partial^+ E}{\partial \xi} = \sum_p \frac{\partial^+ E_p}{\partial \xi} \quad (2.16)$$

$$\frac{\partial^+ E_p}{\partial \xi} = \in_{i_\ell}^\ell \cdot \frac{\partial o_{i_\ell}^\ell}{\partial \xi} \quad (2.17)$$

Si todos los τ parámetros $\theta \in [\xi, \Theta]$ de la red adaptativa IT2FNN se modifican utilizando el algoritmo BP con las técnicas de optimización numérica de la Tabla 2.1, se necesita evaluar la

matriz jacobiana $J(\theta)$, la matriz hessina $H(\theta)$ y el vector gradiente $g(\theta)$ de acuerdo a las siguientes ecuaciones:

$$g(\theta) = J^T(\theta) \cdot e(\theta) \quad (2.18)$$

$$H(\theta) \approx J^T(\theta) \cdot J(\theta) \quad (2.19)$$

donde la matriz jacobiana se define en función del error e_p de la red adaptativa IT2FNN como;

$$J = \begin{bmatrix} \frac{\partial e_1}{\partial \theta_1} & \cdots & \frac{\partial e_1}{\partial \theta_j} & \cdots & \frac{\partial e_1}{\partial \theta_\tau} \\ \vdots & \cdots & \vdots & \cdots & \vdots \\ \frac{\partial e_p}{\partial \theta_1} & \cdots & \frac{\partial e_p}{\partial \theta_j} & \cdots & \frac{\partial e_p}{\partial \theta_\tau} \\ \vdots & \cdots & \vdots & \cdots & \vdots \\ \frac{\partial e_q}{\partial \theta_1} & \cdots & \frac{\partial e_q}{\partial \theta_j} & \cdots & \frac{\partial e_q}{\partial \theta_\tau} \end{bmatrix} \quad (2.20)$$

En el apéndice A.1 se listan los pseudo-códigos de algunas técnicas de optimización heurística numérica para el entrenamiento BP de las arquitecturas IT2FNN-1:A2C0, IT2FNN-2:A2C0, IT2FNN-1:A2C1, IT2FNN-2:A2C1 y IT2FNN-3:A2C1 [51, 54].

2.2.3 Algoritmo de aprendizaje híbrido con HGA.

En un algoritmo genético jerárquico (HGA), se mantiene el cómputo del GA convencional pero la estructura genética del cromosoma se organiza jerárquicamente. Los HGA operan con una estructura adaptable que permite al algoritmo explorar en un espacio mucho más grande y le permite encontrar mejores soluciones. El cromosoma consiste ahora de genes paramétricos y genes de control. La activación de los genes paramétricos es gobernada por el valor de los genes del control, si el gen de control está en '1' entonces el gen paramétrico que éste gobierna estará

activo; un '0' desactiva el gen paramétrico (anula esa característica en el fenotipo). Un cambio en los genes de control impacta en los genes nivelados más abajo en la jerarquía cuando se evalúa y decodifique el genotipo. Ésta es la razón de por qué el HGA puede alcanzar una arquitectura del sistema mínimo. Esta estructura jerárquica implica que el cromosoma tiene más información que en las codificaciones de un GA convencional, técnica que permite obtener una solución con el número mínimo de funciones de membresía tipo-2 por intervalos (IT2MF), reglas y sintonizar los parámetros de la red adaptativa IT2FNN. En el caso particular de entrenar evolutivamente una IT2FNN debemos considerar la estructura jerárquica del cromosoma en un HGA, esta deberá contener la información del IT2FIS de TSK y los parámetros genéticos.

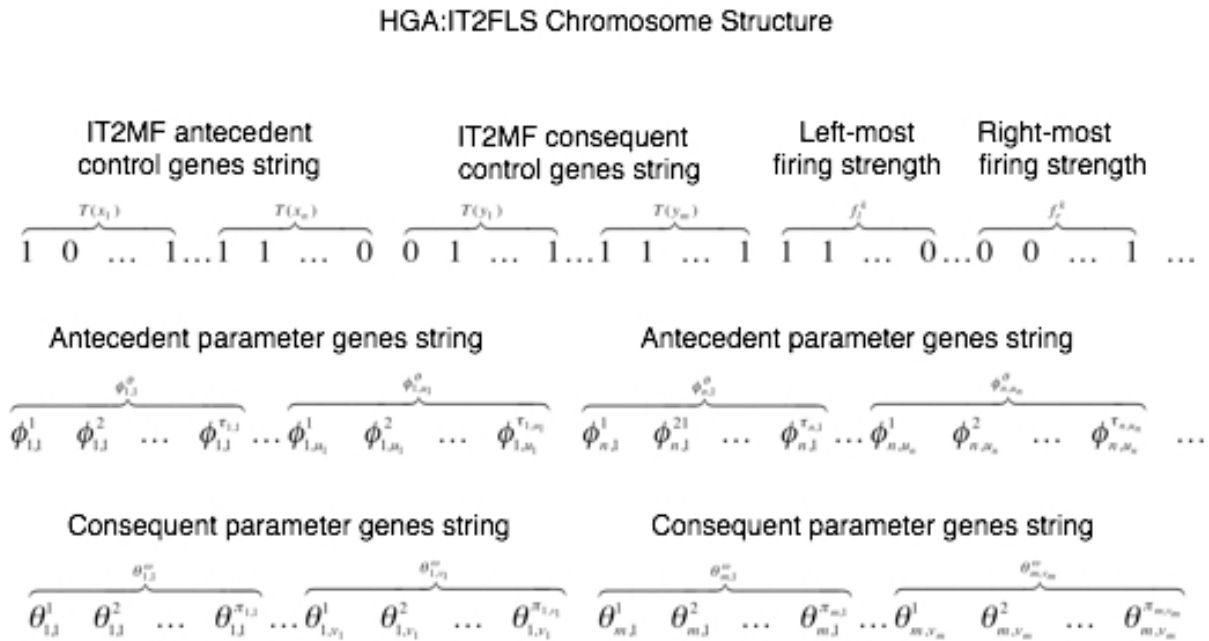


Figura 2.15 Estructura jerárquica de un cromosoma para codificar un HGA:IT2FLS

El cromosoma para codificar un HGA-IT2FNN (Figura 2.15) consiste en genes paramétricos (parámetros de las IT2MF de los antecedentes-consecuentes y parámetros de mutación-cruce) y genes de control (Activación de las IT2MF de los antecedentes-consecuentes y las fuerzas de

disparo inferior y superior). La activación de las fuerzas de disparo de las reglas f_l^k y f_r^k es gobernada por el valor de los genes de control, si el gen de control está en '1' entonces el gen paramétrico que éste gobierna activa la fuerza de disparo superior $\overline{f}^k$; un '0' activa el gen paramétrico que gobierna la fuerza de disparo inferior $\underline{f}^k$.

El algoritmo de aprendizaje híbrido HGA-IT2FNN, utiliza un HGA típico codificando toda la información de una IT2FNN en un cromosoma. En cada generación se evalúa la función objetivo compuesta por una pasada hacia adelante de la IT2FNN. En la pasada hacia adelante, para cada vector de entrada, se identifican los parámetros de la IT2FNN en forma evolutiva. Una alternativa de cómputo más eficaz es codificar la IT2FNN parcialmente (excepto los consecuentes). En cada generación al evaluar la función objetivo se actualizan los consecuentes usando el algoritmo RLS (HGA-IT2FNN+RLS)

2.2.4 Algoritmo de aprendizaje híbrido con PSO.

En la terminología utilizada en PSO, Kennedy y Eberhart [88, 89] introducen el término general partícula o agente para representar a los peces, pájaros, abejas, hormigas o cualquier otro tipo de individuos que exhiban un comportamiento social como grupo, en forma de una colección de agentes que interactúan entre sí. De acuerdo con los fundamentos teóricos del método, el movimiento de cada una de estas partículas hacia un objetivo común en dos dimensiones está condicionado por dos factores básicos, la memoria autobiográfica de la partícula o nostalgia y la influencia social de todo el enjambre. A nivel computacional, como método de optimización, esta filosofía puede extenderse a un espacio N -dimensional de acuerdo con el problema bajo análisis. La posición instantánea de cada una de las partículas de la población en el espacio N -dimensional representa una solución potencial, siendo N el número de

incógnitas del problema original. Básicamente, el proceso evolutivo se reduce a mover cada partícula dentro del espacio de soluciones con una velocidad que variará de acuerdo a su velocidad actual, a la memoria de la partícula y a la información global que comparte el resto del enjambre, utilizando una función de aptitud para cuantificar la calidad de cada partícula en función de la posición que ésta ocupe.

En la formulación de PSO se define la velocidad de partícula como el único operador disponible para controlar la evolución de la optimización. De forma genérica, supongamos una población de n_s partículas donde cada partícula del enjambre se identifica con dos variables de estado inicializadas aleatoriamente dentro del espacio n_x -dimensional que establece el problema a optimizar: un vector velocidad, $v_i = (v_{i,1}, \dots, v_{i,j}, \dots, v_{i,n_x})$, y un vector de posición, $x_i = (x_{i,1}, \dots, x_{i,j}, \dots, x_{i,n_x})$. Este último corresponde a una solución potencial al problema de optimización. Los límites de los parámetros a optimizar, $x_i(0) \sim U(x_{\min}, x_{\max})$, conforman en su conjunto el espacio de búsqueda al cual debe restringirse el movimiento del enjambre. Adicionalmente, cada partícula mantiene en memoria información de la posición espacial asociada con la mejor solución históricamente visitada por ésta, $y_i = (y_{i,1}, \dots, y_{i,j}, \dots, y_{i,n_x})$, y también conoce la posición de la mejor partícula o solución encontrada por todos sus congéneres, $\hat{y} = (\hat{y}_1, \dots, \hat{y}_j, \dots, \hat{y}_{n_x})$. El movimiento del enjambre se realiza en pasos temporales, que se traducen a nivel de algoritmo en iteraciones contiguas.

En cada iteración, t , cada una de las partículas de la población recorre el espacio de soluciones con una velocidad v_i hacia nuevas posiciones x_i , de acuerdo con su propia experiencia y_i , y con la experiencia aportada por el mejor de sus convecinos, $\hat{y}$. En las primeras versiones del algoritmo [88], esta formulación se reduce a las siguientes ecuaciones;

$$v_{i,j}(t+1) = v_{i,j}(t) + c_1 r_{1,j}(t)[y_{i,j}(t) - x_{i,j}(t)] + c_2 r_{2,j}(t)[\hat{y}_j(t) - x_{i,j}(t)] \quad (2.21)$$

$$x_i(t+1) = x_i(t) + v_i(t+1) \quad (2.22)$$

En la ecuación (2.21), $v_{i,j}(t)$ y $x_i(t)$ representan, respectivamente, la velocidad y posición en la iteración o instante de tiempo t de la partícula i en la dimensión n_x -ésima del espacio de búsqueda. Los factores c_1 y c_2 son las denominadas constantes de aceleración cognitiva y social, que determinan en que medida influyen sobre el movimiento de la partícula su propia memoria y la cooperación entre individuos, respectivamente. Los términos $r_{1,j}(t), r_{2,j}(t) \sim U(0,1)$, son dos números aleatorios uniformemente distribuidos entre 0 y 1, cuyo objetivo es emular el comportamiento estocástico y un tanto impredecible que exhibe la población del enjambre. Después de calcular la nueva velocidad de la partícula i en la dimensión n_x , la nueva posición $x_{i,j}(t+1)$ se actualiza directamente de acuerdo con la ecuación (2.22), donde se asume que la velocidad se aplica durante un cierto período de tiempo típicamente de valor unitario. El proceso descrito se extiende al espacio n_x -dimensional, de forma que se van componiendo iterativamente nuevos vectores de posición $x_i(t)$, utilizando, como en cualquier otro método de cómputo evolutivo, una función de aptitud para ponderar la calidad de dicha solución parcial, actualizando los vectores y_i y $\hat{y}$ si se detectan resultados mejores (Apéndice A.2).

El movimiento de los agentes sobre el espacio de soluciones y , en consecuencia, el rendimiento del algoritmo, está condicionado por el grado de contribución de las tres componentes de la velocidad en la ecuación (2.21), denominadas por este orden como momento, hábito o inercia para considerar la tendencia de la partícula; memoria, nostalgia o autoaprendizaje para incluir la experiencia de la propia partícula, y cooperación, conocimiento social, conocimiento de grupo o información compartida, para reflejar el intercambio de información y el comportamiento social como grupo.

Para acotar la velocidad de la partícula se especifica un valor máximo, $v_{\max}$, que restringe la velocidad en cada dimensión al intervalo $[-v_{\max}, v_{\max}]$. Si el valor de $v_{\max}$ es demasiado grande las partículas pueden sobrepasar e ignorar continuamente la zona con la solución global. En el extremo opuesto, si $v_{\max}$ toma valores extremadamente pequeños las partículas explorarán el espacio de soluciones muy lentamente y podrán quedar atrapadas alrededor de soluciones locales, incapaces de librarse de la base de atracción. Sin embargo, si no se introdujese el parámetro de control $v_{\max}$, el enjambre no convergería hacia un punto, sino que sufriría el fenómeno conocido como explosión del PSO, consistente en un comportamiento oscilatorio y creciente de la posición de las partículas, careciendo el método de potencial alguno como algoritmo de optimización.

El efecto de $v_{\max}$, fuertemente vinculado con la naturaleza del problema a optimizar, le convierte en el parámetro principal a sintonizar en PSO, sin ninguna regla aparente que dicte las pautas a seguir en su elección para asegurar la convergencia del enjambre. Con el objetivo de reducir el efecto de $v_{\max}$ y perfeccionar el control del alcance de la búsqueda sobre el espacio de soluciones, surgen versiones mejoradas del algoritmo que incorporan el concepto de peso inercial [91]. Básicamente, para cada instante temporal, llámese iteración en el caso computacional, en el cálculo de la nueva velocidad de partícula en cada dimensión, $v_{in}(k+1)$, el modelo de peso inercial introduce como factor el así denominado peso inercial, w , tal que

$$v_{i,j}(t+1) = w \cdot v_{i,j}(t) + c_1 r_{1,j}(t)[y_{i,j}(t) - x_{i,j}(t)] + c_2 r_{2,j}(t)[\hat{y}_j(t) - x_{i,j}(t)] \quad (2.23)$$

donde $w \in [0,1]$ para favorecer la convergencia de la partícula. Con este factor se controla la tendencia de la partícula a continuar en la dirección en la que se estaba moviendo y se regula la relación entre capacidad de exploración del espacio de búsqueda y habilidad de convergencia hacia las soluciones locales o global, dictadas por el segundo y tercer sumando en la ecuación

(2.23). En el modelo de PSO con peso inercial la trascendencia del factor $v_{\max}$ se reduce, asignándole en unos casos el valor del rango dinámico de cada variable x_n , y en otros la mitad de dicho rango [89, 91]; prescindiendo así de la necesidad de sintonizarlo. Aún en el modelo con peso inercial y dependiendo del problema a optimizar, el límite $v_{\max}$ puede resultar imprescindible para evitar que las partículas oscilen demasiado rápido en las proximidades de una región de interés sin explorarla adecuadamente.

Una alternativa al modelo inercial lo constituye el denominado factor de constricción introducido por Clerc en [90]. Partiendo de la necesidad de encontrar un razonamiento a la necesidad del límite $v_{\max}$ y a la explosión del PSO, se iniciaron estudios para analizar la trayectoria que describen las partículas. La formulación de Ozcan y Mohan en [91] propone que las partículas se desplazan describiendo ondas sinusoidales, cambiando su frecuencia y amplitud aleatoriamente en su búsqueda de la posición óptima. En [90, 91] se incluye una formulación mucho más exhaustiva, que deriva en la obtención de un modelo generalizado para PSO en el cual se puede controlar la explosión y la convergencia del algoritmo. El modelo propuesto por Clerc [91] más utilizado es el descrito como constricción de tipo 1, regido por las ecuaciones

$$v_{i,j}(t+1) = \chi \cdot \{v_{i,j}(t) + \varphi_1 r_{1,j}(t)[y_{i,j}(t) - x_{i,j}(t)] + \varphi_2 r_{2,j}(t)[\hat{y}_j(t) - x_{i,j}(t)]\} \quad (2.24)$$

donde χ es el denominado coeficiente de constricción, el cual se expresa como

$$\chi = \frac{2\kappa}{2 - \varphi - \sqrt{\varphi^2 - 4\varphi}} \quad ; \kappa \in [0,1] \quad , \quad \varphi = \varphi_1 + \varphi_2 \quad , \quad \varphi > 4 \quad (2.25)$$

Tal y como se indica en la ecuación (2.25), el coeficiente de constricción no está definido para $\varphi \leq 4$. A medida que φ toma valores por encima de 4.0, χ decrece paulatinamente, acentuando el efecto de amortiguamiento del movimiento. En este modelo de constricción típicamente se utilizan valores de $\kappa=1$ y $\varphi=4.1$ ($\varphi_1=\varphi_2=2.05$), lo cual reporta un valor para el coeficiente de

constricción de $\chi \approx 0.729$. Algebraicamente, esta constricción es equivalente a utilizar el modelo inercial de la ecuación (2.23) con $w=0.729$ y $c_1=c_2=1.49445$. La utilización de ambos modelos de PSO, con peso inercial y con factor de constricción, está muy extendida en la bibliografía y aunque es el modelo de Clerc el que ofrece mejores resultados [88-91], los parámetros óptimos del algoritmo están íntimamente ligados con la naturaleza del problema físico a optimizar, tal y como ya se comentó con anterioridad.

Selección de parámetros

Como en cualquier otro método de cómputo evolutivo, en PSO el ajuste de los parámetros condiciona el rendimiento posterior del algoritmo de optimización. Aunque existen múltiples estudios que intentan generalizar la selección de estos parámetros [91], dicha selección está íntimamente ligada con la naturaleza del problema a optimizar, siendo preciso lograr un balance óptimo entre exploración y convergencia. Al margen de los parámetros intrínsecos del algoritmo, debe resaltarse la importancia que adquiere la función de aptitud. La utilización de una función de aptitud que no introduzca una métrica adecuada para pesar la bondad de cada partícula hará emerger carencias del PSO que erróneamente pueden asociarse con el algoritmo en sí, aún siendo más propias de los defectos de la propia función de aptitud. En esta sección se resumen los valores típicos para los parámetros del algoritmo.

En lo referente al peso inercial, se utilizan comúnmente diferentes valores dependiendo del compromiso que se establezca entre exploración y rapidez de convergencia. Un peso inercial alto reduce la influencia de la nostalgia y de la cooperación sobre el movimiento de las partículas, favoreciendo la exploración. Por el contrario, un peso inercial muy bajo promueve la convergencia hacia la solución propuesta por la propia partícula, y por la mejor solución de conjunto. Se sugiere un peso inercial constante de valor 0.8 y se utiliza un valor constante para w

de 0.4. Por otra parte, para aunar exploración y convergencia del espacio de soluciones, se suele utilizar una variante con un peso inercial que decrece linealmente entre $w_{\max}$ y $w_{\min}$ durante el transcurso de la simulación, de acuerdo con la ecuación (2.26), con valores típicos de 0.9 y 0.4 para $w_{\max}$ y $w_{\min}$, respectivamente.

$$w = w_{\max} - (w_{\max} - w_{\min}) \cdot \frac{t}{n_t} \quad (2.26)$$

El inconveniente del peso inercial decreciente está relacionado con la selección del número de iteraciones. Si el número máximo de iteraciones es demasiado grande, el PSO puede quedarse estancado a la espera de que el peso inercial decrezca para converger hacia las regiones de interés. Por el contrario, un número excesivamente reducido de iteraciones puede forzar la convergencia prematura hacia una solución local. Paralelamente al peso inercial decreciente, se propone una nueva alternativa basada en utilizar un peso inercial creciente e inspirada en el comportamiento del recocido simulado. Para eliminar en cierta medida la dependencia del número de iteraciones y combinar exploración y convergencia, se considera un peso inercial aleatorio de valor $w=0.5+rnd/2.0$, donde *rnd* representa a un número aleatorio con distribución uniforme $U[0,1]$.

La selección del peso inercial no se debe dissociar de la elección de las constantes de aceleración cognitiva y social c_1 y c_2 , respectivamente. Se considera un valor de 2.0 para ambas tasas de aprendizaje, con el objetivo de que estos factores estocásticos tengan en media un valor de 1.0, de forma que los agentes sobrevuelen el objetivo la mitad del tiempo. Esta versión suele combinarse con el peso inercial lineal decreciente. Como alternativa, se combina el peso inercial de valor $w=0.5+rnd/2.0$ con un valor de 1.49445 para c_1 y c_2 .

