

Centro Nacional de Investigación y Desarrollo Tecnológico

Subdirección Académica

Departamento de Ingeniería Electrónica

TESIS DE MAESTRÍA EN CIENCIAS

**Estimador M de Huber para la Identificación de un Modelo en Red
Neuronal**

presentada por
Ing. Carlos Jesús Zúñiga Aguilar

como requisito para la obtención del grado de
Maestro en Ciencias en Ingeniería Electrónica

Director de tesis
Dr. Víctor Manuel Alvarado Martínez

Codirectora de tesis
Dra. María Guadalupe López López

Cuernavaca, Morelos, México. Junio de 2016.

SEP

SECRETARÍA DE
EDUCACIÓN PÚBLICA



TECNOLÓGICO NACIONAL DE MÉXICO
Centro Nacional de Investigación y Desarrollo Tecnológico

Cuernavaca, Mor., 13 de junio de 2016.

OFICIO No. DIE/126/2016

Asunto: Aceptación de documento de tesis

DR. GERARDO VICENTE GUERRERO RAMÍREZ
SUBDIRECTOR ACADÉMICO
PRESENTE

Por este conducto, los integrantes de Comité Tutorial del **C. Carlos Jesús Zúñiga Aguilar** con número de control **M14CE038** de la Maestría en Ciencias en Ingeniería Electrónica, le informamos que hemos revisado el trabajo de tesis profesional titulado **“Estimador M de Huber para la Identificación de un Modelo en Red Neuronal”** y hemos encontrado que se han realizado todas las correcciones y observaciones que se le indicaron, por lo que hemos acordado aceptar el documento de tesis y le solicitamos la autorización de impresión definitiva.

DIRECTOR DE TESIS

Dr. Víctor Manuel Alvarado Martínez
Doctor en Ciencias en Ingeniería Electrónica
Cédula profesional 8031070

CODIRECTOR DE TESIS

Dra. Ma. Guadalupe López López
Doctora en Ciencias en Ingeniería Química
Cédula profesional 7980045

REVISOR 1

Dr. Juan Reyes Reyes
Doctor en Ciencias en la Especialidad de Control Automático
Cédula profesional 4214833

REVISOR 2

Dr. Carlos Daniel García Beltrán
Doctor en Ciencias en Ingeniería Electrónica
Cédula profesional 8699605

C.p. Lic. Guadalupe Garrido Rivera.- Jefa del Departamento de Servicios Escolares.
Estudiante
Expediente

CMAZ/lrr

cenidet[®]
Centro Nacional de Investigación
y Desarrollo Tecnológico

Interior Internado Palmira S/N, Col. Palmira. C.P. 62490 Cuernavaca, Mor.
Tels. (01)777 362-77-70 Ext. 4106, e-mail: direccion@cenidet.edu.mx
www.cenidet.edu.mx





Cuernavaca, Mor., 14 de junio de 2016
OFICIO No. SAC/207/2016

Asunto: Autorización de impresión de tesis

ING. CARLOS JESÚS ZÚÑIGA AGUILAR
CANDIDATO AL GRADO DE MAESTRO EN CIENCIAS
EN INGENIERÍA ELECTRÓNICA
P R E S E N T E

Por este conducto, tengo el agrado de comunicarle que el Comité Tutorial asignado a su trabajo de tesis titulado **“Estimador M de Huber para la Identificación de un Modelo en Red Neuronal”**, ha informado a esta Subdirección Académica, que están de acuerdo con el trabajo presentado. Por lo anterior, se le autoriza a que proceda con la impresión definitiva de su trabajo de tesis.

Esperando que el logro del mismo sea acorde con sus aspiraciones profesionales, reciba un cordial saludo.

ATENTAMENTE

“CONOCIMIENTO Y TECNOLOGIA AL SERVICIO DE MEXICO”

DR. GERARDO VICENTE GUERRERO RAMÍREZ
SUBDIRECTOR ACADÉMICO



SEP TecNM
CENTRO NACIONAL
DE INVESTIGACIÓN
Y DESARROLLO
TECNOLÓGICO
SUBDIRECCIÓN
ACADÉMICA

C.p. Lic. Guadalupe Garrido Rivera.- Jefa del Departamento de Servicios Escolares.
Expediente

GVGR/mcr

*“Mira que te mando que te esfuerces y seas valiente; no temas ni desmayes, porque
Jehová tu Dios estará contigo en dondequiera que vayas.”
Josué 1:9*

Dedicatoria

Para mis padres Elena y Cirino, que me regalaron la oportunidad de vivir, que han estado conmigo como fieles compañeros a lo largo de esta difícil vida y que sin importar qué tan sucio estuviera siempre vieron en mí “un hermoso regalo”. Ustedes son el pilar más grande en mi vida, gracias por tratar de estar ahí siempre, por escucharme, por calmarme, por sufrir conmigo. Gracias por sostenerme, por aguantarme, por tratar de hacerme sentir en familia en cualquier lugar en el que esté.

Para mis hermanos María, Jesús y América. Hermanos míos, ustedes para mí han sido el motor más potente que haya podido presenciar. Gracias por esa motivación inigualable que se han encargado inyectar en mí, por esas ganas de vivir la vida a toda costa, por cuidarme sin importar el qué, el cómo y el por qué y principalmente, gracias por otorgarme la dicha de ser el tío de tres maravillosos niños. Para mis sobrinos, esos niños que no inspiran nada más que amor e inspiración para seguir adelante y enfrentarme contra lo que sea.

Para el más gran descubrimiento que he hecho en mi corta carrera, el descubrimiento más importante de mi vida. Porque solo en las misteriosas sendas del amor es donde no se puede encontrar ninguna lógica, ninguna razón.

Estoy aquí gracias a ustedes, ustedes son mi razón de ser. En realidad son todas mis razones. ¡Gracias!

Agradecimientos

A lo largo de mis estudios de maestría recibí el apoyo, comprensión, motivación de familiares, amigos, compañeros y profesores. A cada persona que estuvo ahí para mí les brindo mi más sincero y noble agradecimiento.

Le agradezco mis asesores el Dr. Víctor Manuel Alvarado Martínez y a la Dra. María Guadalupe López López por haberme guiado durante todo este tiempo, por haberme introducido en la senda de la investigación científica y por toda la confianza inyectada en mí. A mis revisores, el Dr. Alejandro Rodríguez Palacios y al Dr. Carlos Daniel García Beltrán por sus comentarios asertivos y sus observaciones con la finalidad de perfeccionar este trabajo. Al CENIDET y al Dr. Víctor Hugo Olivares Peregrino por haberme brindado la oportunidad de realizar mis estudios de maestría. Al Dr. Hector Manuel Romero Ugalde le agradezco infinitamente por haberme acogido en su hogar durante mi estancia, por haberme instruido en el desarrollo de mi tesis, por su confianza, ímpetu, paciencia y amistad. Muchas gracias a todos mis profesores quienes me motivaron a seguir adelante como el Dr. Juan Reyes, el Dr. Enrique Quintero, el Dr. Astorga y el Dr. Vicente Guerrero.

Agradezco enormemente al CONCAYT por haberme apoyado económicamente durante dos años con el fin de que pudiera realizar y culminar mis estudios de maestría y por el apoyo que recibí para realizar una estancia en el extranjero.

Al personal del CENIDET, compañeros y amigos: Lore, Sari, Gongora, Garrido, Anita, Alfredo, Adriana, Heda, Alberto, Joachin, Julio, al “tigre”, Mota, Schadt, Jorge, Mael, Aldo, Turi, Juan Pablo, LeoDan, Audomaro, Chuy, Diego, Moi, Jashiel, Eider, Paty, Ivonne, Adrián, Oscar y Emmanuel. Mis queridos amigos, Susana, Didier y Montse que siempre estuvieron ahí para escucharme, apoyarme y animarme. ¡Muchas gracias!

Resumen

En esta tesis se presenta una metodología para realiza la identificación de sistemas con valores atípicos con el uso de redes neuronales artificiales y el estimador M de Huber. Se obtuvieron conjuntos de datos entrada-salida de sistemas SISO con y sin valores atípicos; con este conjunto de datos se entrenó a una red neuronal fuera de línea con el fin de optimizar los pesos sinápticos que la conforman y de esta manera lograr que se ajuste la salida de la red a la salida del sistema real.

Se estudiaron en gran medida las redes neuronales artificiales, el estimador M de Huber y algunos de los algoritmos de optimización más utilizados y eficientes para encontrar la manera de que estos aspectos funcionaran en conjunto.

Se compararon los resultados del estimador M de Huber contra el estimador L_2 para comprobar que la función de Huber es capaz de realizar una estimación igual o más eficiente bajo las mismas condiciones, es decir, la pruebas se realizaron con la misma estructura de red neuronal, el mismo número de neuronas, el mismo número de regresores y el mismo algoritmo de optimización. También se compararon los resultados obtenidos de la identificación con la función de Huber y la red neuronal seleccionada contra la red neuronal de MATLAB para la identificación de sistemas no lineales, NLARX.

Finalmente, con los resultados obtenido de la identificación de sistemas con y sin outliers mediante el uso del estimador M de Huber, el estimador L_2 y el NLARX se realizaron las conclusiones para verificar si el uso de la función de Huber es eficiente para enfrentar datos con valores atípicos.

Abstract

In this thesis it is presented a methodology to perform systems identification with outliers based in the use of artificial neural networks and M Huber estimator. An input-output data set of SISO systems with and without outliers was obtained; an offline neural network was trained with this data set in order to optimize the synaptic weights which conform it in such a way to achieve the adjustment of the network output to the real system output.

Artificial neural networks, the M Huber estimator and several of the most used and efficient optimization algorithms were studied in great extent in order to find the way that this aspects functioned together.

The results of the M Huber estimator and the L_2 estimator were compared in order to prove that the Huber function is able to perform an estimation that is equal or more efficient under the same conditions, namely, tests were performed with the same neural network structure, the same number of neurons, the same number of regressors and the same optimization algorithm. Also, the results obtained by the identification using Huber function and the selected neural network against the MATLAB neural network for nonlinear systems identification (NLARX) were compared.

Finally, the results obtained by the identification of systems with and without outlier through the use of M Huber estimator, the L_2 estimator and the NLARX, conclusions were made to verify whether the use of Huber function is efficient to deal with data with outliers.

Índice general

Dedicatoria	IV
Agradecimientos	V
Resumen	VI
Abstract	VII
Índice de figuras	XI
Índice de tablas	XIV
Lista de símbolos	XV
Lista de símbolos	XVI
1. INTRODUCCIÓN	1
1.1. Identificación de Sistemas	2
1.1.1. Sistema	3
1.1.2. Modelo Matemático	3
1.1.3. Proceso de Identificación	4
1.2. Estado del Arte	7
1.2.1. Identificación de Sistemas	7
1.2.2. Identificación de Sistemas con Redes Neuronales	7
1.2.3. Estimador M de Huber	8
1.3. Planteamiento del problema	10
1.4. Objetivos	11

1.4.1. Objetivo general	11
1.4.2. Objetivos específicos	11
1.5. Alcance de la Tesis	11
1.6. Estrategia para desarrollar las metodologías propuestas	12
1.7. Organización de la Tesis	13
2. REDES NEURONALES ARTIFICIALES	15
2.1. Neuronas biológicas	16
2.2. Neuronas Artificiales	17
2.2.1. Aplicaciones de las redes neuronales	17
2.2.2. Arquitecturas de redes neuronales	18
2.2.3. Perceptrón	20
2.2.4. Adaline	23
2.2.5. RNA unicapa	24
2.2.6. Perceptrón Multicapa	26
2.3. Identificación de sistemas utilizando redes neuronales	28
2.3.1. Selección de estructura neuronal	28
2.3.2. Entrenamiento de una red neuronal	29
2.3.3. Algoritmos de optimización	31
2.4. Red neuronal recurrente 2nn - 2 - 1: Precisión	36
2.5. Red neuronal recurrente 2-1: Simplicidad	39
2.6. Red neuronal recurrente 2-2-1: Costo computacional	40
2.7. Proceso de Reducción	41
3. ESTIMADOR M DE HUBER	46
3.1. “ <i>Outliers</i> ” o valores atípicos	47
3.1.1. Tipos de “outliers”	47
3.1.2. Causas	49
3.1.3. Detección	50
3.2. Estimadores	51
3.3. Estimador M de Huber ($L_1 - L_2$)	54

4. Metodología para la identificación de sistemas mediante redes neuronales y el estimador M de Huber	60
4.1. Metodología para la identificación de sistemas con redes neuronales y el estimador M de Huber	61
4.2. Índices de desempeño	62
4.2.1. Media del error cuadrático	62
4.2.2. Media del error	63
4.2.3. Desviación estándar	63
4.2.4. FIT	63
4.2.5. Integral del error absoluto	64
4.2.6. Integral del error al cuadrado	64
4.3. Cálculo del orden de una RNA	64
4.4. Cálculo de parámetros de la función de Huber	67
5. Pruebas y resultados	69
5.1. Identificación de un brazo robot	69
5.1.1. El sistema	70
5.1.2. Experimentos	70
5.1.3. Selección de κ	72
5.1.4. Identificación del sistema	75
5.1.5. Comentarios	78
5.2. Wiener - Hammerstein Benchmark	79
5.2.1. El sistema	79
5.2.2. Validación	79
5.2.3. Experimentos	81
5.2.4. Selección de κ	85
5.2.5. Identificación del sistema	87
5.2.6. Comentarios	91
5.3. Identificación de ducto acústico	92
5.3.1. El sistema	92
5.3.2. Experimentos	93
5.3.3. Identificación del sistema	94
5.3.4. Comentarios	97

6. Conclusiones y trabajos futuros	99
6.1. Conclusiones	99
6.2. Trabajos futuros	101
A. Funciones de activación	103
B. Modelo NLARX	104
C. Pesos sinápticos para estimación y optimización de pesos sinápticos	106
C.1. Brazo Robot	106
C.1.1. Entrenamiento de la red neuronal	107
C.1.2. Reducción del modelo y valores de pesos sinápticos	109
C.2. Sistema Wiener-Hammerstein	111
C.2.1. Entrenamiento de la red neuronal	111
C.3. Ducto acústico	116
C.3.1. Entrenamiento de la red neuronal	116
D. Estancia	120
Bibliografía	132

Índice de figuras

1.1. Sistema dinámico con entrada $u(t)$, perturbación $e(t)$ y salida $y(t)$. . .	3
1.2. Identificación de sistemas.	4
1.3. Proceso de identificación.	6
2.1. Neurona biológica.	16
2.2. Redes neuronales estáticas.	19
2.3. Redes Neuronales recurrentes.	20
2.4. Red Neuronal celular.	20
2.5. Neurona artificial.	21
2.6. Neurona Artificial Adaline.	23
2.7. Red Neuronal artificial de una capa.	25
2.8. Perceptrón de dos capas.	27
2.9. Proceso de entrenamiento.	31
2.10. Comportamiento del coeficiente de aprendizaje.	33
2.11. Comportamiento del coeficiente de aprendizaje.	35
2.12. Red neuronal recurrente 2nn-2-1.	38
2.13. Red neuronal recurrente 2-1. Donde f puede ser definida como φ_1 o φ_2	39
2.14. Red neuronal recurrente 2-2-1.	41
3.1. Ejemplo de valores atípicos puntuales. Puntos $O1$ y $O2$ difieren significativamente de las regiones $G1$ y $G2$	48
3.2. Ejemplo de outlier contextual. Es un conjunto de datos mostrados en serie de tiempo donde los outliers tienen una amplitud distante a los demás valores de amplitud.	49
3.3. Ejemplo de outlier colectivo (puntos en verde).	50

3.4. Gráficas de estimadores M.	58
3.5. Gráficas de estimadores M.	59
4.1. Comportamiento de la media del error al cuadrado cambiando los valores de na , nb y considerando que $nk = 4$	65
4.2. Comportamiento del FIT temporal cambiando los valores de na , nb y considerando que $nk = 4$	66
4.3. Función de sintonización con dos intervalos. El intervalo clásico $\gamma \in [1, 1.5]$ y un intervalo extendido $\gamma \in (0, 0.2]$	68
4.4. Función de sintonización con dos intervalos. El intervalo clásico $\gamma \in [1, 1.5]$ y un intervalo extendido $\gamma \in (0, 0.2]$	68
5.1. Brazo Robot	70
5.2. Señal de salida original y señal de salida con valores atípicos añadidos.	71
5.3. Análisis de índices para la selección de ordenes	73
5.4. Análisis de índices para la selección de κ	74
5.5. Respuesta en la frecuencia y temporal del modelo neuronal RHNN	75
5.6. Respuesta en la frecuencia y temporal del modelo neuronal RUNN	76
5.7. Respuesta en la frecuencia y temporal del modelo neuronal NLARX	76
5.8. Respuesta en la frecuencia y temporal del modelo neuronal RHNN	77
5.9. Respuesta en la frecuencia y temporal del modelo neuronal RUNN	77
5.10. Respuesta en la frecuencia y temporal del modelo neuronal NLARX	78
5.11. Sistema Wiener-Hammerstein conectado en cascada por un bloque dinámico, uno estático y un bloque dinámico lineal $G_1(s)f[\cdot]G_2(s)$	79
5.12. Circuito utilizado para construir el sistema estático no lineal.	80
5.13. Señal original sin outliers y con outliers añadidos.	80
5.14. (a) Media del error, (b) desviación estándar del error, (c) RMSe del error y (d) FIT temporal	83
5.15. (a) FIT frecuencial, (b) Error a bajas frecuencias, (c) error a altas frecuencias y (d) error medio cuadrático	84
5.16. Suma del error cuadrático medio para la selección de κ	85
5.17. Desviación estándar del error para la selección de κ	86
5.18. Media del error para la selección de κ	86

5.19. Respuesta temporal del sistema.	88
5.20. Respuesta en la frecuencia de los modelos neuronales.	89
5.21. Respuesta frecuencial de los modelos neuronales en presencia de outliers.	90
5.22. Respuesta temporal de los modelos neuronales en presencia de outliers.	91
5.23. Respuesta temporal de los modelos neuronales en presencia de outliers.	92
5.24. Señal de salida original y señal de salida con valores atípicos añadidos.	93
5.25. Respuesta temporal de los sistemas.	94
5.26. Respuesta en la frecuencia de los modelos neuronales.	95
5.27. Respuesta en la frecuencia y temporal del modelo neuronal NLARX.	96
5.28. Respuesta en la frecuencia y temporal del modelo neuronal RUNN.	96
5.29. Respuesta en la frecuencia y temporal del modelo neuronal RHNN.	97
A.1. Funciones de activación más comunes	103
B.1. Estructura de RNA NLARX.	104

Índice de tablas

3.1. Estimadores comúnmente utilizados.	53
5.1. Comparativa de modelos neuronales y otras metodologías.	76
5.2. Comparativa de modelos neuronales.	77
5.3. Comparativa de modelos neuronales y otras metodologías.	90
5.4. Comparativa de modelos neuronales.	91
5.5. Comparativa de modelos neuronales.	94
5.6. Comparativa de modelos neuronales.	95

Lista de símbolos

w_i	Peso sináptico.
Ω	Matriz de pesos sinápticos.
$\bar{\Omega}$	Matriz de pesos sinápticos extendida.
Γ	Matriz de pesos sinápticos.
$\bar{\Gamma}$	Matriz de pesos sinápticos extendida.
n	Número de entradas de la red neuronal.
Z_h	Umbral o bias.
\hat{y}	Salida estimada o salida de la red neuronal.
n_a	Número de regresos en la señal de salida.
n_b	Número de regresos en la señal de entrada.
n_k	Tiempo muerto o regresor natural en la entrada.
$y(k)$	Señal de salida real o medición.
$e(k)$	Error.
k	Instante de tiempo o iteración.
N	Número total de datos para la estimación.
R	Modifica la dirección de búsqueda.
η	Tamaño de paso o coeficiente de aprendizaje.
μ	Tamaño de paso o coeficiente de aprendizaje.
E	Función objetivo.
γ	Factor de Huber.
η_0	Valor inicial de η .
μ_0	Valor inicial de μ .
s	Señal interna de la red neuronal.
φ	Función de Activación.
T	Señal interna de la red neuronal.
r_a	Señal interna de la red neuronal.
r_b	Señal interna de la red neuronal.

$J(\cdot)$	Matriz Jacobiana.
nn	Número de neuronas.
J_u	Matriz de regresión de la entrada.
$J_{\hat{y}}$	Matriz de regresión de la salida.
X	Peso sináptico.
Z_b	Peso sináptico.
Z_a	Peso sináptico.
Z_h	Peso sináptico.
V_{b_i}	Peso sináptico.
V_{a_i}	Peso sináptico.
W_{b_i}	Pesos sinápticos.
W_{a_i}	Pesos sinápticos.
W_B	Pesos sinápticos.
W_A	Pesos sinápticos.
V_B	Pesos sináptico.
V_A	Pesos sináptico.
H	Peso sináptico.
W_{bh_i}	Peso sináptico.
W_{ah_i}	Pesos sinápticos.
V_{bh}	Peso sináptico.
V_{ah}	Peso sináptico.
μ_t	Valor de la media de la señal de error.
s_t	Desviación estándar de la señal de error.
e_{RMSt}	Raíz media cuadrática (RMS) de la señal de error.
e_{RMSe}	Raíz media cuadrática (RMS) de la señal de error.
n_p	Número de parámetros.
$FITF$	Medida de parentesco frecuencial.
E_{x1}	Respuesta en la frecuencia de la señal de salida real.
E_{x2}	Respuesta en la frecuencia de la señal de salida estimada.

Abreviaciones

MLP	Perceptrón multi-capas.
RNA	Red neuronal artificial.
RHNN	Red neuronal recurrente Huberiana.
RUNN	Red neuronal recurrente de Ugalde.
EMR	Estimador M robusto.
NLARX	Red neuronal ARX no lineal.
MSE	Error medio cuadrático.
GA	Algoritmos genéticos.
PSO	Optimización por enjambre de partículas.
SVR	Support Vector Regression
LM	Levenberg-Marquardt.
BP	Propagación hacia atrás.
FFT	Transformada rápida de Fourier.

Capítulo 1

INTRODUCCIÓN

En la actualidad se utilizan las ventajas del control automático para el mejoramiento de la calidad, productividad, eficiencia, seguridad, costo-beneficio, etc., de procesos. Muchas de las técnicas de control se basan en una representación matemática llamado modelo matemático. Este modelo matemático asemeja el comportamiento de un sistema real para realizar análisis, detección de fallas, estimación de variables que no pueden ser medidas, optimización, robustez, predicción, simulación, etc.

El modelo matemático puede obtenerse mediante distintas técnicas o la mezcla de ellas. Una es analizando las leyes físicas que rigen el comportamiento del sistema y la segunda es la identificación de sistemas; el objetivo de la identificación de sistemas es el de obtener modelos matemáticos que asemejen el comportamiento del sistema o proceso mediante el uso datos de entrada o variables manipulables y señales de salida obtenidas directamente de la medición del sistema mediante diversas estructuras.

Entre las estructuras para realizar la identificación están los modelos ARX, ARMAX, OE, BJ, AR, ARMA, etc., la identificación de sistemas en espacio de estados, redes neuronales artificiales, etc. Las redes neuronales artificiales (RNA) se han caracterizado por ser útiles en distintas áreas como la de procesamiento de información, minería de datos, clasificación, regresión, control, identificación de sistemas, entre otros. El funcionamiento de las RNA trata de asemejar el comportamiento del sistema nervioso del ser humano debido a la distribución de información que con estas

es posible de realizar.

Una RNA está compuesta por capas, la capa que procesa las señales de entrada es llamada “capa de entrada”, la capa que arroja una señal o dato estimado es llamada “capa de salida” y la capa que se encuentra entre la capa de entrada y salida es llamada “capa oculta”. Cada una de estas capas está compuesta por neuronas que contienen funciones de activación, umbrales y pesos sinápticos (w). Los pesos sinápticos permiten que la RNA se adapte según las necesidades que el diseñador se encuentre enfrentando, la adaptación de estos últimos dependen de una función objetivo que es minimizada mediante el uso de algoritmos de optimización como lo son Gauss-Newton, gradiente descendente, gradiente estocástico, SVR (support vector regression), ELM (extreme learning machine), PSO (particle swarm optimization), Levenberg-Marquadt, etc. Algunos de los algoritmos de optimización necesitan del cambio de la función objetivo con respecto al parámetro que se desea estimar; comúnmente la función que se desea minimizar es el clásico error medio cuadrático. La desventaja de utilizar al error medio cuadrático (EMC) es que se ve afectado al procesar señales contaminadas por valores atípicos (*outliers*).

Como se ha mencionado con anterioridad, para realizar la identificación de sistemas se necesitan señales de entrada y salida. En la práctica, las señales tienen valores inusuales, muy grandes o muy pequeños; los outliers pueden ser causados por un comportamiento inusual de un sistema, por mediciones mal realizadas, etc. Los outliers son observaciones que se desvían de las otras o datos que parecen inconsistentes con el conjunto de datos. Para hacer frente a este comportamiento en las señales se propone cambiar la función objetivo por un estimador robusto como la función de Huber, la cual dará robustez en la optimización de los pesos sinápticos que conforman a la red neuronal artificial.

1.1. Identificación de Sistemas

En el diseño de controladores basados en modelo ya sea en tiempo discreto o tiempo continuo, se requiere un modelo matemático de la planta que se desea con-

trolar de tal manera que este caracterice su comportamiento dinámico. El modelo que ofrece la identificación de sistemas facilita al diseñador probar su controlador antes de ser implementado físicamente.

1.1.1. Sistema

Un sistema es una porción del universo que consta de elementos que interactúan entre sí. Puede tratarse de un volumen definido en el espacio, puede ser una masa de aire que cambia de volumen y de ubicación, puede tratarse de una roca, de un automóvil, de un circuito eléctrico, de una computadora, de un sistema de aire acondicionado, etc., la lista es bastante larga. Las señales que pueden ser observadas por el diseñador y que son de total interés en la implementación de controladores se denominan señales de *salida*. A las señales que pueden ser manipuladas libremente por el diseñador se le denominan señales de *entrada*. El resto de las señales que interactúan con el sistema pero que no pueden ser observadas, medidas o manipuladas se les conoce como *perturbaciones*. En la figura 1.1 se puede observar un esquema de las variables que se mencionaron con anterioridad.

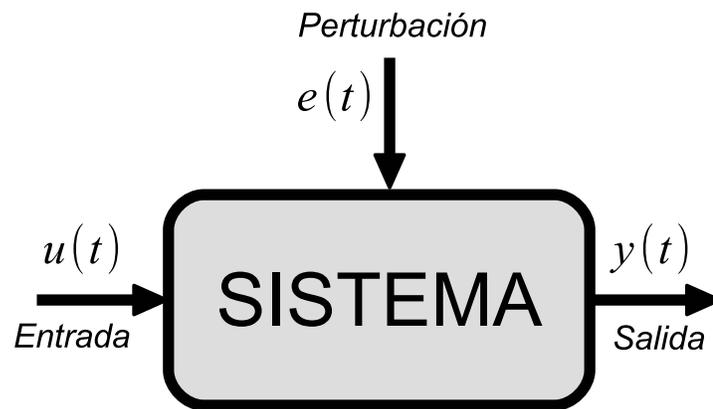


Figura 1.1: Sistema dinámico con entrada $u(t)$, perturbación $e(t)$ y salida $y(t)$

1.1.2. Modelo Matemático

Un modelo matemático puede ser lineal, no lineal, continuo o discreto, variante o invariante en el tiempo o dinámico y capaz representar el comportamiento de

un sistema físico. Se conocen dos métodos para obtener el modelo matemático de un sistema o proceso. El primero consiste en relacionar las leyes físicas que rigen el comportamiento dinámico del sistema; el otro método es la identificación de sistemas la cual es una herramienta que ayuda en la estimación y construcción de modelos a partir de la captura o adquisición de señales de entrada y salida obtenidas de la experimentación del sistema que se requiere modelar (ver figura 1.2).

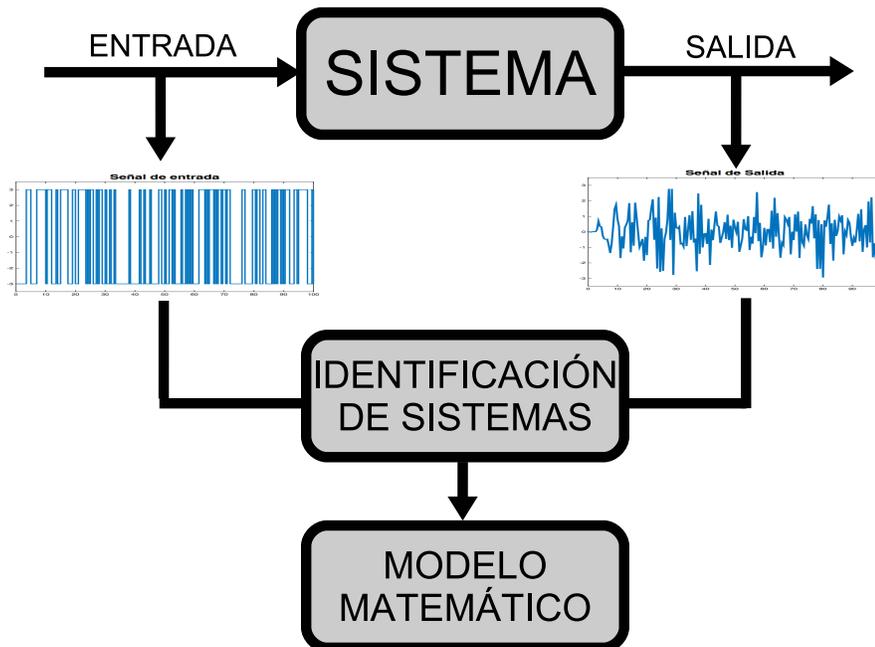


Figura 1.2: Identificación de sistemas.

1.1.3. Proceso de Identificación

Ljung [43] propone una metodología para realizar la identificación de un sistema a partir de la captura de señales de entrada y salida del sistema físico o proceso. Tal metodología se presenta a continuación:

1. *Obtención de datos de entrada y salida:* Para ello se debe excitar o alimentar el sistema mediante la aplicación de una señal de entrada y registrar el comportamiento de la entrada y salida durante un intervalo de tiempo.

2. *Tratamiento previo de los datos registrados:* Los datos registrados están generalmente acompañados de valores numéricos indeseables que pueden ser necesariamente corregidos antes de realizar la identificación del modelo, esto permitirá facilitar y mejorar el proceso de identificación.
3. *Elección de la estructura del modelo:* Se debe seleccionar la estructura de modelo que se utilizará para caracterizar el comportamiento del sistema físico. Este punto tiende a facilitarse en gran medida si se tiene cierto conocimiento sobre las leyes físicas que rigen al proceso.
4. *Obtención de los parámetros del modelo:* Se procede a la estimación de los parámetros que conforman a la estructura del modelo seleccionado en el punto anterior del tal manera que este se ajuste mejor a la respuesta del modelo real.
5. *Validación del modelo:* El último paso consiste en verificar si el modelo propuesto de haber realizado la estimación cumple con los estándares deseados. Si se llega a la conclusión de que el modelo no es válido, se debe revisar las posibles causas:
 - a) El conjunto de datos de entrada y salida no proporciona suficiente información sobre la dinámica del sistema real.
 - b) La estructura escogida no es capaz de proporcionar una buena descripción del modelo.
 - c) El criterio de ajuste de parámetros seleccionado no es el es adecuado.

Dependiendo la causa, deberá repetirse el proceso de identificación desde el punto que corresponde. Por ende, puede decirse que el proceso de identificación es iterativo. Los pasos expuestos con anterioridad se pueden apreciar en la figura 1.3.

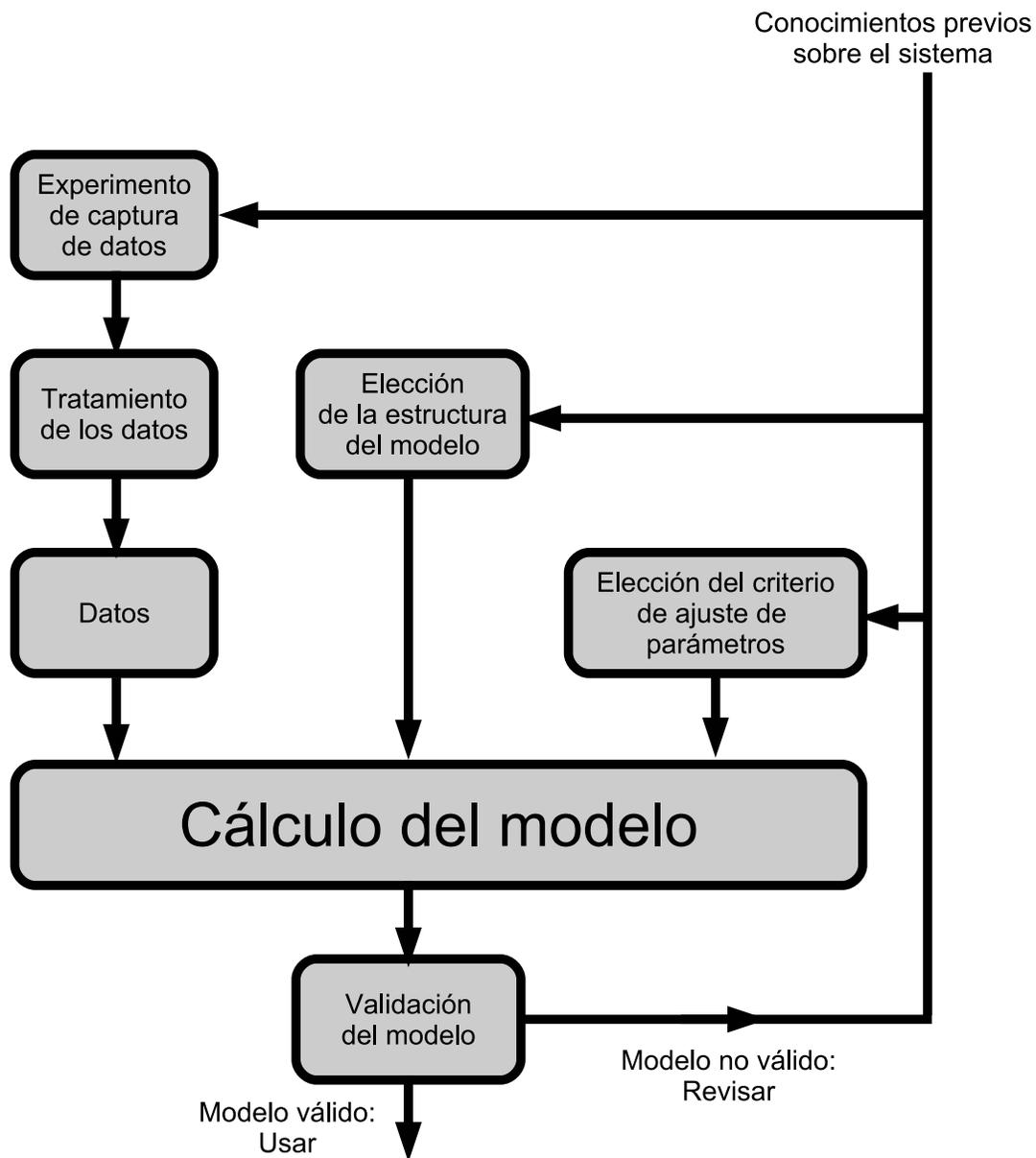


Figura 1.3: Proceso de identificación.

1.2. Estado del Arte

1.2.1. Identificación de Sistemas

El término *Identificación de Sistemas* fue introducido por Zadeh [72] con la estimación de modelos para sistemas dinámicos.

Se conocen dos vías para el desarrollo de la metodología y teoría: Una de ellas se basa en la predicción del error donde se utiliza el análisis y estudios estadísticos. Este enfoque junto con la teoría básica fueron publicados por Åström y Bohlin [9] al igual que Ljung [43]. La otra opción es mediante la realización de modelos en espacio de estado lineales ante la respuesta al impulso Ho y Kalman [32] seguido por Akaike [3, 4]. Lo último, dio lugar a los llamados métodos de subespacio tal como lo realizaron Larimore [41], Van Overschee y De Moor [67].

A principios de los 70's se buscaba diseñar controladores para los modelos resultantes del proceso de identificación. Se enlazó el control con la identificación creando un procedimiento de control adaptable. El pionero en ese área es Kalman [39] seguidos por Åström y Wittenmark [10].

1.2.2. Identificación de Sistemas con Redes Neuronales

A principios de los años 70's Rosenblatt [60] desarrolló un perceptrón dando origen a las redes neuronales artificiales. Al pasar los años las redes neuronales fueron perdiendo interés por la comunidad, no fue hasta los años 90's cuando Cybenko [26] retomó el interés en las redes neuronales en su publicación; demostró que con una combinación lineal finita de funciones unitarias pueden aproximar cualquier función continua con n variables reales. Las pruebas fueron realizadas con una red neuronal de una capa y mínimos cuadrados para estimar los pesos sinápticos adecuados para realizar la aproximación. La técnica que propuso Cybenko ha sido de gran utilidad dentro de la comunidad de redes neuronales. Inclusive, se pueden encontrar trabajos recientes donde se escribe de una manera parecida la teoría que Cybenko propone pero bajo un nombre totalmente diferente, "Extreme Learning Machine" [33–35, 49]. Pasó poco tiempo para que otro trabajo revolucionara el uso de las redes neurona-

les en la comunidad de control. Narendra y Parthasarathy [52] demostraron que las redes neuronales son capaces de identificar y controlar sistemas no lineales.

En Yu et al. [71] realizaron la identificación en línea con el uso de una red neuronal dinámica. Para asegurarse de que el algoritmo de optimización no divergiera, realizaron un análisis con la función de Lyapunov con el fin de asegurar que el algoritmo fuese estable. También consideraron la ecuación algebraica de Riccati para obtener el error de identificación. En Yu [70] se utilizan redes neuronales discretas y recurrentes con un entrenamiento robusto con el fin de garantizar la estabilidad en la estimación utilizando la función de Lyapunov. Para la optimización de los pesos sinápticos de la red neuronal, se utilizó el método del gradiente descendente considerando la reducción de la norma infinito (L_∞).

Las RNA han tenido un gran desarrollo en las últimas décadas debido a su capacidad para aproximar funciones no lineales, procesamiento de información, a su capacidad de “aprender” o ajustarse con la ayuda de la estimación de los pesos sinápticos utilizando algoritmos de optimización [7, 16, 17, 27, 31, 40, 50, 54, 56, 62, 64]. Un trabajo realizado por Romero et al. [56, 57] ofrece una metodología para realizar la identificación de sistemas con el uso de redes neuronales. En el mismo trabajo se plantea un enfoque de reducción algebraica con el fin de realizar la identificación con un número de parámetros considerablemente pequeño pero obteniendo el mismo grado de precisión tal como lo haría un número de neuronas considerablemente alto.

1.2.3. Estimador M de Huber

En estadística, el estimador M es una clase amplia de estimadores, que son obtenidos como la minimización de la suma de funciones de residuos. Técnicas como la conocida, mínimos cuadrados, son ejemplos de estimadores.

En 1964 Huber et al. [37] propuso una generalización para estimación de parámetros de un modelo de máxima verosimilitud con la reducción de una función de la forma

$$\sum_{i=1}^n \rho_{\gamma}(e_i, \theta) \quad (1.1)$$

donde ρ_{γ} es una función simétrica de γ argumentos, La función ρ_{γ} , o su derivado, Ψ , pueden ser elegidos de tal manera que el estimador proporcione propiedades deseables (en términos de sesgo y eficiencia) cuando los datos son distribuidos bajo el supuesto deseado. Las soluciones de la forma

$$\hat{\theta} = \underset{\theta}{\text{mín}} \left(\sum_{i=1}^n \rho_{\gamma}(e_i, \theta) \right) \quad (1.2)$$

son llamadas M-estimadores (“M” de “máxima verosimilitud” [36]); otros tipos de estimadores robustos incluyendo los L-estimadores, R-estimadores y S-estimadores, también son estimadores de máxima verosimilitud.

Como se mencionó con anterioridad, los estimadores son de la forma (1.1), inclusive la técnica de mínimos cuadrados cumple con las características de un estimador M debido a que se busca minimizar la norma euclidiana o norma dos (L_2), los estimadores (L_1), los estimadores ($L_1 - L_2$), los estimadores p (L_p), de Cauchy, German-McClure, Welsch, Tukey, Huber, entre otros. El estimador M de Huber es uno de los casos donde se mezcla el comportamiento de la norma uno (con un límite o penalización en función del factor γ) y la norma dos dentro de una función compuesta por las mismas.

La norma dos ha sido utilizada en innumerables ocasiones para la estimación de parámetros donde los datos no se encuentran contaminados con valores atípicos [13, 27, 42, 54, 56, 65, 69]. El uso de la norma dos con la adición de outliers hace que la estimación se realice de manera ineficiente debido a que la reducción de la norma tiende a diverger en la etapa de optimización. Se ha comprobado que la estimación de los parámetros que conforma a un sistema estimado o propuesto también puede ser realizada mediante el uso de la norma uno; En Alvarado y Carmona [5, 6] se demostró que la identificación de un sistema acústico sin presencia de valores atípicos con una estructura *Output Error* puede realizarse minimizando la norma uno.

Los outliers o valores atípicos son observaciones, mediciones o muestras numéricamente distantes del resto de los datos. Cuando se trata de identificar sistemas los

cuales se encuentran contaminados por outliers o que estos surgen por naturaleza del sistema se pueden encontrar complicaciones en la optimización de parámetros debido a que el residuo (e_i , ver 1.1 o 1.2) tiende a tener un valor numéricamente grande. Con el fin de lidiar con el efecto de los outliers se han utilizado los M-estimadores ya sea para incluir el efecto de los valores atípicos dentro de la estimación [20–24, 58] o de minimizar el efecto de los mismos o filtrándolos de la población [12, 17, 19, 28, 40].

1.3. Planteamiento del problema

Para la implementación de controladores basados en modelo es necesario contar con un modelo matemático que caracterice el comportamiento del sistema. La obtención de un modelo matemático puede realizarse de distintas maneras. Como ya se ha mencionado, una de las técnicas de modelado hace uso de las leyes físicas que rigen al sistema pero, en algunas ocasiones, el proceso y tiempo de modelado puede llegar a ser extenso, tedioso y a veces impreciso. Por otra parte la técnica de identificación, no requiere un conocimiento a profundidad sobre el proceso que se desea modelar, además, el tiempo para la obtención del modelo es corto en relación del tiempo con el modelado por leyes físicas y el modelo obtenido por identificación puede representar de manera adecuada al proceso real Ljung [43].

Las RNA son una herramienta eficiente para realizar la identificación de sistemas caja negra. Comúnmente, los parámetros de un modelo neuronal se obtienen con la reducción de la norma dos. Sin embargo, cuando se utilizan una gran cantidad de datos contaminados (outliers) como señales para realizar la identificación del sistema, la estimación con el estimador L_2 se vuelve ineficiente. Con el fin de introducir en la estimación los valores atípicos y obtener un modelo matemático capaz de asemejar el comportamiento del sistema real y facilitar el proceso de modelado, se propone utilizar una RNA robusta introduciendo al estimador M de Huber en el proceso de optimización.

1.4. Objetivos

1.4.1. Objetivo general

Probar algoritmos de optimización para el entrenamiento de una red neuronal artificial con el fin de realizar la identificación de sistemas no lineales con outliers.

1.4.2. Objetivos específicos

1. Implementar la red neuronal artificial de Romero et al. [56] usando el método de Levenberg-Marquadt para optimizar los pesos sinápticos.
2. Implementar la red neuronal de Romero et al. [56] utilizando el estimador robusto M de Huber para la introducción de los valores atípicos en la estimación.
3. Identificar una serie de sistemas con outliers usando los algoritmos desarrollados y determinar la aportación de los métodos de optimización seleccionados con relación a los métodos tradicionales (mínimos cuadrados, gradiente descendente, Gauss-Newton, etc.), en función de algunos de los siguientes aspectos:
 - Criterio para determinar el orden de la red neuronal.
 - Ajuste a los parámetros de los métodos de optimización
 - Para el caso del estimador M de Huber, la determinación del procedimiento para conmutar entre la norma uno y la norma dos ($L_1 - L_2$).
4. Probar la efectividad del estimador M de Huber para realizar la identificación de sistemas con señales contaminadas por outliers.

1.5. Alcance de la Tesis

En esta Tesis se pretende desarrollar las siguientes actividades para poder cumplir con los objetivos antes definidos.

- Aplicar la metodología propuesta por Romero [54] para la identificación de sistemas dinámicos no lineales mediante el uso de redes neuronales artificiales.

- Programar distintas estructuras de redes neuronales para la identificación de sistemas.
- Implementar algoritmos de optimización para estimar los pesos sinápticos adecuados con el fin de realizar una identificación eficiente por medio de la programación de funciones y programas realizados en MATLAB.
- Implementar el estimador M de Huber para agregar el comportamiento de los outliers dentro del modelo neuronal mediante el uso de funciones.
- Realizar la identificación de distintos sistemas dinámicos con outliers con el fin de comprobar que el estimador M de Huber como función objetivo es eficiente en comparación con la clásica norma dos.

1.6. Estrategia para desarrollar las metodologías propuestas

Para desarrollar esta Tesis se realizaron los siguientes pasos:

- Estudio de identificación de sistemas.

Se realizó una búsqueda bibliográfica y un estudio de las técnicas de identificación de sistemas de manera general, esto con el objetivo de implementar los conocimientos adquiridos en una red neuronal.

- Estudio de redes neuronales.

Se buscó información general de las redes neuronales, se estudiaron diferentes características, algoritmos de entrenamiento, arquitecturas etc. Después se estudió la red neuronal que propone Romero et al. [56] con el fin aprovechar la capacidad de de la RNA de obtener un alto grado de precisión y bajando su complejidad. También se estudió la configuración NARX y Hammerstein-Wiener que propone [27].

- Estudio de algoritmos de optimización.

Se buscó información de distintos algoritmos de optimización para poder estimar los pesos sinápticos adecuados de tal manera que el modelo neuronal asemejara el comportamiento del sistema dinámico con outliers. Entre los algoritmos que se probaron se encuentran en algoritmo de Gauss-Newton, gradiente descendente y Levenberg-Marquadt.

- Estudio del estimador M de Huber.

Se buscó información general del estimador M de Huber como características, parámetros, función, etc. La búsqueda bibliográfica se centró en la aplicación del estimador M de Huber para la identificación de sistemas dinámicos con outliers artificiales [20–22].

- Validación de modelos neuronales.

Se desarrollaron modelos neuronales de sistemas dinámicos con outliers artificiales. Se verificó que los modelos obtenidos generaran cierto grado de exactitud con los sistemas que se desean identificar con la diferencia entre las dos señales de salida. De esa comparación se obtuvieron índices de medición como el error medio cuadrático, la desviación estándar del error, etc.

- Desarrollo de algoritmos.

Con los conocimientos adquiridos acerca de redes neuronales, algoritmos de optimización y estimador M de Huber, se programarán funciones que realizan el procesamiento de información y que se encargan de combinar las tres bases anteriores. También, se realizaron funciones capaces de entregar una medición comparativa entre el modelo real y el estimado como la desviación estándar, el error cuadrático medio, FIT, etc.

1.7. Organización de la Tesis

En el capítulo dos, se muestra información general de redes neuronales artificiales, la identificación de sistemas con redes neuronales y la red neuronal con la que

se trabajó. También, se muestra un apartado que habla acerca de algoritmos de optimización cuyo objetivo es el de encontrar los pesos sinápticos adecuados para que el modelo neuronal se asemeje al comportamiento físico real.

La explicación del funcionamiento del estimador M de Huber y su implementación, se expresa en el capítulo tres.

En el capítulo cuatro se explica la metodología para realizar la identificación de sistemas con el estimador M de Huber. También se observará la forma en que se calculan los ordenes de la red neuronal y de los parámetros de la función de Huber.

En el capítulo cinco se muestran los resultados obtenidos realizando la identificación de tres sistemas diferentes con y sin valores atípicos y el desempeño de estimación realizado mediante L_2 .

El capítulo seis está conformado por las conclusiones de este tema de tesis y trabajos futuros.

Capítulo 2

REDES NEURONALES ARTIFICIALES

El cerebro humano es la mejor computadora jamás creada, es capaz de adquirir información de cualquiera de los sentidos, procesar tal información y ejecutar una acción según las condiciones en las que el sujeto se encuentre. Se estima que el cerebro humano está constituido por 10^{11} neuronas y que cada neurona se comunica con otra neurona mediante impulsos eléctricos (llamados sinápsis) con el uso de ramificaciones llamadas dendritas. El funcionamiento del cerebro del ser humano es altamente complejo, con el simple hecho de recordar la forma en que vestía la maestra de preescolar, andar en bicicleta, reconocer rostros en la oscuridad, entre otras, se alcanza a apreciar que efectivamente, el cerebro humano es capaz de realizar tareas altamente complicadas. Lo más interesante es que el cerebro aprende sin instrucciones explícitas de ninguna clase y crea representaciones internas que hacen posibles todas estas actividades.

En este capítulo se presenta un estudio sobre redes neuronales artificiales (NN: Neural Networks), se describen aspectos como su funcionamiento, el proceso de identificación mediante el uso de las redes, algoritmos de optimización y la estructura de red neuronal general que se utilizó para el desarrollo de este trabajo de tesis.

2.1. Neuronas biológicas

El cerebro está constituido por un gran número de elementos (10^{11} aproximadamente) altamente interconectados (aproximadamente 10^4 conexiones por elemento) llamados neuronas. Una neurona biológica consta de tres partes esenciales, las dendritas, el cuerpo de la célula o soma y el axón. Las dendritas son ramificaciones nerviosas que cargan de señales eléctricas el cuerpo de la célula y se encuentran alrededor de la célula y del axón. El axón es una fibra nerviosa más larga que las dendritas encargado de llevar la señal desde el cuerpo de la célula hacia otras neuronas. El cuerpo de la célula o soma es el encargado de realizar la suma de todas las señales entrantes, también se encarga de estimular a la célula de tal manera que esta produzca una señal hacia otra neurona. A la “unión” entre el axón de una célula con una dendrita de otra célula se le llama sinápsis. En la figura 2.1 se pueden ver las partes principales que conforman a una neurona biológica.

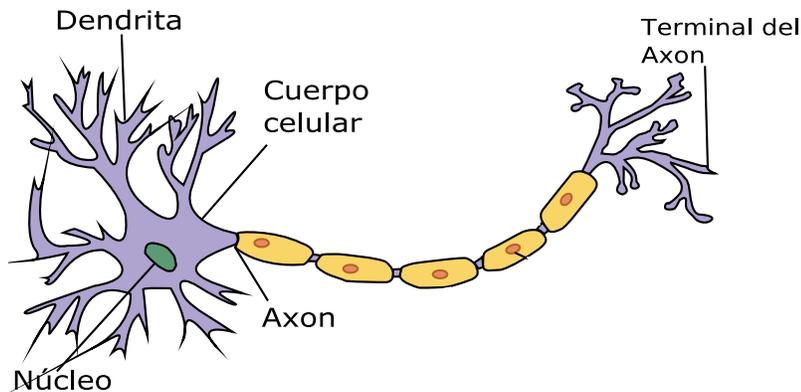


Figura 2.1: Neurona biológica.

Todas las neuronas conducen información de forma similar, la información viaja a través del axón mediante impulsos eléctricos denominados potenciales de acción. Los potenciales de acción alcanzan una amplitud máxima de $100mV$ y duran aproximadamente $1ms$. Cuando la célula se encuentra en reposo el potencial de acción se ve reducido hasta los $-70mV$.

2.2. Neuronas Artificiales

Las redes neuronales artificiales (RNA) se definen como sistemas de mapeos no lineales cuya estructura se basa en la distribución de información como en el sistema nervioso de humanos y animales. Constan de un número grande de procesadores simples ligados por conexiones con pesos. Las unidades de procesamiento se denominan neuronas. Cada unidad recibe entradas de otros nodos y juntos, generan una salida simple ya sea un escalar o un vector que depende de la información recibida. Las neuronas artificiales también pueden nombrarse como nodos, neuronodos, celda, unidad o elemento de procesamiento (PE). Las neuronas artificiales simples fueron introducidas por McCulloch y Pitts [48] a inicios de los años 40's. Una red neuronal está constituida por los siguientes elementos:

1. Un conjunto de unidades de procesamiento o neuronas (ver figura 2.5).
2. Un estado de activación para cada unidad, equivalente a la salida de la unidad o neurona.
3. Conexiones entre las neuronas, generalmente definidas por el producto de un escalar (pesos sináptico) que determina el efecto de una señal de entrada a una próxima unidad o neurona.
4. Una función de activación que actualiza el nuevo nivel de activación basándose en la entrada efectiva y la activación anterior.
5. Un método para reunir información, correspondiente a la regla de aprendizaje o algoritmo de optimización
6. Un ambiente donde la RNA se encontrará interactuando, es decir, señales de entrada, salida e incluso señales de error.

2.2.1. Aplicaciones de las redes neuronales

Desde hace más de tres décadas, las redes neuronales se han utilizado en distintos ámbitos de la ciencia. Algunas áreas donde se aplican las redes neuronales artificiales son:

- Automóviles: Sistemas de piloto automático. Detección de fallas por reconocimiento extremo de vibraciones, detección de fallas en inyectores, etc., mediante el uso de reconocimiento de patrones.
- Electrónica: Predicción de secuencia de códigos. Control de procesos, Análisis de fallas. Visión artificial. Reconocimiento de voz.
- Manufactura: Control de la producción y del proceso, Análisis y diseño de producto. Diagnóstico de fallas en el proceso y maquinarias. Identificación de partículas en tiempo real.
- Medicina: Análisis de células portadoras de cáncer. Análisis de electroencefalograma y de electrocardiograma. Reconocimiento de infartos mediante ECG. Diseño de prótesis. Optimización de tiempo de trasplante.
- Robótica: Control dinámico de trayectoria. Robots elevadores. Controladores. Sistemas ópticos.
- Telecomunicaciones: Compresión de datos e imágenes. Automatización de servicios de información. Traslación en tiempo real de lenguaje hablado.
- Control: Diseño de controladores para una planta en específico o de planta completa. Identificación de sistemas. Detección de fallas. Control Robusto. Controladores adaptables.

2.2.2. Arquitecturas de redes neuronales

La arquitectura de una red neuronal se encuentra en función de la forma en que esta se conecte. El producto de conectar neuronas artificiales con otras será el de obtener capas de neuronas artificiales conectadas entre sí y, por ende, puede mejorar el procesamiento de la información. Existen diversas arquitecturas de redes neuronales, no se podría decir que el número de configuraciones es finito debido a que se siguen proponiendo diferentes tipos de interconexiones neuronales. Cabe mencionar que en algunas ocasiones la salida de una neurona artificial puede ser la entrada hacia una neurona destino, es decir, la salida de un nodo puede ser la

entrada de otro nodo, o incluso ser la entrada de sí mismo (en el caso de las RNA's recurrentes). Entre las configuraciones principales se encuentran:

1. Redes neuronales estáticas
2. Redes neuronales recurrentes
3. Redes neuronales celulares

Se puede observar (ver figura 2.2) que en las redes neuronales estáticas no existe ningún lazo de retroalimentación, es decir, la conexión se encuentra en cascada. El entrenamiento de este tipo de redes neuronales tiende a ser más sencillo que el resto debido a que se puede utilizar la regla de la cadena en la derivación de funciones para la optimización de los pesos sinápticos.

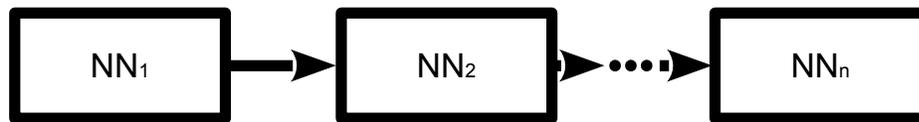


Figura 2.2: Redes neuronales estáticas.

En las redes neuronales recurrentes existe uno o varios lazos de retroalimentación. En algunas ocasiones esta retroalimentación es información de importancia para que el comportamiento de la RNA cambie, es decir, realiza un cambio de decisión para efectuar una acción. Usualmente, el entrenamiento de las redes neuronales recurrentes al igual que las redes neuronales celulares se ve complicado en comparación de las redes neuronales estáticas por tal motivo se han implementado diferentes algoritmos de optimización [25, 38, 46, 51] que ayudan a obtener los pesos sinápticos adecuados de tal manera que la RNA cumpla con los estándares por la que fue hecha. En la figura 2.3 se muestra una configuración simple de una red neuronal recurrente.