Trasladando los principios del modelo inercial al modelo propuesto por Clerc, la selección de w , c_1 y c_2 se traduce a los parámetros χ , φ_1 y φ_2 . Como extensión a la construcción más utilizada,

con valores $\chi=0.729$ y $\varphi_1=\varphi_2=2.05$, se plantean alternativas que muestran un rendimiento superior para problemas muy concretos [90, 91], manteniendo el mismo valor para el factor de constricción ($\chi=0.729$), y con valores $\varphi_1=2.8$ y $\varphi_2=1.3$.

Otro parámetro que precisa ser seleccionado con extrema rigurosidad es el tamaño de la población. Poblaciones excesivamente grandes exploran minuciosamente el espacio de búsqueda, pero como contrapartida, el coste computacional asociado con el aumento del número de evaluaciones de la función de aptitud puede resultar excesivamente elevado. Estudios paramétricos revelan que una población de alrededor de 30 partículas es suficiente para múltiples problemas, y típicamente se utilizan poblaciones que oscilan entre 10 y 50 partículas, o entre 100 y 200 partículas para abordar problemas más complejos [90, 91].

2.3 Aproximador universal.

La red adaptativa IT2FNN puede ser utilizada como un aproximador universal de cualquier tipo de función. Esta sección lleva a cabo tal demostración con el fin de presentar las bases y fundamentos para su posterior aplicación.

La IT2FNN presenta una característica interesante cuando se utilizan reglas TSK de primer orden por intervalos, las cuales tienen un poder ilimitado de aproximación de funciones no-lineales a partir de patrones de datos. Sin embargo, deben cumplirse ciertos requerimientos que establece el teorema de Stone-Weierstrass [92, 93] para la estimación de dichas funciones, el cual se presenta a continuación.

Teorema de Stone – Weierstrass

Sea el dominio $\mathbf{D}$ un espacio compacto de N dimensiones y sea $\mathfrak{S}$ un conjunto de funciones continuas evaluadas en el dominio real en $\mathbf{D}$, satisfaciendo los siguientes criterios:

1. **Función Identidad:** La constante $f(x) = 1$ está en $\mathfrak{F}$.
2. **Separabilidad:** Para dos puntos $x_1 \neq x_2$ en D , existe una f en $\mathfrak{F}$ tal que $f(x_1) \neq f(x_2)$.
3. **Inclusión Algebraica:** Si f y g son dos funciones en $\mathfrak{F}$, entonces (fg) y $(af + bg)$ están en $\mathfrak{F} \forall a, b \in \mathfrak{R}$.

Entonces $\mathfrak{F}$ es denso en $C(D)$, el conjunto de funciones continuas evaluadas en el dominio real en D . En otras palabras, para cualquier $\varepsilon > 0$ y cualquier función g en $C(D)$, existe una función f en $\mathfrak{F}$ tal que $|g(x) - f(x)| < \varepsilon$.

Entonces, el teorema de Stone-Weierstrass permite garantizar la existencia de una estimación adecuada de una función lineal o no-lineal de cualquier orden definida como se mostró con anterioridad, siempre y cuando se cumplan los criterios de dicho teorema. De tal manera, se requiere analizar si IT2FNN cumple con los criterios requeridos con el fin de demostrar formalmente que puede utilizarse, efectivamente, como un aproximador universal. Los siguientes puntos encierran pues, el análisis de los criterios que marca el teorema de Stone-Weierstrass para determinar si la red adaptativa IT2FNN cumple o no con todos y cada uno de ellos.

Función Identidad

Inicialmente, debe garantizarse que el sistema de inferencia difuso es capaz de calcular la función identidad $f(x) = 1$, por lo que una forma práctica de calcularla es estableciendo los consecuentes de todas las premisas de la base de reglas como un intervalo en $[1-\delta, 1+\delta]$ donde δ es un fuente de incertidumbre; teniendo así, obligatoriamente una salida total unitaria hecho que hace que IT2FNN cumpla con este criterio.

Separabilidad

Este criterio marca que debe ser posible calcular funciones que arrojan valores distintos para entradas distintas. Así pues, es obvio que IT2FNN permite cumplir con este criterio de forma natural con los parámetros apropiados.

Inclusión algebraica

Finalmente, debe cumplirse con el último criterio del teorema de Stone-Weierstrass que, como se sabe, comprende un aspecto aditivo y multiplicativo. En esencia se marca que el IT2FIS debe ser invariante bajo la adición y multiplicación. Por lo tanto, deberán cumplirse de manera separada ambos aspectos para finalmente demostrar que se cumple este criterio en su totalidad.

Para efecto de la demostración, supongamos que se tienen dos IT2FIS S y $\bar{S}$ con dos reglas cada uno y continuando con la notación mostrada para la definición de la estructura de IT2FNN.

Inclusión algebraica aditiva

$$S : z = \frac{1}{2} \left[\frac{\bar{f}^1 y_l^1 + \underline{f}^2 y_l^2}{\bar{f}^1 + \underline{f}^2} + \frac{\bar{f}^1 y_r^1 + \underline{f}^2 y_r^2}{\bar{f}^1 + \underline{f}^2} \right] \quad (2.27)$$

$$\bar{S} : \bar{z} = \frac{1}{2} \left[\frac{\bar{g}^1 \bar{y}_l^1 + \underline{g}^2 \bar{y}_l^2}{\bar{g}^1 + \underline{g}^2} + \frac{\bar{g}^1 \bar{y}_r^1 + \underline{g}^2 \bar{y}_r^2}{\bar{g}^1 + \underline{g}^2} \right] \quad (2.28)$$

sustituyendo S y $\bar{S}$ en $az + b\bar{z}$ y simplificando los términos algebraicos se obtiene;

$$\begin{aligned} az + b\bar{z} = & \frac{1}{2} \frac{\bar{f}^1 \bar{g}^1}{(\bar{f}^1 + \underline{f}^2)(\bar{g}^1 + \underline{g}^2)} (ay_l^1 + b\bar{y}_l^1) + \frac{1}{2} \frac{\bar{f}^1 \underline{g}^2}{(\bar{f}^1 + \underline{f}^2)(\bar{g}^1 + \underline{g}^2)} (ay_l^1 + b\bar{y}_l^2) + \\ & \frac{1}{2} \frac{\underline{f}^2 \bar{g}^1}{(\bar{f}^1 + \underline{f}^2)(\bar{g}^1 + \underline{g}^2)} (ay_l^2 + b\bar{y}_l^1) + \frac{1}{2} \frac{\underline{f}^2 \underline{g}^2}{(\bar{f}^1 + \underline{f}^2)(\bar{g}^1 + \underline{g}^2)} (ay_l^2 + b\bar{y}_l^2) + \end{aligned}$$

$$\begin{aligned}
& \frac{1}{2} \frac{\underline{f}^1 \underline{g}^1}{\left(\underline{f}^1 + \underline{f}^2\right)\left(\underline{g}^1 + \underline{g}^2\right)} \left(ay_r^1 + b\bar{y}_r^1\right) + \frac{1}{2} \frac{\underline{f}^1 \underline{g}^2}{\left(\underline{f}^1 + \underline{f}^2\right)\left(\underline{g}^1 + \underline{g}^2\right)} \left(ay_r^1 + b\bar{y}_r^2\right) + \\
& \frac{1}{2} \frac{\underline{f}^2 \underline{g}^1}{\left(\underline{f}^1 + \underline{f}^2\right)\left(\underline{g}^1 + \underline{g}^2\right)} \left(ay_r^2 + b\bar{y}_r^1\right) + \frac{1}{2} \frac{\underline{f}^2 \underline{g}^2}{\left(\underline{f}^1 + \underline{f}^2\right)\left(\underline{g}^1 + \underline{g}^2\right)} \left(ay_r^2 + b\bar{y}_r^2\right)
\end{aligned} \tag{2.29}$$

Entonces, se puede construir un IT2FIS con 4 reglas (8 nodos en un IT2FNN en la capa de reglas) que calcule $az + b\bar{z}$ con las condiciones de disparo $\bar{f}^1 \underline{g}^1, \bar{f}^1 \underline{g}^2, \underline{f}^2 \underline{g}^1, \underline{f}^2 \underline{g}^2$ y la salida de las reglas $ay_i^j + b\bar{y}_i^j$ (para $i, j = 1, 2$) respectivamente. Asimismo $\underline{f}^1 \underline{g}^1, \underline{f}^1 \underline{g}^2, \bar{f}^2 \underline{g}^1, \bar{f}^2 \underline{g}^2$ y $ay_r^i + b\bar{y}_r^j$ (para $i, j = 1, 2$) respectivamente.

Inclusión algebraica multiplicativa

Sustituyendo S y $\bar{S}$ en $z\bar{z}$ y simplificando los términos algebraicos se obtiene;

$$\begin{aligned}
z\bar{z} = & \frac{1}{4} \frac{\bar{f}^1 \underline{g}^1}{\left(\bar{f}^1 + \underline{f}^2\right)\left(\underline{g}^1 + \underline{g}^2\right)} y_i^1 \bar{y}_i^1 + \frac{1}{4} \frac{\bar{f}^1 \underline{g}^2}{\left(\bar{f}^1 + \underline{f}^2\right)\left(\underline{g}^1 + \underline{g}^2\right)} y_i^1 \bar{y}_i^2 + \\
& \frac{1}{4} \frac{\underline{f}^2 \underline{g}^1}{\left(\bar{f}^1 + \underline{f}^2\right)\left(\underline{g}^1 + \underline{g}^2\right)} y_i^2 \bar{y}_i^1 + \frac{1}{4} \frac{\underline{f}^2 \underline{g}^2}{\left(\bar{f}^1 + \underline{f}^2\right)\left(\underline{g}^1 + \underline{g}^2\right)} y_i^2 \bar{y}_i^2 + \\
& \frac{1}{4} \frac{\bar{f}^1 \underline{g}^1}{\left(\bar{f}^1 + \underline{f}^2\right)\left(\underline{g}^1 + \underline{g}^2\right)} y_i^1 \bar{y}_r^1 + \frac{1}{4} \frac{\bar{f}^1 \underline{g}^2}{\left(\bar{f}^1 + \underline{f}^2\right)\left(\underline{g}^1 + \underline{g}^2\right)} y_i^1 \bar{y}_r^2 + \\
& \frac{1}{4} \frac{\underline{f}^2 \underline{g}^1}{\left(\bar{f}^1 + \underline{f}^2\right)\left(\underline{g}^1 + \underline{g}^2\right)} y_i^2 \bar{y}_r^1 + \frac{1}{4} \frac{\underline{f}^2 \underline{g}^2}{\left(\bar{f}^1 + \underline{f}^2\right)\left(\underline{g}^1 + \underline{g}^2\right)} y_i^2 \bar{y}_r^2 + \\
& \frac{1}{4} \frac{\underline{f}^1 \underline{g}^1}{\left(\underline{f}^1 + \bar{f}^2\right)\left(\underline{g}^1 + \underline{g}^2\right)} y_r^1 \bar{y}_i^1 + \frac{1}{4} \frac{\underline{f}^1 \underline{g}^2}{\left(\underline{f}^1 + \bar{f}^2\right)\left(\underline{g}^1 + \underline{g}^2\right)} y_r^1 \bar{y}_i^2 +
\end{aligned}$$

$$\begin{aligned}
& \frac{1}{4} \frac{\bar{f}^2 \bar{g}^{-1}}{(\underline{f}^1 + \bar{f}^2)(\bar{g}^{-1} + \underline{g}^2)} y_r^2 \bar{y}_l^1 + \frac{1}{4} \frac{\bar{f}^2 \underline{g}^2}{(\underline{f}^1 + \bar{f}^2)(\bar{g}^{-1} + \underline{g}^2)} y_r^2 \bar{y}_l^2 + \\
& \frac{1}{4} \frac{\underline{f}^1 \underline{g}^1}{(\underline{f}^1 + \bar{f}^2)(\bar{g}^{-1} + \underline{g}^2)} y_r^1 \bar{y}_r^1 + \frac{1}{4} \frac{\underline{f}^1 \bar{g}^{-2}}{(\underline{f}^1 + \bar{f}^2)(\bar{g}^{-1} + \underline{g}^2)} y_r^1 \bar{y}_r^2 + \\
& \frac{1}{4} \frac{\bar{f}^2 \underline{g}^1}{(\underline{f}^1 + \bar{f}^2)(\bar{g}^{-1} + \underline{g}^2)} y_r^2 \bar{y}_r^1 + \frac{1}{4} \frac{\bar{f}^2 \bar{g}^{-2}}{(\underline{f}^1 + \bar{f}^2)(\bar{g}^{-1} + \underline{g}^2)} y_r^2 \bar{y}_r^2
\end{aligned} \tag{2.30}$$

Al igual que en la inclusión algebraica aditiva, puede generarse un par de IT2FIS que sean capaces de calcular ahora $z\bar{z}$ con las condiciones de disparo $\{[\underline{f}^i, \bar{f}^i] \times [\underline{g}^j, \bar{g}^j]\}$ y la salida de las reglas $\{[y_l^i, y_r^i] \times [\bar{y}_l^j, \bar{y}_r^j]\}$ con $(i, j = 1, 2)$ respectivamente.

De esta manera, puede observarse que los IT2FIS diseñados para calcular $\hat{y}$ son exactamente los mismos que para calcular S y $\bar{S}$ si y solo si los conjuntos difusos tipo-2 por intervalos utilizados para representar las entradas, son invariantes para la multiplicación. Las funciones de membresía tipo-2 por intervalos que hacen esto posible son las IT2MF gaussianas (igausstype2, igaussmtype2, igaussstype2, etc.), campanas generalizadas (igbelltype2, igbellmtype2, igbellstype2, etc.), triangulares (itritype2 y itritype2), por mencionar algunas. Entonces, queda apropiadamente demostrado que es IT2FNN una herramienta adecuada para estimar funciones lineales o no-lineales comportándose así como un aproximador universal.

CAPÍTULO 3. EXPERIMENTOS REALIZADOS.

En este capítulo se presentan los resultados de las simulaciones obtenidas utilizando los modelos IT2FNN-2:A2C0 (IT2FNN-0), IT2FNN-1:A2C1 (IT2FNN-1), IT2FNN-2:A2C1 (IT2FNN-2) y IT2FNN-3:A2C1 (IT2FNN-3) para aproximar funciones no lineales y pronosticar la serie de tiempo de Mackey-Glass con $\tau = 13, 17, 30, 60, 100$ y diferentes valores de razón señal a ruido, $\text{SNR}(\text{dB}) = 0, 10, 20, 30$ como fuentes de incertidumbre.

3.1 Condiciones generales de los experimentos

En los siguientes experimentos se usa el método híbrido de Levenberg-Marquardt (Apéndice A.1) para adaptar los parámetros del modelo IT2FNN, puesto que sus características de eficiencia y robustez lo hacen especialmente adecuado para el aprendizaje híbrido IT2FNN. Además se propone la alternativa de aprendizaje HGA (Toolbox GA de Matlab) y PSO (Apéndice A.2). Las arquitecturas propuestas del modelo IT2FNN se validan usando validación cruzada (10-fold cross-validation) [113, 114] considerando los valores estadísticos de la suma de cuadrados residuales (**SSE**) o la raíz cuadrada de la media de cuadrados residuales (**RMSE**) en la fase de entrenamiento o prueba; Criterio de Información de Akaike (**AIC**) y Prueba **F** [111, 112].

La evaluación del procedimiento de validación cruzada se realiza usando la función *crossvalind* de la Toolbox de Matlab con sintaxis: $[train, test] = \text{crossvalind}('Kfold', N, K)$ donde **train** y **test** son los índices del conjunto de datos de entrenamiento y prueba respectivamente, N es el numero de datos y K es la partición de los datos. La señal libre de ruido (**s**) se afectan con ruido usando la función de Matlab *awgn* con sintaxis: $x = \text{awgn}(s, snr, sigpower, state, powertype)$.

El criterio de información de Akaike (*AIC*) y prueba *F* se evalúan usando la Toolbox estadística de Matlab.

Experimento 1. Identificación de una función no-lineal de una variable.

En este experimento se aproxima una función no lineal $f : \Re \rightarrow \Re$, utilizando para ello un modelo IT2FNN de una entrada y una salida, 50 muestras de entrenamiento con validación cruzada (10 fold cross-validation). Los valores de la variable independiente u para el conjunto de entrenamiento se eligen en el intervalo $u \in [-1, 1]$. La función f se define de la siguiente manera:

$$f(u) = 0.6 \sin(\pi u) + 0.3 \sin(3\pi u) + 0.1 \sin(5\pi u) + \eta \quad (3.1)$$

donde η es una componente de ruido uniforme.

Experimento 2. Identificación de una función no-lineal de dos variables.

Se utiliza el modelo IT2FNN con dos entradas y una salida para aproximar la función no lineal **sinc** : $\Re^2 \rightarrow \Re$, definida como:

$$z = \text{sinc}(x, y) = \frac{\sin(x) \sin(y)}{xy} + \eta \quad (3.2)$$

El conjunto de entrenamiento consiste en 121 muestras $x \times y \in [-10, 10] \times [-10, 10]$ con validación cruzada (10 fold cross-validation).

Experimento 3. Identificación de una función no-lineal de tres variables.

El tercer problema de prueba es la función de Sugeno $f : \Re^3 \rightarrow \Re$ propuesta en (Sugeno and Kang, 1988) [92, 93, 98], y se define como sigue:

$$f(x_1, x_2, x_3) = \left(1 + \sqrt{x_1} + \frac{1}{x_2} + \frac{1}{\sqrt{x_3}} \right)^2 + \eta \quad (3.3)$$

Se trata de un problema utilizado con frecuencia para probar la capacidad de aproximación de

sistemas adaptativos de aprendizaje. Los datos de entrenamiento se generan de la siguiente manera: Se generan 216 muestras para entrenamiento con validación cruzada (10 fold cross-validation) y 125 para prueba. Los valores de las variables independientes (x_1, x_2, x_3) para el conjunto de entrenamiento se eligen en el intervalo $(x_1, x_2, x_3) \in [1, 6]$ y los del conjunto de prueba de la misma forma en el intervalo $(x_1, x_2, x_3) \in [1.5, 5.5]$.

Experimento 4. Pronóstico de la serie de tiempo caótica de Mackey-Glass.

La serie de tiempo caótica Mackey-Glass es un benchmark muy conocido para el modelado de sistemas, que ha sido ampliamente utilizado en la literatura de este tipo de problemas [110]. Esta serie de tiempo esta descrita por la siguiente ecuación diferencial;

$$\dot{x}(t) = \frac{0.1x(t-\tau)}{1 + x^{10}(t-\tau)} - 0.1x(t) \quad (3.4)$$

Se generan 1200 muestras a partir de la condición inicial $x(0)=1.2$ y $\tau=13, 17, 30, 60, 100$ utilizando el método de Runge-Kutta de 4° orden y se agregan diferentes niveles de ruido uniforme. Para realizar comparaciones justas con otros trabajos, se elige un vector de entrada-salida para el modelo IT2FNN con el siguiente formato;

$$[x(t-18), x(t-12), x(t-6), x(t); x(t+6)]$$

Se han seleccionado 500 para entrenamiento con validación cruzada (10 fold cross-validation), y las ultimas 500 para prueba.

3.2 Experimentos de IT2FNN.

Experimento 1. Identificación de una función no-lineal de una variable.

En este experimento se aproxima una función no lineal $f : \Re \rightarrow \Re$, utilizando para ello un modelo IT2FNN de una entrada y una salida, 50 muestras de entrenamiento con validación

cruzada (10 fold cross-validation) con niveles de ruido uniforme, 6 IT2MF tipo igaussmtype2, 6 reglas y 50 épocas. Una vez identificados los modelos ANFIS y IT2FNN, se realizó una comparación considerando los valores estadísticos RMSE con validación cruzada (10 fold cross-validation). La Tabla 3.1 y la Figura 3.1 muestran los valores RMSE obtenidos para los modelos ANFIS y IT2FNN. Se observa que las arquitecturas del modelo IT2FNN identifican mejor la función no-lineal respecto a ANFIS.