Las redes neuronales celulares consisten en la conexión de varias neuronas llamadas células, las células solo se encuentran conectadas con otras neuronas que se encuentren cerca, es decir, no utilizan ningún lazo para realimentar o retroalimentar la información, sin embargo, una neurona o célula que se encuentre distante de otra aún así le puede causar cierta variación a su comportamiento. Existen diversas maneras de conectar las redes neuronales celulares, las más utilizadas son las

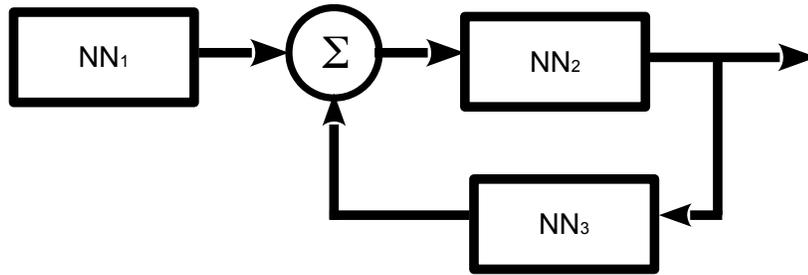


Figura 2.3: Redes Neuronales recurrentes.

que se encuentran conectadas de manera rectangular, hexagonal, triangular o que se encuentre basado en otro tipo de polígono regular. En la figura 2.4 se aprecia la una estructura rectangular de una red neuronal celular. Para este tema de tesis se estará utilizando la arquitectura de una red neuronal estática y recurrente que facilitará el proceso de identificación mostrado en [54, 56, 57].

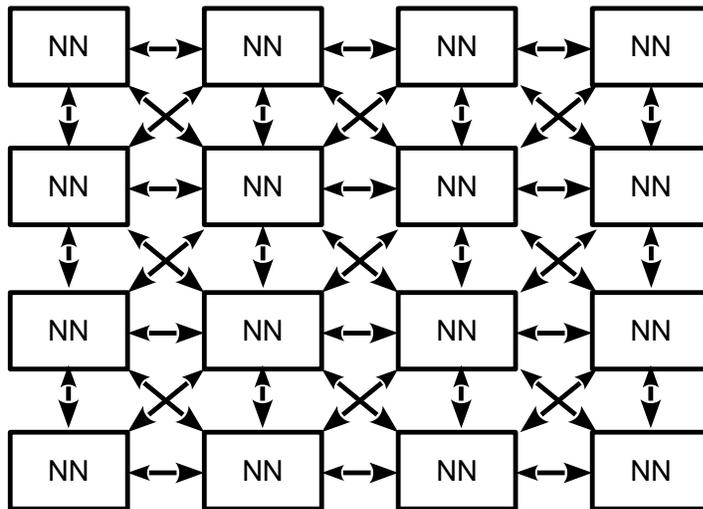


Figura 2.4: Red Neuronal celular.

2.2.3. Perceptrón

El perceptrón es la red neuronal más utilizada y la más simple. El perceptrón fue desarrollado por Rosenblatt [59] con el fin de realizar clasificación, es decir, Rosebnlatt desarrolló un algoritmo capaz de generar un criterio para seleccionar

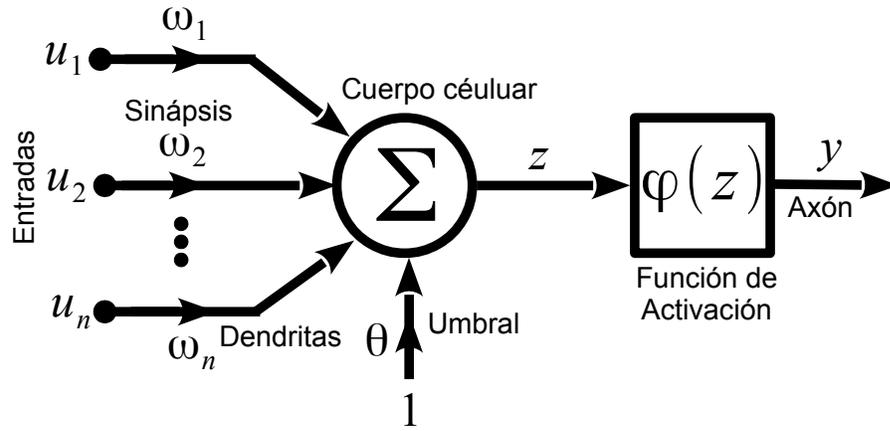


Figura 2.5: Neurona artificial.

un subconjunto de un conjunto de datos. En la figura 2.5 se muestra la estructura general de una neurona artificial de tipo perceptrón. Las entradas u_1, u_2, \dots, u_n (dendritas en la neurona biológica) reciben información proveniente de otra neurona o información externa. Esta información es ponderada por pesos sinápticos $\omega_1, \omega_2, \dots, \omega_n$ (sinápsis), si los pesos sinápticos tienen una magnitud positiva quiere decir que son excitativos pero, si son de magnitud negativa son inhibitorios. Después, la suma de los pesos sinápticos y los umbrales es procesada mediante una función de activación produciendo una señal y (axón) que puede conducir a otra neurona o puede ser la salida del modelo neuronal. La ecuación 2.1 es la representación matemática de neurona artificial mostrada en la figura 2.5.

$$y(k) = \varphi \left(\sum_{i=1}^n \omega_i u_i(k) + \theta_0 \right) \quad (2.1)$$

donde, u_i son las señales de entrada, ω_i son pesos sinápticos, n es el número de entradas a la neurona, θ es un umbral, y es la señal de salida y φ es la función de activación; la intención de la función de activación es modelar el comportamiento no lineal de los sistemas. Las funciones sigmoideas son una clase general de las funciones monóticamente decrecientes con un dominio en $(-\infty, \infty)$. Nótese que, si la compensación o bias θ cambia, la función de activación cambia su cruce en cero ya sea hacia la izquierda o la derecha. En la mayoría de los algoritmos de optimización

para redes neuronales (incluyendo los de propagación hacia atrás BP por sus siglas en ingles), la derivada de $\varphi(\cdot)$ es necesaria, entonces la función de activación que se seleccione debe ser diferenciable.

La expresión para la salida de la neurona $y(k)$ en el instante k ($y(t)$ en el caso continuo) puede ser calculada definiendo un vector columna de nn pesos sinápticos $\bar{\omega} \in \mathbb{R}$ como

$$u(k) = \begin{bmatrix} u_1 & u_2 & \cdots & u_n \end{bmatrix}^T \quad \bar{\omega}(k) = \begin{bmatrix} \omega_1 & \omega_2 & \cdots & \omega_n \end{bmatrix}^T. \quad (2.2)$$

Entonces, es posible escribir la ecuación 2.1 en forma matricial como

$$y(k) = \varphi(\bar{\omega}^T u(k)) + \theta_0 \quad (2.3)$$

para lidiar con el efecto del bias lo que se realiza es cambiar la dimensión de $\bar{\omega}(k)$ y de las entradas $u(k)$ de tal manera que $(u(k), \bar{\omega}) \in \mathbb{R}^{n+1}$, entonces

$$\begin{aligned} u(k) &= \begin{bmatrix} 1 & u^T \end{bmatrix}^T = \begin{bmatrix} 1 & u_1 & u_2 & \cdots & u_n \end{bmatrix} \\ \bar{\omega}(k) &= \begin{bmatrix} 1 & \bar{\omega}^T \end{bmatrix}^T = \begin{bmatrix} \theta_0 & \omega_1 & \omega_2 & \cdots & \omega_n \end{bmatrix} \end{aligned} \quad (2.4)$$

la ecuación 2.4 permite realizar una notación más compacta de la neurona mostrada en la figura 2.5 como

$$y(k) = \varphi(\omega(k)u(k)^T). \quad (2.5)$$

Sabiendo que el vector de entrada $u(k) \in \mathbb{R}^n$ y el vector de pesos sinápticos $\bar{\omega}(k) \in \mathbb{R}$ han sido aumentados por una unidad y un valor de umbral (θ_0), respectivamente, para incluir el umbral dentro del modelo mostrado en (2.5). Se puede decir vagamente que $u(k)$ y $\bar{\omega}$ son elementos de \mathbb{R} . El vector de salida $y(k)$ se refiere al “mecanismo de recuperación celular”. Describe cómo la salida es reconstruida por medio del uso de las señales de entrada y los valores de los pesos sinápticos de cada celda, nodo, neurona, unidad o elemento.

2.2.4. Adaline

Otro de los modelos neuronales clásicos es el Adaline (**ADA**ptative **LIN**ear **E**lement). Fue desarrollado por el profesor Bernie Widrow. La estructura del Adaline es prácticamente idéntica al perceptrón simple, pero es un mecanismo físico capaz de realizar aprendizaje. El modelo neuronal está basado en la teoría propuesta por [48]. La RNA Adaline está compuesta por una sola capa de n neuronas y m entradas.

Las m entradas representan a un vector x de entrada que pertenece a un espacio vectorial \mathbb{R}^m . Por cada neurona, existe un vector w de pesos sinápticos que indican la fuerza de conexión entre los valores de entrada y la neurona. Al igual que el perceptrón, la Adeline contiene un umbral θ . Adaline está compuesta por dos partes: una combinación lineal adaptable encargada de ajustar los pesos sinápticos según lo que se desea realizar y una comparación entre la salida análoga de la RNA (r) y la señal de salida real después de haber sido evaluada por la función de activación y por último un bloque básico que muchos sistemas adaptables contienen. En este caso, Adeline utiliza una función signo con el fin de cuantificar solamente dos casos, es decir, $\hat{y} = \text{sign}(r)$ y que $\hat{y} \in \{-1, 1\}$.

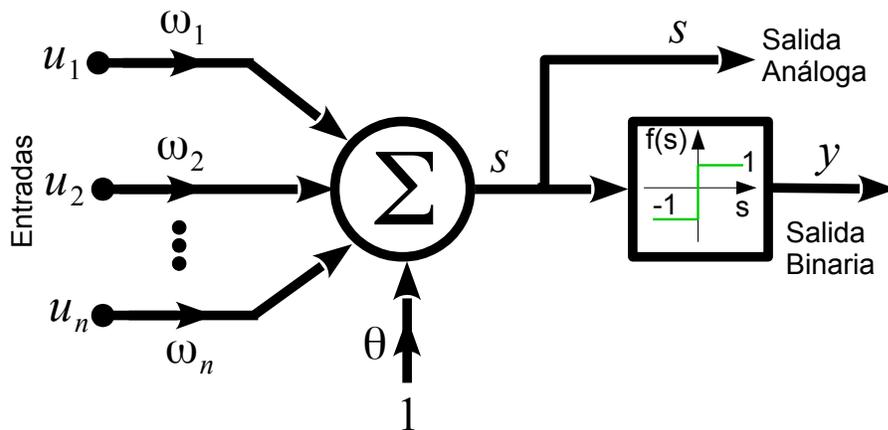


Figura 2.6: Neurona Artificial Adaline.

En la figura 2.6 se muestra la estructura general de una neurona Adaline, se puede observar que la función de activación es una función signo. Con anterioridad se mencionó que las funciones de activación (Vea Anexo A) deben ser suaves, cuando se usa la terminología “suaves” hace referencia a que deben ser diferenciables o de

por lo menos C^1 , es decir, de clase uno debido a que la mayoría de los algoritmos de optimización requieren de la razón de cambio de la función de activación con respecto al parámetro estimado. En este caso, la Adaline tiene una función que no es diferenciable cuando $s = 0$, por tanto se utilizan otros métodos de optimización para poder estimar los pesos sinápticos y umbrales que la conforman. Para ajustar los pesos sinápticos se utiliza la regla Delta Widrow-Hoff que es una simple variación de mínimos cuadrados LMS (Least Mean Square).

Con ayuda en la figura 2.6 se puede observar que

$$\begin{aligned} s &= \sum_{i=1}^n \omega_i u_i + \theta \\ y &= \text{sign} \left(\sum_{i=1}^n \omega_i u_i + \theta \right) \end{aligned} \quad (2.6)$$

Si se define un vector $W = [\omega_1 \ \omega_2 \ \cdots \ \omega_n]^T$, definiendo un vector de error como $e_i = r_i - y_i$ y una función objetivo como $E_N = \frac{1}{2} \sum_{i=1}^N e_i^2$ entonces

$$\hat{W} = \arg \min_W (E_N) \quad (2.7)$$

2.2.5. RNA unicapa

Por cuestiones de síntesis se tomará en cuenta que $\varphi(\cdot)$ es igual a una expresión similar como se mostró en la ecuación (2.1). La figura 2.7 muestra una RNA constituida por \mathcal{Q} neuronas, todas están siendo alimentadas por el mismo vector de entrada $u_j(k)$ y se produce una señal de salida $y(k)$ por neurona. Se define a la figura 2.7 como una RNA de una capa. La ecuación que describe el modelo neuronal de la figura 2.7 está dado por

$$\hat{y}_q(k) = \varphi \left(\sum_{j=1}^n \omega_{qj} u_j(k) + \theta_{q0} \right), \quad q = 1, 2, \dots, \mathcal{Q} \quad (2.8)$$

Para sintetizar el modelo neuronal de la ecuación 2.8, es conveniente escribir los pesos sinápticos y los bías en forma de matriz y de vector, respectivamente. Definiendo la matriz de pesos sinápticos y de bias como

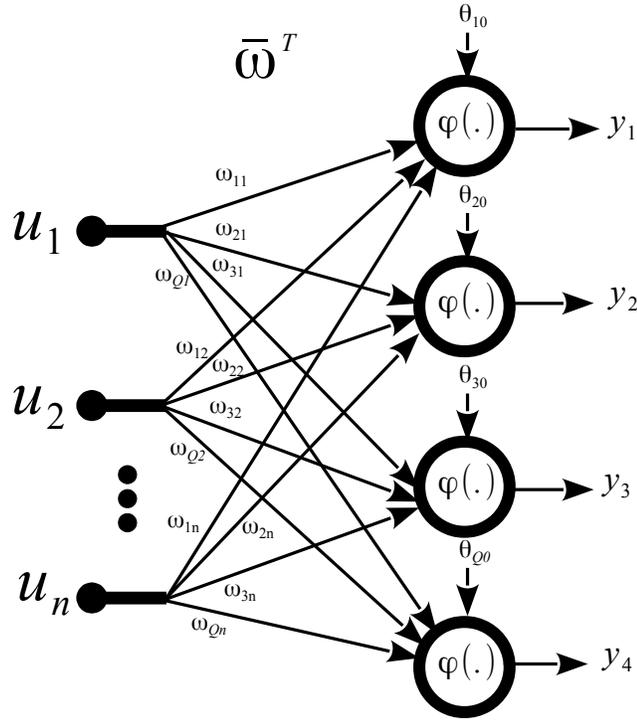


Figura 2.7: Red Neuronal artificial de una capa.

$$\bar{\omega}^T \equiv \begin{bmatrix} \omega_{11} & \omega_{12} & \cdots & \omega_{1n} \\ \omega_{21} & \omega_{22} & \cdots & \omega_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \omega_{Q1} & \omega_{Q2} & \cdots & \omega_{Qn} \end{bmatrix}, \quad b_\theta = \begin{bmatrix} \theta_{10} \\ \theta_{20} \\ \vdots \\ \theta_{Q0} \end{bmatrix}. \quad (2.9)$$

Se define al vector de salida como $y(k) = [y_1 \ y_2 \ \cdots \ y_Q]^T$ entonces

$$\hat{y}(k) = \bar{\varphi}(\bar{\omega}^T u(k) + b_\theta) \quad (2.10)$$

si se define al vector del argumento de la función de activación como w , entonces $w \equiv [w_1 \ w_2 \ \cdots \ w_Q]^T$ y entonces si $\bar{\varphi} \equiv [\bar{\varphi}(w)_1 \ \bar{\varphi}(w)_2 \ \cdots \ \bar{\varphi}(w)_Q]^T$.

Se puede refinar la ecuación (2.10) mediante la inserción del vector de umbral (b_θ) como la primer columna de la matriz $\bar{\omega}$ obteniendo una matriz aumentada con los parámetros fundamentales que conforman a la RNA

$$\Omega^T \equiv \begin{bmatrix} \theta_{10} & \omega_{11} & \cdots & \omega_{1n} \\ \theta_{20} & \omega_{21} & \cdots & \omega_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \theta_{Q0} & \omega_{Q1} & \cdots & \omega_{Qn} \end{bmatrix}. \quad (2.11)$$

Después, la salida de la RNA puede ser expresado en términos de los los vectores de entrada aumentados $u(k)$ como

$$\hat{y}(k) = \bar{\varphi}(\Omega^T u(k)) \quad (2.12)$$

2.2.6. Perceptrón Multicapa

Para la identificación de sistemas y el control de sistemas, las arquitecturas de redes neuronales mas utilizadas son las que se encuentran en cascada y las recurrentes (una combinación de una conexión seriada con un lazo de retroalimentación). En este tipo de configuraciones, generalmente la estructura total de la red se encuentra dividida por secciones o capas que contienen un número finito de neuronas. Por ejemplo, la figura 2.8 muestra una red neuronal tipo perceptrón con dos capas conectadas secuencialmente. La RNA que se muestra en la figura 2.8 está compuesta por dos capas de neuronas, la primer capa contiene \mathcal{P} neuronas, esta alimenta a la capa de siguiente que está compuesta por \mathcal{Q} neuronas. El modelo neuronal de la figura 2.8 se encuentra expresado por la ecuación 2.13.

Cada capa es numerada $(0, 1, 2, \dots, \mathcal{N})$. La capa 0, se le conoce como la capa de entrada, es la capa encargada de recibir información, ya sea información externa o proveniente de otra red neuronal. La capa de entrada alimenta a la capa siguiente, es decir, la capa uno; las capas que se encuentran entre la capa de entrada y la capa \mathcal{N} son llamadas capas ocultas. Las neuronas que se encuentran en las capas ocultas son llamadas neuronas ocultas, y las neuronas que conforman la capa de entrada y salida son neuronas de entrada y salida, respectivamente. Para definir el número de neuronas en cada capa se utilizará la siguiente notación: $n_1-n_2-\dots-n_{\mathcal{N}}$, donde n_1 es el número de neuronas en la capa de entrada o la primer capa oculta, n_2 el número de neuronas en la segunda capa oculta y $n_{\mathcal{N}}$ es el número de neuronas en la capa de salida.

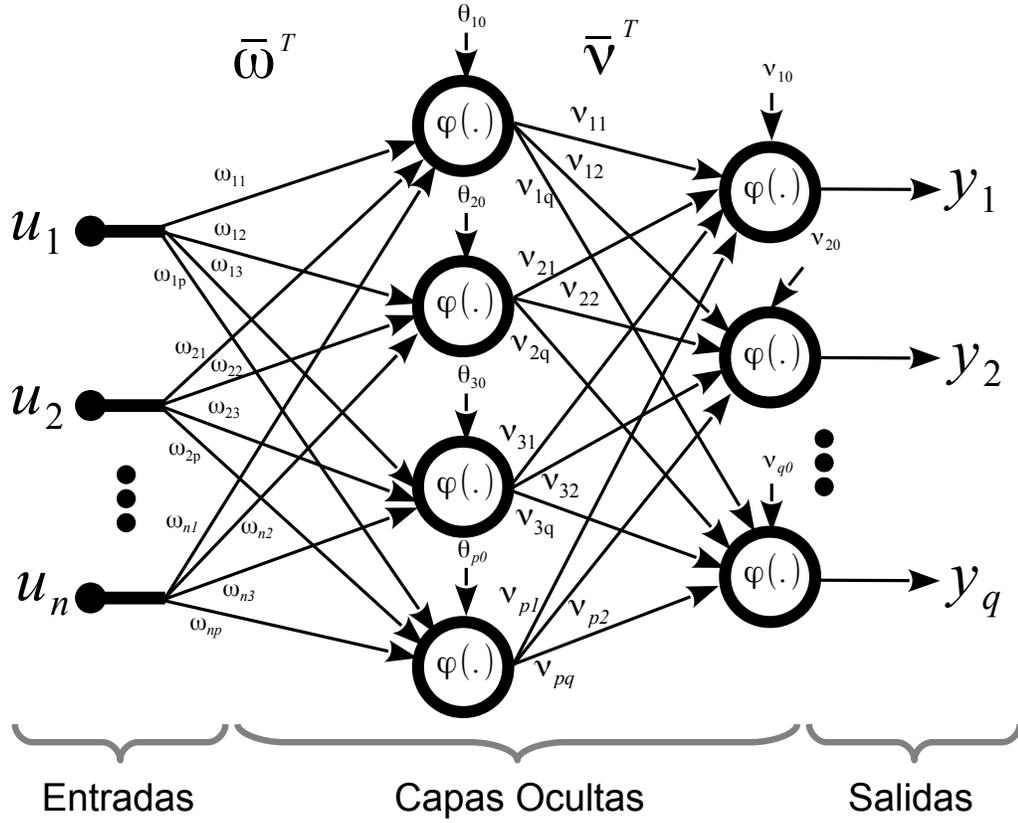


Figura 2.8: Perceptrón de dos capas.

Comúnmente se utilizan algoritmos de *Back-Propagation* (BP) para realizar la optimización de los pesos sinápticos que conforman a la RNA.

$$y_i = \varphi \left(\sum_{p=1}^{\mathcal{P}} \nu_{ip} \varphi \left(\sum_{j=1}^n \omega_{pj} u_j + \theta_{p0} \right) + \nu_{i0} \right), \quad i = 1, 2, \dots, \mathcal{Q} \quad (2.13)$$

definiendo la salidas de la capa oculta como z_q permite que

$$\begin{aligned} z_q &= \varphi \left(\sum_{j=1}^n \omega_{pj} u_j + \theta_{p0} \right), \quad p = 1, 2, \dots, \mathcal{P} \\ y_i &= \varphi \left(\sum_{p=1}^{\mathcal{P}} \nu_{ip} z_q + \nu_{i0} \right), \quad i = 1, 2, \dots, \mathcal{Q} \end{aligned} \quad (2.14)$$

Si se definieran matrices aumentadas de los pesos sinápticos de cada capa, es decir, una matriz que contenga los pesos sinápticos de la primera capa junto con los umbrales y otra matriz con la misma estructura se tendría que

$$\Omega^T \equiv \begin{bmatrix} \theta_{10} & \omega_{11} & \cdots & \omega_{1n} \\ \theta_{20} & \omega_{21} & \cdots & \omega_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \theta_{p0} & \omega_{p1} & \cdots & \omega_{pn} \end{bmatrix}, \quad \Gamma^T \equiv \begin{bmatrix} \nu_{10} & \nu_{11} & \cdots & \nu_{1n} \\ \nu_{20} & \nu_{21} & \cdots & \nu_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \nu_{p0} & \nu_{p1} & \cdots & \nu_{pn} \end{bmatrix}. \quad (2.15)$$

Se puede expresar el modelado neuronal de una forma compacta si utilizamos las matrices definidas en 2.15. De tal manera que

$$y = \varphi(\Omega^T \varphi(\Gamma^T u)) \quad (2.16)$$

2.3. Identificación de sistemas utilizando redes neuronales

Se deben considerar dos aspectos que son de suma importancia al realizar la identificación de sistemas mediante el uso de redes neuronales. Uno de ellos es la estructura de la red neuronal correcta y el entrenamiento u optimización de la misma. Para este tema de tesis se escogió una estructura de red neuronal propuesta en Romero et al. [56].

2.3.1. Selección de estructura neuronal

La selección de la estructura de la red neuronal a utilizar es la decisión más importante para realizar la identificación. Como se mencionó con anterioridad, las redes neuronales con realimentación y recurrentes son las más utilizadas para realizar la identificación de sistemas. Una vez que la estructura de la red ha sido seleccionada es importante definir el número de capas, el número de neuronas en cada capa, el tipo de función de activación. Generalmente, la selección de los aspectos anteriores se basan en el conocimiento previo del sistema. Pero, como el objetivo de este tema

de tesis es el modelar sistemas de tipo caja negra con valores atípicos, la selección de los aspectos anteriores se vuelve tediosa y se realiza a prueba y error.

La selección del número de capas y el número de neuronas debe ser escogido de tal manera que se pueda obtener un balance entre complejidad y precisión. Cuando se dice complejidad se refiere a que la estructura de la red sea “fácilmente diferenciable”.

Con respecto a las funciones de activación, las funciones de activación permite que la RNA aproxime a comportamientos no lineales si esta es no lineal. Comúnmente se seleccionan funciones sigmoideas o infinitamente diferenciables (C^∞). La función de activación más utilizada es la $\tanh(\bullet)$ pero en el caso de utilizar redes neuronales de base radial utilizan funciones campana (haciendo referencia a la campana de Gauss). La $\tanh(\bullet)$ es utilizada debido a que se caracteriza por ser saturada, es decir, cuando el argumento de la función es mayor a tres, la función tiende a ser igual a uno. Otra de las ventajas es que se puede obtener la derivada de la función mediante el uso de identidades trigonométricas (sabiendo que la derivada de $\tanh(\bullet)$ es igual a $\text{sech}^2(\bullet)$).

La red neuronal que se utilizó para el desarrollo de este tema de tesis es la que se propuso en Romero et al. [56, 57], Romero y Corbier [58]. Más adelante se hablará acerca de las configuraciones que esta red puede tener como también de las propiedades de reducción con la que esta cuenta considerando suposiciones algebraicas.

2.3.2. Entrenamiento de una red neuronal

La otra parte fundamental para realizar la identificación de sistemas mediante el uso de redes neuronales es el proceso de entrenamiento. En este proceso se aplican algoritmos de optimización cuyo objetivo es el de encontrar el mejor grupo de parámetros que conforman al modelo propuesto con la reducción de una función E_N . El proceso de entrenamiento se muestra en la figura 2.9.

Después de haber seleccionado la red neuronal que se desea utilizar para realizar la identificación del sistema. La RNA recibe las entradas pasadas $\begin{bmatrix} u(k-1) & \dots & u(k-nb) \end{bmatrix}$ y salidas pasadas $\begin{bmatrix} y(k-1) & \dots & y(k-nb) \end{bmatrix}$ del sistema real, se generan pesos sinápticos aleatorios (parámetros de la red neuronal) y así se genera una salida $\hat{y}(k)$ (salida de la RNA). La señal de salida estimada (\hat{y}) sustraerá a la señal real del sistema ($y(k)$) generando un error de predicción $e(k)$. La señal de error ($e(k)$), las entradas y

salidas pasadas son procesadas en un algoritmo de optimización, el cual se encarga de generar nuevos pesos sinápticos de tal manera que $\hat{y}(k) \approx y(k)$.

Una manera de actualizar los pesos sinápticos es calcular su variación e ir acumulando los resultados y solamente entonces se procede a la actualización e los mismos. Este esquema se suele denominar *aprendizaje por lotes* (batch) [47]. Otra alternativa consiste en actualizar los pesos tras la presentación de cada patrón y se denomina *aprendizaje en serie*, la segunda opción es la que se utilizó en este trabajo de tesis. Los algoritmos de entrenamiento, aprendizaje u optimización se encargan de adaptar los pesos sinápticos de la arquitectura de la red neuronal seleccionada, es de suma importancia que para esta instancia ya se haya seleccionado la arquitectura de la red neuronal debido a que el algoritmo de optimización necesita forzosamente una estructura seleccionada. La adaptación se realiza mediante la reducción o minimización de una función objetivo (MSE 'ver 2.17', SVM, **función de Huber** 'ver 2.18', etc). La selección de la función objetivo depende del usuario, es decir, si los datos no se encuentran "contaminados" por valores atípicos se puede utilizar la suma del error cuadrático (MSE por sus siglas en inglés: mean squared error). Pero, si los datos se encuentran "contaminados" por valores atípicos, una función robusta como la función de Huber es capaz de facilitar la estimación de los parámetros que conforman a la red neuronal.

$$E(\hat{w}, u^N, y^N, e) = \frac{1}{2}e^2(k, \hat{w}) \quad (2.17)$$

$$E(\hat{w}, u^N, y^N, e) = \begin{cases} \frac{1}{2}e^2(k, \hat{w}) & \text{para } |e(k, \hat{w})| \leq \gamma \\ \gamma|e(k, \hat{w})| - \frac{1}{2}\gamma^2 & \text{para } |e(k, \hat{w})| > \gamma \end{cases} \quad (2.18)$$

En la ecuaciones 2.17 y 2.18 se muestran algunos ejemplos de funciones objetivo donde, N es el número total de datos y $e(k, \hat{w})$ es el error de predicción que puede ser calculado como (2.19). Como se mencionó con anterioridad, el objetivo de los algoritmos de optimización es el de obtener los pesos sinápticos adecuados de tal manera que $\hat{y}(k) \approx y(k)$ para ello, es necesario la minimización de una función objetivo como se muestra en (2.20).

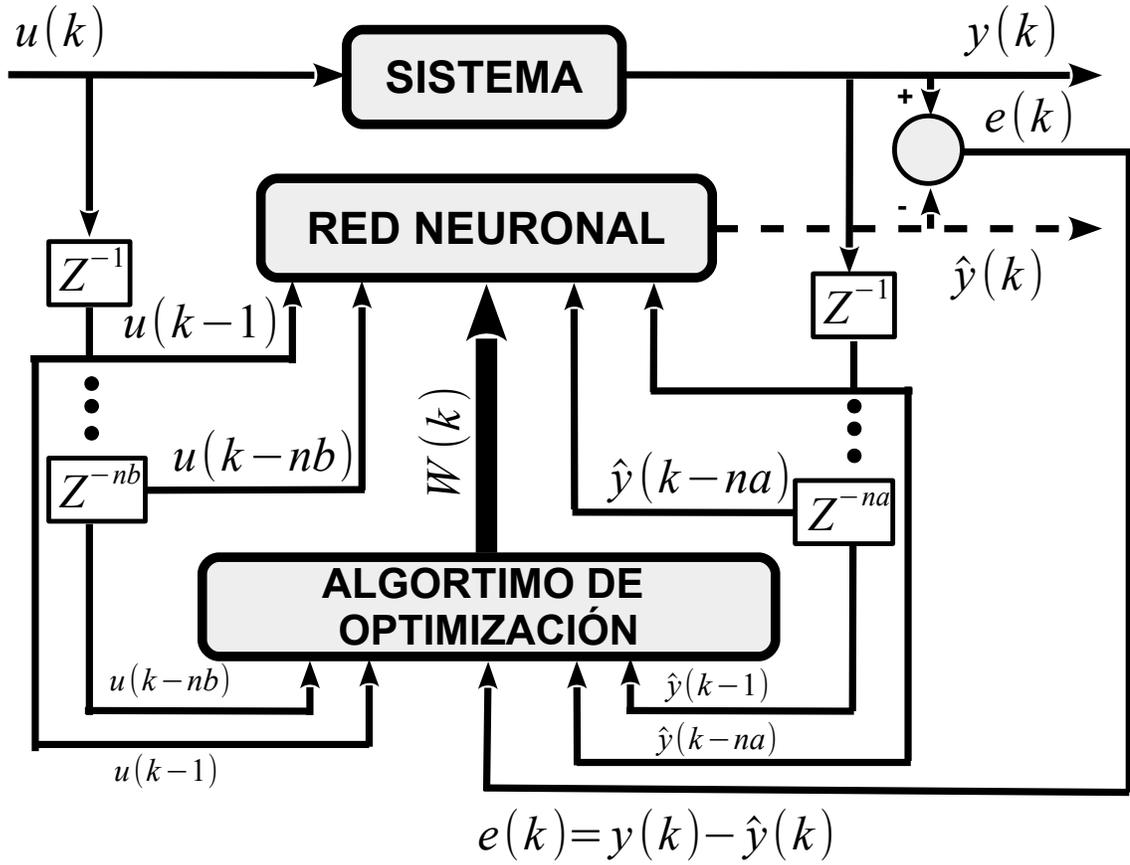


Figura 2.9: Proceso de entrenamiento.

$$e(k, \hat{w}) = y(k) - \hat{y}(k, \hat{w}) \quad (2.19)$$

$$\hat{W}(k) = \underset{W}{\text{mín}}(E(\hat{w}, u^N, y^N, e)) \quad (2.20)$$

2.3.3. Algoritmos de optimización

Los algoritmos de optimización, aprendizaje, entrenamiento, etc. son usados para adaptar los pesos sinápticos de una arquitectura de red neuronal definida. El objetivo de esta adaptación es minimizar la función objetivo por ejemplo (2.17 y 2.18). Los algoritmos de aprendizaje comúnmente utilizados para la adaptación de los pesos sinápticos son los de propagación hacia atrás o retropropagación (RP o BP), que se

basan principalmente en la regla de la cadena para obtener las derivadas parciales correspondientes al parámetro que se desea estimar. La forma general de los algoritmos para adaptar los pesos sinápticos está dada por la ecuación (2.21).

$$\hat{w}(k+1) = \hat{w}(k) - \eta[R^{-1}] \frac{\partial E}{\partial \hat{w}} \quad (2.21)$$

donde R modifica el tipo del algoritmo, η es el factor de aprendizaje, $\frac{\partial E}{\partial \hat{w}}$ es el gradiente de la función objetivo con respecto al parámetro que se desea estimar ($\hat{w}(k)$) en la k -ésima iteración. Existen diversos algoritmos de optimización, como el gradiente descendente, Gauss-Newton, Levenberg-Marquadt, SVR, ELM, PSO, etc.

Gradiente Descendente

Como se mencionó con anterioridad, la ecuación (2.21) generaliza a gran parte de los algoritmos de optimización. En este caso si $R = 1$ se obtiene el algoritmo del gradiente descendente como se expresa en la ecuación (2.22).

$$\hat{w}(k+1) = \hat{w}(k) - \eta \frac{\partial E}{\partial \hat{w}} \quad (2.22)$$

donde $\hat{w}(k+1)$ es el valor del parámetro en la siguiente época, ciclo o iteración, $\hat{w}(k)$ es el valor del parámetro actual, $\frac{\partial E}{\partial \hat{w}}$ es la razón de cambio de la función costo con respecto al parámetro que se desea estimar y η es el factor de aprendizaje.

En la ecuación (2.22), el parámetro η juega un papel importante en la adaptación de los pesos sinápticos pero es de mayor importancia su papel en el algoritmo de optimización debido a que define la velocidad de convergencia del algoritmo. En los algoritmos de optimización el valor de η debe encontrarse delimitado en un rango pequeño. Si se define un valor de η relativamente pequeño, significa que la velocidad de convergencia será pequeña pero, si se define el valor de η relativamente grande, entonces se tendrá una convergencia rápida pero podrían presentarse complicaciones como oscilaciones en el comportamiento de la función costo con respecto al número de épocas o también el algoritmo podría diverger. Para evadir la problemática que produce la selección de un valor de η constante, se implementó un algoritmo de “búsqueda y convergencia” sugerido en [44] y [18]. El algoritmo es sencillo de implementar; primeramente busca un valor relativamente alto para el coeficiente

de aprendizaje de tal manera que discrimine los mínimos locales y conforme pasan las iteraciones correspondientes al número de datos de procesamiento el valor del coeficiente η disminuye tal como se muestra en la figura 2.10. El algoritmo puede ser implementado bajo numerosas expresiones, para este tema de tesis se utilizó la expresión de la ecuación (2.23).

$$\eta(k) = \eta_0 \frac{1}{1 + \frac{k}{10 * k_0}} \quad (2.23)$$

donde η_0 es un valor inicial propuesto por el usuario, k_0 es igual a $\frac{N}{3}$, N es el número total de datos que se está utilizando para realizar la estimación del sistema y k es el k -ésimo instante de tiempo.

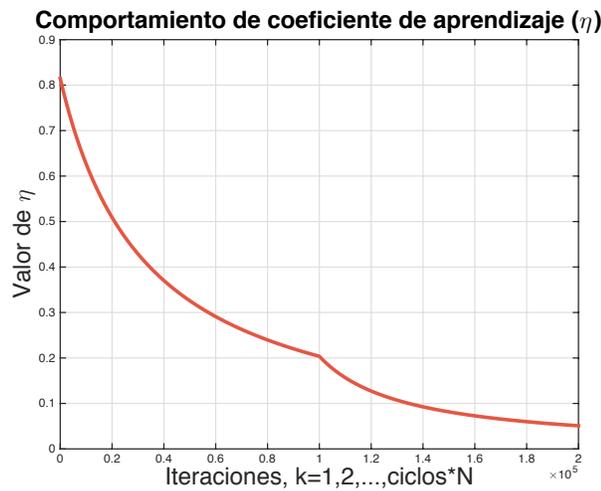


Figura 2.10: Comportamiento del coeficiente de aprendizaje.

Newton

La idea tras el método de Newton es tal que una función objetivo $E(\hat{w})$ debe ser minimizada con una aproximación local mediante una función cuadrática. La función $E(\hat{w})$ cerca al punto $\hat{w}(k)$ ($k = 1, 2, \dots, N$) puede ser aproximada mediante la serie truncada de Taylor:

$$E(\hat{w}) \cong E(\hat{w}(k)) + \Delta \hat{w}^T \nabla E(\hat{w}(k)) + \frac{1}{2} \Delta \hat{w}^T \nabla^2 E(\hat{w}(k)) \Delta \hat{w}. \quad (2.24)$$

Realizando ciertas operaciones se tiene que:

$$\hat{w}(k+1) = \hat{w}(k) - [\nabla^2 E(\hat{w})]^{-1} \nabla E(\hat{w}). \quad (2.25)$$

donde $[\nabla^2 E(\hat{w})]$ se puede definir como la matriz Hessiana de la función objetivo con respecto al parámetro que se desea optimizar y $\nabla E(\hat{w})$ es la matriz Jacobiana de la función objetivo.

Utilizando la generalización de la ecuación (2.21), considerando que $R = \nabla^2 E(\hat{w})$ y que el valor de $\eta = 1$ se puede llegar a la misma expresión (2.25).

En general, el método de Newton tiene un resultado de convergencia rápido, pero requiere la evaluación de la primera y segunda derivada de la función objetivo y del cálculo del inverso de la matriz Hessiana lo cual contrae problemas de singularidades. Además, si el punto inicial (\hat{w}_0) se encuentra distante del mínimo global, la matriz Hessiana podría no ser positiva definida y el algoritmo podría diverger.

Levenberg-Marquardt

El algoritmo de Levenberg-Marquardt es una modificación al algoritmo de Newton, el cual evita ciertas desventajas que tiene el algoritmo de Newton. La técnica de Levenberg-Marquardt [46] aproxima el comportamiento de la matriz Hessiana por la matriz $\nabla^2 E = (J(\hat{w})^T J(\hat{w}) + \mu I)$, donde μ es un factor positivo con efectos (para optimización) parecidos al comportamiento de η en el algoritmo del gradiente descendente e I es la matriz identidad. Partiendo del método de Newton mostrado en la ecuación (2.25), se tiene que $\nabla E(\hat{w}) \approx 2J(\hat{w})^T e(\hat{w})$ donde $J(\hat{w})^T$ es la matriz Jacobiana transpuesta de la función objetivo, $e(\hat{w})$ son los residuos de la estimación (ver ecuación 2.19) y $\nabla^2 E(\hat{w}) \approx 2J(\hat{w})^T J(\hat{w})$. Realizando ciertas operaciones se llega a la expresión mostrada en la ecuación (2.26) la cual es el algoritmo de Levenberg-Marquardt.

$$\hat{w}(k+1) = \hat{w}(k) - [J(\hat{w})^T J(\hat{w}) + \mu I]^{-1} J(\hat{w})^T e(\hat{w}) \quad (2.26)$$

Nótese que $[J(\hat{w})^T J(\hat{w}) + \mu I]$ es una matriz simétrica no singular, aún si la matriz Hessiana (aproximada por $J(\hat{w})^T J(\hat{w})$) es singular. Además, esta matriz será positiva definida con la selección apropiada de μ . Por otro lado, el parámetro μ debería ser pe-

queño para asegurar una convergencia rápida. El algoritmo de Levenberg-Marquardt posee una similitud al algoritmo de Newton. Además, el algoritmo puede converger a un valor óptimo a pesar que el valor inicial \hat{w}_0 sea relativamente pobre (es decir, que el valor inicial se encuentre distante del valor óptimo \hat{w}^*) al igual que el método del gradiente descendente. En conclusión, si el valor de $\mu \rightarrow \infty$ el algoritmo de Levenberg-Marquardt tenderá a comportarse como el método del gradiente descendente pero, si $\mu \rightarrow 0$ el algoritmo tendrá las capacidades de convergencia iguales al algoritmo de Newton. Para asegurar la mejor selección del valor de μ se utilizó un algoritmo de “búsqueda y convergencia” [18, 44] al igual que en el caso del gradiente descendente considerando un perfil de curva diferente. El algoritmo de búsqueda y convergencia para μ está dado por la ecuación (2.27) y visto en la figura 2.11

$$\mu = \mu_0 \left(1 + \frac{k}{5 * k_0} \right) \quad (2.27)$$

donde μ_0 es el valor inicial que puede ser calculado por la traza de la matriz Hessiana [44], k_0 es igual $\frac{N}{3}$, N es el número total de datos que se están utilizando para realizar la estimación del sistema y k es el k-ésimo instante de tiempo discreto.



Figura 2.11: Comportamiento del coeficiente de aprendizaje.

2.4. Red neuronal recurrente 2nn - 2 - 1: Precisión

La figura 2.12 muestra una red neuronal de tres capas. Se puede observar que la primera capa está compuesta por $2 * nn$ neuronas, la segunda capa por dos neuronas y la capa de salida por una neurona. Al tener un número elevado de neuronas se puede asegurar una mejor precisión en la estimación del sistema físico. nn representa el número de neuronas que el usuario utilice, es decir, puede variar el número de neuronas en la primera según las características de diseño que el usuario requiera. El número de neuronas encargadas de procesar la entrada y salida también está en función del número nn ; se puede ver que la red neuronal 2nn-2-1 se encuentra seccionada en dos partes, la primer sección se encarga de procesar la información de la señal de entrada del sistema real, la segunda sección procesa la información de la señal producida por el sistema, es decir, la señal de salida. El modelo matemático de esta red neuronal se encuentra expresado en la ecuación 2.28

$$\begin{aligned}
 \hat{y}(k) &= X\varphi_3(T) \\
 T &= Z_b\varphi_2(r_b) + Z_a\varphi_2(r_a) + Z_h \\
 r_b &= \sum_{i=1}^{nn} V_{bi}\varphi_1(J_u W_{bi}) \\
 r_a &= \sum_{i=1}^{nn} V_{ai}\varphi_1(J_{\hat{y}} W_{ai})
 \end{aligned} \tag{2.28}$$

donde:

$$\left\| \begin{aligned}
 J_u &= \begin{bmatrix} u(k-1-nk) & u(k-2-nk) & \dots & u(k-nb-nk) \end{bmatrix} \in \mathbb{R}^{N \times nb} \\
 J_{\hat{y}} &= \begin{bmatrix} \hat{y}(k-1) & \hat{y}(k-2) & \dots & \hat{y}(k-na) \end{bmatrix} \in \mathbb{R}^{N \times na} \\
 W_{b_i} &= \begin{bmatrix} W_{b_{i,1}} & W_{b_{i,2}} & \dots & W_{b_{i,nb}} \end{bmatrix}^T \in \mathbb{R}^{nb \times 1} \quad i = 1, 2, \dots, nn \\
 W_{a_i} &= \begin{bmatrix} W_{a_{i,1}} & W_{a_{i,2}} & \dots & W_{a_{i,na}} \end{bmatrix}^T \in \mathbb{R}^{na \times 1} \quad i = 1, 2, \dots, nn \text{ y}
 \end{aligned} \right.$$

$X, Z_b, Z_a, V_{b_i}, V_{a_i}, W_{b_i}, W_{a_i}, Z_h \in \mathbb{R}$ y son pesos sinápticos y N corresponde al número de datos utilizados para realizar la estimación.

Partiendo del modelo mostrado en 2.28 se pueden definir modelos diferentes cambiando solamente la función de activación por ejemplo:

Si $\varphi_3(z) = \varphi_2(z) = z$ y $\varphi_1(z) = f(z)$, es decir, una función no lineal:

$$\begin{aligned}
 \hat{y}(k) &= XT \\
 T &= Z_b r_b + Z_a r_a + Z_h \\
 r_b &= \sum_{i=1}^{nn} V_{bi} \varphi_1(J_u W_{b_i}) \\
 r_a &= \sum_{i=1}^{nn} V_{ai} \varphi_1(J_{\hat{y}} W_{a_i})
 \end{aligned} \tag{2.29}$$

Si $\varphi_3(z) = \varphi_1(z) = z$ y $\varphi_2(z) = f(z)$

$$\begin{aligned}
 \hat{y}(k) &= XT \\
 T &= Z_b \varphi_2(r_b) + Z_a \varphi_2(r_a) + Z_h \\
 r_b &= \sum_{i=1}^{nn} V_{bi} J_u W_{b_i} \\
 r_a &= \sum_{i=1}^{nn} V_{ai} J_{\hat{y}} W_{a_i}
 \end{aligned} \tag{2.30}$$

Se pueden definir a todas las funciones de activación como lineales y así crear una estructura lineal de tipo ARX. Es decir, si $\varphi_3(z) = \varphi_2(z) = \varphi_1(z) = z$

$$\begin{aligned}
 \hat{y}(k) &= XT \\
 T &= Z_b(r_b) + Z_a(r_a) + Z_h \\
 r_b &= \sum_{i=1}^{nn} V_{bi} J_u W_{b_i} \\
 r_a &= \sum_{i=1}^{nn} V_{ai} J_{\hat{y}} W_{a_i}
 \end{aligned} \tag{2.31}$$

Seleccionando a $\varphi_2(z) = \varphi_1(z) = z$ y $\varphi_3(z) = f(z)$

$$\begin{aligned}
 \hat{y}(k) &= X\varphi_3(T) \\
 T &= Z_b(r_b) + Z_a(r_a) + Z_h \\
 r_b &= \sum_{i=1}^{nn} V_{bi}(J_u W_{b_i}) \\
 r_a &= \sum_{i=1}^{nn} V_{ai}(J_{\hat{y}} W_{a_i})
 \end{aligned} \tag{2.32}$$

Como se puede observar, cambiando solamente las funciones de activación puede cambiar el modelo neuronal radicalmente.

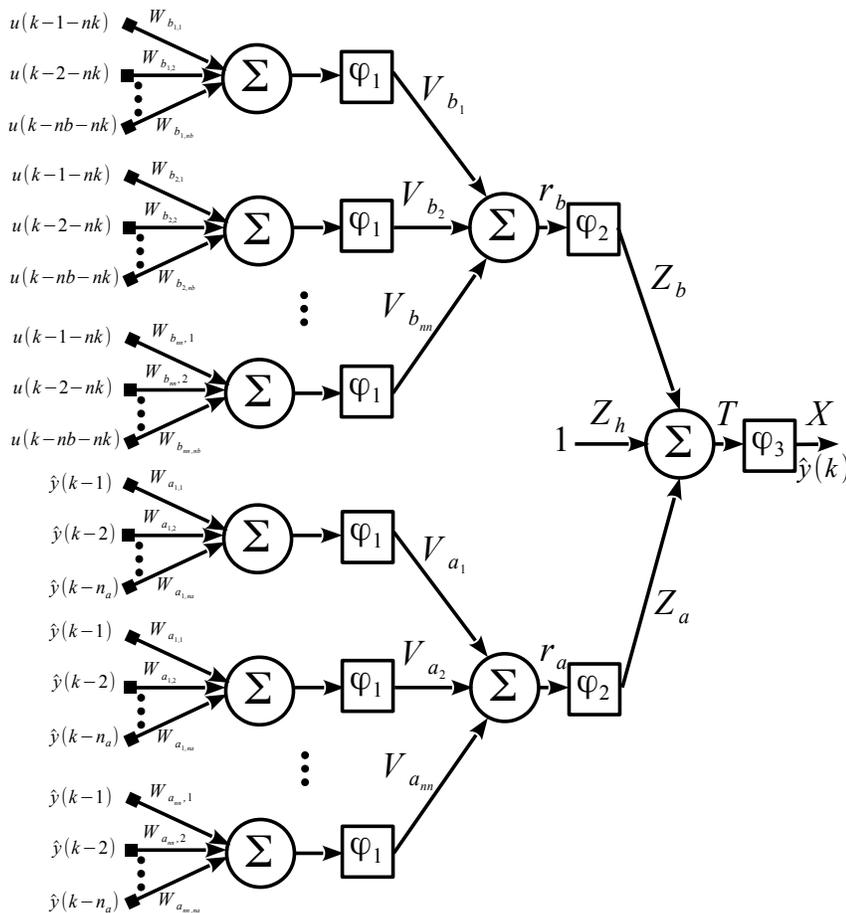


Figura 2.12: Red neuronal recurrente 2nn-2-1.

2.5. Red neuronal recurrente 2-1: Simplicidad

La arquitectura de la red neuronal 2nn-2-1 le permite ser reducida a una red neuronal con un número de parámetros más pequeño pero sin perder la precisión que la red original es capaz de producir. En la figura 2.13 se puede observar la estructura de la red neuronal. La estructura es sencilla: consta de dos neuronas en la primera capa y en la capa de salida por una neurona. El modelo matemático de esta red neuronal se muestra en la ecuación 2.33.

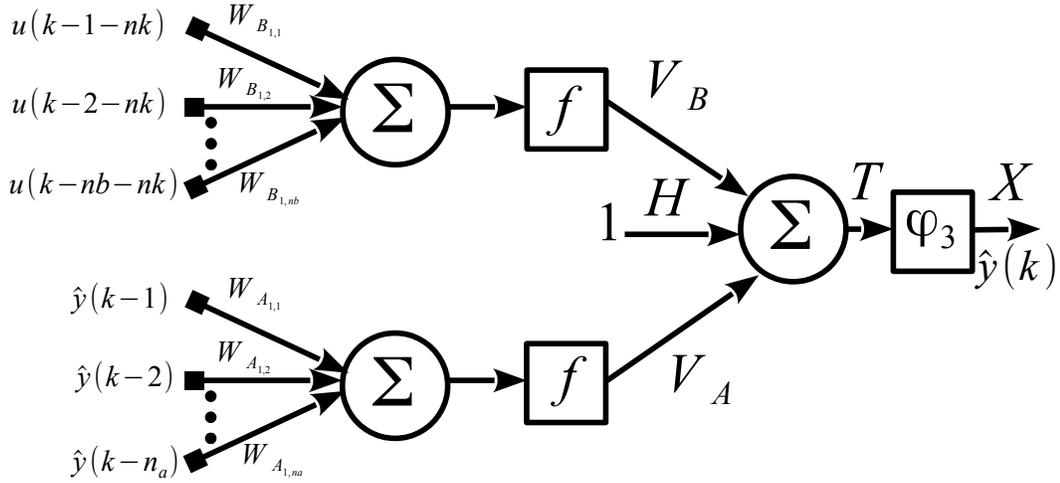


Figura 2.13: Red neuronal recurrente 2-1. Donde f puede ser definida como φ_1 o φ_2

$$\hat{y}(k) = X\varphi_3(T)$$

$$T = V_B f \left(\sum_{i=1}^{nn} J_u W_{B,1,i} \right) + V_A f \left(\sum_{i=1}^{nn} J_{\hat{y}} W_{A,1,i} \right) + H \quad (2.33)$$

$$T = V_B f(J_u W_B) + V_A f(J_{\hat{y}} W_A) + H$$

donde:

$$\begin{aligned}
J_u &= \begin{bmatrix} u(k-1-nk) & u(k-2-nk) & \cdots & u(k-nb-nk) \end{bmatrix} \in \mathbb{R}^{N \times nb} \\
J_{\hat{y}} &= \begin{bmatrix} \hat{y}(k-1) & \hat{y}(k-2) & \cdots & \hat{y}(k-na) \end{bmatrix} \in \mathbb{R}^{N \times na} \\
W_B = W_{B_{1,i}} &= \begin{bmatrix} W_{B_{1,1}} & W_{B_{1,2}} & \cdots & W_{B_{1,nb}} \end{bmatrix}^T \in \mathbb{R}^{nb \times 1} \quad i = 1, 2, \dots, nb \\
W_A = W_{A_{1,i}} &= \begin{bmatrix} W_{A_{1,1}} & W_{A_{1,2}} & \cdots & W_{A_{1,na}} \end{bmatrix}^T \in \mathbb{R}^{na \times 1} \quad i = 1, 2, \dots, na
\end{aligned}$$

V_B, V_A y $H \in \mathbb{R}$ y son pesos sinápticos.

2.6. Red neuronal recurrente 2-2-1: Costo computacional

En Romero [55] se comprobó que la siguiente estructura de red neuronal hace frente al costo computacional, es decir, el tiempo de estimación es relativamente competente contra el tiempo de estimación que requieren los algoritmos desarrollados por softwares de cálculo numérico como lo es MATLAB. En este tema de tesis no se considera el costo computacional, sino la estructura de la red para realizar la identificación de los sistemas que se muestran en el capítulo cinco. La red neuronal está compuesta por dos neuronas en la capa de entrada (una neurona para procesar las señales de entrada y otra neurona para procesar la información que produce la salida del sistema que se desea estimar). En la capa oculta está conformada por dos neuronas más, esto con el fin de poder agilizar el adelanto o atraso de las funciones de activación y, por último, en la capa de salida se puede encontrar una neurona. En la figura 2.14 se puede observar la arquitectura que se ha descrito.

El modelo neuronal de la estructura mostrada en la figura 2.14 se puede apreciar en la ecuación (2.34).

$$\begin{aligned}
\hat{y}(k) &= X\varphi_3(T) \\
T &= Z_b\varphi_2(rb) + Z_a\varphi_2(ra) + Z_h \\
r_b &= V_b\varphi_1\left(\sum_{i=1}^{nb} J_u W_{b_{1,i}}\right) + V_{bh} \\
r_a &= V_a\varphi_1\left(\sum_{i=1}^{na} J_{\hat{y}} W_{a_{1,i}}\right) + V_{ah}
\end{aligned} \tag{2.34}$$

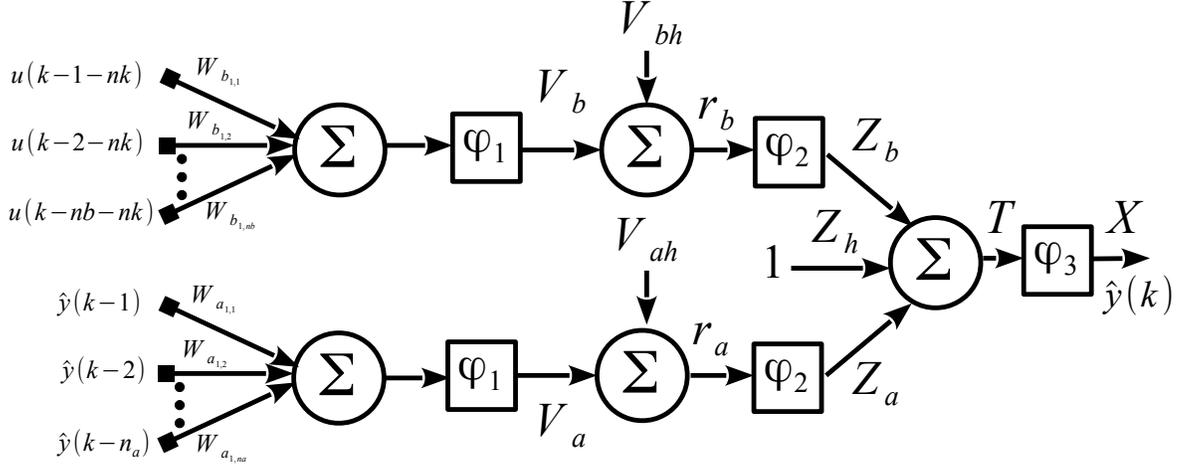


Figura 2.14: Red neuronal recurrente 2-2-1.

$$\begin{aligned}
 J_u &= \begin{bmatrix} u(k-1-nk) & u(k-2-nk) & \cdots & u(k-nb-nk) \end{bmatrix} \in \mathbb{R}^{N \times nb} \\
 J_{\hat{y}} &= \begin{bmatrix} \hat{y}(k-1) & \hat{y}(k-2) & \cdots & \hat{y}(k-na) \end{bmatrix} \in \mathbb{R}^{N \times na} \\
 W_{b_{1,i}} &= \begin{bmatrix} W_{b_{1,1}} & W_{b_{1,2}} & \cdots & W_{b_{1,nb}} \end{bmatrix}^T \in \mathbb{R}^{nb \times 1} \quad i = 1, 2, \dots, nb \\
 W_{a_{1,i}} &= \begin{bmatrix} W_{a_{1,1}} & W_{a_{1,2}} & \cdots & W_{a_{1,na}} \end{bmatrix}^T \in \mathbb{R}^{na \times 1} \quad i = 1, 2, \dots, na
 \end{aligned}$$

$V_b, V_a, V_{bh}, V_{ah}, X, Z_a, Z_b$ y $Z_h \in \mathbb{R}$ y son pesos sinápticos.