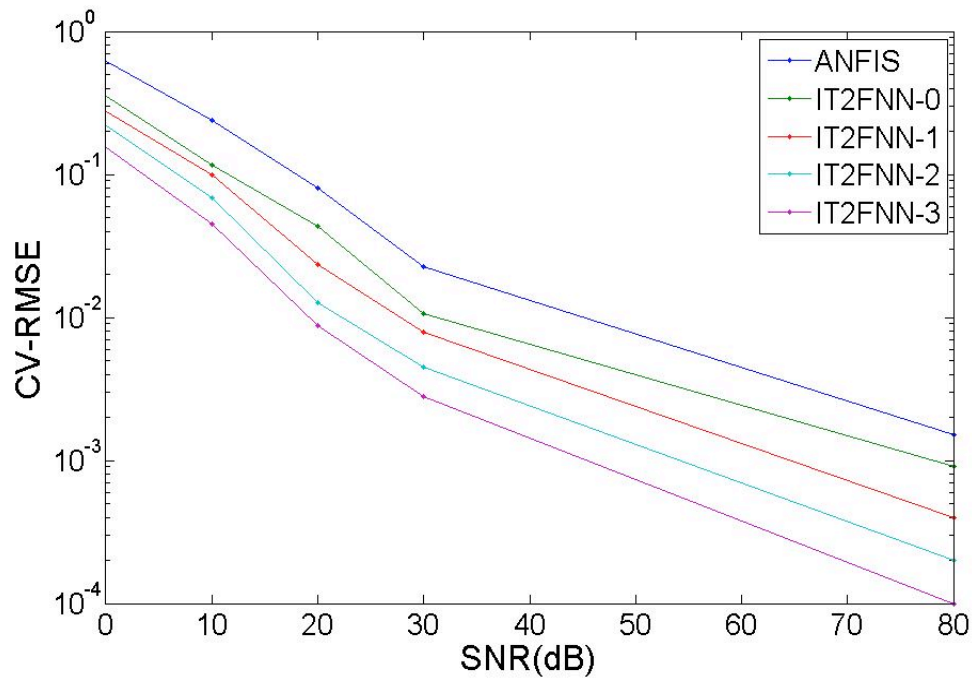


Figura 3.1 Valores RMSE de los modelos ANFIS y IT2FNN evaluados con validación cruzada para la identificación no lineal del experimento 1.

Tabla 3.1 Valores RMSE de los modelos ANFIS y IT2FNN con validación cruzada para la identificación no-lineal del experimento 1.

SNR(dB)	ANFIS	IT2FNN-0	IT2FNN-1	IT2FNN-2	IT2FNN-3
0	0.6156	0.3532	0.2764	0.2197	0.1535
10	0.2375	0.1153	0.0986	0.0683	0.0453
20	0.0806	0.0435	0.0234	0.0127	0.0087
30	0.0225	0.0106	0.0079	0.0045	0.0028
libre	0.0015	0.0009	0.0004	0.0002	0.0001

Experimento 2. Identificación de una función no-lineal de dos variables.

Se utiliza el modelo IT2FNN con dos entradas y una salida para aproximar la función no lineal **sinc** : $\Re^2 \rightarrow \Re$. El conjunto de entrenamiento consiste en 121 muestras $x \times y \in [-10,10] \times [-10,10]$ con validación cruzada (10 fold cross-validation), 4 IT2MF tipo igaussmtype2 por entrada, 16 reglas y 50 épocas. Identificados los modelos ANFIS y IT2FNN, se realizó una comparación considerando los valores estadísticos RMSE con validación cruzada (10 fold cross-validation). La Tabla 3.2 y la Figura 3.2 muestran los valores RMSE obtenidos para los modelos ANFIS y IT2FNN. Se observa que las arquitecturas del modelo IT2FNN identifican mejor la función no-lineal respecto a ANFIS.

Tabla 3.2 Valores RMSE de los modelos ANFIS y IT2FNN con validación cruzada para la identificación no-lineal del experimento 2.

SNR(dB)	ANFIS	IT2FNN-0	IT2FNN-1	IT2FNN-2	IT2FNN-3
0	0.7512	0.5523	0.4386	0.3972	0.2897
10	0.2258	0.1713	0.1302	0.1039	0.0897
20	0.0933	0.0637	0.0458	0.0376	0.0281
30	0.0476	0.0329	0.0249	0.0189	0.0145
libre	0.0373	0.0269	0.0183	0.0158	0.0139

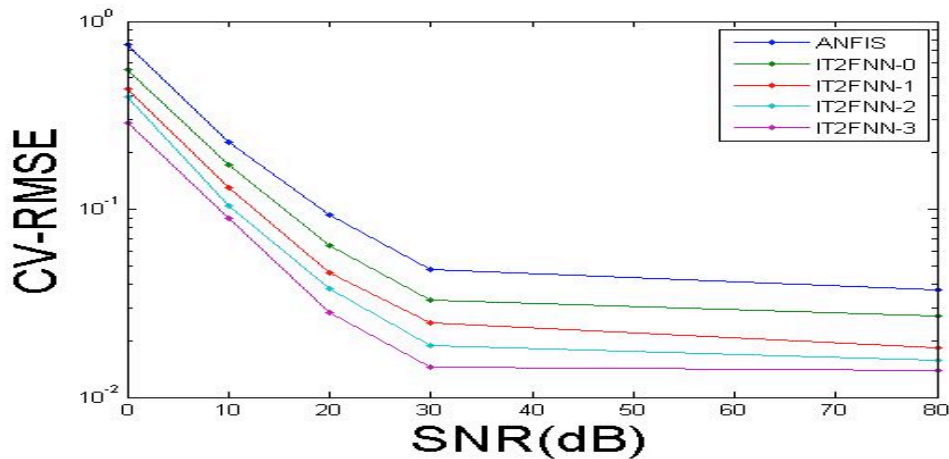


Figura 3.2 Valores RMSE de los modelos ANFIS y IT2FNN evaluados con validación cruzada para la identificación no lineal del experimento 2.

Experimento 3. Identificación de una función no-lineal de tres variables.

Se utiliza el modelo IT2FNN con tres entradas y una salida para aproximar la función no lineal de Sugeno $f: \mathfrak{R}^3 \rightarrow \mathfrak{R}$. Se generan 216 muestras para entrenamiento con validación cruzada (10 fold cross-validation) y 125 para prueba; 2 IT2MF tipo igaussmtype2 por entrada, 8 reglas y 50 épocas. Identificados los modelos ANFIS y IT2FNN, se realizó una comparación con los valores estadísticos RMSE con validación cruzada (10 fold cross-validation). La Tabla 3.3 y la Figura 3.3 muestran los valores RMSE obtenidos para los modelos ANFIS y IT2FNN. Se observa que las arquitecturas del modelo IT2FNN identifican mejor la función no-lineal respecto a ANFIS.

Tabla 3.3 Valores RMSE determinados en los modelos ANFIS y IT2FNN para la identificación no-lineal del experimento 3, con validación cruzada.

SNR(dB)	ANFIS	IT2FNN-0	IT2FNN-1	IT2FNN-2	IT2FNN-3
0	1.0432	0.7203	0.6523	0.5512	0.5267
10	0.3096	0.2798	0.2583	0.2464	0.2344
20	0.1703	0.1637	0.1592	0.1465	0.1387
30	0.1526	0.1408	0.1368	0.1323	0.1312
libre	0.1503	0.1390	0.1323	0.1304	0.1276

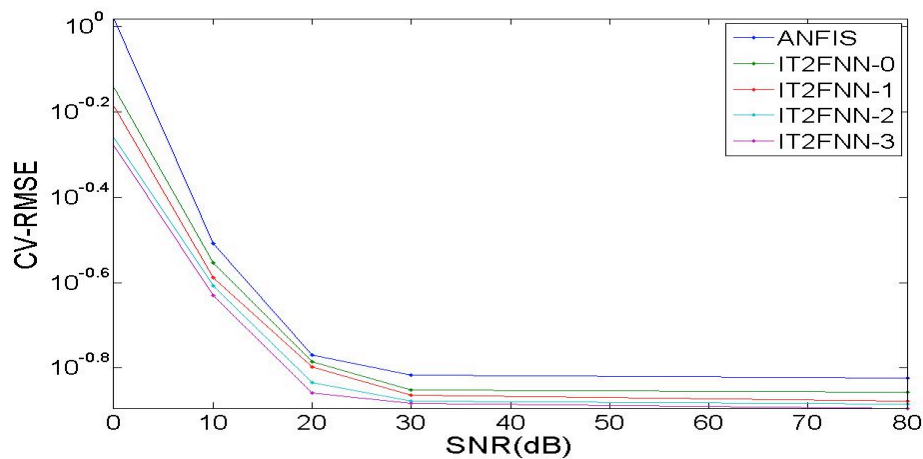


Figura 3.3 Valores RMSE determinados en los modelos ANFIS y IT2FNN para la identificación no-lineal del experimento 3, con validación cruzada.

Experimento 4. Pronóstico de la serie de tiempo caótica de Mackey-Glass.

Se utiliza el modelo IT2FNN con cuatro entradas y una salida para el pronóstico de la serie de tiempo caótica de Mackey-Glass, se seleccionan 500 datos para entrenamiento y 500 para prueba con validación cruzada (10 fold cross-validation), 2 IT2MF por entrada tipo igaussmtype2, 16 reglas y 50 épocas. Se identifican los modelos ANFIS y IT2FNN, comparando los valores estadísticos RMSE con validación cruzada (10 fold cross-validation). Las Tablas 3.4-3.8 y las Figuras 3.4-3.13 muestran el número de puntos ζ que están fuera del intervalo de incertidumbre $\tilde{Y}(x) \in [\hat{y}_l(x), \hat{y}_r(x)]$ evaluado por el modelo IT2FNN, los valores RMSE de entrenamiento (TRN) y prueba (CHK) obtenidos para los modelos ANFIS y IT2FNN. Se observa que las arquitecturas del modelo IT2FNN pronostican mejor la serie caótica de Mackey-Glass.

Tabla 3.4 Valores RMSE (TRN/CHK) y ζ determinados en los modelos ANFIS y IT2FNN con validación cruzada para el pronóstico de la serie de tiempo caótica de Mackey-Glass con $\tau=13$.

SNR(dB)	ANFIS			IT2FNN-0			IT2FNN-1			IT2FNN-2			IT2FNN-3		
	TRN	CHK	ζ	TRN	CHK	ζ	TRN	CHK	ζ	TRN	CHK	ζ	TRN	CHK	ζ
0	0.2209	0.2519	NA	0.2012	0.2131	25	0.1974	0.2108	23	0.1612	0.1839	22	0.1413	0.1538	19
10	0.0718	0.0784	NA	0.0654	0.0723	18	0.0628	0.0705	17	0.0524	0.0572	16	0.0423	0.0487	14
20	0.0242	0.0264	NA	0.0211	0.0237	13	0.0197	0.0208	12	0.0177	0.0193	12	0.0137	0.0169	9
30	0.0083	0.0090	NA	0.0075	0.0081	9	0.0071	0.0077	9	0.0061	0.0066	10	0.0036	0.0056	7
libre	0.0002	0.0002	NA	0.0001	0.0001	5	0.0001	0.0001	4	0.0001	0.0001	2	0.0001	0.0001	2

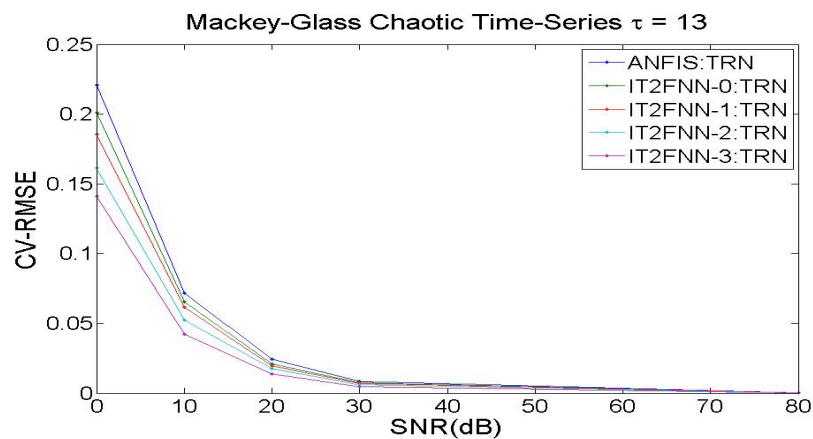


Figura 3.4 Valores RMSE (TRN) determinados en los modelos ANFIS y IT2FNN con validación cruzada para el pronóstico de la serie de tiempo caótica de Mackey-Glass con $\tau=13$.

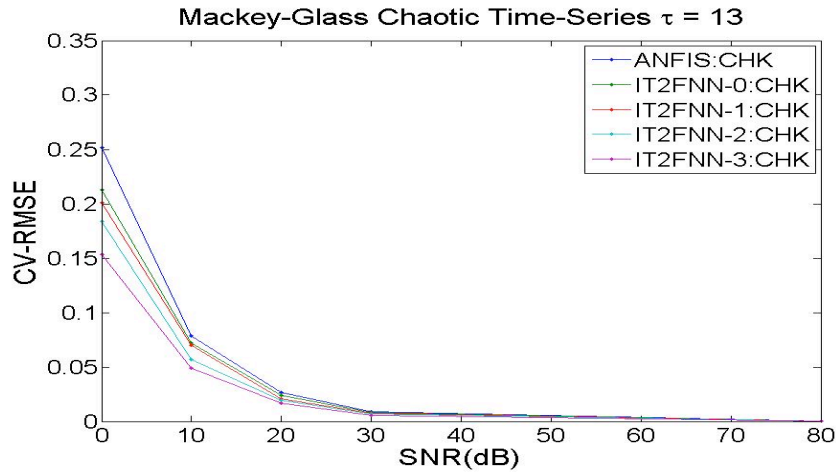


Figura 3.5 Valores RMSE (CHK) determinados en los modelos ANFIS y IT2FNN con validación cruzada para el pronóstico de la serie de tiempo caótica de Mackey-Glass con $\tau = 13$.

Tabla 3.5 Valores RMSE (TRN/CHK) y ζ determinados en los modelos ANFIS y IT2FNN con validación cruzada para el pronóstico de la serie de tiempo caótica de Mackey-Glass con $\tau = 17$.

SNR(dB)	ANFIS			IT2FNN-0			IT2FNN-1			IT2FNN-2			IT2FNN-3		
	TRN	CHK	ζ	TRN	CHK	ζ	TRN	CHK	ζ	TRN	CHK	ζ	TRN	CHK	ζ
0	0.2643	0.2917	NA	0.2357	0.2576	28	0.2089	0.2334	25	0.1883	0.1947	24	0.1558	0.1632	20
10	0.0902	0.0989	NA	0.0776	0.0837	23	0.0672	0.0875	19	0.0540	0.0592	18	0.0427	0.0549	15
20	0.0316	0.0345	NA	0.0289	0.0327	17	0.0268	0.0318	15	0.0189	0.0207	14	0.0134	0.0148	11
30	0.0110	0.0118	NA	0.0088	0.0107	13	0.0056	0.0078	11	0.0066	0.0071	10	0.0041	0.0043	8
libre	0.0020	0.0021	NA	0.0017	0.0019	7	0.0016	0.0018	6	0.0014	0.0015	4	0.0011	0.0013	3

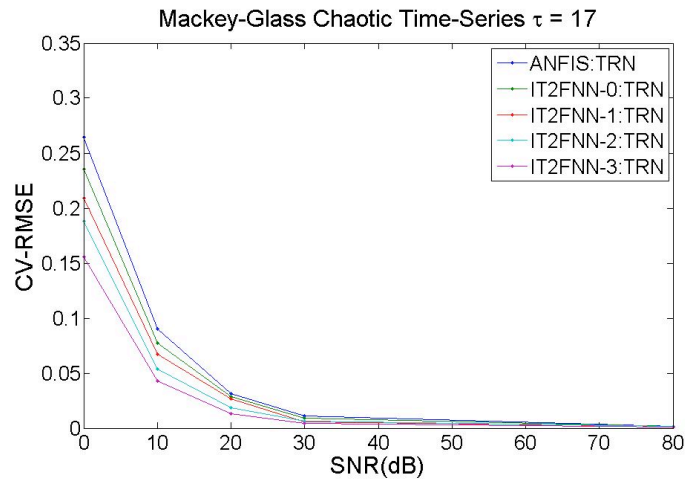


Figura 3.6 Valores RMSE (TRN) determinados en los modelos ANFIS y IT2FNN con validación cruzada para el pronóstico de la serie de tiempo caótica de Mackey-Glass con $\tau = 17$.

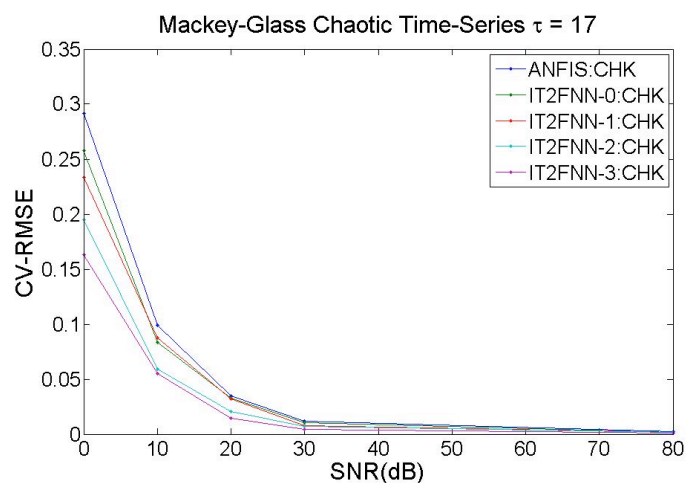


Figura 3.7 Valores RMSE (CHK) determinados en los modelos ANFIS y IT2FNN con validación cruzada para el pronóstico de la serie de tiempo caótica de Mackey-Glass con $\tau=17$.

Tabla 3.6 Valores RMSE (TRN/CHK) y ζ determinados en los modelos ANFIS y IT2FNN con validación cruzada para el pronóstico de la serie de tiempo caótica de Mackey-Glass con $\tau=30$.

SNR(dB)	ANFIS			IT2FNN-0			IT2FNN-1			IT2FNN-2			IT2FNN-3		
	TRN	CHK	ζ	TRN	CHK	ζ	TRN	CHK	ζ	TRN	CHK	ζ	TRN	CHK	ζ
0	0.3416	0.3800	NA	0.2543	0.2722	36	0.2097	0.2605	33	0.1719	0.1801	30	0.1521	0.1708	27
10	0.1582	0.1714	NA	0.1278	0.1385	29	0.1123	0.1203	27	0.0812	0.0892	26	0.0733	0.0773	23
20	0.0876	0.0943	NA	0.0633	0.0719	25	0.0517	0.0611	22	0.0415	0.0447	20	0.0338	0.0419	18
30	0.0635	0.0672	NA	0.0512	0.0555	19	0.0413	0.0429	17	0.0301	0.0318	16	0.0249	0.0302	8
libre	0.0585	0.0619	NA	0.0444	0.0467	9	0.0326	0.0359	8	0.0297	0.0289	6	0.0212	0.0241	5

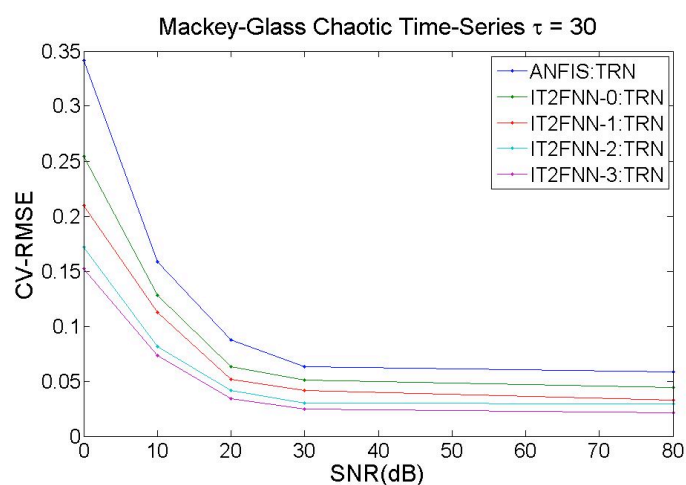


Figura 3.8 Valores RMSE (TRN) determinados en los modelos ANFIS y IT2FNN con validación cruzada para el pronóstico de la serie de tiempo caótica de Mackey-Glass con $\tau=30$.

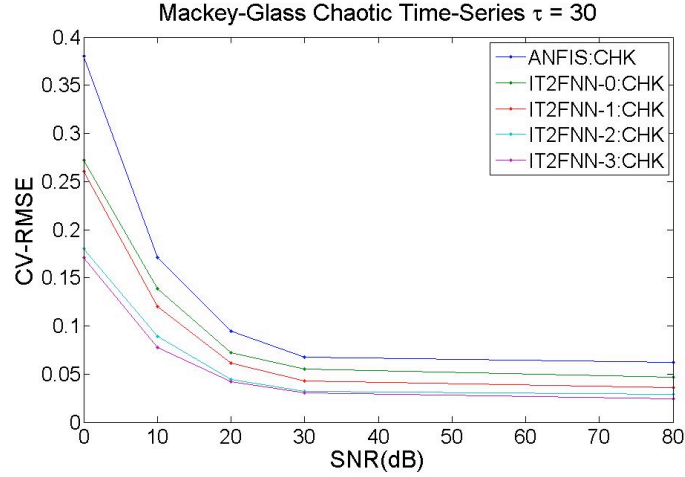


Figura 3.9 Valores RMSE (CHK) determinados en los modelos ANFIS y IT2FNN con validación cruzada para el pronóstico de la serie de tiempo caótica de Mackey-Glass con $\tau=30$.

Tabla 3.7 Valores RMSE (TRN/CHK) y ζ determinados en los modelos ANFIS y IT2FNN con validación cruzada para el pronóstico de la serie de tiempo caótica de Mackey-Glass con $\tau=60$.