En la siguiente sección se considerará el modelo neuronal (2.28) para realizar el proceso de reducción algebraica de tal manera que el modelo resultante guarde el mismo grado de precisión pero reduciendo el número de parámetros original.

2.7. Proceso de Reducción

Este proceso de reducción permite que el modelo neuronal 2nn-2-1 (ver ecuación 2.28) pueda ser simplificado a un modelo neuronal con un número de parámetros reducido (un modelo neuronal como 2.33). El proceso de reducción se basa en dos suposiciones, la primera se refiere en la arquitectura de la red neuronal y la segunda se refiere los valores de inicialización de los pesos sinápticos (valor inicial del peso sináptico, es decir, \hat{w}_0) y al entrenamiento de la red neuronal.

Suposición 1: Todas las funciones de activación de al menos una capa deben ser lineales, es decir, $\varphi_1(x) = x$ o $\varphi_2(x) = x$ o $\varphi_3(x) = x$ siempre y cuando la estructura neuronal se asemeje a la mostrada en la figura 2.12.

Suposición 2: Se deben inicializar los pesos sinápticos de una red neuronal $2nn - 2 - 1$ (ver ecuación 2.28) con el mismo valor grupo a grupo, es decir, $V_{b_1}(1) = V_{b_i}(1)$, $V_{a_1}(1) = V_{a_i}(1)$, $W_{b_1}(1) = W_{b_i}(1)$ y $W_{a_1}(1) = W_{a_i}(1)$ con $i = 2, 3, \dots, nn$.

Teorema 1. Considerando la red neuronal $2nn - 2 - 1$ cuyo modelo matemático se muestra en 2.28 y forma en la figura 2.12, si se cumple la suposición 1 y la suposición 2, entonces el modelo neuronal puede ser reducido a un modelo equivalente $2 - 1$ cuyo modelo matemático se muestra en 2.33 y estructura en la figura 2.13.

Demostración. Con el fin de satisfacer la suposición 1, se selecciona $\varphi_3(z) = \varphi_2(z) = z$ y $\varphi_1(z) = \tanh(z)$, entonces la ecuación 2.28 se transforma de tal manera que:

$$\begin{aligned} \hat{y}(k) &= XT \\ T &= Z_b r_b + Z_a r_a + Z_h \\ r_b &= \sum_{i=1}^{nn} V_{b_i} \tanh(J_u W_{b_i}) \\ r_a &= \sum_{i=1}^{nn} V_{a_i} \tanh(J_{\hat{y}} W_{a_i}) \end{aligned} \quad (2.35)$$

de acuerdo a la suposición 2, $V_{b_1}(1) = V_{b_i}(1)$, $V_{a_1}(1) = V_{a_i}(1)$, $W_{b_1}(1) = W_{b_i}(1)$ y $W_{a_1}(1) = W_{a_i}(1)$ con $i = 2, 3, \dots, nn$.

Ahora, utilizando el algoritmo de Levenberg-Marquardt que fue el algoritmo propuesto para realizar la optimización de los pesos sinápticos minimizando el error cuadrático medio (2.17) se tiene que:

$$W_{a_i}(k+1) = W_{a_i}(k) - [J(W_{a_i})^T J(W_{a_i}) + \mu I]^{-1} J(W_{a_i})^T e(k) \quad (2.36)$$

$$W_{b_i}(k+1) = W_{b_i}(k) - [J(W_{b_i})^T J(W_{b_i}) + \mu I]^{-1} J(W_{b_i})^T e(k) \quad (2.37)$$

$$V_{a_i}(k+1) = V_{a_i}(k) - [J(V_{a_i})^T J(V_{a_i}) + \mu I]^{-1} J(V_{a_i})^T e(k) \quad (2.38)$$

$$V_{b_i}(k+1) = V_{b_i}(k) - [J(V_{b_i})^T J(V_{b_i}) + \mu I]^{-1} J(V_{b_i})^T e(k) \quad (2.39)$$

$$Z_a(k+1) = Z_a(k) - [J(Z_a)^T J(Z_a) + \mu I]^{-1} J(Z_a)^T e(k) \quad (2.40)$$

$$Z_b(k+1) = Z_b(k) - [J(Z_b)^T J(Z_b) + \mu I]^{-1} J(Z_b)^T e(k) \quad (2.41)$$

$$Z_h(k+1) = Z_h(k) - [J(Z_h)^T J(Z_h) + \mu I]^{-1} J(Z_h)^T e(k) \quad (2.42)$$

$$X(k+1) = X(k) - [J(X)^T J(X) + \mu I]^{-1} J(X)^T e(k) \quad (2.43)$$

donde:

$$\begin{aligned} J(W_{a_i}) &= -e(k) X Z_a V_{a_i} \operatorname{sech}^2(J_{\hat{y}} W_{a_i}) J_{\hat{y}} \\ J(W_{b_i}) &= -e(k) X Z_b V_{b_i} \operatorname{sech}^2(J_u W_{b_i}) J_u \\ J(V_{a_i}) &= -e(k) X Z_a \tanh(J_{\hat{y}} W_{a_i}) \\ J(V_{b_i}) &= -e(k) X Z_b \tanh(J_u W_{b_i}) \\ J(Z_a) &= -e(k) X r_a \\ J(Z_b) &= -e(k) X r_b \\ J(Z_h) &= -e(k) X \\ J(X) &= -e(k) T \end{aligned}$$

e $i = 1, 2, \dots, nn$.

Una vez que el modelo neuronal dado por 2.35 es entrenado bajo la suposición uno y dos se tiene que:

$$\begin{aligned} \hat{y}(k) &= \tilde{X} T \\ T &= \tilde{Z}_b r_b + \tilde{Z}_a r_a + \tilde{Z}_h \\ r_b &= \sum_{i=1}^{nn} \tilde{V}_{b_i} \tanh(J_u \tilde{W}_{b_i}) \\ r_a &= \sum_{i=1}^{nn} \tilde{V}_{a_i} \tanh(J_{\hat{y}} \tilde{W}_{a_i}). \end{aligned} \quad (2.44)$$

La notación \tilde{X} quiere decir que el pesos sináptico X ha sido entrenado. Debido a que los pesos sinápticos fueron inicializados según la suposición dos y fueron en-

trenados bajo el mismo algoritmo de optimización, los pesos sinápticos finales son: $\tilde{V}_{b_1} = \tilde{V}_{b_i}$, $\tilde{V}_{a_1} = \tilde{V}_{a_i}$, $\tilde{W}_{b_1} = \tilde{W}_{b_i}$ y $\tilde{W}_{a_1} = \tilde{W}_{a_i}$ con $i = 2, 3, \dots, nn$. Ahora, sustituyendo r_a y r_b en T , y T en $\hat{y}(k)$, se tiene que:

$$\hat{y}(k) = \sum_{i=1}^{nn} \tilde{V}_{b_i} \tilde{X} \tilde{Z}_b \tanh(J_u \tilde{W}_{b_i}) + \sum_{i=1}^{nn} \tilde{V}_{a_i} \tilde{X} \tilde{Z}_a \tanh(J_{\hat{y}} \tilde{W}_{a_i}) + \tilde{X} \tilde{Z}_h \quad (2.45)$$

Dada la ecuación 2.45 se puede realizar cambio de variable, de tal manera que:

$$\begin{aligned} \tilde{H} &= \tilde{X} \tilde{Z}_h \\ \tilde{V}_{B_i} &= \tilde{V}_{b_i} \tilde{X} \tilde{Z}_b \\ \tilde{V}_{A_i} &= \tilde{V}_{a_i} \tilde{X} \tilde{Z}_a \end{aligned}$$

donde $\tilde{V}_{B_1} = \tilde{V}_{B_i}$ y $\tilde{V}_{A_1} = \tilde{V}_{A_i}$ ($i = 2, 3, \dots, nn$) aplicando nuevamente la suposición 2.

Como se puede observar en la ecuación 2.45, la red neuronal inicial ($2nn - 2 - 1$) es reducida a una red neuronal de dos capas. Es importante mencionar que el modelo siempre es el mismo, solamente se utilizan herramientas algebraicas para reducir el modelo neuronal. Después de haber realizado el cambio de variables, el modelo (2.45) puede ser descrito como:

$$\hat{y}(k) = \sum_{i=1}^{nn} \tilde{V}_{B_i} \tanh(J_u \tilde{W}_{b_i}) + \sum_{i=1}^{nn} \tilde{V}_{A_i} \tanh(J_{\hat{y}} \tilde{W}_{a_i}) + \tilde{H} \quad (2.46)$$

Debido a que la suposición 2, establece que los pesos sinápticos deben ser iguales por lotes o grupos y que se entrenaron bajo el mismo algoritmo de adaptación, entonces $\tilde{V}_{B_1} = \tilde{V}_{B_i}$, $\tilde{V}_{A_1} = \tilde{V}_{A_i}$, $\tilde{W}_{a_1} = \tilde{W}_{a_i}$ y $\tilde{W}_{b_1} = \tilde{W}_{b_i}$ ($i = 2, 3, \dots, nn$) y, por ende, la siguiente operación algebraica es válida:

$$\begin{aligned} \sum_{i=1}^{nn} \tilde{V}_{B_1} \tanh(J_u \tilde{W}_{b_i}) &= \tilde{V}_B \tanh(J_u \tilde{W}_B) \\ \sum_{i=1}^{nn} \tilde{V}_{A_1} \tanh(J_{\hat{y}} \tilde{W}_{a_i}) &= \tilde{V}_A \tanh(J_{\hat{y}} \tilde{W}_A) \end{aligned}$$

donde:

$$\begin{aligned}\tilde{V}_B &= nn \times \tilde{V}_{B_1} \\ \tilde{V}_A &= nn \times \tilde{V}_{A_1} \\ \tilde{W}_B &= \tilde{W}_{b_1} \\ \tilde{W}_A &= \tilde{W}_{a_1}\end{aligned}$$

Al sustituir las operaciones algebraicas anteriores en la ecuación (2.46), resulta el modelo esperado mostrado en la ecuación (2.47).

$$\hat{y}(k) = \tilde{V}_B \tanh(J_u \tilde{W}_B) + \tilde{V}_A \tanh(J_{\hat{y}} \tilde{W}_A) + \tilde{H} \quad (2.47)$$

En conclusión, se puede observar que la ecuación correspondiente al modelo neuronal de la figura 2.12 y ecuación (2.28) ha sido reducido a un modelo neuronal de dos capas; con dos neuronas en la capa de entrada y una sola neurona en la capa de salida al igual que el modelo neuronal 2 – 1 mostrado en la figura 2.13. Además, el modelo correspondiente a la figura 2.13 (ver ecuación 2.33) es similar al modelo resultante después de la reducción de capas y neuronas (ver ecuación 2.47). Dado el modelo obtenido se puede concluir que la demostración ha sido realizada.

□

Capítulo 3

ESTIMADOR M DE HUBER

Las redes neuronales son una herramienta multiusos. Para este trabajo de tesis se utilizaron para realizar la identificación de sistemas. En el capítulo dos se habló de como realizar la identificación de sistemas mediante el uso de redes neuronales, también se describieron dos aspectos importantes para realizar la identificación; una de ellas es la selección de la estructura y la segunda es la adaptación de los pesos sinápticos. Comúnmente, la adaptación de los pesos sinápticos se realiza mediante la minimización del estimador L_2 , es eficiente en muchos de los casos pero, cuando se trabaja con datos corrompidos, contaminados, con efectos unívocos llamados “outliers” o valores atípicos, la estimación mediante L_2 se vuelve ineficiente debido a que los residuos creados por la comparación de las observaciones y de la estructura estimada tienden a ser numéricamente grandes, esto debido a que la estimación con el uso de L_2 discrimina el efecto o comportamiento del outlier.

Generalmente hay dos formas de realizar la estimación con valores atípicos: una de ellas consiste en remover o filtrar los valores atípicos del conjunto de datos utilizados para la identificación antes de realizar la estimación paramétrica [17, 19, 28, 40]. La segunda manera es utilizar el estimador M de Huber que es la minimización de una función compuesta por el estimador L_1 y el estimador L_2 [20–24, 58].

3.1. “Outliers” o valores atípicos

Un valor atípico es una observación numéricamente distante de un conjunto de observaciones [29]. Los outliers pueden ocurrir en cualquier distribución, pero a menudo se deben a que hay errores en la medición o simplemente así es como se comporta el experimento. Para hacerle frente al comportamiento a los valores atípicos comúnmente se desprenden del conjunto de datos (filtran) o se usan herramientas estadísticas robustas (función de Huber) [1].

3.1.1. Tipos de “outliers”

Los valores atípicos pueden ser clasificados en tres clases: outliers puntuales, outliers contextuales y outliers colectivos .

Outliers puntuales

Los outliers puntuales son observaciones cuyo comportamiento o valor puede ser caracterizado como anómalo con respecto a un conjunto de datos. Este es el tipo más simple de outlier y el que con frecuencia sucede [14, 15, 30]. Por ejemplo, en la figura 3.1, los puntos $O1$ y $O2$ son valores atípicos puntuales debido a que son significativamente diferentes al conjunto de datos normales $G1$ y $G2$.

Outliers contextuales

Si un dato es anómalo en un contexto específico, entonces se define como un outlier contextual. El contexto es referido al tipo de valor atípico que se refiere. La noción de contexto es referida por la estructura del conjunto de datos y tiene que ser especificada como una parte de la formulación del problema. Cada dato es definido con dos atributos:

- Atributos contextuales: Los cuales son usados para determinar un contexto (o vecindad) para una instancia específica; por ejemplo, conjunto de datos (t y $f(t)$), o longitud y latitud, etc.

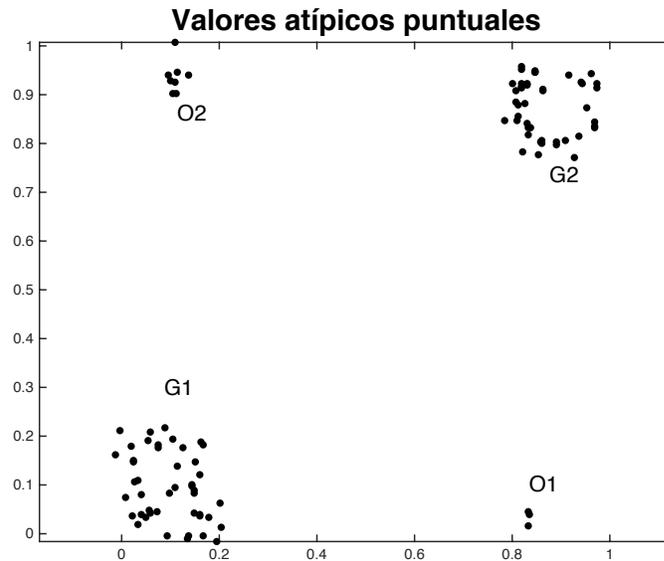


Figura 3.1: Ejemplo de valores atípicos puntuales. Puntos $O1$ y $O2$ difieren significativamente de las regiones $G1$ y $G2$.

- Atributos de conducta: Se usan para definir otras características de datos, específico al problema entre manos (al problema en cuestión); por ejemplo, número de ventas en un lugar específico.

Los outliers contextuales son detectados mediante datos correspondientes a su comportamiento en un contexto específico. Por lo tanto, un dato puede ser considerado un outlier en un contexto, pero puede ser considerado normal en un contexto diferente. En la figura 3.2 se puede observar un ejemplo de outliers contextuales donde se puede apreciar el comportamiento de una señal respiratoria [1].

Outliers colectivos

Si un subconjunto de datos son anómalos con respecto al conjunto entero de datos, este subconjunto se le define como outlier colectivo. Los datos atípicos individuales no pueden definirse colectivos por sí mismos, pero si se presentan de manera colectiva, es decir, un subconjunto de datos anómalos en un conjunto de datos sí. Los outliers colectivos pueden ocurrir solo en conjunto de datos los cuales se encuentran

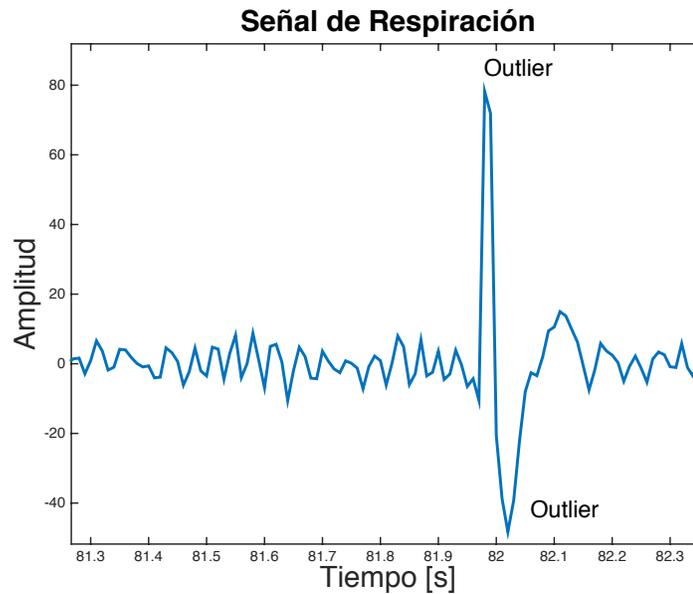


Figura 3.2: Ejemplo de outlier contextual. Es un conjunto de datos mostrados en serie de tiempo donde los outliers tienen una amplitud distante a los demás valores de amplitud.

relacionados. Por ejemplo, en la figura 3.3 muestra el comportamiento de un diodo túnel; se puede observar que existen un grupo de datos superiores e inferiores a la media.

3.1.2. Causas

Los outliers pueden aparecer debido a diferentes causas anómalas. Un aparato para la captura de mediciones puede sufrir un descompuesto, errores en la transmisión o transcripción de datos. Los valores atípicos pueden suceder en el conjunto de datos de manera abrupta como un comportamiento fraudulento, errores humanos, error del instrumento o simplemente a través de desviaciones en las ocurridas de manera natural. También puede ser que una muestra haya sido contaminada por elementos fuera de la población que se está examinando.

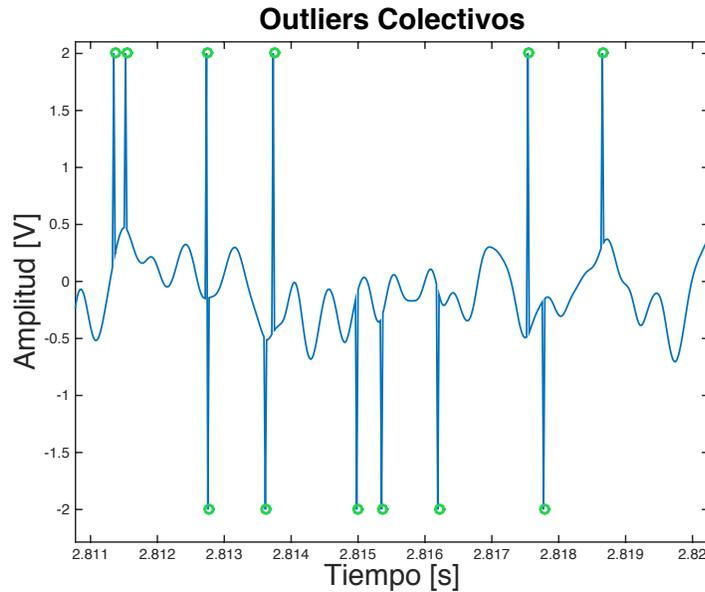


Figura 3.3: Ejemplo de outlier colectivo (puntos en verde).

3.1.3. Detección

No existe un método matemático que pueda definir cuándo un dato se comporta de manera “normal” y cuándo no lo está haciendo. La detección de outliers es un estudio vasto y existen distintas maneras de realizar la detección [11, 14, 15, 30, 61]. Algunos de los métodos son gráficos mediante gráficas probabilísticas de distribución normal. Otros son los basados en modelo; mediante identificación de sistemas pero con datos de distribución normal e identificando qué dato es no deseado con base en promedio y la desviación estándar.

Como se mencionó con anterioridad, la estadística proporciona criterios para detectar estos valores atípicos. Uno de los métodos más utilizados es el que utiliza el concepto de cuartil de un conjunto de datos; se tiene un conjunto de datos, este debe ser ordenado de menor a mayor y el cuartil uno (Q_1) es el valor tal que desde el primer valor hacia su izquierda se encuentra la primera cuarta parte de los valores del conjunto de datos. El cuartil dos (Q_2) es el valor tal que desde ese valor hacia su izquierda se encuentran la primera mitad de los valores de este conjunto de datos. Y finalmente, el cuartil tres (Q_3) es el valor tal que desde ese valor hacia su izquierda se encuentra las tres cuartas partes del conjunto de datos. Los outliers pueden ser

clasificados como moderados o extremos; para identificar si un outlier es moderado o extremo pueden aplicarse las expresiones en (3.1) y (3.2) respectivamente.

Detección de outliers moderados:

$$\begin{aligned}\bar{\Psi}_m &= Q_3 + 1.5(Q_3 - Q_1) \\ \underline{\Psi}_m &= Q_1 - 1.5(Q_3 - Q_1)\end{aligned}\tag{3.1}$$

Los valores que sean menores que $\underline{\Psi}_m$ o mayores que $\bar{\Psi}_m$ se consideran valores outliers.

Detección de outliers extremos:

$$\begin{aligned}\bar{\Psi}_e &= Q_3 + 3(Q_3 - Q_1) \\ \underline{\Psi}_e &= Q_1 - 3(Q_3 - Q_1)\end{aligned}\tag{3.2}$$

Los valores que sean menores que $\underline{\Psi}_e$ o mayores que $\bar{\Psi}_e$ se consideran valores outliers.

3.2. Estimadores

Los estimadores son la columna vertebral en la identificación de sistemas. Definiendo a $e(k)$ como un valor residual en el k -ésimo instante de tiempo, es decir, la diferencia entre la observación k -ésima y el valor estimado ($\hat{y}(k)$); el método de mínimos cuadrados por ejemplo minimiza $1/N \sum_{k=1}^N e(k)^2$, el cual es inestable si los datos se encuentran contaminados por outliers debido a que los datos atípicos dan un efecto de gran valor numérico ($e(k)$ tiende a un valor grande) en la minimización y por ende los parámetros estimados son distorsionados.

Los estimadores M son una amplia clase de estimadores, los cuales se obtienen mediante la suma mínima de funciones dependientes de residuales. Los estimadores M robustos (EMR) tratan de reducir el efecto de los outliers en la estimación reemplazando el cuadrado de los residuales ($e(k)^2$ en la estimación) por otra expresión que se encuentra en función de los residuales tal como se muestra en 3.3.

$$\text{mín} \sum_{k=1}^N \rho_\gamma(e(k)),\tag{3.3}$$

donde ρ es una función simétrica, definida positiva con un mínimo global en cero,

y es elegida de tal manera que reduzca el incremento cuadrático.

Ahora, definiendo un vector de parámetros a estimar $\hat{\mathbf{w}} = [w_1 \ \cdots \ w_n]^T$. El EMR de $\hat{\mathbf{w}}$ basado en la función $\rho_\gamma(e(k))$ es el vector $\hat{\mathbf{w}}$ el cual es la solución de las siguientes n ecuaciones:

$$\sum_{k=1}^N \psi(e(k)) \frac{\partial e(k)}{\partial w_i} = 0, \quad i = 1, \dots, n, \quad (3.4)$$

donde la derivada parcial $\frac{\partial \rho_\gamma(x)}{\partial x} = \psi(x)$ es llamada *función de influencia*. Ahora, si se define una *función peso* como en (3.5)

$$\Phi(x) = \frac{\psi(x)}{x}, \quad (3.5)$$

entonces la ecuación 3.4 se puede describir como la ecuación 3.6

$$\sum_{k=1}^N \Phi(e(k)) e(k) \frac{\partial e(k)}{\partial w_i} = 0, \quad i = 1, \dots, n. \quad (3.6)$$

La *función de influencia* ψ mide la influencia de un dato en función del parámetro estimado. Por ejemplo, para mínimos cuadrados con $\rho_\gamma(e(k)) = \frac{e(k)^2}{2}$, la función influencia $\psi(e(k)) = e(k)$, se puede ver cómo el error crece linealmente con respecto al tamaño del residuo, confirmando que no se encuentra acotado, es decir, si $e(k) \rightarrow \infty \Rightarrow \rho_\gamma(e(k)) \rightarrow \infty$ así que no existe ninguna robustez en la estimación con mínimos cuadrados. Cuando un estimador es robusto, puede ser inferido tal que cualquier dato es insuficiente para producir un valor constante significante. Existen diversas características que un EMR debe cumplir:

- El estimador debe tener una función influencia limitada. Es decir, que a partir de cierto valor la función comienza a converger.
- La función objetivo (ρ_γ) del vector de parámetros $\hat{\mathbf{w}}$ debe tener un mínimo único [73]. Esto requiere que la función ρ_γ es convexa con la variable $\hat{\mathbf{w}}$. Esto es necesario debido a que no es suficiente con definir que la función ρ_γ tenga un mínimo único. La restricción de convexidad es equivalente a imponer que $\frac{\partial^2 \rho_\gamma(\cdot)}{\partial \hat{\mathbf{w}}^2}$ es positivo definido.

- Siempre que $\frac{\partial^2 \rho_\gamma(\cdot)}{\partial \hat{\mathbf{w}}^2}$ sea singular, la función objetivo debe tener una gradiente, es decir, $\frac{\partial \rho_\gamma(\cdot)}{\partial \hat{\mathbf{w}}} \neq \mathbf{0}$. Esto permite por lo menos realizar una aproximación de $\frac{\partial^2 \rho_\gamma(\cdot)}{\partial \hat{\mathbf{w}}^2}$.

Tabla 3.1: Estimadores comúnmente utilizados.

Tipo	$\rho_\gamma(x)$	$\psi(x)$	$\Phi(x)$
L_2	$\frac{x^2}{2}$	x	1
L_1	$ x $	$\text{sign}(x)$	$\frac{1}{ x }$
L_p	$\frac{ x ^v}{v}$	$\text{sign}(x) x ^{v-1}$	$ x ^{v-2}$
Huber ($L_1 - L_2$)	$\begin{cases} \frac{x^2}{2} & x \leq \gamma \\ \gamma \left(x - \frac{\gamma}{2} \right) & x \geq \gamma \end{cases}$	$\begin{cases} x & x \leq \gamma \\ \gamma \text{sign}(x) & x \geq \gamma \end{cases}$	$\begin{cases} \frac{1}{ x } & x \leq \gamma \\ \frac{\gamma}{ x } & x \geq \gamma \end{cases}$
Cauchy	$\frac{c^2}{2} \log(1 + (x/c)^2)$	$\frac{x}{1 + (x/c)^2}$	$\frac{1}{1 + (x/c)^2}$
Gema y McClure	$\frac{x^2/2}{1 + x^2}$	$\frac{x}{(1 + x^2)^2}$	$\frac{1}{(1 + x^2)^2}$
Welsch	$\frac{c^2}{2} (1 - e^{-(x/c)^2})$	$xe^{-(x/c)^2}$	$e^{-(x/c)^2}$
Tukey	$\begin{cases} \frac{c^2}{2} (1 - (1 - (x/c)^2)^3) & x \leq c \\ \frac{c^2}{6} & x \geq c \end{cases}$	$\begin{cases} x(1 - (x/c)^2)^2 & x \leq c \\ 0 & x \geq c \end{cases}$	$\begin{cases} (1 - (x/c)^2)^2 & x \leq c \\ 0 & x \geq c \end{cases}$

En la tabla 3.1 se puede observar una lista de varias funciones ρ , la función de influencia y la función de peso. En la figura 3.4 y 3.5 se muestra gráficamente el comportamiento de cada una de estas funciones. Nótese que no todas las funciones cumplen satisfactoriamente con los requerimientos anteriores.

Los estimadores de mínimos cuadrados (L_2) no son robustos debido a que su función de influencia no está delimitada.

Los estimadores de mínimos absolutos (L_1) no son estables porque la función ρ no es estrictamente convexa. En realidad, la segunda derivada cuando $x = 0$ no

existe, otorgando un valor indefinido. Los estimadores absolutos reducen la influencia de errores numéricamente grandes, pero aún así estos errores tienen influencia en el comportamiento de la función ρ porque esta última carece de un punto de corte.

Los estimadores de mínimos “p” o mínima potencia son una familia de funciones. Si $v = 2$ entonces se realizaría la estimación con mínimos cuadrados (L_2) y si $v = 1$ entonces se realizaría con mínimos absolutos. Entre más pequeño sea el valor de v el estimador será más robusto. La selección de un valor óptimo para v ha sido investigado, y el valor de v debe encontrarse al rededor de 1.2 para esperar una estimación efectiva [53, 73].

La función de Cauchy, también llamada función Lorenziana, no garantiza una solución única. Para cubrir el 95% de los datos con una distribución normal y con ruido Gaussiano se debe definir que $c = 2.3849$, en el caso de la función de Welsch $c = 2.9846$ y con la función de Tukey $c = 4.6851$.

3.3. Estimador M de Huber ($L_1 - L_2$)

Con anterioridad se ha definido lo que es un estimador M (ver ecuación 3.3). También se habló de las restricciones que la función $\rho_\gamma(\cdot)$ debe cumplir. El estimador M de Huber cumple con las características siguientes:

- La función $\rho_\gamma(\cdot)$ es derivable, es decir, existe $\psi(\cdot)$.
- La función $\rho_\gamma(\cdot)$ es simétrica o par, es decir, $\rho_\gamma(x) = \rho_\gamma(-x)$.
- La función $\rho_\gamma(\cdot)$ es convexa, es decir, es creciente.
- Se cumple que $\rho_\gamma(0) = 0$.

En la tabla 3.1 se mostró un ejemplo de la función de Huber. Para este tema de tesis se utilizó la función expresada en (2.18) la cual en seguida se puede encontrar expresada nuevamente:

$$\rho_\gamma(\hat{w}, u^N, y^N, e) = \begin{cases} \frac{1}{2}e^2(k, \hat{w}) & \text{para } |e(k, \hat{w})| \leq \gamma \\ \gamma|e(k, \hat{w})| - \frac{1}{2}\gamma^2 & \text{para } |e(k, \hat{w})| > \gamma \end{cases}$$

donde, \hat{w} son pesos sinápticos (parámetros de las redes neuronales), u^N es una señal de entrada, y^N es la señal de salida del sistema, $e(k)$ son los residuos o señal de error y γ es el factor de Huber compuesto por el producto de dos factores κ y $\sqrt{\lambda_N^e}$; κ es un factor escalar de suma importancia para lograr la estimación mediante la función de Huber y λ_N^e es la varianza obtenida mediante el uso de los residuos generados en la estimación utilizando solamente el estimador L_2 calculada por

$$\lambda_N^e = \frac{\sum_{k=1}^N (e(k) - \bar{e}(k))^2}{N - 1} \quad (3.7)$$

Se sabe que existe una función $\psi(e(k, \hat{w}))$ de tal manera que cumpla con la expresión mostrada en (3.4), la forma para expresar a la función de influencia $\psi(e(k, \hat{w}))$ se puede observar en la igualdad (3.8).

$$\psi(e(k, \hat{w})) = \frac{\partial \rho_\gamma(e(k, \hat{w}))}{\partial e(k, \hat{w})} = \begin{cases} e(k, \hat{w}), & \text{si } |e(k, \hat{w})| \leq \gamma \\ \gamma, & \text{si } e(k, \hat{w}) > \gamma \\ -\gamma, & \text{si } e(k, \hat{w}) < \gamma \end{cases} \quad (3.8)$$

De tal manera, si se desea entrenar un modelo neuronal como el mostrado en la ecuación (2.28) mediante el uso del algoritmo de Levenberg-Marquardt [46] se tendría que cada peso sináptico se entrena tal como se muestra en (2.36) a la (2.43). Utilizando el estimador M de Huber se tendrían tres casos distintos para realizar la optimización; cada caso repercute en las jacobianas utilizadas para obtener los valores adecuados para la estimación.

Caso 1: Si $|e(k, \hat{w})| \leq \gamma$, el cual es equivalente a utilizar el clásico error al cuadrado o L_2 se tiene que:

$$J(W_{a_i}) = -e(k)X Z_a V_{a_i} \text{sech}^2(J_{\hat{y}} W_{a_i}) J_{\hat{y}} \quad (3.9)$$

$$J(W_{b_i}) = -e(k)X Z_b V_{b_i} \text{sech}^2(J_u W_{b_i}) J_u \quad (3.10)$$

$$J(V_{a_i}) = -e(k)X Z_a \tanh(J_{\hat{y}} W_{a_i}) \quad (3.11)$$

$$J(V_{b_i}) = -e(k)X Z_b \tanh(J_u W_{b_i}) \quad (3.12)$$

$$J(Z_a) = -e(k)Xr_a \quad (3.13)$$

$$J(Z_b) = -e(k)Xr_b \quad (3.14)$$

$$J(Z_h) = -e(k)X \quad (3.15)$$

$$J(X) = -e(k)T \quad (3.16)$$

Caso 2: Si $e(k, \hat{w}) > \gamma$

$$J(W_{a_i}) = -\gamma X Z_a V_{a_i} \operatorname{sech}^2(J_{\hat{y}} W_{a_i}) J_{\hat{y}} \quad (3.17)$$

$$J(W_{b_i}) = -\gamma X Z_b V_{b_i} \operatorname{sech}^2(J_u W_{b_i}) J_u \quad (3.18)$$

$$J(V_{a_i}) = -\gamma X Z_a \tanh(J_{\hat{y}} W_{a_i}) \quad (3.19)$$

$$J(V_{b_i}) = -\gamma X Z_b \tanh(J_u W_{b_i}) \quad (3.20)$$

$$J(Z_a) = -\gamma X r_a \quad (3.21)$$

$$J(Z_b) = -\gamma X r_b \quad (3.22)$$

$$J(Z_h) = -\gamma X \quad (3.23)$$

$$J(X) = -\gamma T \quad (3.24)$$

Caso 3: Si $e(k, \hat{w}) < -\gamma$

$$J(W_{a_i}) = \gamma X Z_a V_{a_i} \operatorname{sech}^2(J_{\hat{y}} W_{a_i}) J_{\hat{y}} \quad (3.25)$$

$$J(W_{b_i}) = \gamma X Z_b V_{b_i} \operatorname{sech}^2(J_u W_{b_i}) J_u \quad (3.26)$$

$$J(V_{a_i}) = \gamma X Z_a \tanh(J_{\hat{y}} W_{a_i}) \quad (3.27)$$

$$J(V_{b_i}) = \gamma X Z_b \tanh(J_u W_{b_i}) \quad (3.28)$$

$$J(Z_a) = \gamma X r_a \quad (3.29)$$

$$J(Z_b) = \gamma X r_b \quad (3.30)$$

$$J(Z_h) = \gamma X \quad (3.31)$$

$$J(X) = \gamma T \quad (3.32)$$

Cómo se puede observar, el estimador M de Huber hace que la optimización de los pesos sinápticos no diverja por los valores de amplitud que podría arrojar la señal de error ante un valor atípico (ver ecuaciones (3.17)-(3.32)) [58]. En el siguiente capítulo se propone una metodología para realizar la estimación de sistemas con valores atípicos, también se profundiza en gran medida con respecto a la selección del valor de κ para utilizar la función de Huber.

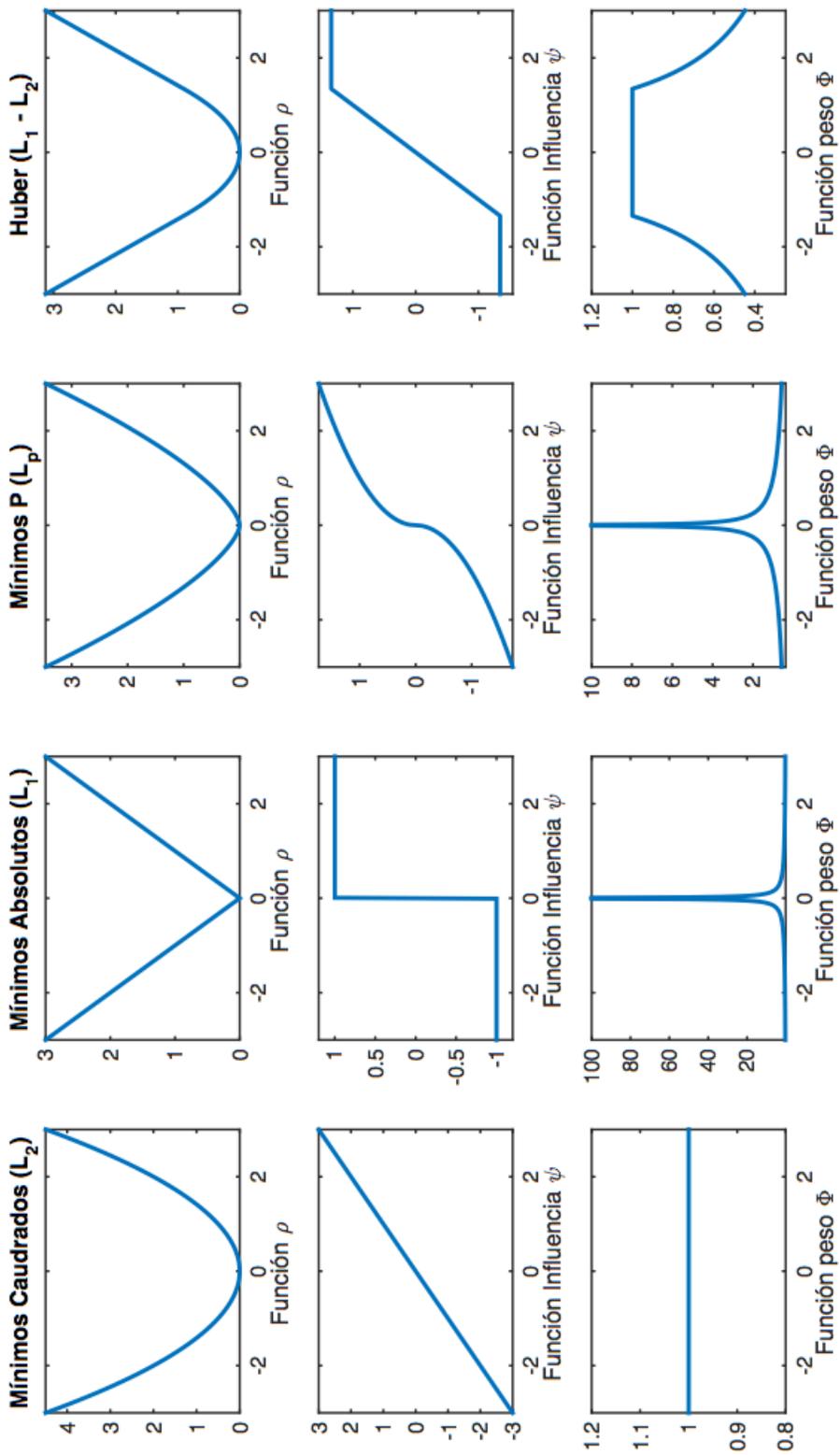


Figura 3.4: Gráficas de estimadores M.

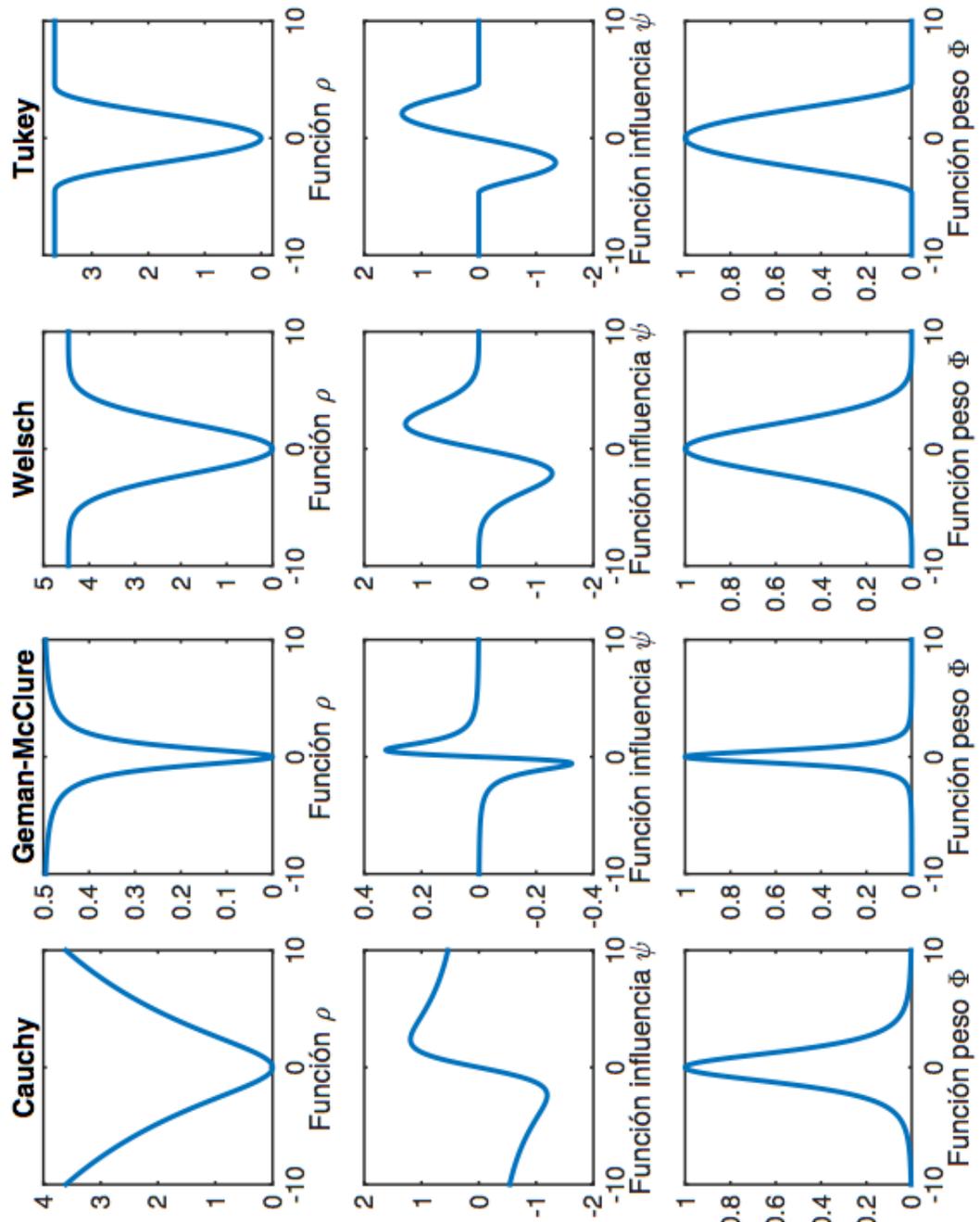


Figura 3.5: Gráficas de estimadores M.

Capítulo 4

Metodología para la identificación de sistemas mediante redes neuronales y el estimador M de Huber

En los capítulos dos y tres se explicaron respectivamente las redes neuronales que se utilizaron para realizar la identificación de sistemas y el estimador robusto M de Huber para introducir el efecto de los valores atípicos en la estimación.

Para calcular el valor de los órdenes (n_a , n_b y n_k) que la red neuronal necesita para realizar la estimación se realizan q número de modelos. El valor de q depende del número de iteraciones para cada valor de n_a , n_b y n_k . El proceso de selección de órdenes entregará un valor de n_a , n_b y n_k subóptimo y se seleccionarán aquellos valores que con ayuda de índices de desempeño puedan realizar la mejor estimación del sistema.

La función de Huber está compuesta por un parámetro desconocido κ ; en este capítulo se hablará del proceso de selección de κ que es similar al proceso de selección de ordenes.

4.1. Metodología para la identificación de sistemas con redes neuronales y el estimador M de Huber

Se propone una metodología para realizar la identificación de sistemas con redes neuronales y el estimador M de Huber.

1. *Obtención del conjunto de datos entrada-salida:* Para ello se debe excitar o alimentar al sistema mediante la aplicación de una señal de entrada y registrar el comportamiento de la entrada y la salida durante un intervalo de tiempo antes definido.
2. *Observación de los datos:* Para utilizar el estimador M de Huber primero se debe verificar que los datos que han sido capturados en el paso anterior contienen valores atípicos. Si no es así, se puede utilizar el estimador L_2 (ver tabla 3.1) que comúnmente se utiliza.
3. *Selección de neuronas y funciones de activación:* Definir las funciones de activación de la red neuronal $2nn - 2 - 1$ respetando la suposición uno y dos para realizar el proceso de reducción de la red neuronal. Entre las posibles combinaciones de funciones de activación se pueden encontrar las que se muestran en las ecuaciones 2.29 - 2.32. Para concluir con este paso, se debe definir el número de neuronas encargadas del procesamiento de la señal de entrada y la señal de salida.
4. *Encontrar el orden de la red neuronal:* Este paso consiste en encontrar el número de regresores en la señal de entrada nb y el número de regresores en la señal de salida na . También debe ser seleccionado el número de atrasos naturales en la señal de entrada nk . En la sección 4.3 se explica más detalladamente cómo es que se realiza el cálculo de los órdenes de una red neuronal.
5. *Entrenamiento de la RNA:* Si los datos no se encuentran contaminados por valores atípicos o que estos no aparecen de manera natural, entonces se puede realizar el entrenamiento normal y utilizando el estimador L_2 . Si los datos

se encuentran con valores atípicos, entonces es necesario utilizar la función de Huber, de la cual, se deben encontrar dos parámetros de suma importancia; uno de ellos es la raíz cuadrada de la varianza del error, esta debe obtenerse al haber realizado la estimación con el estimador L_2 . La explicación para encontrar los parámetros de la función de Huber se encuentran en la subsección 4.4.

6. *Validación:* Consiste en evaluar diversos estándares estadísticos con el fin de comparar el desempeño que la red neuronal está ofreciendo en función de la señal de error. Cabe destacar que el error es calculado mediante la ecuación 2.19. Los estándares estadísticos son llamados índices de desempeño, los cuales se pueden ver en la sección 4.2. También se puede validar el modelo neuronal comparando la respuesta en frecuencia de la salida real y la salida de la RNA.

Si modelo neuronal no es el adecuado se debe a que el conjunto de datos de entrada-salida no proporciona la suficiente información del sistema, el algoritmo de optimización diverge o es necesario aumentar el número de épocas o ciclos.

4.2. Índices de desempeño

Los índices de desempeño son estándares estadísticos o funciones cuya principal función es la de indicar numéricamente el desempeño de un sistema. También tienen una gran utilidad en la comparación de respuestas de sistemas, es decir, comparando la señal de salida real y la señal de salida estimada, para este caso, una red neuronal. Si el indicador arroja un valor numéricamente pequeño quiere decir que la estimación se está realizando de manera apropiada ya que la señal de salida real es similar a la señal generada por la red neuronal. A continuación se muestran algunos de los índices de desempeño.

4.2.1. Media del error cuadrático

El MSE (por sus siglas en ingles: mean square error) es un indicador de la norma dos del error. La forma para calcular el MSE se encuentra expresado en la ecuación 4.1.

$$MSE = \frac{1}{N} \sum_{k=1}^N e^2(k) \quad (4.1)$$

donde N es el número de muestras tomadas, $e(k)$ es el error de estimación calculado mediante la ecuación (2.19).

4.2.2. Media del error

Calcula la media o promedio del error. Entre más cerca del cero se encuentre la media del error mejor será la estimación. En la ecuación 4.2 se puede observar cómo se calcula la media del error.

$$\bar{e} = \frac{1}{N} \sum_{i=1}^N e_i \quad (4.2)$$

4.2.3. Desviación estándar

La desviación estándar es una medida de dispersión que permite conocer que tanto se aleja la media del parámetro de cada uno de los demás datos. La desviación estándar para datos normalmente distribuidos se muestra en la ecuación 4.3.

$$\sigma_N^e = \sqrt{\frac{\sum_{i=1}^N (e_i - \bar{e})^2}{N - 1}} \quad (4.3)$$

4.2.4. FIT

El FIT es una medida de parentesco entre dos señales. El valor del FIT es arrojado en porcentaje y se calcula mediante la siguiente igualdad:

$$FIT = 100 \left(1 - \frac{\|\hat{y} - y_r\|}{\|y_r - \bar{y}_r\|} \right) \quad (4.4)$$

donde \hat{y} es la señal estimada o la señal de salida de la red neuronal, y_r es la señal de salida real y \bar{y}_r es la media de la señal de salida real.

En algunos casos se puede encontrar una medida semejante al FIT en el dominio de la frecuencia. Esta técnica no es de lo más correcta pero puede utilizarse siempre y cuando se comparen las magnitudes frecuenciales de la señal de salida real y la señal de salida estimada.

4.2.5. Integral del error absoluto

Pondera los errores hacia el mismo sentido, es decir, no importa si los valores del error son negativos; el IAE (por sus siglas en ingles Integral Absoulte Error) suma la magnitud del error sin importar el sentido.

$$\begin{aligned} IAE &= \int_0^{\infty} |e(t)| dt \\ IAE &= \sum_{k=0}^N |e(k)| \end{aligned} \quad (4.5)$$

4.2.6. Integral del error al cuadrado

La principal característica del ISE (por sus siglas en ingles Integrated Square Error) es que para errores con una magnitud numéricamente grande el resultado será del mismo modo al igual que con errores cuya magnitud es numéricamente pequeña.

$$\begin{aligned} ISE &= \int_0^{\infty} e(t)^2 dt \\ ISE &= \sum_{k=0}^N e(k)^2 \end{aligned} \quad (4.6)$$

4.3. Cálculo del orden de una RNA

El cálculo del orden de la red neuronal es un paso fundamental para realizar una estimación eficiente. Se propone la siguiente metodología para encontrar el valor de los ordenes:

Seleccionar para cada orden (na , nb y nk) un límite superior $\bar{\beta}$ cuyo valor sea entero positivo y un valor inferior $\underline{\beta}$ cuyo valor sea entero positivo. Es decir, $\underline{\beta} \leq na \leq \bar{\beta}$,

$$\underline{\beta} \leq nb \leq \bar{\beta} \text{ y } \underline{\beta} \leq nk \leq \bar{\beta}.$$

Se comienza a cambiar el valor de na de $\underline{\beta}$ a $\bar{\beta}$ en intervalos de una sola unidad considerando a nb y nk con su valor inferior en el comienzo de la búsqueda. Después de que $na > \bar{\beta}$ ha generado $\bar{\beta} - \underline{\beta}$ modelos diferentes ya que cada valor diferente de na , nb o nk corresponde a un modelo diferente. Después de que $na > \bar{\beta}$ cambia el valor de nb una unidad más hasta llegar a $\bar{\beta}$; en cuanto $nb > \bar{\beta}$, el valor de nk aumenta una unidad y así sucesivamente.

Cabe mencionar que el rango para nk puede ser diferente al rango de na y nb ; se puede considerar al rango de nk más pequeño que el rango de na y nb .

Por cada iteración realizada se calcula cualquiera de los índices de desempeño que se quiera mejorar. Es importante considerar que si los tres ordenes tienen el mismo rango se tendrían $(\bar{\beta} - \underline{\beta})^3$ modelos distintos.

En la figura 4.1 se puede apreciar la forma en que se analizó el comportamiento de uno de los índices de desempeño conforme cambiaba el valor de na y nb considerando que $nk = 4$.

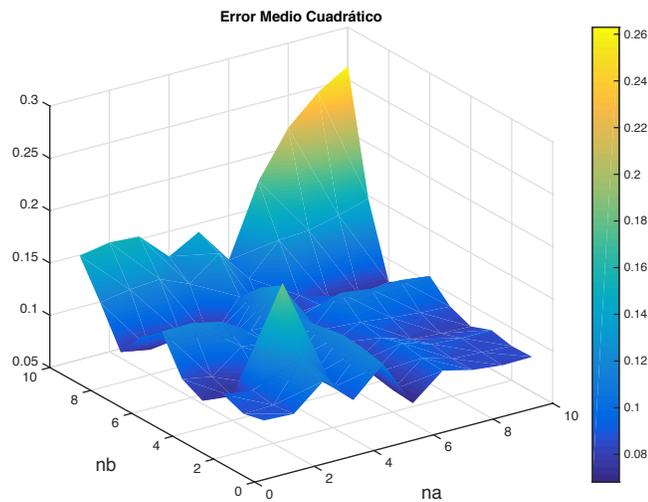


Figura 4.1: Comportamiento de la media del error al cuadrado cambiando los valores de na , nb y considerando que $nk = 4$.

En la figura 4.1 se puede observar que entre el mejor valor para na podría ser

cuatro o seis y que justamente en esos valores nb podría tomar el valor de siete. Con el fin de no generar un gran número de retardos se optó por seleccionar los valores de $nk = 4$, $na = 4$ y $nb = 7$ para este experimento. También se pueden analizar otros comportamientos, por ejemplo en la figura 4.2 se muestra el FIT temporal que resultó del cambio constante de na y nb , manteniendo el valor de nk fijo.

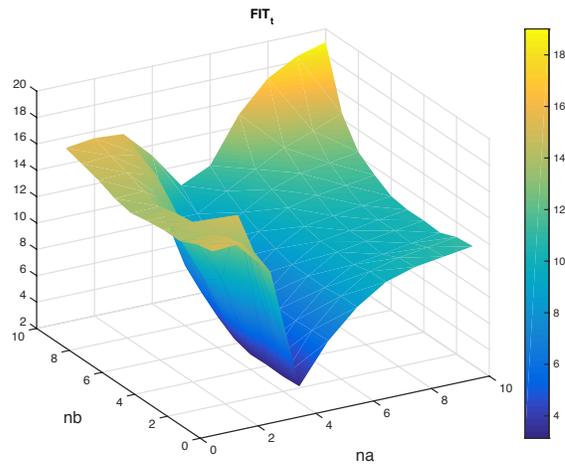


Figura 4.2: Comportamiento del FIT temporal cambiando los valores de na , nb y considerando que $nk = 4$.

Se puede observar en la figura que para el valor de $na = 4$ el FIT temporal es ineficiente pero, si el valor de na y nb aumenta, el FIT comienza a aumentar significativamente. Es ahí donde el usuario debe tomar la decisión de realizar más retardos en la estimación o conformarse con un índice menos favorable pero con un número menor de retardos. Cabe mencionar que si se tiene un número de retardos alto, el número de parámetros a estimar también tiende a crecer.

Otra opción para mejorar el desempeño de la estimación es volver a realizar la optimización de los pesos sinápticos pero ahora considerando los pesos sinápticos que se obtuvieron con anterioridad, es decir, tomar el valor de los pesos sinápticos obtenidos y definir esos valores como un nuevo valor de inicialización.

4.4. Cálculo de parámetros de la función de Huber

Cómo se mostró en 2.18, la función de Huber consta de un parámetro desconocido γ . Este parámetro está en función del producto de dos factores, uno de ellos es la raíz cuadrada de la varianza (ver ecuación 3.7) del error generado por una estimación realizada mediante el estimador M de L_2 y un factor escalar κ .

La función de Huber [36] es una parábola con vértice en cero, y su crecimiento lineal esta dado por $|e(k)| > \gamma$. Para una distribución normal se recomienda que el valor de γ se igual a 1.345 o por lo menos que se encuentre en un intervalo $1 \leq \gamma \leq 1.5$ o $1.5 \leq \gamma \leq 2$ [21, 24]. En Corbier y Ugalde [24] se propone que el valor inferior del intervalo de γ sea cercano a cero y que el superior sea igual a dos; este nuevo intervalo es una combinación del intervalo clásico y un intervalo extendido o recomendado expresado en una función de sintonización $f(\gamma)$ tal como se muestra en la figura 4.3.

Cabe mencionar que es de suma importancia la selección del valor de γ ya que define con qué estimador se realizará la optimización ya sea con L_1 o con L_2 . En todos los métodos propuestos se plantea la idea de establecer un intervalo para el valor de γ pero, como ya se ha mencionado, γ es un valor que está en función de otros dos valores de los cuales uno permanecerá constante (la raíz cuadrada de la varianza) y el otro puede variar su valor (κ). Por lo tanto se realiza la búsqueda del valor κ .