SNR(dB)	ANFIS			IT2FNN-0			IT2FNN-1			IT2FNN-2			IT2FNN-3		
	TRN	CHK	ζ	TRN	CHK	ζ	TRN	CHK	ζ	TRN	CHK	ζ	TRN	CHK	ζ
0	0.3403	0.3714	NA	0.2388	0.2687	37	0.2027	0.2135	35	0.1495	0.1536	34	0.1244	0.1444	31
10	0.1544	0.1714	NA	0.1312	0.1456	33	0.1069	0.1119	30	0.0805	0.0972	28	0.0532	0.0629	25
20	0.0929	0.1007	NA	0.0708	0.0781	28	0.0566	0.0594	26	0.0424	0.0485	24	0.0342	0.0355	21
30	0.0788	0.0847	NA	0.0526	0.0582	19	0.0411	0.0468	18	0.0309	0.0332	16	0.0227	0.0316	14
libre	0.0749	0.0799	NA	0.0414	0.0477	10	0.0321	0.0353	9	0.0251	0.0319	6	0.0215	0.0293	4

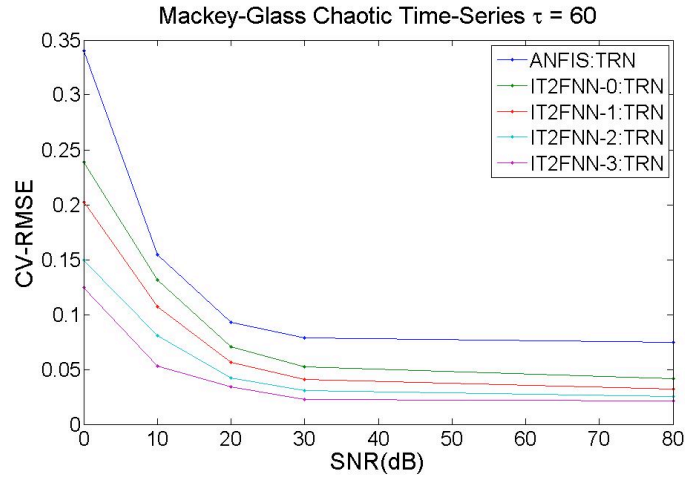


Figura 3.10 Valores RMSE (TRN) determinados en los modelos ANFIS y IT2FNN con validación cruzada para el pronóstico de la serie de tiempo caótica de Mackey-Glass con $\tau=60$.

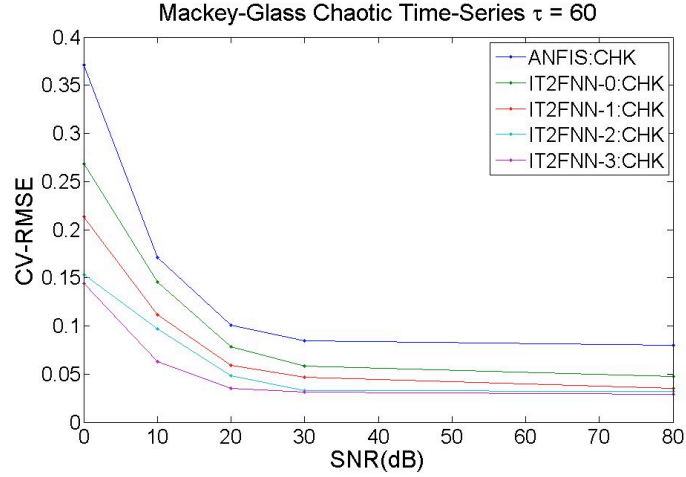


Figura 3.11 Valores RMSE (CHK) determinados en los modelos ANFIS y IT2FNN con validación cruzada para el pronóstico de la serie de tiempo caótica de Mackey-Glass con $\tau=60$.

Tabla 3.8 Valores RMSE (TRN/CHK) y ζ determinados en los modelos ANFIS y IT2FNN con validación cruzada para el pronóstico de la serie de tiempo caótica de Mackey-Glass con $\tau=100$.

SNR(dB)	ANFIS			IT2FNN-0			IT2FNN-1			IT2FNN-2			IT2FNN-3		
	TRN	CHK	ζ	TRN	CHK	ζ	TRN	CHK	ζ	TRN	CHK	ζ	TRN	CHK	ζ
0	0.4253	0.4677	NA	0.3017	0.3322	50	0.2258	0.2976	48	0.1751	0.1899	46	0.1311	0.1377	39
10	0.1839	0.2035	NA	0.1445	0.1497	42	0.1133	0.1235	40	0.0724	0.0783	38	0.0527	0.0554	31
20	0.1035	0.1227	NA	0.0828	0.0898	38	0.0687	0.0779	36	0.0412	0.0570	34	0.0258	0.0311	28
30	0.0706	0.0758	NA	0.0563	0.0681	29	0.0406	0.0473	27	0.0313	0.0398	24	0.0179	0.0212	19
libre	0.0617	0.0669	NA	0.0415	0.0483	13	0.0307	0.0397	11	0.0189	0.0296	8	0.0155	0.0133	6

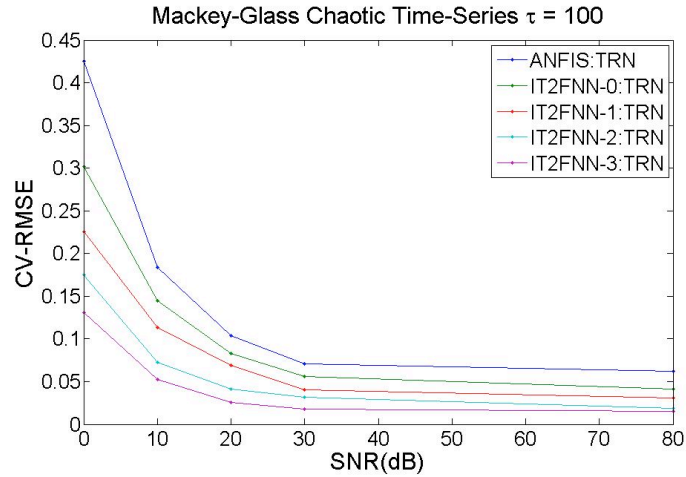


Figura 3.12 Valores RMSE (TRN) determinados en los modelos ANFIS y IT2FNN con validación cruzada para el pronóstico de la serie de tiempo caótica de Mackey-Glass con $\tau=100$.

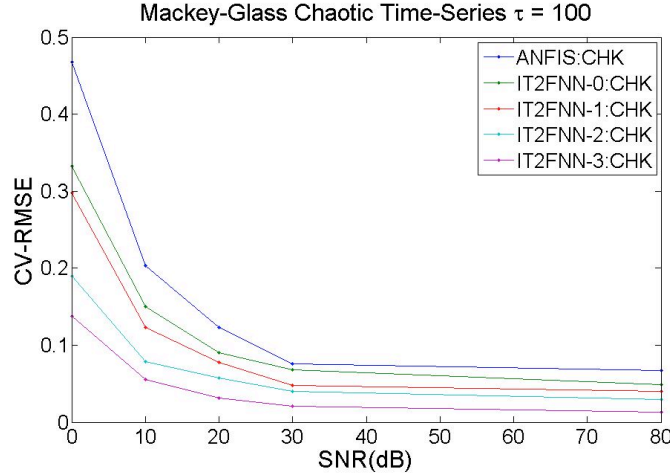


Figura 3.13 Valores RMSE (CHK) determinados en los modelos ANFIS y IT2FNN con validación cruzada para el pronóstico de la serie de tiempo caótica de Mackey-Glass con $\tau=100$.

3.3 Experimentos de HGA-IT2FNN.

Experimento 1. Identificación de una función no-lineal de una variable.

En este experimento se aproxima una función no lineal $f: \mathfrak{R} \rightarrow \mathfrak{R}$, utilizando para ello un modelo HGA-IT2FNN de una entrada y una salida, 50 muestras de entrenamiento con validación cruzada (10 fold cross-validation) con niveles de ruido uniforme; el mejor modelo resultó con 7/16 igaussmtype2 para x_1 , 7/16 reglas, 0.04472 mutación, 0.7163 cruce, 3 subpoblaciones, 20 individuos por subpoblación, 8 individuos por migración y 500 generaciones.

Una vez identificados los modelos HGA-ANFIS y HGA-IT2FNN, se realizó una comparación considerando las medidas RMSE con validación cruzada (10 fold cross-validation). La Tabla 3.9 y la Figura 3.14 muestran los valores RMSE obtenidos para los modelos HGA-ANFIS y HGA-IT2FNN. Se observa que las arquitecturas del modelo HGA-IT2FNN identifican mejor la función no-lineal respecto a HGA-ANFIS.

Tabla 3.9 Valores RMSE de los modelos HGA-ANFIS y HGA-IT2FNN con validación cruzada para la identificación no-lineal del experimento 1.

SNR(dB)	HGA-ANFIS	HGA-IT2FNN-0	IT2FNN-1	HGA-IT2FNN-2	HGA-IT2FNN-3
0	0.6133	0.3412	0.2694	0.2167	0.1525
10	0.2295	0.1133	0.0926	0.0683	0.0423
20	0.0796	0.0401	0.0214	0.0127	0.0063
30	0.0215	0.0101	0.0065	0.0045	0.00224
libre	0.0013	0.0007	0.0003	0.0001	0.000083

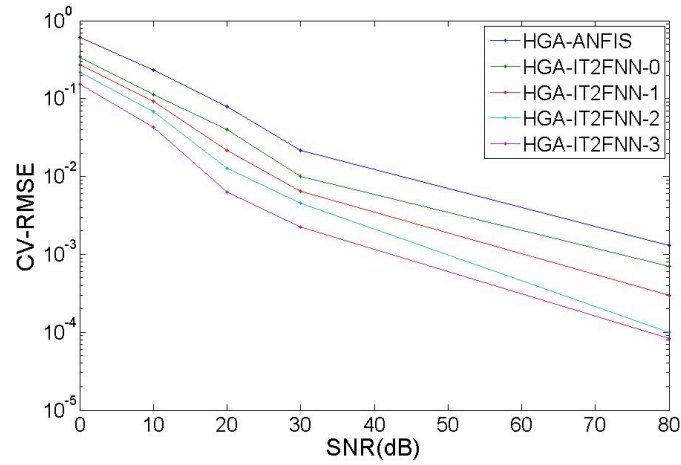


Figura 3.14 Valores RMSE de los modelos HGA-ANFIS y HGA-IT2FNN evaluados con validación cruzada para la identificación no lineal del experimento 1.

Experimento 2. Identificación de una función no-lineal de dos variables.

Se utiliza el modelo HGA-IT2FNN con dos entradas y una salida para aproximar la función no lineal **sinc** : $\Re^2 \rightarrow \Re$. El conjunto de entrenamiento consiste en 121 muestras $x \times y \in [-10,10] \times [-10,10]$ con validación cruzada (10 fold cross-validation); el mejor modelo resulto con 4/6 igaussmtype para x_1 , 5/6 igaussmtype2 para x_2 , 20/36 reglas, 0.05432 mutación, 0.7893 cruce, 3 subpoblaciones, 30 individuos por subpoblación, 10 individuos por migración y 500 generaciones.

La Tabla 3.10 y la Figura 3.15 muestran los valores RMSE obtenidos para los modelos HGA-ANFIS y HGA-IT2FNN. Se observa que las arquitecturas del modelo HGA-IT2FNN identifican mejor la función no-lineal respecto a HGA-ANFIS.

Tabla 3.10 Valores RMSE de los modelos HGA-ANFIS y HGA-IT2FNN con validación cruzada para la identificación no-lineal del experimento 2.

SNR(dB)	HGA-ANFIS	HGA-IT2FNN-0	IT2FNN-1	HGA-IT2FNN-2	HGA-IT2FNN-3
0	0.7196	0.5518	0.4246	0.2712	0.2185
10	0.2034	0.1605	0.1321	0.0911	0.0685
20	0.0712	0.0521	0.0433	0.0311	0.0247
30	0.0253	0.0216	0.0168	0.0144	0.0123
libre	0.0192	0.0165	0.0145	0.0137	0.0117

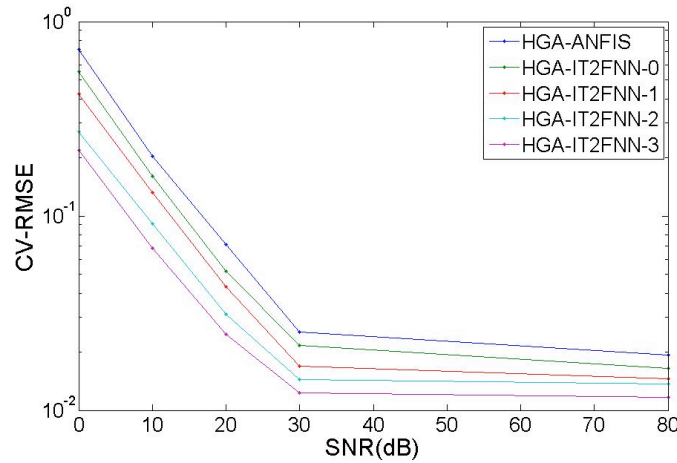


Figura 3.15 Valores RMSE de los modelos HGA-ANFIS y HGA-IT2FNN evaluados con validación cruzada para la identificación no lineal del experimento 2.

Experimento 3. Identificación de una función no-lineal de tres variables.

Se utiliza el modelo HGA-IT2FNN con tres entradas y una salida para aproximar la función no lineal de Sugeno $f: \mathfrak{R}^3 \rightarrow \mathfrak{R}$. Se generan 216 muestras para entrenamiento con validación cruzada (10 fold cross-validation) y 125 para prueba; el mejor modelo resultó con 2

igaussmtype2 para x_1 , 3 igaussmtype2 para x_2 , 2 igaussmtype2 para x_3 , 12/64 reglas, 0.06321 mutación, 0.7977 cruce, 5 subpoblaciones, 30 individuos por subpoblación, 12 individuos por migración y 500 generaciones.

La Tabla 3.11 y la Figura 3.16 muestran los valores RMSE obtenidos para los modelos HGA-ANFIS y HGA-IT2FNN. Se observa que las arquitecturas del modelo HGA-IT2FNN identifican mejor la función no-lineal respecto a HGA-ANFIS.

Tabla 3.11 Valores RMSE de los modelos HGA-ANFIS y HGA-IT2FNN con validación cruzada para la identificación no-lineal del experimento 3.

SNR(dB)	HGA-ANFIS	HGA-IT2FNN-0	HGA-IT2FNN-1	HGA-IT2FNN-2	HGA-IT2FNN-3
0	0.9187	0.7145	0.6712	0.5232	0.5937
10	0.2987	0.2737	0.2513	0.2333	0.2132
20	0.1698	0.1579	0.1474	0.1418	0.1287
30	0.1467	0.1436	0.1369	0.1288	0.1219
libre	0.1427	0.1398	0.1267	0.1222	0.1161

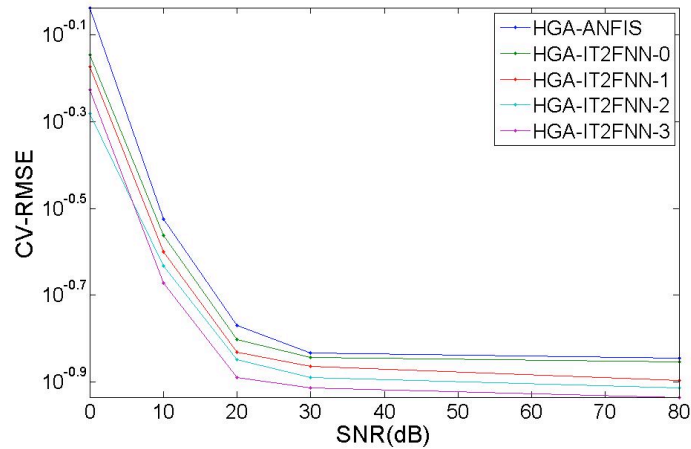


Figura 3.16 Valores RMSE de los modelos HGA-ANFIS y HGA-IT2FNN evaluados con validación cruzada para la identificación no lineal del experimento 3.

3.4 Experimentos de PSO-IT2FNN.

Experimento 1. Identificación de una función no-lineal de una variable.

En este experimento se aproxima una función no lineal $f : \Re \rightarrow \Re$, utilizando para ello un modelo PSO-IT2FNN de una entrada y una salida, 50 muestras de entrenamiento con validación cruzada (10 fold cross-validation) con niveles de ruido uniforme, se usa el modelo de Clerc con $\chi = 0.729, \varphi = 4.1, \varphi_1 = 2.8, \varphi_2 = 1.3$, población de 30 partículas y 2000 épocas.

Una vez identificados los modelos PSO-ANFIS y PSO-IT2FNN, se realizó una comparación considerando las medidas RMSE con validación cruzada (10 fold cross-validation). La Tabla 3.12 y la Figura 3.17 muestran los valores RMSE obtenidos para los modelos PSO-ANFIS y PSO-IT2FNN. Se observa que las arquitecturas del modelo PSO-IT2FNN identifican mejor la función no-lineal respecto a PSO-ANFIS.

Tabla 3.12 Valores RMSE de los modelos PSO-ANFIS y PSO-IT2FNN con validación cruzada para la identificación no-lineal del experimento 1.

SNR(dB)	PSO-ANFIS	PSO-IT2FNN-0	PSO-IT2FNN-1	PSO-IT2FNN-2	PSO-IT2FNN-3
0	0.6121	0.3407	0.2654	0.2147	0.1521
10	0.2271	0.1129	0.0911	0.0671	0.0411
20	0.0767	0.0387	0.0183	0.0116	0.0041
30	0.0223	0.0169	0.0123	0.0037	0.00212
libre	0.00095	0.00045	0.000201	0.000087	0.000053

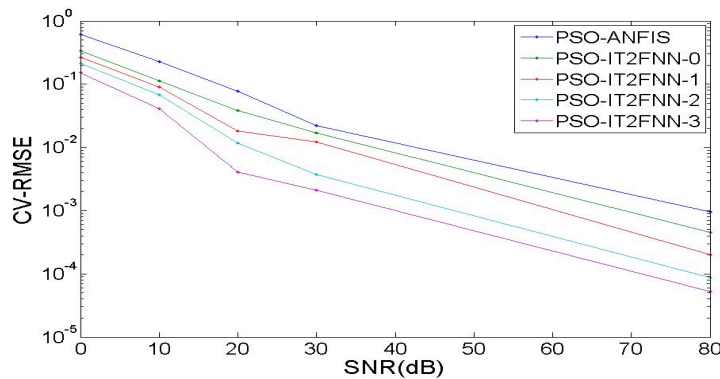


Figura 3.17 Valores RMSE de los modelos PSO-ANFIS y PSO-IT2FNN evaluados con validación cruzada para la identificación no lineal del experimento 1.

Experimento 2. Identificación de una función no-lineal de dos variables.

Se utiliza el modelo PSO-IT2FNN con dos entradas y una salida para aproximar la función no lineal **sinc** : $\Re^2 \rightarrow \Re$. El conjunto de entrenamiento consiste en 121 muestras $x \times y \in [-10,10] \times [-10,10]$ con validación cruzada (10 fold cross-validation), se usa el modelo de Clerc con $\chi = 0.729$, $\varphi = 4.1$, $\varphi_1 = 2.8$, $\varphi_2 = 1.3$, población de 40 partículas y 2000 épocas.

La Tabla 3.13 y la Figura 3.18 muestran los valores RMSE obtenidos para los modelos PSO-ANFIS y PSO-IT2FNN. Se observa que las arquitecturas del modelo PSO-IT2FNN identifican mejor la función no-lineal respecto a PSO-ANFIS.

Tabla 3.13 Valores RMSE de los modelos PSO-ANFIS y PSO-IT2FNN con validación cruzada para la identificación no-lineal del experimento 2.

SNR(dB)	PSO-ANFIS	PSO-IT2FNN-0	PSO-IT2FNN-1	PSO-IT2FNN-2	PSO-IT2FNN-3
0	0.7167	0.5510	0.4147	0.3211	0.2148
10	0.2054	0.1698	0.1322	0.0980	0.0747
20	0.0787	0.0603	0.0425	0.0305	0.0278
30	0.0422	0.0321	0.0233	0.0186	0.0172
libre	0.0303	0.0244	0.0191	0.0178	0.0168

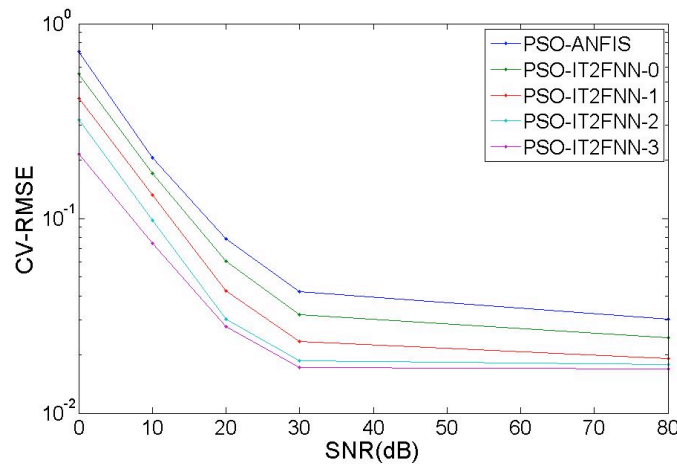


Figura 3.18 Valores RMSE de los modelos PSO-ANFIS y PSO-IT2FNN evaluados con validación cruzada para la identificación no lineal del experimento 2.

Experimento 3. Identificación de una función no-lineal de tres variables.

Se utiliza el modelo PSO-IT2FNN con tres entradas y una salida para aproximar la función no lineal de Sugeno $f: \mathfrak{R}^3 \rightarrow \mathfrak{R}$. Se generan 216 muestras para entrenamiento con validación cruzada (10 fold cross-validation) y 125 para prueba; se usa el modelo de Clerc con $\chi = 0.729, \varphi = 4.1, \varphi_1 = 2.8, \varphi_2 = 1.3$, población de 45 partículas y 2000 épocas.

La Tabla 3.14 y la Figura 3.19 muestran los valores RMSE obtenidos para los modelos PSO-ANFIS y PSO-IT2FNN. Se observa que las arquitecturas del modelo PSO-IT2FNN identifican mejor la función no-lineal respecto a PSO-ANFIS.

Tabla 3.14 Valores RMSE de los modelos PSO-ANFIS y PSO-IT2FNN con validación cruzada para la identificación no-lineal del experimento 3.

SNR(dB)	PSO-ANFIS	PSO-IT2FNN-0	PSO-IT2FNN-1	PSO-IT2FNN-2	PSO-IT2FNN-3
0	0.9177	0.7840	0.6674	0.5227	0.5927
10	0.2969	0.2730	0.2489	0.2315	0.2105
20	0.1673	0.1570	0.1416	0.1352	0.1229
30	0.1463	0.1351	0.1283	0.1233	0.1137
libre	0.1411	0.1325	0.1237	0.1113	0.1031

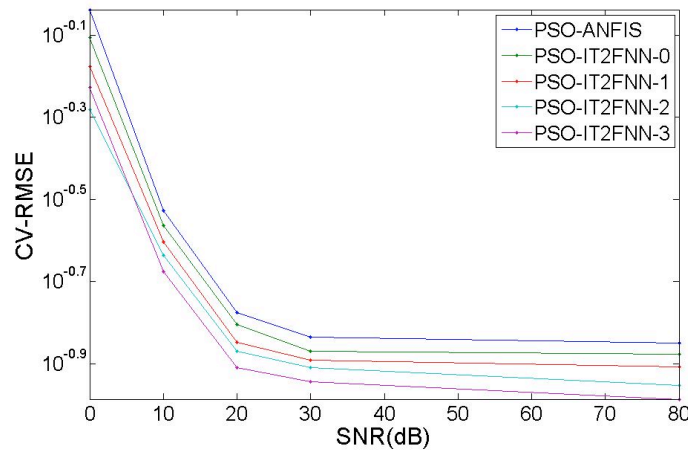


Figura 3.19 Valores RMSE de los modelos PSO-ANFIS y PSO-IT2FNN evaluados con validación cruzada para la identificación no lineal del experimento 3.

CAPÍTULO 4. DISCUSIÓN DE RESULTADOS

En este capítulo se discuten los resultados de los experimentos propuestos en base a los valores estadísticos de la suma de cuadrados residuales (**SSE**) o la raíz cuadrada de la media de cuadrados residuales (**RMSE**) obtenidos con validación cruzada (**CV**), Criterio de Información de Akaike (**AIC**) y Prueba **F** [111, 112] usando el siguiente procedimiento para seleccionar la mejor arquitectura de los modelos IT2FNN, HGA-IT2FNN y PSO-IT2FNN:

1. Identificar los diferentes modelos.

2. Calcular los valores estadísticos $SSE = \sum_{p=1}^q (y_p - \hat{y}_p)^2$ y $RMSE = \sqrt{\frac{SSE}{q}}$

3. El modelo que produce una menor **SSE** o **RMSE** identifica mejor a los datos experimentales. Si los modelos en comparación tienen el mismo número de parámetros y por tanto los mismos grados de libertad, las **SSE** y **RMSE** pueden compararse directamente.

4. Cuando los modelos en comparación difieren en el número de parámetros debe tenerse en cuenta que los modelos más complejos permiten en general obtener **SSE** menores. Si el modelo más sencillo permite obtener una **SSE** más pequeña es un indicativo claro de que el modelo de mayor complejidad no supone ninguna mejora. En aquellos casos en los que el modelo de más complejidad proporciona una suma de cuadrados más pequeña, se debe contrastar si ésta reducción en la suma de cuadrados tiene o no significación estadística. Para ello se dispone de dos pruebas adicionales que se describen a continuación:

a. Criterio de información de Akaike.

El denominado valor *AIC* (*Criterio de Información de Akaike*) es un parámetro que se calcula a partir de la suma de cuadrados residuales *SSE* mediante la siguiente expresión:

$$AIC = q \ln(SSE) + 2p \quad (4.1)$$

donde *SSE* es la suma de cuadrados residual, *q* el número de puntos experimentales utilizados y *p* el número de parámetros del modelo. El término *2p* es el encargado de *penalizar* a los modelos de mayor complejidad puesto que se considera mejor modelo aquel que proporciona un valor de *AIC* más pequeño [112].

b. Prueba F.

La prueba *F*, se utiliza para comparar varianzas (sumas de cuadrados divididas por sus grados de libertad). Para establecer si la mejora producida por el modelo mas complejo es estadísticamente significativa se calcula el estadístico *F* mediante la siguiente ecuación:

$$F = \frac{(SSE_s - SSE_c) / (v_s - v_c)}{SSE_c / v_c} \quad (4.2)$$

donde *SSE_s* y *SSE_c* corresponden a las sumas de cuadrados residuales del modelo sencillo y complejo respectivamente, *v_s* son los grados de libertad del modelo sencillo y *v_c* los del modelo mas complejo. Para calcular los grados de libertad asociados a las sumas de cuadrados residuales se utiliza la relación: número de puntos (*q*)-número de parámetros (*p*). El valor de *F* calculado se contrasta con el correspondiente valor tabulado $F_{1-\alpha}(v_s - v_c, v_c)$ para el valor de probabilidad deseado (en general $\alpha = 0.05$) y los grados de libertad del numerador $v_s - v_c$ y del denominador v_c . Si la *F* calculada es menor que la $F_{1-\alpha}$ tabulada se considera que la mejora en la suma de cuadrados no presenta significación estadística.

Entonces, si la F calculada es mayor que la $F_{1-\alpha}$ tabulada se concluye que la mejora es estadísticamente significativa [111].

4.1 Análisis de resultados obtenidos con IT2FNN.

A continuación se hace un análisis para la validación del modelo IT2FNN en base a los resultados de los valores de AIC y la prueba estadística F , comparando los resultados de las arquitecturas del modelo IT2FNN con respecto al modelo ANFIS.

Experimento 1. Identificación de una función no-lineal de una variable.

La Tabla 4.1 muestra en base a AIC que la arquitectura IT2FNN-3 identificó mejor la función no-lineal que ANFIS en este experimento. En la Tabla 4.2 las celdas sombreadas indican que no existe mejora significativa entre las arquitecturas IT2FNN-1 (SNR=0, 10, 30), IT2FNN-2 (SNR=0) con ANFIS.

Tabla 4.1 Valores del Criterio de Información de Akaike (AIC) para los modelos ANFIS y IT2FNN del experimento 1.

SNR(dB)	ANFIS	IT2FNN-0	IT2FNN-1	IT2FNN-2	IT2FNN-3
0	199.04	156.37	155.36	131.94	95.37
10	101.89	42.18	50.22	12.77	-29.11
20	-8.34	-57.25	-96.49	-158.82	-197.41
30	-138.49	-201.26	-207.25	-264.65	-313.05
libre	-414.71	-452.81	-511.53	-582.23	-652.93

Tabla 4.2 Valores estadísticos F para los modelos ANFIS/IT2FNN del experimento 1.

SNR(dB)	ANFIS/IT2FNN-0 $F_{0.95}(7,20) = 2.5140$	ANFIS/IT2FNN-1 $F_{0.95}(19,8) = 3.1613$	ANFIS/IT2FNN-2 $F_{0.95}(19,8) = 3.1613$	ANFIS/IT2FNN-3 $F_{0.95}(19,8) = 3.1613$
0	5.8222	1.6676	2.8847	6.3509
10	9.2656	2.0219	4.6702	11.1525
20	6.9518	4.5744	16.5379	35.7173
30	10.0160	2.9944	10.1053	26.7675
libre	5.0794	5.5000	23.2632	94.3158

Experimento 2. Identificación de una función no-lineal de dos variables.

En la Tabla 4.3 las celdas sombreadas muestran en base a AIC que las arquitecturas IT2FNN-0 (SNR=10) y IT2FNN-3(SNR=0, 20, 30 y libre de ruido) identifican mejor la función no-lineal que ANFIS en este experimento. En la Tabla 4.4 se observa que existe una mejora significativa de la arquitectura IT2FNN-0 en relación a ANFIS. La prueba F para el contraste de las arquitecturas ANFIS/IT2FNN-1, ANFIS/IT2FNN-2 y ANFIS/IT2FNN-3 no se evaluaron por no cumplir con el criterio de tamaño de muestra ($q-p>2$).

Tabla 4.3 Valores del Criterio de Información de Akaike (AIC) para los modelos ANFIS y IT2FNN del experimento 2.

SNR(dB)	ANFIS	IT2FNN-0	IT2FNN-1	IT2FNN-2	IT2FNN-3
0	639.06	582.62	622.84	598.85	522.47
10	348.17	299.32	379.11	353.14	328.00
20	134.28	59.93	76.09	28.35	-42.13
30	17.57	-57.57	-71.39	-138.11	-202.24
libre	16.30	-65.73	-40.41	-181.46	-212.47

Tabla 4.4 Valores estadísticos F para los modelos ANFIS/IT2FNN del experimento 2.

	ANFIS/IT2FNN-0
SNR(dB)	$F_{0.95}(9,48) = 2.0817$
0	4.5331
10	3.9335
20	6.1082
30	6.1819
libre	6.8574

Experimento 3. Identificación de una función no-lineal de tres variables.

La Tabla 4.5 muestra en base a AIC que la arquitectura IT2FNN-3 identificó mejor la función no-lineal que ANFIS en este experimento. En la Tabla 4.6 las celdas sombreadas indican que no existe mejora significativa entre las arquitecturas IT2FNN-0 (SNR=30 y libre de ruido) y IT2FNN-1 (SNR=20) con ANFIS.

Tabla 4.5 Valores del Criterio de Información de Akaike (AIC) para los modelos ANFIS y IT2FNN del experimento 3.

SNR(dB)	ANFIS	IT2FNN-0	IT2FNN-1	IT2FNN-2	IT2FNN-3
0	2105.52	2034.03	2076.96	2044.80	2024.88
10	1277.04	1222.02	1231.49	1199.33	1165.28
20	869.40	856.44	901.43	844.73	789.99
30	794.56	795.93	798.01	775.20	769.51
libre	784.20	796.84	775.20	765.34	750.53

Tabla 4.6 Valores estadísticos F para los modelos ANFIS/IT2FNN del experimento 3.

SNR(dB)	ANFIS/IT2FNN-0 $F_{0.95}(7,290) = 2.0412$	ANFIS/IT2FNN-1 $F_{0.95}(39,258) = 1.4465$	ANFIS/IT2FNN-2 $F_{0.95}(39,258) = 1.4465$	ANFIS/IT2FNN-3 $F_{0.95}(39,258) = 1.4465$
0	11.8039	2.4267	3.3210	3.9186
10	9.2946	2.8887	3.8288	4.9256
20	3.4079	0.9547	2.3240	3.8808
30	1.5632	1.6164	2.1859	2.3341
libre	0.1659	1.9226	2.1732	2.5631

Experimento 4. Pronóstico de la serie de tiempo caótica de Mackey-Glass.

En la Tabla 4.7 las celdas sombreadas muestran en base a AIC que las arquitecturas IT2FNN-0 (TRN/CHK) libre de ruido y IT2FNN-3 (SNR = 0, 10, 20, 30) pronostican mejor la serie caótica de Mackey-Glass con $\tau=13$ que ANFIS. En la Tabla 4.8 se observa que no existe una mejora significativa en las arquitecturas IT2FNN-1 (TRN) a SNR=0, IT2FNN-1 (TRN/CHK) a SNR=10, 30 con ANFIS.

Tabla 4.7 Valores del Criterio de Información de Akaike (AIC) de TRN/CHK y ζ determinados en los modelos ANFIS y IT2FNN con validación cruzada para el pronóstico de la serie de tiempo caótica de Mackey-Glass con $\tau=13$.

SNR(dB)	ANFIS			IT2FNN-0			IT2FNN-1			IT2FNN-2			IT2FNN-3		
	TRN	CHK	ζ	TRN	CHK	ζ	TRN	CHK	ζ	TRN	CHK	ζ	TRN	CHK	ζ
0	1789.26	1920.58	NA	1713.85	1771.31	25	1854.78	1920.46	23	1652.19	1783.94	22	1520.43	1605.20	19
10	665.43	753.37	NA	590.07	690.37	18	709.50	825.16	17	528.46	616.10	16	314.34	455.23	14
20	-422.10	-335.09	NA	-541.18	-424.98	13	-449.83	-395.50	12	-556.89	-470.35	12	-813.06	-603.14	9
30	-1492.20	-1411.23	NA	-1575.55	-1498.59	9	-1470.36	-1389.23	9	-1622.16	-1543.38	10	-2149.52	-1707.68	7
libre	-5217.89	-5217.89	NA	-5893.04	-5893.04	5	-5733.04	-5733.04	4	-5733.04	-5733.04	2	-5733.04	-5733.04	2

Tabla 4.8 Valores estadísticos F (TRN/CHK) y ζ determinados en los modelos ANFIS y IT2FNN con validación cruzada para el pronóstico de la serie de tiempo caótica de Mackey-Glass con $\tau=13$.

SNR(dB)	ANFIS/IT2FNN-0			ANFIS/IT2FNN-1			ANFIS/IT2FNN-2			ANFIS/IT2FNN-3		
	$F_{0.95}(9,395) = 1.9036$			$F_{0.95}(89,315) = 1.3070$			$F_{0.95}(89,315) = 1.3070$			$F_{0.95}(89,315) = 1.3070$		
$\tau=13$	TRN	CHK	ζ	TRN	CHK	ζ	TRN	CHK	ζ	TRN	CHK	ζ
0	9.0153	17.4370	25	0.8929	1.5147	23	3.1070	3.1014	22	5.1109	5.9550	19
10	9.0102	7.7183	18	1.0871	0.8377	17	3.1059	3.1097	16	6.6581	5.6333	14
20	13.8436	10.5696	13	1.8016	2.1623	12	3.0768	3.0830	12	7.5043	5.0975	9
30	9.8623	10.2949	9	1.2975	1.2960	9	3.0133	3.0421	10	15.2743	5.6024	7
libre	131.6667	131.6667	5	10.6180	10.6180	4	10.6180	10.6180	2	10.6180	10.6180	2

En la Tabla 4.9 las celdas sombreadas muestran en base a AIC que la arquitectura IT2FNN-3 (TRN/CHK) pronostica mejor la serie caótica de Mackey-Glass con $\tau=17$ que ANFIS. En la Tabla 4.10 se observa que no existe una mejora significativa en las arquitecturas IT2FNN-1 (CHK) a SNR=10, 20 y libre de ruido con ANFIS.

Tabla 4.9 Valores del Criterio de Información de Akaike (AIC) de TRN/CHK y ζ determinados en los modelos ANFIS y IT2FNN con validación cruzada para el pronóstico de la serie de tiempo caótica de Mackey-Glass con $\tau=17$.

SNR(dB)	ANFIS			IT2FNN-0			IT2FNN-1			IT2FNN-2			IT2FNN-3		
	TRN	CHK	ζ	TRN	CHK	ζ	TRN	CHK	ζ	TRN	CHK	ζ	TRN	CHK	ζ
$\tau=17$															
0	1968.63	2067.27	NA	1872.11	1960.96	28	1911.40	2022.30	25	1807.59	1841.01	24	1618.12	1664.53	20
10	893.58	985.66	NA	761.12	836.79	23	777.22	1041.19	19	558.53	650.47	18	323.75	575.06	15
20	-155.29	-67.49	NA	-226.61	-103.08	17	-142.05	29.02	15	-491.29	-400.32	14	-835.20	-735.82	11
30	-1210.56	-1140.35	NA	-1415.70	-1220.21	13	-1707.68	-1376.33	11	-1543.38	-1470.36	10	-2019.46	-1971.84	8
libre	-2915.30	-2866.51	NA	-3059.82	-2948.60	7	-2960.45	-2842.66	6	-3093.98	-3024.99	4	-3335.14	-3168.09	3

Tabla 4.10 Valores estadísticos F (TRN/CHK) y ζ determinados en los modelos ANFIS y IT2FNN con validación cruzada para el pronóstico de la serie de tiempo caótica de Mackey-Glass con $\tau=17$.

SNR(dB)	ANFIS/IT2FNN-0			ANFIS/IT2FNN-1			ANFIS/IT2FNN-2			ANFIS/IT2FNN-3		
	$F_{0.95}(9,395) = 1.9036$			$F_{0.95}(89,315) = 1.3070$			$F_{0.95}(89,315) = 1.3070$			$F_{0.95}(89,315) = 1.3070$		
$\tau=17$	TRN	CHK	ζ	TRN	CHK	ζ	TRN	CHK	ζ	TRN	CHK	ζ
0	11.2972	12.3887	28	2.1262	1.9890	25	3.4336	4.4051	24	6.6461	7.7678	20
10	15.4097	17.3879	23	2.8374	0.9823	19	6.3359	6.3387	18	12.2542	7.9467	15
20	8.5838	4.9648	17	1.3814	0.6265	15	6.3547	6.2921	14	16.1434	15.6932	11
30	24.6875	9.4877	13	10.1169	4.5609	11	6.2921	6.2368	10	21.9371	23.1138	8
libre	16.8570	9.7261	7	1.9909	1.2781	6	3.6838	3.3978	4	8.1609	5.6964	3

En la Tabla 4.11 las celdas sombreadas muestran en base a AIC que la arquitectura IT2FNN-3 (TRN/CHK) pronostica mejor la serie caótica de Mackey-Glass con $\tau=30$ que ANFIS. En la Tabla 4.12 se observa que existe una mejora significativa en todas arquitecturas del modelo IT2FNN (TRN/CHK) para todos los valores SNR con respecto a ANFIS.

Tabla 4.11 Valores del Criterio de Información de Akaike (AIC) de TRN/CHK y ζ determinados en los modelos ANFIS y IT2FNN con validación cruzada para el pronóstico de la serie de tiempo caótica de Mackey-Glass con $\tau=30$.

SNR(dB)	ANFIS			IT2FNN-0			IT2FNN-1			IT2FNN-2			IT2FNN-3		
	TRN	CHK	ζ	TRN	CHK	ζ	TRN	CHK	ζ	TRN	CHK	ζ	TRN	CHK	ζ
0	2225.19	2331.72	NA	1948.06	2016.09	36	1915.23	2132.15	33	1716.46	1763.06	30	1594.09	1710.04	27
10	1455.41	1535.55	NA	1260.02	1340.42	29	1290.72	1359.54	27	966.46	1060.43	26	864.11	917.24	23
20	864.33	938.03	NA	557.43	684.83	25	515.01	682.06	22	295.24	369.52	20	90.01	304.83	18
30	542.59	599.22	NA	345.29	425.93	19	290.41	328.42	17	-25.93	29.02	16	-215.58	-22.61	8
libre	460.58	517.07	NA	202.79	253.29	9	53.86	150.29	8	-39.30	-66.61	6	-376.45	-248.24	5

Tabla 4.12 Valores estadísticos F (TRN/CHK) y ζ determinados en los modelos ANFIS y IT2FNN con validación cruzada para el pronóstico de la serie de tiempo caótica de Mackey-Glass con $\tau=30$.