Al igual que en la búsqueda de los valores de na , nb y nk , κ cambiará su valor en un intervalo parecido a los antes propuestos de tal manera que por cada valor de κ se producirá un modelo diferente y al final, utilizando los índices de desempeño, se elegirá el valor de κ que mejor ajuste el modelo neuronal a la salida del sistema real.

Para cada modelo identificado se optó por variar el valor de κ en un intervalo de 0.01 hasta 2 en espacios de 0.01 unidades. En la figura 4.4 se muestra uno de

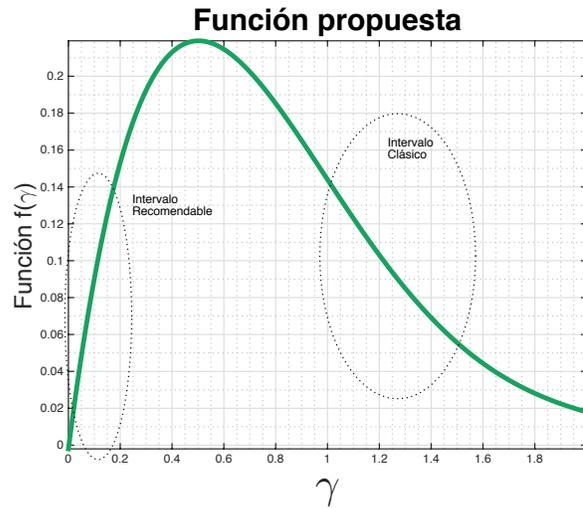


Figura 4.3: Función de sintonización con dos intervalos. El intervalo clásico $\gamma \in [1, 1.5]$ y un intervalo extendido $\gamma \in (0, 0.2]$.

los índices de desempeño que se utilizaron para la selección de κ . Cabe resaltar que con estos doscientos modelos diferentes se estarían calculando al rededor de $(\bar{\beta} - \underline{\beta})^3 + 200$ modelos distintos con el fin de encontrar el mejor candidato para realizar la estimación.

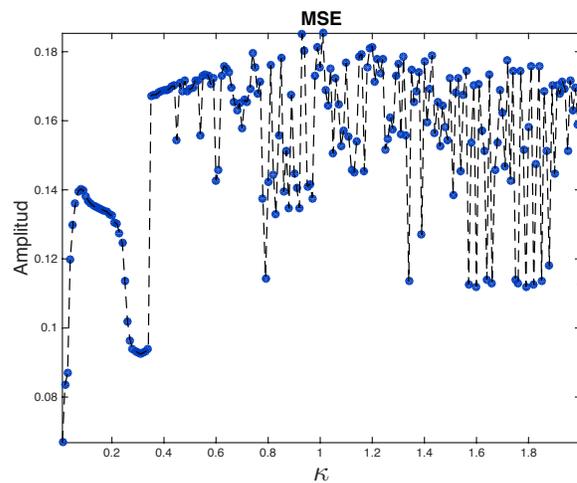


Figura 4.4: Función de sintonización con dos intervalos. El intervalo clásico $\gamma \in [1, 1.5]$ y un intervalo extendido $\gamma \in (0, 0.2]$.

Capítulo 5

Pruebas y resultados

En esta sección se presentan los resultados con la implementación de la teoría explicada con la identificación de sistemas caja gris utilizando redes neuronales artificiales. Se consideraron dos funciones objetivo, la primera es la robusta función de Huber expresada en la ecuación (2.18) y la segunda es la norma dos del error, es decir la expresión mostrada en (2.17).

Para validar los resultados, se comparan la señal producida por el modelo neuronal propuesto con respecto a la caja de herramientas de identificación de MATLAB como el NLARX y la red neuronal que se implementó pero realizando la minimización de L_2 . Se agregarán valores atípicos a las señales de salida real de los sistemas a identificar con el fin de ver el efecto de las dos distintas funciones costo (2.17 y 2.18). El modelo neuronal que se ha implementado con el estimador M de Huber se llama RHNN y el mismo modelo neuronal implementado pero con el estimador M (L_2) se le llama RUNN.

5.1. Identificación de un brazo robot

Con el fin de aplicar la teoría que se ha visto hasta el momento primeramente se identificó un sistema electromecánico tal como se muestra a continuación.

5.1.1. El sistema

Los datos caracterizan el movimiento de un brazo robot midiendo la aceleración de uno de los eslabones cuando un motor aplica un par mecánico (figura 5.1). Este sistema es un ejemplo tomado del DaISy (Database for Identification of Systems) [2].

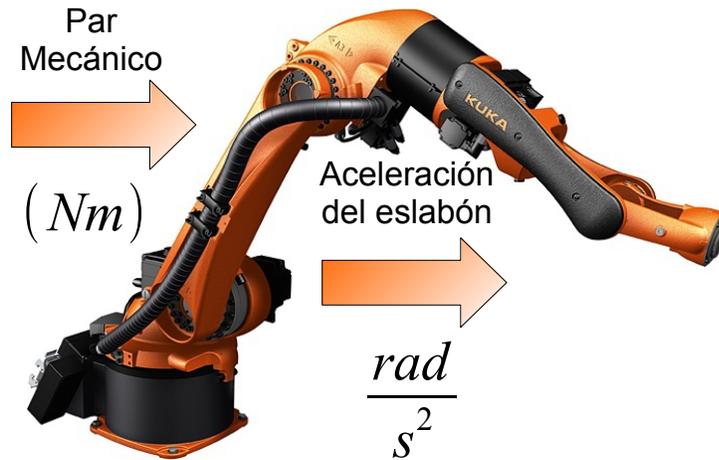


Figura 5.1: Brazo Robot

5.1.2. Experimentos

El sistema fue identificado utilizando el modelo neuronal tal como la expresión (5.1). Para estimar los parámetros del modelo neuronal se utilizó el estimador M de Huber y el algoritmo de Levenberg-Marquadt. Se añadieron valores atípicos a la señal de salida real con posición aleatoria tal como se muestra en la figura 5.2.

$$\begin{aligned}
 r_b &= \sum_{i=1}^{nn} V_{b_i} \tanh(J_u W_{b_i}) \\
 r_a &= \sum_{i=1}^{nn} V_{a_i} \tanh(J_y W_{a_i}) \\
 T &= Z_b(r_b) + Z_a(r_a) \\
 \hat{y}(k) &= X \tanh(T)
 \end{aligned} \tag{5.1}$$

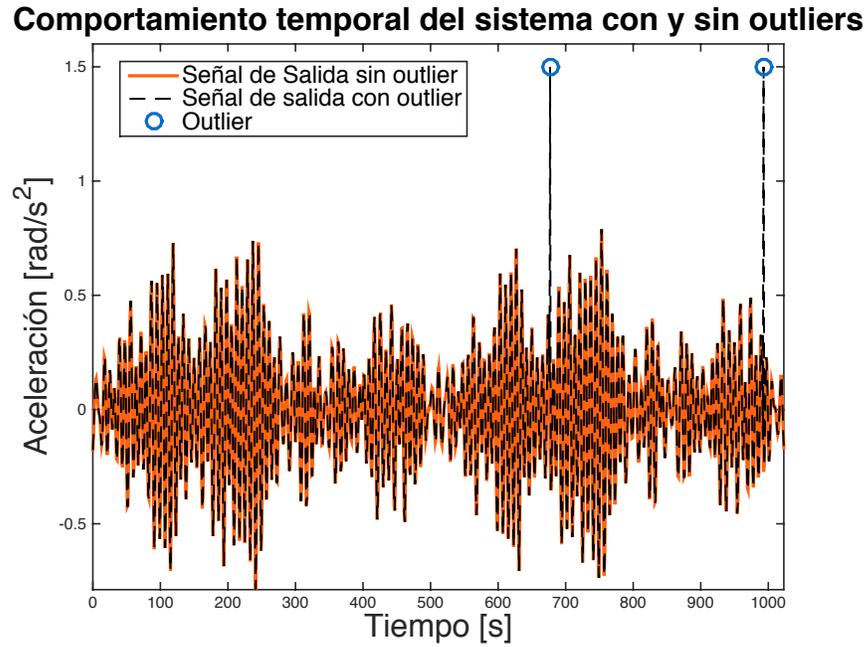


Figura 5.2: Señal de salida original y señal de salida con valores atípicos añadidos.

Una vez que el modelo neuronal fue entrenado, se redujo bajo las suposiciones uno y dos y se obtuvo un modelo neuronal reducido como se muestra en (5.2).

$$\hat{y}(k) = X \tanh \left(\tilde{Z}_B \tanh(J_u W_b) + \tilde{Z}_A \tanh(J_y W_a) \right) \quad (5.2)$$

Se realizó la identificación del sistema con el uso del estimador M de Huber, el estimador M de L_2 con el uso de las redes neuronales vistas en la sección 2.4. También se realizó la identificación del sistema mediante el toolbox de identificación de sistemas no lineales que ofrece el software de cálculo numérico MATLAB llamado NLARX.

Selección de ordenes na , nb y nk

Cumpliendo con lo establecido en la sección 4.3, se definió un intervalo de na , nb y nk . Donde el intervalo de na y nb son iguales y se encuentran entre cinco y 10. El intervalo de nk se encuentra entre uno; la selección de estos intervalos hace que

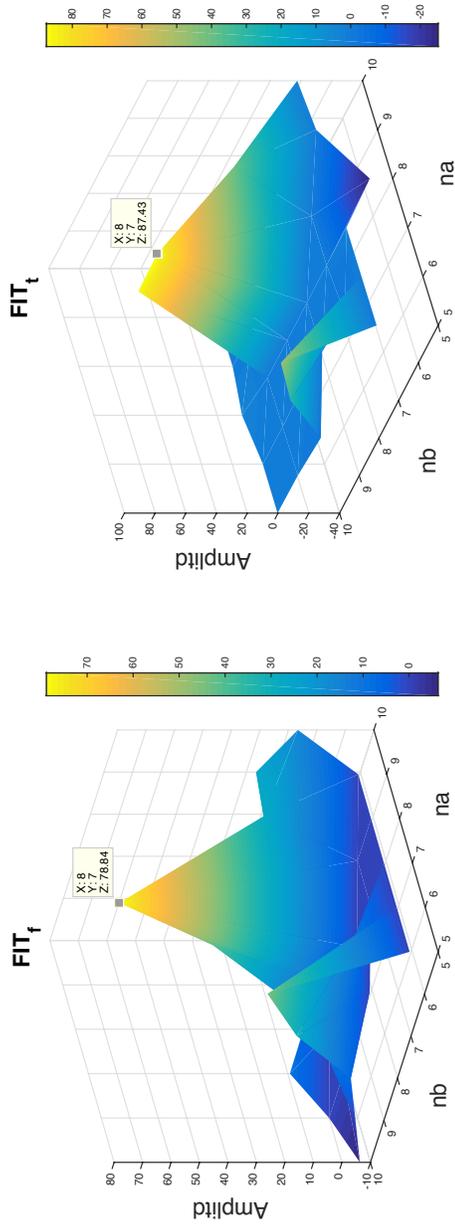
se tenga el comportamiento de 108 diferentes modelos. En la figura 5.3 se pueden observar las gráficas de los comportamientos del fit en la frecuencia (ver ecuación 5.8), el fit temporal y la desviación estándar respectivamente.

En este caso se torna sencillo realizar el análisis ya que solamente hay un máximo en algunas de las figuras (a y b donde se busca el número más cercano al 100%) y hay un mínimo en la desviación estándar. Dadas esas gráficas se escogió el valor de $na = 8$, $nb = 7$ y $nk = 1$.

5.1.3. Selección de κ

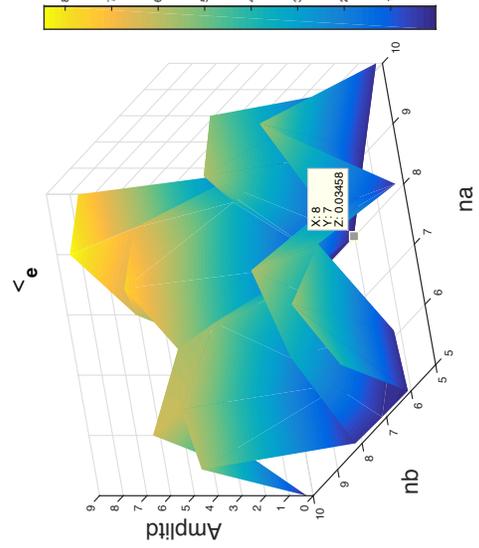
Después de haber seleccionado el valor del número de retrocesos de la señal de entrada y de salida, se debe seleccionar el valor de κ el cual es esencial para realizar la estimación con el estimador M de Huber. Aplicando la misma metodología para la selección de los ordenes de la red neuronal, se iterará el valor de κ de 0.1 a 2 en espacios de 0.1.

En la figura 5.4 se muestran los diferentes valores de los índices a través del cambio de κ . Se puede observar que entre los valores de 0.2 y 0.3 puede ser el valor adecuado para κ .



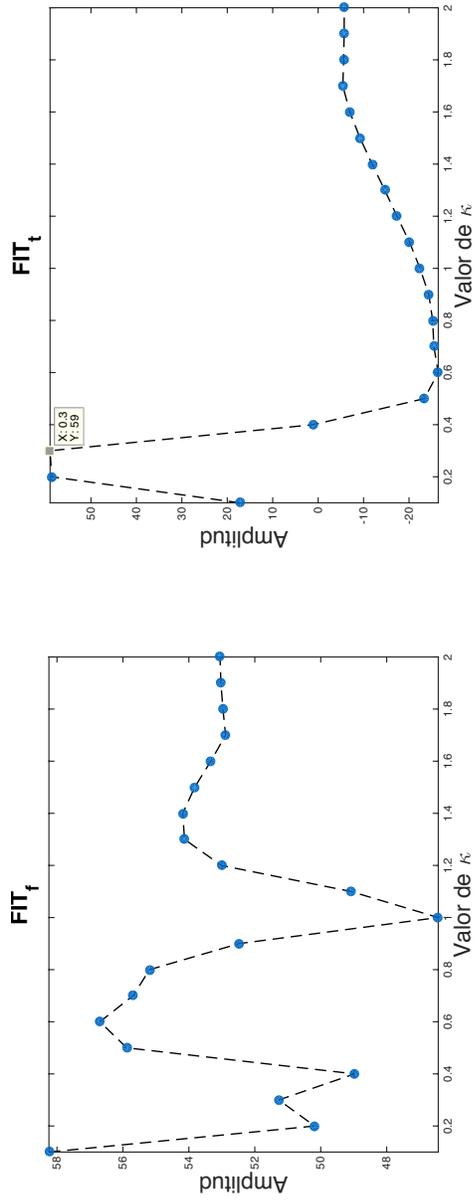
(a) Comportamiento del fit en la frecuencia

(b) Comportamiento del fit en la temporal



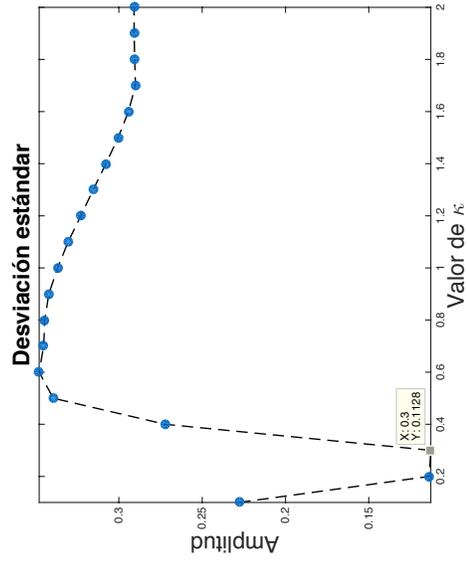
(c) Desviación estándar (σ_e).

Figura 5.3: Análisis de índices para la selección de ordenes



(a) Comportamiento del fit en la frecuencia

(b) Comportamiento del fit en la temporal



(c) Desviación estándar (σ_e).

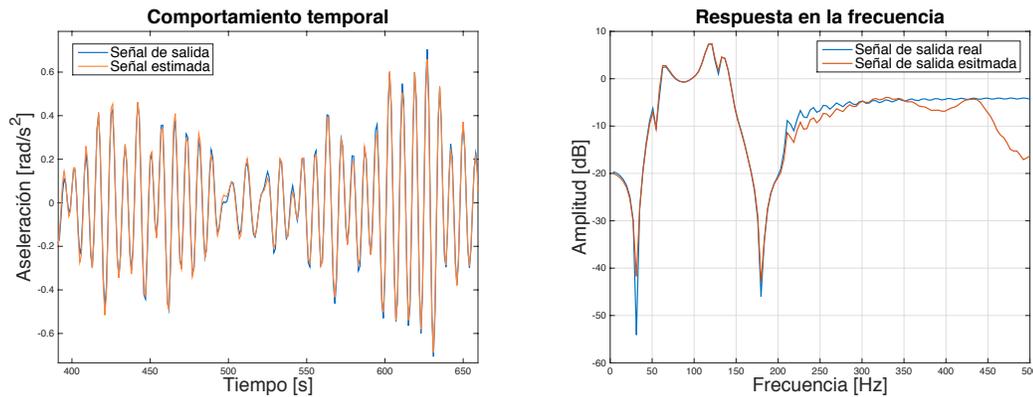
Figura 5.4: Análisis de índices para la selección de κ

5.1.4. Identificación del sistema

Para realizar la identificación del sistema se seleccionaron los valores de $na = 8$, $nb = 7$ y $nk = 1$. El número de neuronas para el procesamiento de la señal de entrada fue de 40 al igual que el número de neuronas encargadas de procesar las señales de salida. La respuesta temporal de cada sistema se puede observar en la figura 5.5, 5.6, 5.7 al igual que la respuesta en la frecuencia sin valores atípicos agregados.

En la tabla 5.1 se pueden encontrar las comparaciones de los tres modelos neuronales distintos. Cabe recalcar que el modelo RHNN ha sido optimizado bajo el mismo algoritmo de entrenamiento pero, este cuenta con el estimador M de Huber. Los otros modelos han sido optimizados con el estimador de L_2 .

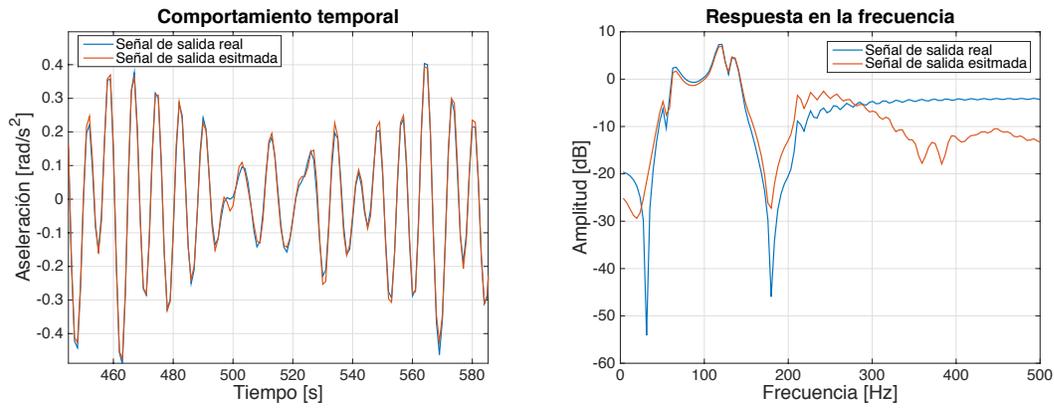
En la tabla 5.1 se puede observar claramente que el modelo RHNN es capaz de realizar una estimación favorable, es decir, la optimización con el estimador M de Huber es capaz de igualar los resultados o por lo menos estar numéricamente cercanos a ellos en comparación con el estimador M de L_2 .



(a) Respuesta temporal del modelo neuronal (b) Respuesta en la frecuencia del modelo neuronal

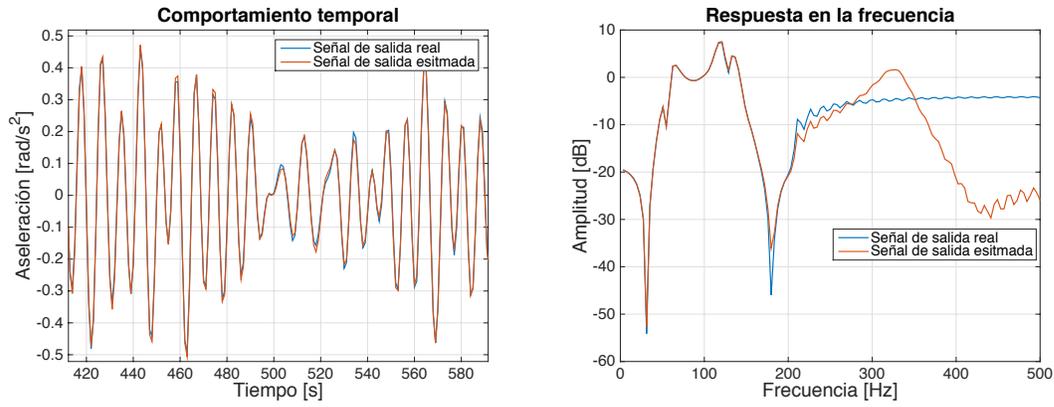
Figura 5.5: Respuesta en la frecuencia y temporal del modelo neuronal RHNN

A continuación se realizará la estimación del sistema con la adición de outliers.



(a) Respuesta temporal del modelo neuronal (b) Respuesta en la frecuencia del modelo neuronal

Figura 5.6: Respuesta en la frecuencia y temporal del modelo neuronal RUNN

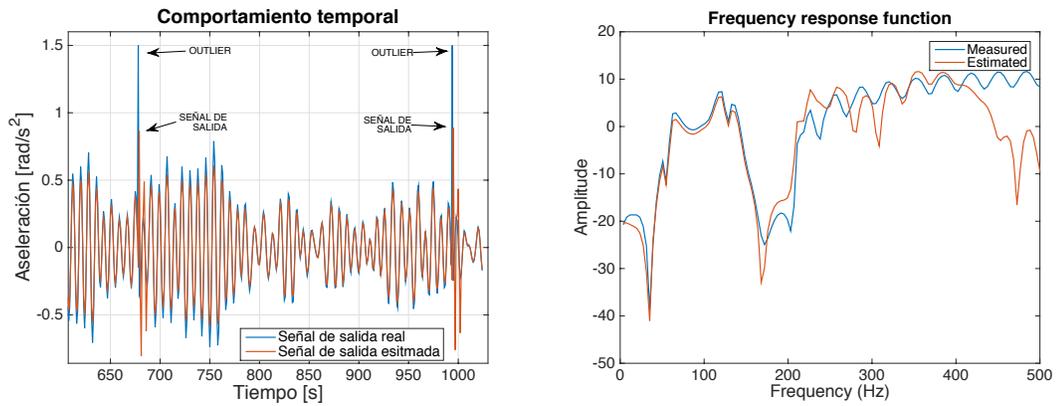


(a) Respuesta temporal del modelo neuronal (b) Respuesta en la frecuencia del modelo neuronal

Figura 5.7: Respuesta en la frecuencia y temporal del modelo neuronal NLARX

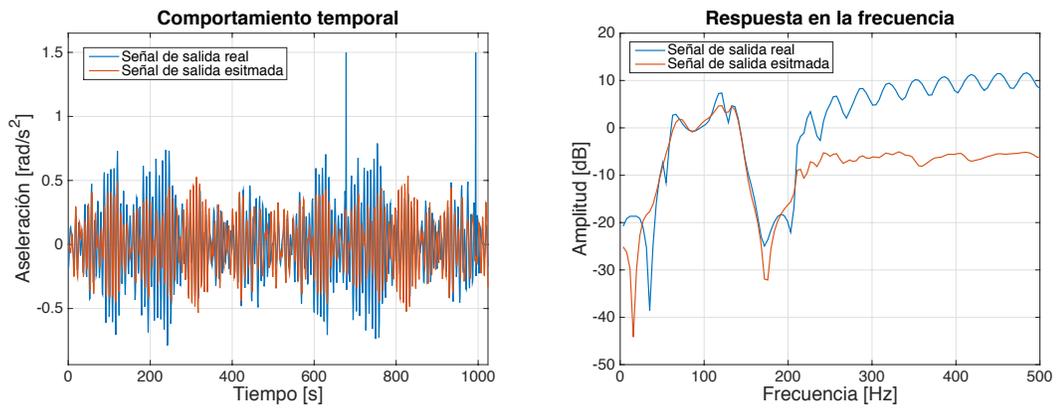
Tabla 5.1: Comparativa de modelos neuronales y otras metodologías.

Ref.	n_p	μ_t	s_t	e_{RMS_t}	e_{RMS_e}	FIT_F %	FIT_t %
RHNN	603	-0.0000588	0.0346	0.0346	0.0346	78.84	87.42
RUNN	603	-0.000187	0.0232	0.0232	0.0232	63.6265	91.5537
NLARX	551	-0.0000944	0.0234	0.0234	0.0234	52.9350	91.4759



(a) Respuesta temporal del modelo neuronal (b) Respuesta en la frecuencia del modelo neuronal

Figura 5.8: Respuesta en la frecuencia y temporal del modelo neuronal RHNN

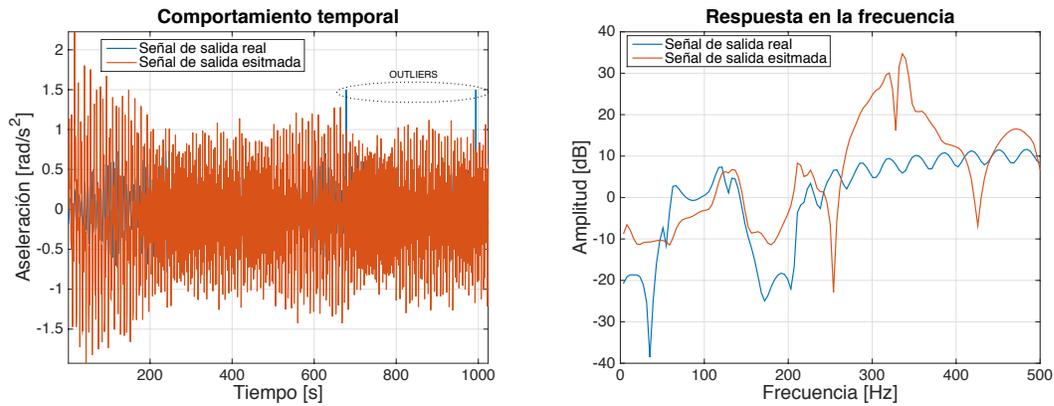


(a) Respuesta temporal del modelo neuronal (b) Respuesta en la frecuencia del modelo neuronal

Figura 5.9: Respuesta en la frecuencia y temporal del modelo neuronal RUNN

Tabla 5.2: Comparativa de modelos neuronales.

Ref.	n_p	μ_t	s_t	e_{RMS_t}	e_{RMS_e}	$FIT_F \%$	$FIT_t \%$
RHNN	603	0.0027	0.1109	0.1109	0.1109	47.2869	60.7529
RUNN	603	0.0036	0.2549	0.2549	0.2549	14.7326	9.8288
NLARX	551	0.1649	0.7548	0.7726	0.772	-380.3377	-173.2950



(a) Respuesta temporal del modelo neuronal (b) Respuesta en la frecuencia del modelo neuronal

Figura 5.10: Respuesta en la frecuencia y temporal del modelo neuronal NLARX

5.1.5. Comentarios

En la tabla 5.2 se puede observar la eficiencia de la función de Huber. En la misma tabla, se puede apreciar que hay una diferencia inminente del modelo RHNN con respecto a los modelos RUNN y NLARX los cuales fueron optimizados mediante el uso del estimador L_2 . Las medidas de parentesco (FIT) mostradas en la tabla 5.2 tal vez no sean las más efectivas pero, si se comparara el FIT producido por el modelo RHNN y los modelos RUNN y NLARX se podría concluir que el modelo que podría representar al sistema real sería el RHNN.

De manera visual en las figuras 5.8, 5.9 y 5.10 se puede apreciar de manera similar que el modelo más adecuado para representar el comportamiento del sistema es el RHNN. Su comportamiento en la frecuencia asemeja el comportamiento del sistema real y, además, en la figura 5.8(a) se puede observar que el modelo neuronal “modela” al valor atípico, mientras que los otros dos modelos no lo hacen. Los pesos sinápticos estimados se muestran en el apéndice C.

5.2. Wiener - Hammerstein Benchmark

El sistema que a continuación se presenta se mostró ante el IFAC (International Federation of Automatic Control) con el fin de comparar los resultados de las distintas metodologías de la comunidad de control automático. Para más detalles ver [63].