SNR(dB)	ANFIS/IT2FNN-0			ANFIS/IT2FNN-1			ANFIS/IT2FNN-2			ANFIS/IT2FNN-3		
	$F_{0.95}(9,395) = 1.9036$			$F_{0.95}(89,315) = 1.3070$			$F_{0.95}(89,315) = 1.3070$			$F_{0.95}(89,315) = 1.3070$		
$\tau=30$	TRN	CHK	ζ	TRN	CHK	ζ	TRN	CHK	ζ	TRN	CHK	ζ
0	35.3061	41.6464	36	5.8527	3.9920	33	10.4374	12.2172	30	14.3131	13.9798	27
10	23.3632	23.3277	29	3.4845	3.6454	27	9.8952	9.5288	26	12.9471	13.8620	23
20	40.1645	31.6065	25	6.6219	4.8913	22	12.2307	12.2124	20	20.2343	14.3880	18
30	23.6202	20.4550	19	4.8276	5.1452	17	12.2126	12.2661	16	19.4788	13.9852	8
libre	32.3015	33.2196	9	7.8578	6.9830	8	10.1922	12.6977	6	23.4108	19.8096	5

En la Tabla 4.13 las celdas sombreadas muestran en base a AIC que la arquitectura IT2FNN-3 (TRN/CHK) pronostica mejor la serie caótica de Mackey-Glass con $\tau=60$ que ANFIS. En la Tabla 4.14 se observa que existe una mejora significativa en todas arquitecturas del modelo IT2FNN (TRN/CHK) para todos los valores SNR con respecto a ANFIS.

Tabla 4.13 Valores del Criterio de Información de Akaike (AIC) de TRN/CHK y ζ determinados en los modelos ANFIS y IT2FNN con validación cruzada para el pronóstico de la serie de tiempo caótica de Mackey-Glass con $\tau=60$.

SNR(dB)	ANFIS			IT2FNN-0			IT2FNN-1			IT2FNN-2			IT2FNN-3		
	TRN	CHK	ζ	TRN	CHK	ζ	TRN	CHK	ζ	TRN	CHK	ζ	TRN	CHK	ζ
0	2221.38	2308.83	NA	1885.18	2003.14	37	1881.28	1933.19	35	1576.85	1603.90	34	1393.05	1542.14	31
10	1431.10	1535.55	NA	1286.27	1390.41	33	1241.44	1287.15	30	957.81	1146.32	28	543.61	711.09	25
20	923.07	1003.69	NA	669.41	767.54	28	605.56	653.84	26	316.70	451.11	24	101.77	139.08	21
30	758.46	830.66	NA	372.26	473.43	19	285.56	415.43	18	0.30	72.10	16	-308.09	22.71	14
libre	707.70	772.32	NA	132.83	274.48	10	38.40	133.43	9	-207.58	32.15	6	-362.40	-52.86	4

Tabla 4.14 Valores estadísticos F (TRN/CHK) y ζ determinados en los modelos ANFIS y IT2FNN con validación cruzada para el pronóstico de la serie de tiempo caótica de Mackey-Glass con $\tau=60$.

SNR(dB)	ANFIS/IT2FNN-0			ANFIS/IT2FNN-1			ANFIS/IT2FNN-2			ANFIS/IT2FNN-3		
	$F_{0.95}(9,395) = 1.9036$			$F_{0.95}(89,315) = 1.3070$			$F_{0.95}(89,315) = 1.3070$			$F_{0.95}(89,315) = 1.3070$		
$\tau=60$	TRN	CHK	ζ	TRN	CHK	ζ	TRN	CHK	ζ	TRN	CHK	ζ
0	45.2382	39.9611	37	6.4362	7.1711	35	14.7991	17.1536	34	22.9459	19.8744	31
10	16.8940	16.9321	33	3.8441	4.7646	30	9.4810	7.4662	28	26.2727	22.7416	25
20	31.6759	29.0756	28	5.9956	6.6327	26	13.4517	11.7186	24	22.5763	24.9395	21
30	54.6110	49.0667	19	9.4710	8.0537	18	19.4781	19.4969	16	39.1109	21.8887	14
libre	99.7651	79.2546	10	15.7303	14.5935	9	27.9771	18.6648	6	39.4150	22.7802	4

En la Tabla 4.15 las celdas sombreadas muestran en base a AIC que la arquitectura IT2FNN-3 (TRN/CHK) pronostica mejor la serie caótica de Mackey-Glass con $\tau=100$ que ANFIS. En la Tabla 4.16 se observa que existe una mejora significativa en todas arquitecturas del modelo IT2FNN (TRN/CHK) para todos los valores SNR con respecto a ANFIS.

Tabla 4.15 Valores del Criterio de Información de Akaike (AIC) de TRN/CHK y ζ determinados en los modelos ANFIS y IT2FNN con validación cruzada para el pronóstico de la serie de tiempo caótica de Mackey-Glass con $\tau=100$.

SNR(dB)	ANFIS			IT2FNN-0			IT2FNN-1			IT2FNN-2			IT2FNN-3		
	TRN	CHK	ζ	TRN	CHK	ζ	TRN	CHK	ζ	TRN	CHK	ζ	TRN	CHK	ζ
0	2444.34	2539.38	NA	2118.98	2215.29	50	1989.20	2265.30	48	1734.91	1816.05	46	1445.51	1494.63	39
10	1605.94	1707.21	NA	1382.83	1418.18	42	1299.59	1385.79	40	851.76	930.10	38	534.16	584.13	31
20	1031.12	1201.29	NA	825.98	907.13	38	799.30	924.97	36	287.99	612.60	34	-180.08	6.76	28
30	648.58	719.65	NA	440.24	630.53	29	273.32	426.06	27	13.17	253.42	24	-545.65	-376.45	19
libre	513.83	594.75	NA	135.24	286.98	13	-6.19	250.90	11	-491.29	-42.68	8	-689.61	-842.69	6

Tabla 4.16 Valores estadísticos F (TRN/CHK) y ζ determinados en los modelos ANFIS y IT2FNN con validación cruzada para el pronóstico de la serie de tiempo caótica de Mackey-Glass con $\tau=100$.

SNR(dB)	ANFIS/IT2FNN-0			ANFIS/IT2FNN-1			ANFIS/IT2FNN-2			ANFIS/IT2FNN-3		
	$F_{0.95}(9,395) = 1.9036$			$F_{0.95}(89,315) = 1.3070$			$F_{0.95}(89,315) = 1.3070$			$F_{0.95}(89,315) = 1.3070$		
$\tau=100$	TRN	CHK	ζ	TRN	CHK	ζ	TRN	CHK	ζ	TRN	CHK	ζ
0	43.3268	43.1053	50	9.0170	5.2022	48	17.3411	17.9294	46	33.7089	37.2914	39
10	27.1968	37.2147	42	5.7851	6.0705	40	19.2960	20.3677	38	39.5592	44.2169	31
20	24.6875	38.0502	38	4.4939	5.2415	36	18.7967	12.8613	34	53.4196	51.5527	28
30	25.1267	10.4860	29	7.1630	5.5501	27	14.4677	9.2985	24	51.5191	41.7074	19
libre	53.1238	40.3112	13	10.7567	6.5113	11	34.1803	14.5403	8	52.5432	86.0115	6

4.2 Análisis de resultados obtenidos con HGA-IT2FNN.

A continuación se hace un análisis para la validación del modelo HGA:IT2FNN en base a los resultados de los valores de AIC y la prueba estadística F , comparando los resultados de las arquitecturas del modelo HGA-IT2FNN con respecto al modelo HGA-ANFIS.

Experimento 1. Identificación de una función no-lineal de una variable.

La Tabla 4.17 muestra en base a AIC que la arquitectura HGA-IT2FNN-3 identificó mejor la función no-lineal que HGA-ANFIS en este experimento. En la Tabla 4.18 las celdas sombreadas indican que no existe mejora significativa entre las arquitecturas HGA-IT2FNN-1 (SNR=0, 10), HGA-IT2FNN-2 (SNR=0) respecto a HGA-ANFIS.

Tabla 4.17 Valores del Criterio de Información de Akaike (AIC) para los modelos HGA-ANFIS y HGA-IT2FNN del experimento 1.

SNR(dB)	HGA-ANFIS	HGA-IT2FNN-0	HGA-IT2FNN-1	HGA-IT2FNN-2	HGA-IT2FNN-3
0	198.66	152.84	152.74	130.54	94.70
10	98.39	40.40	43.82	12.77	-36.10
20	-9.61	-65.55	-105.60	-158.82	-230.33
30	-143.13	-206.19	-227.14	-264.65	-335.81
libre	-429.31	-478.45	-540.87	-652.93	-671.94

Tabla 4.18 Valores estadísticos F para los modelos HGA:ANFIS/IT2FNN del experimento 1.

SNR(dB)	HGA:ANFIS/IT2FNN-0 $F_{0.95}(7,20) = 2.5140$	HGA:ANFIS/IT2FNN-1 $F_{0.95}(19,8) = 3.1613$	HGA:ANFIS/IT2FNN-2 $F_{0.95}(19,8) = 3.1613$	HGA:ANFIS/IT2FNN-3 $F_{0.95}(19,8) = 3.1613$
0	6.3741	1.7611	2.9515	6.3889
10	8.8658	2.1653	4.3330	11.9732
20	8.4011	5.4045	16.1197	66.7963
30	10.0898	4.1856	9.1904	38.3687
libre	6.9971	7.4854	70.7368	102.8710

Experimento 2. Identificación de una función no-lineal de dos variables.

En la Tabla 4.19 las celdas sombreadas muestran en base a AIC que las arquitecturas HGA-IT2FNN-0 (SNR=10) y HGA-IT2FNN-3 (SNR=0, 20, 30 y libre de ruido) identifican mejor la función no-lineal que HGA-ANFIS en este experimento. En la Tabla 4.20 se observa que existe una mejora significativa de la arquitectura HGA-IT2FNN-0 en relación a HGA-ANFIS. La prueba F para el contraste de las arquitecturas HGA:ANFIS/IT2FNN-1, HGA:ANFIS/IT2FNN-2 y HGA:ANFIS/IT2FNN-3 no se evaluaron por no cumplir con el criterio de tamaño de muestra ($q-p>2$).

Tabla 4.19 Valores del Criterio de Información de Akaike (AIC) para los modelos HGA-ANFIS y HGA-IT2FNN del experimento 2.

SNR(dB)	HGA-ANFIS	HGA-IT2FNN-0	HGA-IT2FNN-1	HGA-IT2FNN-2	HGA-IT2FNN-3
0	628.66	582.40	614.99	582.47	454.22
10	334.50	298.19	366.55	311.40	306.14
20	128.77	53.77	62.51	-17.58	-73.34
30	-1.21	-94.87	-82.32	-203.92	-242.06
libre	-10.94	-88.47	-75.31	-215.98	-254.17

Tabla 4.20 Valores estadísticos F para los modelos HGA:ANFIS/IT2FNN-0 del experimento 2.

SNR(dB)	HGA:ANFIS/IT2FNN-0 $F_{0.95}(9,48) = 2.0817$
0	3.7369
10	3.0215
20	6.1695
30	8.0873
libre	6.4129

Experimento 3. Identificación de una función no-lineal de tres variables.

La Tabla 4.21 muestra en base a AIC que la arquitectura HGA-IT2FNN-3 identificó mejor la función no-lineal que HGA-ANFIS en este experimento. En la Tabla 4.22 las celdas sombreadas indican que no existe mejora significativa entre las arquitecturas HGA-IT2FNN-0 (SNR=30 y libre de ruido) y HGA-IT2FNN-1 (SNR=0) con HGA-ANFIS.

Tabla 4.21 Valores del Criterio de Información de Akaike (AIC) para los modelos HGA-ANFIS y HGA-IT2FNN del experimento 3.

SNR(dB)	HGA-ANFIS	HGA-IT2FNN-0	HGA-IT2FNN-1	HGA-IT2FNN-2	HGA-IT2FNN-3
0	2018.84	2006.97	2060.64	2021.98	1997.10
10	1252.60	1206.99	1212.76	1162.07	1100.62
20	867.39	831.84	830.15	822.50	756.39
30	767.66	767.10	769.51	756.92	719.37
libre	748.81	748.81	745.71	721.04	686.12

Tabla 4.22 Valores estadísticos F para los modelos HGA:ANFIS/IT2FNN del experimento 3.

SNR(dB)	HGA:ANFIS/IT2FNN-0 $F_{0.95}(7,290) = 2.0412$	HGA:ANFIS/IT2FNN-1 $F_{0.95}(39,258) = 1.4465$	HGA:ANFIS/IT2FNN-2 $F_{0.95}(39,258) = 1.4465$	HGA:ANFIS/IT2FNN-3 $F_{0.95}(39,258) = 1.4465$
0	3.2657	0.7410	1.6239	2.2478
10	7.9139	2.7309	4.2288	6.3699
20	6.4798	2.6600	2.8705	4.8999
30	1.8080	1.6554	1.9665	2.9656
libre	1.7366	1.7763	2.4057	3.3786

4.3 Análisis de resultados obtenidos con PSO-IT2FNN.

A continuación se hace un análisis para la validación del modelo PSO-IT2FNN en base a los resultados de los valores de AIC y la prueba estadística F , comparando los resultados de las arquitecturas del modelo PSO-IT2FNN con respecto al modelo PSO-ANFIS.

Experimento 1. Identificación de una función no-lineal de una variable.

La Tabla 4.23 muestra en base a AIC que la arquitectura PSO-IT2FNN-3 identificó mejor la función no-lineal que PSO-ANFIS en este experimento. En la Tabla 4.24 las celdas sombreadas indican que no existe mejora significativa entre las arquitecturas PSO-IT2FNN-0 (SNR=30), PSO-IT2FNN-1 (SNR=0, 10, 30) y PSO-IT2FNN-2 (SNR=0) respecto a PSO-ANFIS.

Tabla 4.23 Valores del Criterio de Información de Akaike (AIC) para los modelos PSO-ANFIS y PSO-IT2FNN del experimento 1.

SNR(dB)	PSO-ANFIS	PSO-IT2FNN-0	PSO-IT2FNN-1	PSO-IT2FNN-2	PSO-IT2FNN-3
0	198.46	152.69	151.22	129.59	94.43
10	97.32	40.04	42.15	10.96	-39.04
20	-13.40	-69.17	-121.56	-168.07	-274.15
30	-139.40	-153.68	-162.09	-284.62	-341.42
libre	-461.30	-523.52	-581.72	-667.14	-717.69

Tabla 4.24 Valores estadísticos F para los modelos PSO:ANFIS/IT2FNN del experimento 1.

SNR(dB)	PSO:ANFIS/IT2FNN-0 $F_{0.95}(7,20) = 2.5140$	PSO:ANFIS/IT2FNN-1 $F_{0.95}(19,8) = 3.1613$	PSO:ANFIS/IT2FNN-2 $F_{0.95}(19,8) = 3.1613$	PSO:ANFIS/IT2FNN-3 $F_{0.95}(19,8) = 3.1613$
0	6.3650	1.8186	3.0012	6.3980
10	8.7034	2.1955	4.4020	12.4344
20	8.3656	6.9754	17.9871	146.9321
30	2.1176	0.9629	14.8737	46.1670
libre	9.8765	8.9847	49.7837	134.8584

Experimento 2. Identificación de una función no-lineal de dos variables.

En la Tabla 4.25 las celdas sombreadas muestran en base a AIC que las arquitecturas PSO-IT2FNN-0 (SNR=10) y PSO-IT2FNN-3 (SNR=0, 20, 30 y libre de ruido) identifican mejor la función no-lineal que PSO-ANFIS en este experimento. En la Tabla 4.26 se observa que existe una mejora significativa de la arquitectura PSO-IT2FNN-0 en relación a PSO-ANFIS. La prueba F para el contraste de las arquitecturas PSO:ANFIS/IT2FNN-1, PSO:ANFIS/IT2FNN-2 y PSO:ANFIS/IT2FNN-3 no se evaluaron por no cumplir con el criterio de tamaño de muestra ($q-p>2$).

Tabla 4.25 Valores del Criterio de Información de Akaike (AIC) para los modelos PSO-ANFIS y PSO-IT2FNN del experimento 2.

SNR(dB)	PSO-ANFIS	PSO-IT2FNN-0	PSO-IT2FNN-1	PSO-IT2FNN-2	PSO-IT2FNN-3
0	627.68	582.05	609.28	582.40	438.55
10	325.25	297.19	366.71	311.20	298.26
20	122.05	46.65	57.99	-22.30	-82.32
30	-6.26	-105.92	-87.46	-217.75	-244.04
libre	-15.23	-89.18	-83.34	-228.67	-273.54

Tabla 4.26 Valores estadísticos F para los modelos PSO:ANFIS/IT2FNN-0 del experimento 2.

SNR(dB)	PSO:ANFIS/IT2FNN-0
	$F_{0.95}(9,48) = 2.0817$
0	3.6901
10	2.4708
20	6.2068
30	8.7703
libre	6.0696

Experimento 3. Identificación de una función no-lineal de tres variables.

La Tabla 4.27 muestra en base a AIC que la arquitectura PSO-IT2FNN-3 identificó mejor la función no-lineal que PSO-ANFIS en este experimento. En la Tabla 4.28 las celdas sombreadas indican que no existe mejora significativa entre las arquitecturas PSO-IT2FNN-0 (SNR=30) y PSO-IT2FNN-1 (SNR=0) con PSO-ANFIS.

Tabla 4.27 Valores del Criterio de Información de Akaike (AIC) para los modelos PSO-ANFIS y PSO-IT2FNN del experimento 3.

SNR(dB)	PSO-ANFIS	PSO-IT2FNN-0	PSO-IT2FNN-1	PSO-IT2FNN-2	PSO-IT2FNN-3
0	2018.10	2006.58	2057.65	2021.57	1996.24
10	1248.48	1205.24	1206.21	1156.79	1091.93
20	857.28	827.94	821.53	789.99	724.94
30	765.80	764.72	754.26	727.15	671.87
libre	741.12	732.52	729.36	710.36	605.13

Tabla 4.28 Valores estadísticos F para los modelos PSO:ANFIS/IT2FNN del experimento 3.

SNR(dB)	PSO:ANFIS/IT2FNN-0 $F_{0.95}(7,290) = 2.0412$	PSO:ANFIS/IT2FNN-1 $F_{0.95}(39,258) = 1.4465$	PSO:ANFIS/IT2FNN-2 $F_{0.95}(39,258) = 1.4465$	PSO:ANFIS/IT2FNN-3 $F_{0.95}(39,258) = 1.4465$
0	3.2189	0.7895	1.6160	2.2508
10	7.5713	2.7976	4.2657	6.5451
20	5.6142	2.6193	3.5142	5.6433
30	1.8736	1.9864	2.6982	4.3374
libre	2.8393	1.9920	2.4854	5.7752

CAPÍTULO 5. CONCLUSIONES

En esta Tesis se ha abordado la modelación de arquitecturas híbridas inteligentes *neuro-genético-difusa tipo-2 por intervalos* (HGA-IT2FNN), *neuro-swarm-difuso tipo-2 por intervalos* (PSO-IT2FNN) para la aplicación de identificación de funciones no-lineales y pronóstico de la serie de tiempo caótica de Mackey-Glass. La conjunción de redes neuronales, sistemas de lógica difusa tipo-2 por intervalos con algoritmos genéticos refuerza el aprendizaje, explicación, flexibilidad, descubrimiento y manejo de fuentes de incertidumbre, características deseables en el procesamiento inteligente de información imperfecta. Las experiencias obtenidas al trabajar con sistemas híbridos IT2FNN, HGA-IT2FNN y PSO-IT2FNN nos indican que éste enfoque es sumamente eficiente por su elevada capacidad de representación, poseen una estructura que las hacen altamente paralelizables en las plataformas actuales.

Los resultados de la serie de tiempo caótica de Mackey-Glass analizada con validación cruzada, AIC y prueba F, muestra que las arquitecturas híbridas IT2FNN, HGA-IT2FNN y PSO-IT2FNN pueden ser derivados como una generalización al caso no lineal de los modelos autorregresivos en el contexto de las series de tiempo. Esta derivación permite especificar de forma práctica una clase mas general de modelos para la identificación y pronóstico de series de tiempo, en la cual, los conjuntos de variables de entrada-salida conforman la base de conocimiento de la dinámica de la serie de tiempo. Esto posibilita la aplicación de esta metodología a series con diversas dinámicas, con un número relativamente bajo de variables causales para explicar su comportamiento bajo los efectos de fuentes de incertidumbre.

El análisis del valor RMSE estimado para los experimentos de identificación de funciones no-lineales con validación cruzada, Criterio de Información de Akaike (AIC) y prueba F para las

arquitecturas híbridas IT2FNN, HGA-IT2FNN y PSO-IT2FNN, demuestran ser aproximadores universales no-lineales para la eficiente identificación de funciones no-lineales.

Estos resultados incentivan la investigación de esta metodología en el modelado de sistemas reales, pero aún esta lejos de ser completa, es necesario investigar en el contexto de inteligencia computacional modelos dinámicos tales como procesos de control, biomedicina, sistemas ambientales, dinámicas de aviones, macroeconomía, procesamiento de señales, etc.

Las arquitecturas híbridas inteligentes IT2FNN, HGA-IT2FNN y PSO-IT2FNN, forman un versátil modelo para la solución de varios problemas de la ciencia y la ingeniería. La aportación de estos modelos ha sido evidenciado por los experimentos de las aplicaciones descritas en este trabajo. En el análisis de contraste estadístico de validación cruzada, AIC y prueba F; las arquitecturas de los modelos híbridos inteligentes IT2FNN, HGA-IT2FNN y PSO-IT2FNN; resultaron en todas las pruebas mejor (altas fuentes de incertidumbre) o igual (pequeñas fuentes de incertidumbre) que el modelo ANFIS, con la desventaja de este, de no poder evaluar el intervalo de incertidumbre de salida, $\tilde{Y} \in [\hat{y}_l, \hat{y}_r]$.

REFERENCIAS

- [1]. Klir, G. J. and T. A Folger, Fuzzy Sets, Uncertainty and Information. Prentice Hall, Englewood Cliffs, NJ, 1988.
- [2]. Klir, G. J. and M. J. Wierman, Uncertainty-Based Information: Elements of Generalized Information Theory. Physica-Verlag / Springer-Verlag, Heidelberg and New York, 1999.
- [3]. Davis, T.J. y Keller, C.P. (1997): "Modelling uncertainty in natural resource analysis using fuzzy sets and Monte Carlo simulation: slope stability prediction", *International Journal of Geographical Information Science*, 11, 5, pp. 409-434.
- [4]. GIL ALUJA, J. (1999): *Elements for a theory of decision in uncertainty*. Boston: Kluwer Academic Publishers.
- [5]. Hagrais, F., Roberts, H., Callaghan, D., "A Type-2 Fuzzy Based System for Handling the Uncertainties in Group Decisions for Ranking Job Applicants within Human Resources Systems". In FUZZ-IEEE 2008 , Hong Kong , June 2008.
- [6]. C. Hwang and F.C-H. Rhee, "Uncertain Fuzzy Clustering: Interval Type-2 Fuzzy", *IEEE Transaction on Fuzzy System*, vol. 15, No. 1, pp. 107-120, 2007.
- [7]. P. Eklund, H. Virtanen and T. Riissanen, On the fuzzy logic nature of neural nets, in: *Proceedings of Neuro-Nimes*, 1991 293–300.
- [8]. K. Hirota and W. Pedrycz, Knowledge-based networks in classification problems, *Fuzzy Sets and Systems*, 51(1992) 1-27.
- [9]. K. Hirota and W. Pedrycz, OR/AND neuron in modeling fuzzy set connectives, *IEEE Transactions on Fuzzy Systems*, 2(1994) 151-161.
- [10]. S. Horikawa, T. Furuhashi and Y. Uchikawa, On fuzzy modeling using fuzzy neural networks with the backpropagation algorithm, *IEEE Transactions on Neural Networks*, 3(1992).
- [11]. H. Ishibuchi, H. Okada and H. Tanaka, Interpolation of fuzzy IF-THEN rules by neural networks, *International Journal of Approximate Reasoning*, 10(1994) 3-27.
- [12]. H. Ishibuchi, K. Kwon and H. Tanaka, A learning algorithm of fuzzy neural networks with triangular fuzzy weights, *Fuzzy Sets and Systems*, 71(1995) 277-293.
- [13]. J.M. Keller and H.Tahani, Implementation of conjunctive and disjunctive fuzzy logic rules with neural networks, *International Journal of Approximate Reasoning*, 6(1992) 221-240.

- [14]. H.K. Kwan and Y. Cai, A fuzzy neural network and its application to pattern recognition, *IEEE Transactions on Fuzzy Systems*, 3(1994) 185-193.
- [15]. Juang Chia Feng, Lin Chin Teng, *An Online Self Constructing Neural Fuzzy Inference Network and its Applications*, *IEEE Transactions on Fuzzy Systems*, Vol 6, No.1, pp. 12-32, 1998.
- [16]. Sulzberger SM, Tschicholg-Gurman NN, Vestli SJ, *FUN: Optimization of Fuzzy Rule Based Systems Using Neural Networks*, In *Proceedings of IEEE Conference on Neural Networks*, San Francisco, pp 312-316, March 1993.
- [17]. Tano S, Oyama T, Arnould T, *Deep combination of Fuzzy Inference and Neural Network in Fuzzy Inference*, *Fuzzy Sets and Systems*, 82(2) pp. 151-160, 1996.
- [18]. Kasabov N and Qun Song, *Dynamic Evolving Fuzzy Neural Networks with 'm-out-of-n' Activation Nodes for On-line Adaptive Systems*, Technical Report TR99/04, Department of information science, University of Otago, 1999.
- [19]. Nauck D, Kruse R, *Neuro-Fuzzy Systems for Function Approximation*, 4th International Workshop Fuzzy-Neuro Systems, 1997.
- [20]. Bherenji H R and Khedkar P, Learning and Tuning Fuzzy Logic Controllers through Reinforcements, *IEEE Transactions on Neural Networks*, Vol (3), pp. 724-740, 1992.
- [21]. Abraham A & Nath B, Evolutionary Design of Neuro-Fuzzy Systems – A Generic Framework, In *Proceedings of the 4th Japan – Australia joint Workshop on Intelligent and Evolutionary Systems*, Japan, November 2000.
- [22]. Chaoui, H., Gueaieb, W., “Type-2 Fuzzy Logic Control of a Flexible-Joint Manipulator”. *JOURNAL OF INTELLIGENT AND ROBOTIC SYSTEMS* 51(2): 159-186, 2008.
- [23]. Wu, D, Tan, W. W., “A simplified type-2 fuzzy logic controller for real-time control”. *ISA TRANSACTIONS* 45(4):503-516, 2006. ISSN: 0019-0578.
- [24]. Castillo, O., Melin, P., Kacprzyk, J., Pedrycz, W., “Type-2 Fuzzy Logic theory and applications”. In *Proceedings of Granular Computing 2007*, pages 145-150, Silicon Valley, CA, USA, November 2007.
- [25]. Gu, L, Zhang, Y. Q., “Web shopping expert using new interval type-2 fuzzy reasoning”. *SOFT COMPUTING* 11(8):741-751, 2007.
- [26]. Jia Zeng, Zhi-Qiang Liu, “Type-2 fuzzy sets for pattern recognition: The state-of-the-art”. *Journal Uncertain System*. 1(3): 163-177, August 2007.
- [27]. Castro, J. R., Castillo, O., Melin, P., Rodriguez-Diaz, A., “Building fuzzy inference systems with a new interval type-2 fuzzy logic toolbox”. In *Transactions on Computational Science. Book Chapter*, Volume 4750: 104-114, June 2008. Springer-Verlag. ISBN: 978-3-540-79298-7, ISSN: 0302-9743 (Print) 1611-3349 (Online).

- [28]. Castro, J. R., Castillo, O., Martinez, L. G., "Interval type-2 fuzzy logic toolbox". *Journal of Engineering Letters* 15(1):89-98, August 2007. ISSN: 1816-0948 (online version); 1816-093X (print version)
- [29]. Mendel, J. M., "Advances in type-2 fuzzy sets and systems". *Information Sciences* 177(1):84-110, January 2007.
- [30]. Mendel, J. M., Wu, H., "New results about the centroid of an interval type-2 fuzzy set, including the centroid of a fuzzy granule". *INFORMATION SCIENCES* 177(2): 360-377, 2007.
- [31]. Mendel, J. M., Wu, H., "Super-Exponential Convergence of the Karnik-Mendel Algorithms for Computing the Centroid of an Interval Type-2 Fuzzy Set". *IEEE TRANSACTIONS ON FUZZY SYSTEMS* 15(2):309-320, 2007.
- [32]. Mendel, J.M., Wu, H., "Type-2 Fuzzistics for Nonsymmetric Interval Type-2 Fuzzy Sets: Forward Problems". *IEEE Transactions on Fuzzy Systems* 15(5):916-930, 2007.
- [33]. Mendel, J.M., D. Wu, "Perceptual Reasoning: A New Computing with Words Engine". In *Proceedings of Granular Computing 2007*, pages 446-451, Silicon Valley, CA, USA, November 2007.
- [34]. Mendoza, O., Melin, P., Licea, G., "Fuzzy Inference Systems Type-1 and Type-2 for Digital Images Edge Detection". *Journal of Engineering Letters* 15(1):45-52, August 2007.
- [35]. Sepulveda, R., Castillo, O., Melin, P., Rodriguez-Diaz, A., Montiel, O., "Experimental study of intelligent controllers under uncertainty using type-1 and type-2 fuzzy logic". *INFORMATION SCIENCES* 177(10):2023-2048, 2007.
- [36]. Singh, M., Srivastava, S., Gupta, J.R.P., Hanmandlu, M., "A new algorithm-based type-2 fuzzy controller for diabetic patient". *INTERNATIONAL JOURNAL OF BIOMEDICAL ENGINEERING AND TECHNOLOGY* 1(1):18-40, 2007.
- [37]. Wu, D., Mendel, J. M., "Uncertainty measures for interval type-2 fuzzy sets". *INFORMATION SCIENCES*, 177(23):5378-5393, 2007.
- [38]. Wu, D., Mendel, J.M., "Aggregation Using the Linguistic Weighted Average and Interval Type-2 Fuzzy Sets". *IEEE TRANSACTIONS ON FUZZY SYSTEMS* 15(6): 1145-1161, 2007.
- [39]. Wu, H.-Y., Wu, Y.-Y., Liu, H.-T., Zhang, H.-J., "Roughness of Type-2 Fuzzy Set Based on Similarity Relation". *INTERNATIONAL JOURNAL OF UNCERTAINTY FUZZINESS AND KNOWLEDGE BASED SYSTEMS* 15(4): 503-518, 2007.
- [40]. Astudillo, L., Castillo, O., Melin, P., Alanis, A., Soria, J., Aguilar, L., "Intelligent Control of an Autonomous Mobile Robot using Type-2 Fuzzy Logic". *Journal of Engineering Letters* 13(2): 93-97, September 2006.

- [41]. Baguley, P., Page, T., Koliza, V., Maropoulos, P., "Time to market prediction using type-2 fuzzy sets". JOURNAL OF MANUFACTURING TECHNOLOGY MANAGEMENT 17(4): 513-520, 2006.
- [42]. Zeng, J., Liu, Z.-Q., "Type-2 Fuzzy Hidden Markov Models and Their Application to Speech Recognition". IEEE TRANSACTIONS ON FUZZY SYSTEMS VOL 14(3): 454-467, 2006.
- [43]. M. Mizumoto and K. Tanaka, "Some properties of fuzzy sets of type-2," Information and Control, vol. 31 pp. 312-340, 1976.
- [44]. J. M. Mendel, Uncertain Rule-Based Fuzzy Logic Systems: Introduction and New Directions, Prentice Hall, NJ. 2001.
- [45]. N. N. Karnik and J. M. Mendel, "Applications of type-2 fuzzy logic systems to forecasting of time-series," Inform. Sci., vol. 120, pp. 89-111, 1999.
- [46]. N. N. Karnik and J. M. Mendel, "Operations on type-2 fuzzy sets," International Journal on Fuzzy Sets and Systems, vol. 122(2), pp. 327-348, 2001.
- [47]. Q. Liang and J. M. Mendel, "Interval type-2 logic systems: theory and design," IEEE Transactions on Fuzzy Systems, vol. 8 no. 5, pp. 535-550, Oct. 2000.
- [48]. H. Wu, J. Mendel, Uncertainty Bounds and Their use in the design of Interval Type-2 Fuzzy Logic Systems, IEEE Transactions on Fuzzy Systems, Vol 10(2) 2002 pp 622-639.
- [49]. D. Wu and J. M. Mendel, "A Vector Similarity Measure for Interval Type-2 Fuzzy Sets and Type-1 Fuzzy Sets," Information Sciences, 178, pp. 381-402, 2008.
- [50]. Lee, C-H., Hu, T-W, Lee, C-T, Lee, Y-C, "A Recurrent Interval Type-2 Fuzzy Neural Network with Asymmetric Membership Functions for Nonlinear System Identification". In FUZZ-IEEE 2008, Hong Kong, June 2008.
- [51]. A Hybrid Learning Algorithm for Interval Type-2 Fuzzy Neural Networks: The Case of Time Series Prediction. Castro, J. R., Castillo, O., Melin, P., Rodriguez-Diaz, A.; In Soft Computing for Hybrid Intelligent Systems. Book Chapter, Volume 154/2008: 363-386, September 2008. Springer Berlin / Heidelberg. ISBN: 978-3-540-70811-7, ISSN: 1860-949X (Print) 1860-9503 (Online).
- [52]. Babek Guirimov, Umit Ilhan, Kaan Uyar. "Neuro-Fuzzy Type 2 System for Loan Assessment". In Forth International Conference on Soft Computing, Computing with Words and Perceptions in System Analysis, Decision and Control , Antalya, Turkey, August 2007.
- [53]. Frank Chung-Hoon Rhee, "Interval Type-2 Fuzzy Membership Function Design and its Application to Radial Basis Function Neural Networks". In Proc. FUZZ-IEEE 2007, pages 2047 - 2052, London, 2007.

- [54]. Hybrid learning algorithm for interval type-2 fuzzy neural networks. Castro, J.R., Castillo, O., Melin, P., Rodriguez-Diaz, A.; In Proceedings of Granular Computing 2007, pages 157-162, Silicon Valley, Ca, USA, November 2007. ISBN: 0-7695-3032-X.
- [55]. Lee, C.-H, Lin, Y.-C, "Type-2 Fuzzy Neuro System Via Input-to-State-Stability Approach". LECTURE NOTES IN COMPUTER SCIENCE (4492):317-327, 2007.
- [56]. Mendez, G. M., "Interval Type-2 ANFIS". Advances in Soft Computing – Innovations in Hybrid Intelligent Systems; Pages 64-71, 2007.
- [57]. Mendez, G. M., Hernandez, M. A., "Modelling and Prediction of the MXNUSD Exchange Rate Using Hybrid RLS_BP Interval Type-1 Non-Singleton Type-2 Fuzzy Logic Systems". Journal of Research in Computer Science 32:204-213, November 2007.
- [58]. Mendez, G.M., Juarez, I.L., "First-order Interval Type-2 TSK Fuzzy Logic Systems Using Hybrid Learning Systems". WSEAS Trans. On Computers 4, April 2005. ISSN 1109-2750.
- [59]. Mendez, G.M., Juarez, I.L., "Orthogonal-BackPropagation Hybrid Learning Algorithm for Interval Type-1 Non-Singleton Type-2 Fuzzy Logic Systems". WSEAS Trans. On Computers 4, March 2005. ISSN 1109-2777.
- [60]. Yu-Ching Lin and Ching-Hung Lee, "System Identification and Adaptive Filter Using a Novel Fuzzy Neuro System", INTERNATIONAL JOURNAL OF COMPUTATIONAL COGNITION, VOL. 5, NO. 1, MARCH 2007
- [61]. C. H. Lee, J. L. Hong, Y. C. Lin, and W. Y. Lai, "Type-2 Fuzzy Neural Network Systems and Learning," International Journal of Computational Cognition, Vol. 1, No. 4, pp. 79-90, 2003.
- [62]. C. H. Wang, C. S. Cheng, and Tsu-Tian Lee, "Dynamical optimal training for interval type-2 fuzzy neural network (T2FNN)," IEEE Trans. on Systems, Man, and Cybernetics Part B: Cybernetics, Vol. 34, No. 3, pp. 1462-1477, 2004.
- [63]. Hagrais, H., "Comments on Dynamical Optimal Training for Interval Type-2 Fuzzy Neural Network (T2FNN)". IEEE TRANSACTIONS ON SYSTEMS MAN AND CYBERNETICS PART B 36(5): 1206-1209, 2006.
- [64]. Mendez, G. M., Leduc, L., "Hybrid Learning Algorithm for Interval Type-2 Fuzzy Logic Systems". Control and Intelligent Systems Journal, Special Issue on Nonlinear Adaptive PID Control Part 2 34(3): 206-215, 2006.
- [65]. Mendez, G. M., Leduc, L., Cavazos, A., Soto, R., "Entry temperature prediction of a hot strip mill by a hybrid learning type-2 FLS". Journal of Intelligence and Fuzzy Systems 17(6):583-596, December 2006.

- [66]. Own, C.-M., Tsai, H.-H., Yu, P.-T., Lee, Y.-J., “Adaptive type-2 fuzzy median filter design for removal of impulse noise”. *Imaging Science* 54(1): 3-18, March 2006. ISSN 1368-2199, Online ISSN: 1743-131X.
- [67]. R. I. John and C. Czarnecki, “A type 2 adaptive fuzzy inferencing system,” in *Proc. IEEE Systems, Man and Cybernetics*, pp. 2068-2073, 1998.
- [68]. Cao, X.-Q., Zeng, J., Yan, H., “Modeling Uncertain Speech Sequences Using Type-2 Fuzzy Hidden Markov Models”. *LECTURE NOTES IN COMPUTER SCIENCE* 4810:315-324, 2007.
- [69]. W. Pedrycz, *Fuzzy Modelling: Paradigms and Practice*. Kluwer Academic Press, 1996.
- [70]. Tang K.-S., K.-F. Man, Z.-F. Liu and S. Kwong (1998), “Minimal fuzzy memberships and rules using hierarchical genetic algorithms”, *IEEE Trans. on Industrial Electronics*, vol. 45, no. 1.
- [71]. C. H. Wang, H.L. Liu and C. T. Lin, “Dynamic optimal Learning rate of A Certain Class of Fuzzy Neural Networks and Its Applications with Genetic Algorithm,” *IEEE Trans. Syst., Man, Cybern.*, vol. 31, no. 3, pp. 467-475, Jun. 2001.
- [72]. Castillo, O., Martinez, A. I., Martinez, A. C., “Evolutionary Computing for Topology Optimization of Type-2 Fuzzy Systems”. In Patricia Melin et al, editor, *Analysis and Design of Intelligent Systems using Soft Computing Techniques*, Number 1 in *Studies in Fuzziness and Soft Computing*, 7, pages 63-75, Springer-Verlag, Germany, 1 edition, June 2007.
- [73]. Castillo, O., Sepulveda, R., Melin, P., Montiel, O., “Evolutionary Optimization of Interval Type-2 Membership Functions”. In Hamid Arabnia, editor, *Proceedings of IC-AI 2006*, pages 558-564, Las Vegas, Nevada, June 2006.
- [74]. Castillo, O., Melin, P., Sepulveda, R., “Evolutionary Optimization of Interval Type-2 Fuzzy Systems for Intelligent Control”. In Cihan H. Dagli, editor, *Smart Engineering System Design*, Number 1, 22, pages 131-136, ASME Press, New York, 1 edition, November 2006.
- [75]. Melin, P., Urias, J., Solano, D., Soto, M., Lopez, M., Castillo, O., “Voice Recognition with Neural Networks, Type-2 Fuzzy Logic and Genetic Algorithms”. *Journal of Engineering Letters* 13(2):108-116, September 2006.
- [76]. Wu, D., Wan Tan, W., “Genetic learning and performance evaluation of interval type-2 fuzzy logic controllers”. *ENGINEERING APPLICATIONS OF ARTIFICIAL INTELLIGENCE* 19(8): 829-841, 2006. ISSN: 0952-1976.
- [77]. G. Ascia, V. Catania, and D. Panno, “An Integrated Fuzzy-GA Approach for Buffer Management,” *IEEE Trans. Fuzzy Syst.*, vol. 14, no. 4, pp. 528-541, Aug. 2006.

- [78]. P.P. Bonissone, R. Subbu, N. Eklund, and T.R. Kiehl, "Evolutionary Algorithms + Domain Knowledge = Real-World Evolutionary Computation", IEEE Trans. Evol Comput., vol. 10, No. 3, pp. 256-280, June 2006.
- [79]. Y.-C. Chiou and L. W. Lan, "Genetic fuzzy logic controller: an iterative evolution algorithm with new encoding method," Fuzzy Sets Syst., vol. 152, no. 3, pp. 617-635, June 2005.
- [80]. J. Holland, *Adaptation in Natural and Artificial Systems*. Ann Arbor: The University of Michigan Press, 1975.
- [81]. W. Pedrycz, *Fuzzy Evolutionary Computation*. Kluwer Academic Publishers, 1997.
- [82]. Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*. New York: Springer-Verlag, 1996.
- [83]. Deb, K., 2005. A population-based algorithm-generator for real-parameter optimization. *Soft Computing*, (in press). Springer.
- [84]. Castillo, O., Huesca, G., Valdez, F., "Evolutionary Computing for Optimizing Type-2 Fuzzy Systems in Intelligent Control of Non-Linear Dynamic Plants". In Proc. North American Fuzzy Info. Processing Society (NAFIPS), pages 247 - 251, Ann Arbor, MI, June 2005.
- [85]. H. Ishibuchi, K. Nozaki, N. Yamamoto, and H. Tanaka, Selecting fuzzy if-then rules for classification problems using genetic algorithms, IEEE Trans. Fuzzy Syst., vol. 3, pp. 260—270, 1995.
- [86]. M. Dorigo and L.M. Ganbardella. Ant colony system: A cooperative learning approach to the travelling salesman problem. IEEE Transaction on Evolutionary Computation, 1(1):53–66, 1997.
- [87]. M. Dorigo, V. Maniezzo & A. Coloni, 1996. "Ant System: Optimization by a Colony of Cooperating Agents", IEEE Transactions on Systems, Man, and Cybernetics–Part B, 26 (1): 29–41.
- [88]. J. Kennedy and R.C. Eberhart. The Particle Swarm: Social Adaptation in Information Processing Systems, chapter 25 in *New Ideas in Optimization*, pages 379–387. McGraw Hill, 1999.
- [89]. J. Kennedy and R.C. Eberhart. *Swarm Intelligence*. Morgan Kaufmann Publishers, 2003.
- [90]. M. Clerc, and J. Kennedy, The Particle Swarm-Explosion, Stability, and Convergence in a Multidimensional Complex Space, IEEE Transactions on Evolutionary Computation, 2002, 6, 58-73.
- [91]. Andries P. Engelbrecht. *Fundamentals of computational swarm intelligence*. John Wiley & Sons, Ltd. ISBN 0-470-09191-6, 2005.

- [92]. J.-S. R. Jang, "ANFIS: adaptive-network-based fuzzy inference system". IEEE Trans. Syst., Man, Cybern., vol. 23, no. 3, pp. 665-684, 1993.
- [93]. J.S.R. Jang, C.T. Sun, and E. Mizutani, Neuro-fuzzy and Soft Computing. New York: Prentice-Hall, 1997.
- [94]. Castillo, O., Melin, P., "Comparison of Hybrid Intelligent Systems, Neural Networks, and Interval Type-2 Fuzzy logic for Time Series Prediction". In Jennie Si, editor, Proc. IJCNN-IEEE 2007, Orlando, USA, August 2007.
- [95]. Mendoza, O., Melin, P., Castillo, O., Licea, G., "Type-2 Fuzzy Systems as a Method for Improving Training Data and Decision Making in Modular Neural Networks for Pattern Recognition". International Journal of Information Technology and Intelligent Computing 1(4):1-10, December 2006.
- [96]. E. H. Mamdani and S. Assilian, "An experiment in linguistic synthesis with a fuzzy logic controller," Int. J. Man-Mach. Stud., vol. 7, pp. 1-13, 1975.
- [97]. S. Russell and P. Norvig, Artificial Intelligence: A Modern Approach. NJ: Prentice-Hall, 2003.
- [98]. T. Takagi and M. Sugeno, "Fuzzy identification of systems and its applications to modeling and control," IEEE Trans. Syst., Man, Cybern., vol. 15, pp. 116–132, Jan.–Feb. 1985.
- [99]. L.X. Wang and J.M. Mendel, "Generating fuzzy rules by learning from examples," IEEE Trans. Syst., Man, Cybern., vol. 22, pp. 1414–1427, 1992.
- [100]. L. A. Zadeh, "Fuzzy Sets," Inform. Control, vol. 8, no. 3, pp. 338-353, June 1965.
- [101]. L. A. Zadeh, "Fuzzy Logic, Neural Networks and Soft Computing ," Communications of the ACM., vol. 37, no. 3, pp. 77-84, 1994.
- [102]. L. A. Zadeh, "Outline of a new approach to the analysis of complex systems and decision processes," IEEE Trans. Syst., Man, and Cybern., vol. 3, no. 1, pp. 28-44, 1973.
- [103]. L.A. Zadeh, The concept of a linguistic variable and its application to approximate reasoning, Part II, Information Sciences 8(4) (1975) 301–357.
- [104]. L.A. Zadeh, Fuzzy logic = computing with words, IEEE Trans. Fuzzy Syst. 4 (1996) 103–111.
- [105]. Hagan, M. T., H. B. Demuth, and M. H. Beale, Neural Network Design, Boston, MA: PWS Publishing, 1996.
- [106]. L.A. Zadeh, From computing with numbers to computing with words—from manipulation of measurements to manipulation of perceptions, IEEE Trans. Circ. Syst.—I: Fundam. Theory Appl. 4 (1999) 105–119.

- [107]. L.A. Zadeh, Towards a generalized theory of uncertainty (GTU)—an outline, *Information Sciences*, Vol. 172 (2005) 1–40, June 2005.
- [108]. L.A. Zadeh, J. Kacprzyk (Eds.), *Computing With Words in Information/Intelligent Systems 1 & 2*, Physica-Verlag, New York, 1999.
- [109]. Simon Haykin. *Adaptive Filter Theory*. Prentice Hall, 2002, ISBN 0-13-048434-2
- [110]. Mackey M.C., Glass L, *Oscillation and Chaos in Physiological Control Systems*, *Science* Vol 197, pp.287-289, 1977.
- [111]. Ljung, L., *System Identification: Theory for the User*, Prentice-Hall, 1987, pp.278-280.
- [112]. Akaike, H. (1974). A new look at the statistical model identification. *IEEE Transactions on automatic control*, AC19, 716-723.
- [113]. Hastie, T. Tibshirani, R, and Friedman, J. (2001) *The Elements of Statistical Learning*, Springer, pp. 214-216.
- [114]. Theodoridis, S. and Koutroumbas, K. (1999) *Pattern Recognition*, Academic Press, pp. 341-342.

APÉNDICE

A.1 Algoritmos de aprendizaje BP con heurísticas elementales de búsqueda.

1. TRNGD: Gradient descent backpropagation.

```
input  $\theta^{(0)}, \eta, tol$   
for  $k=0, 1, \dots$  repeat  
     $E(\theta^{(k)}) = \frac{1}{2} \cdot e^T(\theta^{(k)}) \cdot e(\theta^{(k)})$   
     $g(\theta^{(k)}) = J^T(\theta^{(k)}) \cdot e(\theta^{(k)})$   
     $\Delta\theta^{(k)} = -\eta \cdot g(\theta^{(k)})$   
     $\theta^{(k+1)} = \theta^{(k)} + \Delta\theta^{(k)}$   
end until  $E(\theta^{(k+1)}) < tol$ 
```

2. TRNGDA: Gradient descent with adaptive learning backpropagation.

```
input  $\theta^{(0)}, \eta, \lambda = 1.04, \eta_{dec} = 0.7, \eta_{inc} = 1.05, tol$   
 $E(\theta^{(0)}) = \frac{1}{2} \cdot e^T(\theta^{(0)}) \cdot e(\theta^{(0)})$   
 $g(\theta^{(0)}) = J^T(\theta^{(0)}) \cdot e(\theta^{(0)})$   
for  $k=0, 1, \dots$  repeat  
     $\Delta\theta^{(k+1)} = -\eta \cdot g(\theta^{(k)})$   
     $\theta^{(k+1)} = \theta^{(k)} + \Delta\theta^{(k+1)}$   
     $E(\theta^{(k+1)}) = \frac{1}{2} \cdot e^T(\theta^{(k+1)}) \cdot e(\theta^{(k+1)})$   
    if  $E(\theta^{(k+1)}) / E(\theta^{(k)}) > \lambda$   
         $\eta = \eta \cdot \eta_{dec}$   
    else  
        if  $E(\theta^{(k+1)}) < E(\theta^{(k)})$   
             $\eta = \eta \cdot \eta_{inc}$   
        end  
         $\theta^{(k)} = \theta^{(k+1)}$   
         $E(\theta^{(k)}) = E(\theta^{(k+1)})$ 
```

$g(\theta^{(k+1)}) = J^T(\theta^{(k+1)}) \cdot e(\theta^{(k+1)})$

end

end until $E(\theta^{(k+1)}) < tol$

3. TRNGDM: Gradient descent with momentum backpropagation.

input $\theta^{(0)}, \eta, \beta, tol$

$g(\theta^{(0)}) = J^T(\theta^{(0)}) \cdot e(\theta^{(0)})$

$\Delta\theta^{(0)} = -\eta \cdot g(\theta^{(0)})$

for $k=0, 1, \dots$ **repeat**

$E(\theta^{(k)}) = \frac{1}{2} \cdot e^T(\theta^{(k)}) \cdot e(\theta^{(k)})$

$g(\theta^{(k)}) = J^T(\theta^{(k)}) \cdot e(\theta^{(k)})$

$\Delta\theta^{(k+1)} = \beta \cdot \Delta\theta^{(k)} - (1 - \beta) \cdot \eta \cdot g(\theta^{(k)})$

$\theta^{(k+1)} = \theta^{(k)} + \Delta\theta^{(k+1)}$

end until $E(\theta^{(k+1)}) < tol$

4. TRNGDX: Gradient descent w/momentum and adaptive learning backpropagation.

input $\theta^{(0)}, \eta, \beta, \lambda = 1.04, \eta_{dec} = 0.7, \eta_{inc} = 1.05, tol$

$E(\theta^{(0)}) = \frac{1}{2} \cdot e^T(\theta^{(0)}) \cdot e(\theta^{(0)})$

$g(\theta^{(0)}) = J^T(\theta^{(0)}) \cdot e(\theta^{(0)})$

$\Delta\theta^{(0)} = -\eta \cdot g(\theta^{(0)})$

for $k=0, 1, \dots$ **repeat**

$\Delta\theta^{(k+1)} = \beta \cdot \Delta\theta^{(k)} - (1 - \beta) \cdot \eta \cdot g(\theta^{(k)})$

$\theta^{(k+1)} = \theta^{(k)} + \Delta\theta^{(k+1)}$

$E(\theta^{(k+1)}) = \frac{1}{2} \cdot e^T(\theta^{(k+1)}) \cdot e(\theta^{(k+1)})$

if $E(\theta^{(k+1)}) / E(\theta^{(k)}) > \lambda$

$\eta = \eta \cdot \eta_{dec}$

$\Delta\theta^{(k)} = -\eta \cdot g(\theta^{(k)})$

else

if $E(\theta^{(k+1)}) < E(\theta^{(k)})$

$\eta = \eta \cdot \eta_{inc}$

```

end
 $\theta^{(k)} = \theta^{(k+1)}$ 
 $E(\theta^{(k)}) = E(\theta^{(k+1)})$ 
 $g(\theta^{(k+1)}) = J^T(\theta^{(k+1)}) \cdot e(\theta^{(k+1)})$ 
end
end until  $E(\theta^{(k+1)}) < tol$ 

```

5. TRNOSS: One step secant backpropagation.

```

input  $\theta^{(0)}, \alpha = 0.001, \beta = 0.1, \delta = 0.01, \gamma = 0.1, tol$ 
 $a = 0$ 
for  $k=0,1,\dots$  repeat
  if  $k=0$ 
     $E(\theta^{(k)}) = \frac{1}{2} \cdot e^T(\theta^{(k)}) \cdot e(\theta^{(k)})$ 
     $^{old}E = E(\theta^{(k)})$ 
     $\Delta E = E(\theta^{(k)})$ 
     $g(\theta^{(k)}) = J^T(\theta^{(k)}) \cdot e(\theta^{(k)})$ 
     $^{old}g = g(\theta^{(k)})$ 
     $v = \|g(\theta^{(k)})\|$ 
     $\Delta\theta^{(k)} = -g(\theta^{(k)})$ 
     $\Delta E^{(k)} = g(\theta^{(k)})^T \cdot \Delta\theta^{(k)}$ 
  else
     $\Delta g(\theta^{(k)}) = g(\theta^{(k)}) - ^{old}g$ 
     $^{old}g = g(\theta^{(k)})$ 
     $v = \|g(\theta^{(k)})\|$ 
     $\Delta E(\theta^{(k)}) = E(\theta^{(k)}) - ^{old}E$ 
     $^{old}E = E(\theta^{(k)})$ 
     $\kappa = \chi \cdot \Delta g(\theta^{(k)})$ 
     $\rho = \chi \cdot g(\theta^{(k)})$ 
    if  $\left\lfloor \frac{k}{\tau} \right\rfloor = 0$ 
       $\Delta\theta^{(k)} = -g(\theta^{(k)})$ 
    end if
  end if

```



```

else
     $\varphi = \frac{\rho}{\kappa}$ 
     $\lambda = -\left(1 + \frac{\Delta g(\theta^{(k)})^T \cdot \Delta g(\theta^{(k)})}{\kappa}\right) \varphi + \frac{\Delta g(\theta^{(k)})^T \cdot g(\theta^{(k)})}{\kappa}$ 
     $\Delta \theta^{(k)} = -g(\theta^{(k)}) + \lambda \cdot \chi + \varphi \cdot \Delta g(\theta^{(k)})$ 
end
 $\Delta E^{(k)} = g(\theta^{(k)})^T \cdot \Delta \theta^{(k)}$ 
if  $\Delta E^{(k)} > 0$ 
     $\Delta \theta^{(k)} = -g(\theta^{(k)})$ 
     $\Delta E^{(k)} = g(\theta^{(k)})^T \cdot \Delta \theta^{(k)}$ 
end
 $\sigma = \|\Delta(\theta^{(k)})\|$ 
end
 $a = \min_{\eta_k} E(\theta^{(k)} + \eta_k \cdot g(\theta^{(k)}))$ 
 $\chi = a \cdot \Delta \theta^{(k)}$ 
 $\theta^{(k+1)} = \theta^{(k)} + \chi$ 
end until  $E(\theta^{(k+1)}) < tol$ 

```

6. TRAINLM: Levenberg-Marquardt backpropagation.

```
function [theta, info, perf] = trainlm(namefunobj, it2fnn, opts, varargin)
```

```
% perf = [E, g, mu]
```

```
% opts = [tau, epsilon_1, epsilon_2, epsilon_3, k_max]
```

```
begin
```

```
tau = 10^-3;  epsilon_1 = epsilon_2 = epsilon_3 = 10^-15;  k_max = 100
```

```
k = 0;  v = 2;  theta = it2fnngetx(it2fnn)
```

```
H = J(theta)^T J(theta);  g = J(theta)^T e(theta)
```

```
stop = ||g||_inf <= epsilon_1;  mu = tau * max{h_{i,i}}
```

```
while (not stop) and (k < k_max)
```

```
    k = k + 1;  Solve (H + mu I) Delta theta = -g
```

```
    if ||Delta theta|| <= epsilon_2 (||theta|| + epsilon_2)
```

```

        stop = true
    else
         $\theta_{new} = \theta + \Delta\theta$ 
        
$$\rho = \frac{E(\theta) - E(\theta_{new})}{\frac{1}{2}\Delta\theta^T(\mu\Delta\theta - g)}$$

        if  $\rho > 0$ 
             $\theta = \theta_{new}$ 
             $H = J(\theta)^T J(\theta); \quad g = J(\theta)^T e(\theta)$ 
            stop = ( $\|g\|_\infty \leq \varepsilon_1$ ) or ( $\|e\|^2 \leq \varepsilon_3$ )
             $\mu = \mu * \max\{\frac{1}{3}, 1 - (2\rho - 1)^3\}; \quad v = 2$ 
        else
             $\mu = \mu * v; \quad v = 2 * v$ 
        end
    end
end
end
end

```

7. TRAINHYBRID: Levenberg-Marquardt backpropagation combined with a quasi-Newton method and BFGS updating of the approximate Hessian.

```

function [theta, info, perf] = trainhybrid(namefunobj, it2fnn, opts, varargin)
begin
     $\tau = 10^{-3}; \quad \varepsilon_1 = \varepsilon_2 = \varepsilon_3 = 10^{-15}; \quad \Delta = 0; \quad k_{\max} = 100$ 
     $k = 0; \quad \theta = \text{it2fnngetx}(\text{it2fnn}); \quad B = I$ 
     $H = J(\theta)^T J(\theta); \quad g = J(\theta)^T e(\theta)$ 
    stop =  $\|g\|_\infty \leq \varepsilon_1; \quad \mu = \tau * \max\{h_{i,i}\}; \quad \text{method} = \text{trainlm}$ 
    while (not stop) and ( $k < k_{\max}$ )
         $k = k + 1$ 
        case method of
            trainlm: [theta, stop, better, method, ...] = trainlmstep(namefunobj, it2fnn, Delta, ...)
            trainqn: [theta, stop, better, method, ...] = trainqnstep(namefunobj, it2fnn, Delta, B, ...)
        end
         $h = \theta_{new} - \theta;$ 
         $y = J^T(\theta_{new})J(\theta_{new})h + [J(\theta_{new}) - J(\theta)]^T e(\theta_{new})$ 
        if  $h^T y > 0$ 
             $v = Bh$ 

```

```

        B = B + \left(\frac{y}{h^T y}\right)y^T - \left(\frac{v}{h^T v}\right)v^T
    end
    if better
        \theta = \theta_{new}
    end
end

function [\theta, stop, better, method, ...] = trainlmstep(namefunobj, it2fnn, \Delta, ...)
begin
    k = 0; \theta = it2fnngetx(it2fnn); method = trainlm
    H = J(\theta)^T J(\theta); g = J(\theta)^T e(\theta)
    Solve (H + \mu I) \Delta \theta = -g
    if \|\Delta \theta\| \leq \varepsilon_2 (\|\theta\| + \varepsilon_2)
        stop = true
    else
        \theta_{new} = \theta + \Delta \theta
        \rho = \frac{E(\theta) - E(\theta_{new})}{\frac{1}{2} \Delta \theta^T (\mu \Delta \theta - g(\theta_{new}))}
        if \rho > 0
            better = true; stop = (\|g(\theta_{new})\|_{\infty} \leq \varepsilon_1)
            if \|g(\theta_{new})\|_{\infty} < 0.2 E(\theta_{new})
                k = k + 1
                if k == 3
                    method = trainqn
                else
                    k = 0
                end
            else
                k = 0; better = false
            end
        end
        if method == trainlm
            \Delta = \max \left\{ 1.5 \varepsilon_2 (\|\theta\| + \varepsilon_2), \frac{\|\Delta \theta\|}{5} \right\}
        end
    end
end

function [\theta, stop, better, method, ...] = trainqnstep(namefunobj, it2fnn, \Delta, B, ...)
begin
    \theta = it2fnngetx(it2fnn); method = trainqn; better = false
    g = J(\theta)^T e(\theta)
    Solve B \Delta \theta = -g
    if \|\Delta \theta\| \leq \varepsilon_2 (\|\theta\| + \varepsilon_2)
        stop = true
    else

```

```

    if  $\|\Delta\theta\| > \Delta$ 
        
$$\Delta\theta = \left( \frac{\Delta}{\|\Delta\theta\|} \right) \Delta\theta$$

    end
     $\theta_{new} = \theta + \Delta\theta$ 
    if  $(\|g(\theta_{new})\|_{\infty} \leq \varepsilon_1)$ 
        stop = true
    else
        better =  $(E(\theta_{new}) < E(\theta))$  or  $(E(\theta_{new}) \leq (1 + \delta)E(\theta))$  and
             $(\|g(\theta_{new})\|_{\infty} < \|g(\theta)\|_{\infty})$ 
        if  $\|g(\theta_{new})\|_{\infty} \geq \|g(\theta)\|_{\infty}$ 
            method = trainlm
        end
    end
end
end
end

```

A.2 Algoritmo de aprendizaje PSO

1. TRAINPSOFNN. Pseudo code of generic particle swarm optimizer IT2FNN

```

[optOUT,tr,te]=PSOFNN(funcname,D,mv,VarRange,minmax,PSOparams,plotfcn,PSOseedValue)
// Initialize the particle positions and their velocities
for l = 1 to number of particles n do
    for J = 1 to number of dimensions m do
        X(l,J) = lower limit + (upper limit - lower limit) * uniform random number
        V(l,J) = 0
    enddo
enddo
// Initialize the global and local fitness to the worst possible
fitness_gbest = inf;
for l = 1 to number of particles n do
    fitness_lbest(l) = inf
enddo
// Loop until convergence, in this example a finite number of iterations chosen
for k = 1 to number of iterations t do
    // evaluate the fitness of each particle
    fitness_X = evaluate_fitness(X)
    // Update the local bests and their fitness
    for l = 1 to number of particles n do
        if (fitness_X(l) < fitness_lbest(l))
            fitness_lbest(l) = fitness_X(l)
            for J = 1 to number of dimensions m do
                X_lbest(l,J) = X(l,J)
            enddo
        endif
    enddo
    // Update the global best and its fitness
    [min_fitness, min_fitness_index] = min(fitness_X)
    if (min_fitness < fitness_gbest)
        fitness_gbest = min_fitness
        for J = 1 to number of dimensions m do
            X_gbest(J) = X(min_fitness_index, J)
        enddo
    endif
    // Update the particle velocity and position

```

```

for l = 1 to number of particles n do
  for J = 1 to number of dimensions m do
    R1 = uniform random number
    R2 = uniform random number
    V(l,J) = w*V(l,J) + C1*R1*(X_lbest(l,J) - X(l,J)) + C2*R2*(X_gbest(J) - X(l,J))
    X(l,J) = X(l,J) + V(l,J)
  enddo
enddo
enddo

```