5.2.1. El sistema

El dispositivo bajo pruebas o DUT por sus siglas en inglés, es un circuito electrónico no lineal con una estructura Wiener-Hammerstein como se observa en la figura 5.11.

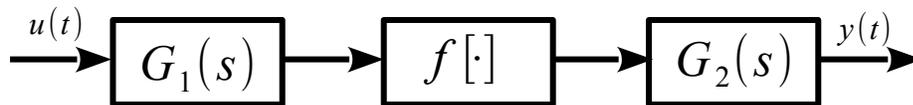


Figura 5.11: Sistema Wiener-Hammerstein conectado en cascada por un bloque dinámico, uno estático y un bloque dinámico lineal $G_1(s)f[\cdot]G_2(s)$

El primer filtro $G_1(s)$ es un filtro de tercer orden Chebyshev (filtro pasa banda de 0.5 dB y una frecuencia de corte de 4.4kHz). El segundo filtro $G_2(s)$ es un filtro de tercer orden Chebyshev (filtro rechaza banda de 40 dB de atenuación comenzando en 5kHz). La no linealidad estática es contruida mediante el circuito de un diodo túnel como el que se muestra en la figura 5.12. El sistema fue excitado por una señal Gaussiana de banda limitada produciendo un total de 188000 muestras correspondientes a 3.6719 segundos de experimentación con 51.2 kHz como tiempo de muestreo. La figura 5.13 muestra dos señales diferentes, la primera es la señal original sin outliers. La segunda es la señal con outliers añadidos.

5.2.2. Validación

La base de datos proporcionada por el benchmark está dividida en dos secciones: con la primer sección se realiza la estimación con $u(t), y(t)$ para $t = 1, 2, \dots, 100000$. Después, el modelo es simulado para obtener una señal de salida estimada $\hat{y}(t)$ producida por la RHNN. La prueba de validación se realiza con el conjunto de datos

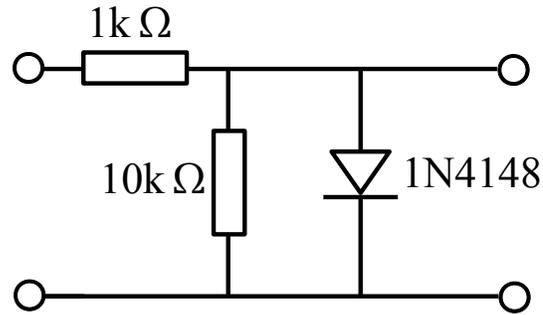


Figura 5.12: Circuito utilizado para construir el sistema estático no lineal.

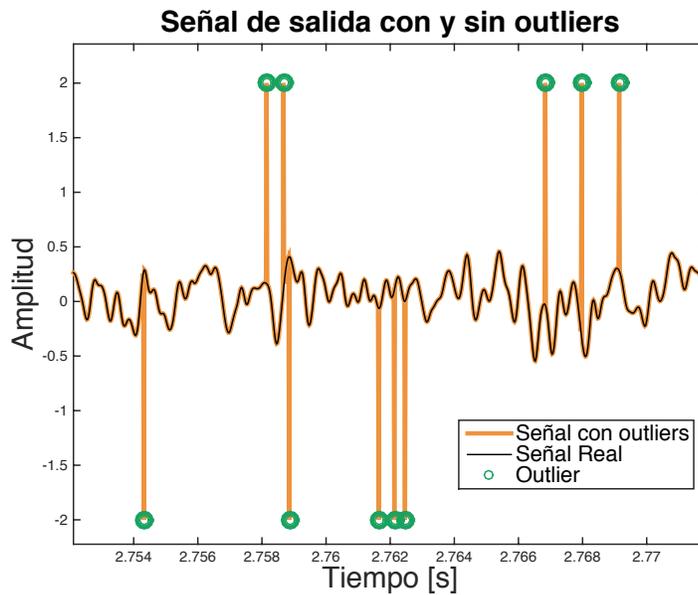


Figura 5.13: Señal original sin outliers y con outliers añadidos.

donde $t = 100001, 100002, \dots, 188000$ [63]. El benchmark establece los índices para realizar la comparación entre modelos. La calidad de precisión se realizará con el análisis de los siguientes indicadores estadísticos:

- El valor promedio del error de simulación:

$$\mu_t = \frac{1}{87000} \sum_{t=101001}^{188000} e_{sim}(t) \quad (5.3)$$

- La desviación estándar del error de simulación:

$$s_t^2 = \frac{1}{87000} \sum_{t=101001}^{188000} (e_{sim}(t) - \mu_t)^2 \quad (5.4)$$

- El valor medio cuadrático del error de simulación:

$$e_{RMSe}^2 = \frac{1}{87000} \sum_{t=101001}^{188000} (e_{sim}(t))^2 \quad (5.5)$$

- El valor medio cuadrático del error de simulación en tal que $t = 1001, 1002, \dots, 100000$:

$$e_{RMSt}^2 = \frac{1}{99000} \sum_{t=1001}^{100000} (e_{sim}(t))^2 \quad (5.6)$$

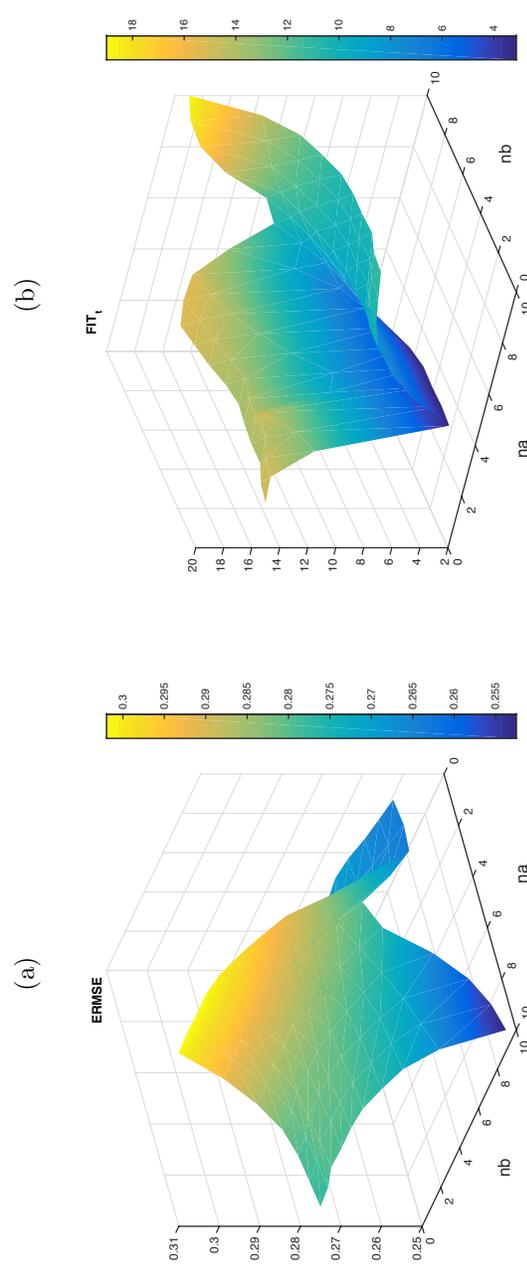
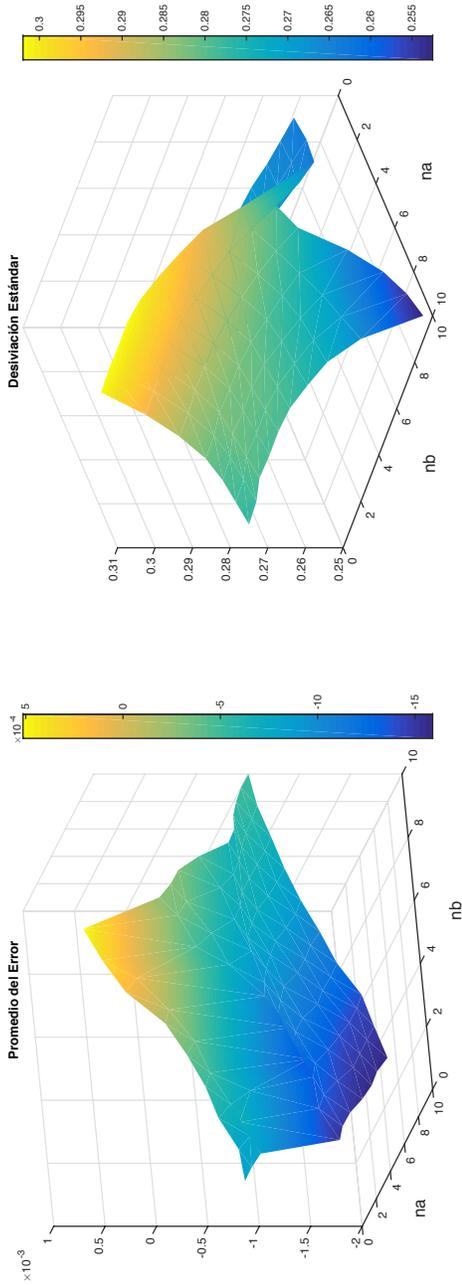
En las ecuaciones (5.3)-(5.6), la sumatoria comienza en $t = 101001$ hasta $t = 188000$ con el fin de comparar los resultados en dicho intervalo con el resultado de (5.6). Esto se realiza para observar el comportamiento de la función estimada fuera del rango de valores que se utilizan para estimar la función.

5.2.3. Experimentos

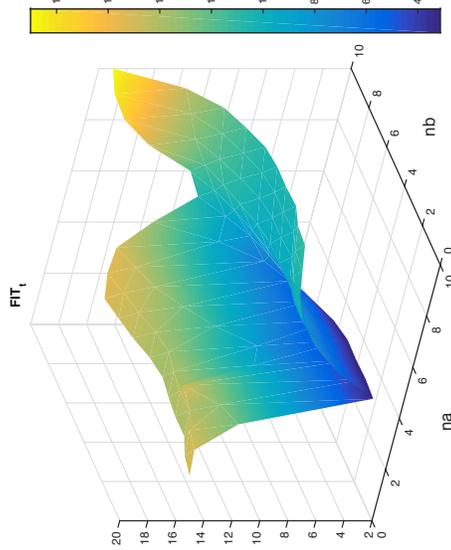
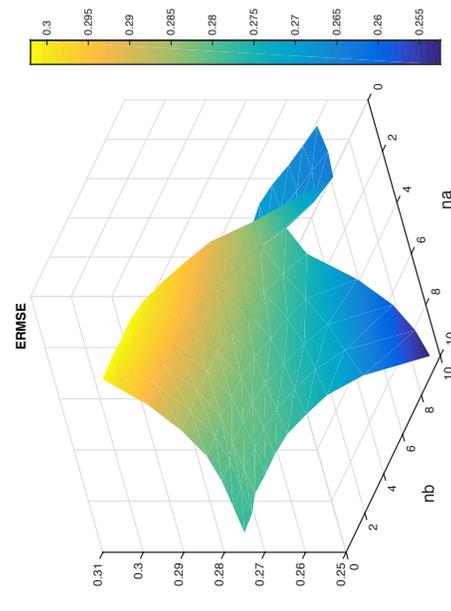
Selección de ordenes n_a , n_b y n_k

Los valores de n_a , n_b y n_k son el número de regresiones en las señales de entrada (n_b) y la salida (n_a). El valor de n_k (regresor natural) se ve reflejado en la señal de entrada. En la figura 2.9 se observa el efecto de estos valores para realizar la identificación de sistemas con RNA's. Para cada sistema, se seleccionaron valores de n_a , n_b y n_k diferentes. Para encontrar el valor subóptimo de los regresores se definieron los siguientes conjuntos $n_a = \{1, 2, 3, \dots, 20\}$, $n_b = \{1, 2, 3, \dots, 20\}$ y $n_k = \{1, \dots, 5\}$. Para seleccionar el modelo adecuado se analiza cada índice mostrado (ver figura 5.14 y 5.15); en el promedio del error, la desviación estándar, el valor ERMSE, el error a bajas frecuencias y altas frecuencias, se busca que el valor de amplitud sea el mínimo posible pero en el caso del FIT temporal y frecuencial se busca el valor máximo posible. En la figura 5.14(a) se observa que en algunas ocasiones el promedio del error

tiende a ser negativo pero lo ideal es que este sea igual a cero o que se encuentre cercano a cero, para ello, el valor de nb debe ser entre ocho y 10, y el valor de na debe ser de uno a tres, pero estos valores afectan que el FIT en la frecuencia (ver figura 5.15(a)) tienda a disminuir y que el error a altas frecuencias aumente (ver figura 5.15(b)). En ocasiones esta manera de seleccionar los ordenes de la red neuronal puede ser ineficiente, se realizaron diversas pruebas y la mejor aproximación al sistema fue un modelo neuronal con $na = 4$, $nb = 7$ y $nk = 4$.



(b)



(d)

Figura 5.14: (a) Media del error, (b) desviación estándar del error, (c) RMSe del error y (d) FIT temporal

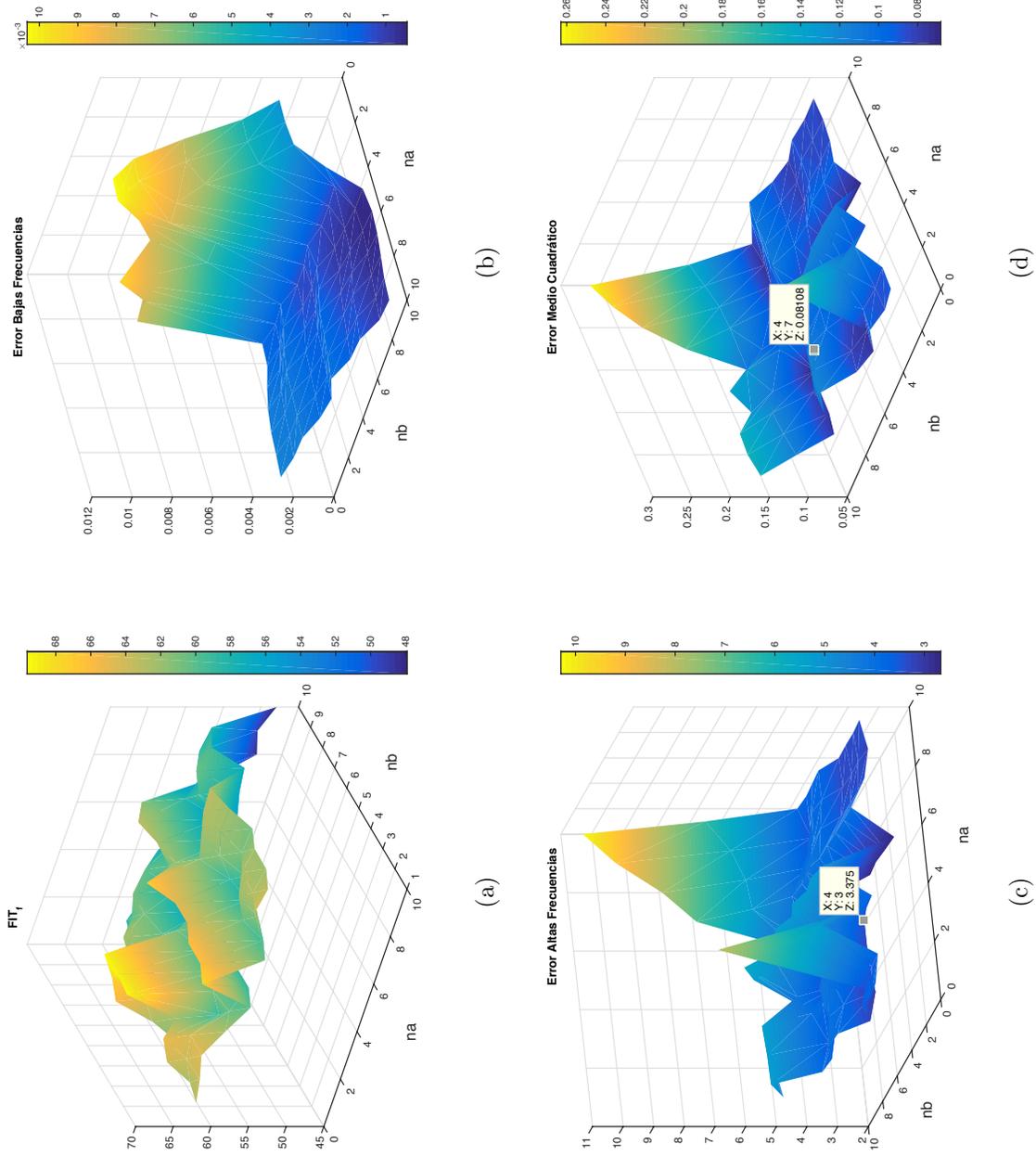


Figura 5.15: (a) FIT frecuencial, (b) Error a bajas frecuencias, (c) error a altas frecuencias y (d) error medio cuadrático

5.2.4. Selección de κ

El valor de κ se selecciona dentro de un intervalo de valores. Se propone un intervalo de $\kappa \in (0, 2]$ para poder percatarse de los efectos que puede producir este parámetro con los datos contaminados artificialmente del sistema Wiener-Hammerstein [63]. Se obtuvieron distintos modelos con diferentes valores de κ , los resultados de la experimentación se observan en las figuras 5.16, 5.17 y 5.18. En la figura 5.16 se observa la media del error producido por la estimación, lo ideal es que esta medida se encuentre variando en cero como la suma del error al cuadrado (figura 5.17). En la figura 5.18 se muestra la desviación estándar la cual se busca que el valor sea cero al igual que el valor ERMS. Dados los índices mostrados en las figuras, se puede concluir que el mejor valor de κ es aquel que se encuentre más cercano a cero. Se mencionó que el valor de κ se encontraba en un intervalo ($\kappa \in (0, 2]$) y se varió en pasos de 0.01 unidades. Observando las figuras 5.16, 5.17 y 5.18 se puede concluir que el valor de κ es de 0.01 ya que este fue el que me mostró un mejor desempeño considerando la desviación estándar y el RMSe.

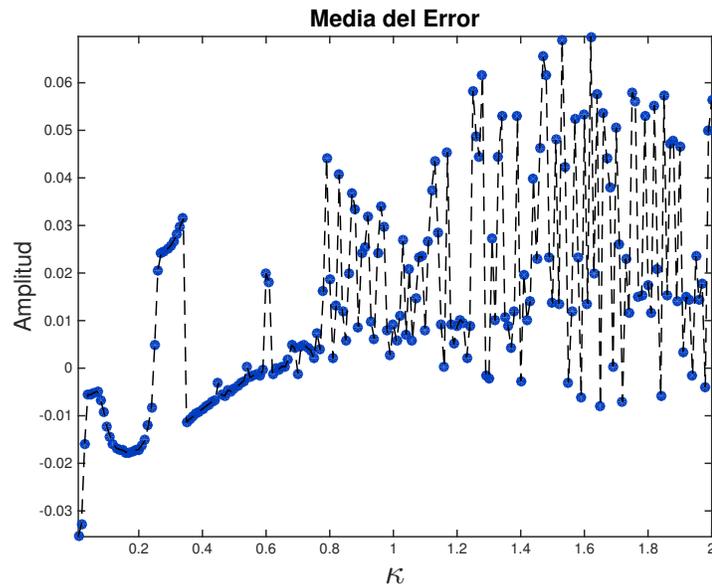


Figura 5.16: Suma del error cuadrático medio para la selección de κ .

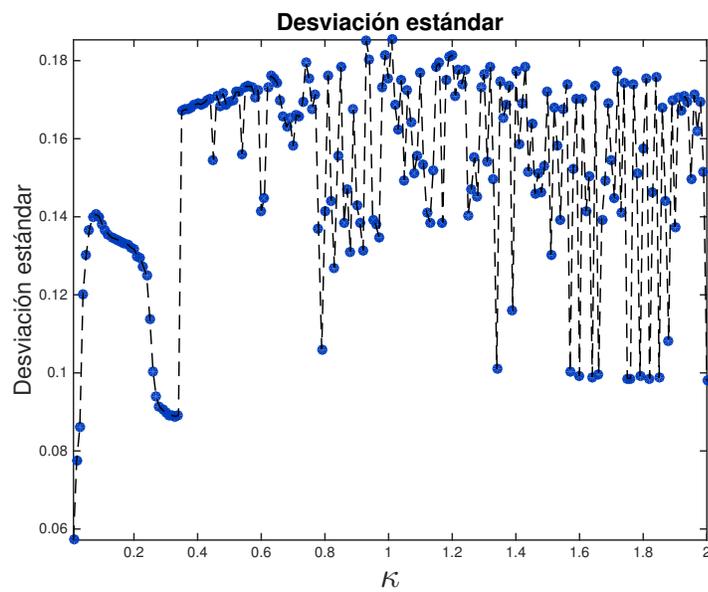


Figura 5.17: Desviación estándar del error para la selección de κ .

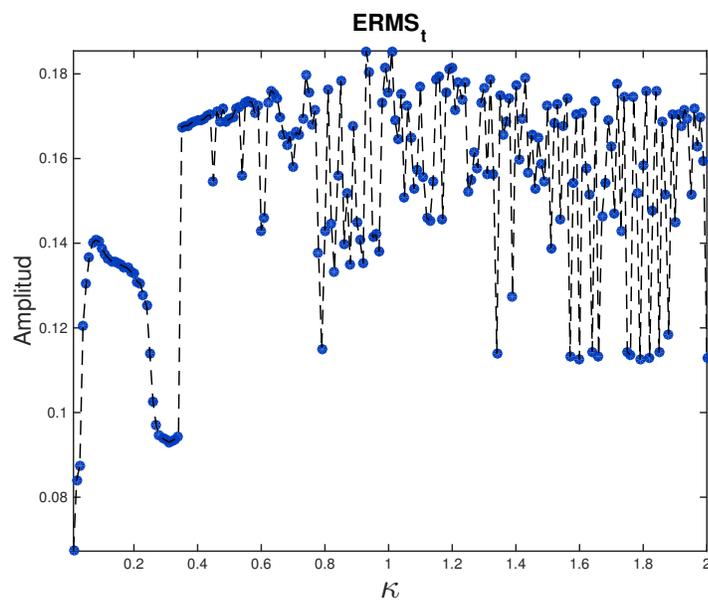


Figura 5.18: Media del error para la selección de κ .

5.2.5. Identificación del sistema

El sistema sin outliers fue identificado utilizando modelo neuronal $2nn - 2 - 1$ dado por la ecuación (2.28). Las funciones de activación se escogieron como $\varphi_1(z) = \tanh(z)$ y $\varphi_2(z) = \varphi_3(z) = z$ también se agregaron umbrales en cada neurona de la capa de entrada, en la capa oculta y en la capa de salida; tales ajustes dan origen al modelo neuronal reducido mostrado en (5.7).

$$\hat{y}(k) = \tilde{V}_B \tanh(J_u \tilde{W}_B + \tilde{W}_{BH}) + \tilde{V}_A \tanh(J_y \tilde{W}_A + \tilde{W}_{AH}) + H \quad (5.7)$$

Se escogieron 40 neuronas para el procesamiento de la señal de entrada y 40 más para el procesamiento de la señal de salida, se seleccionaron los valores de $n_a = 4$, $n_b = 7$ y $n_k = 4$. Los parámetros del modelo neuronal fueron encontrados utilizando el algoritmo de Levenberg-Marquadt y la función de Huber. La estimación se realizó con tan solo 2 épocas. También se identificó al sistema mediante el mismo modelo neuronal pero con el estimador L_2 ; el entrenamiento se realizó bajo las mismas características anteriores.

También se identificó al sistema con el mismo modelo neuronal pero con el estimador L_2 y, de manera similar, se identificó utilizando el toolbox de identificación que ofrece MATLAB 2014b. Para realizar una comparación entre la eficiencia que ofrece el software y nuestra metodología, se utilizó el modelo neuronal no lineal NLARX mostrado en el apéndice B. Se programó el sistema con un parámetro de $nn = 10$ (nn es la cantidad de neuronas en la capa de entrada y la capa de salida). Los parámetros del modelo fueron encontrados utilizando el algoritmo de Levenberg-Marquadt que ofrece MATLAB utilizando la norma L_2 a lo largo de 20 épocas.

Se agregó un parámetro comparativo más. Se midió el parentesco entre las señales deseadas y las estimadas por medio del calculo de un FIT en el dominio de la frecuencia expresado en (5.8). Como parámetro auxiliar en la validación se considera el número de parámetros estimados.

$$FIT_F = 100 \left(1 - \frac{\|F - \hat{F}\|}{\|F - \bar{F}\|} \right) \quad (5.8)$$

donde F es la magnitud de la respuesta en la frecuencia del sistema W-H, \hat{F} es

la magnitud de la respuesta en la frecuencia del modelo neuronal y \bar{F} es la media de F calculado como $\bar{F} = \frac{1}{N_F} \sum_{i=1}^{N_F} F_i$.

En la figura 5.19 se muestra la respuesta temporal del sistema real, del modelo neuronal con el estimador L_2 , el modelo neuronal con el estimador M de Huber y el NLARX de MATLAB. Para realizar una mejor comparación visual del desempeño de cada modelo neuronal se obtuvo la respuesta en la frecuencia de cada modelo (ver figura 5.20). En la figura 5.20 se muestra que el comportamiento de las redes neuronales son similares y aún más importante; la identificación mediante el estimador M de Huber es competente en comparación de las diversas redes neuronales. Mediante los índices de desempeño vistos en la sección de validación (ver sección 5.2.2) se realiza una comparativa de los diferentes modelos neuronales. Tales resultados se muestran en la tabla 5.3. Cabe mencionar que la estimación se realizó sin haber incluido valores atípicos como se muestra en la figura 5.19.

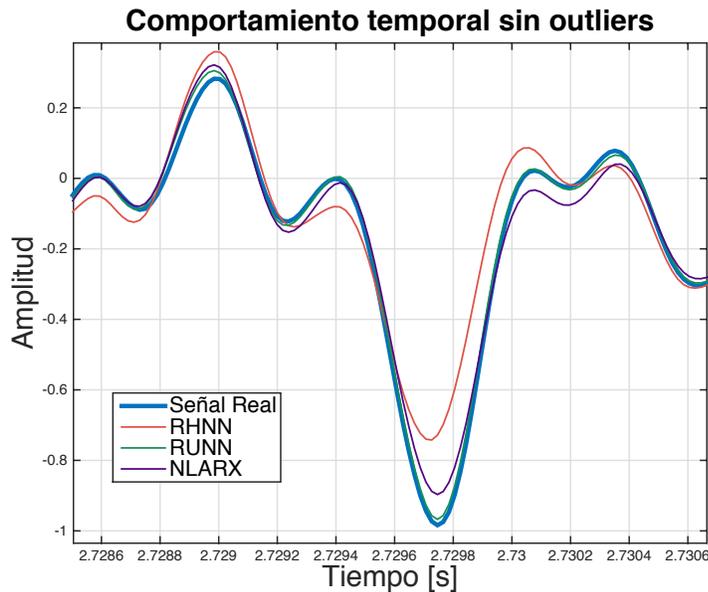
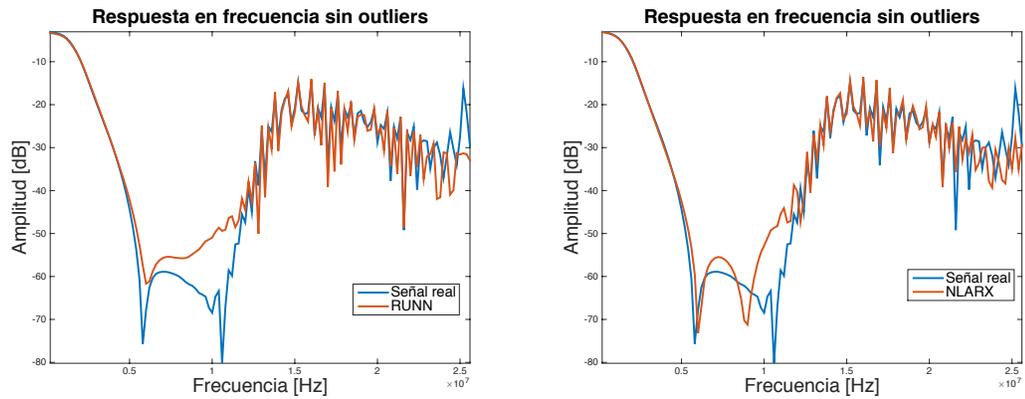


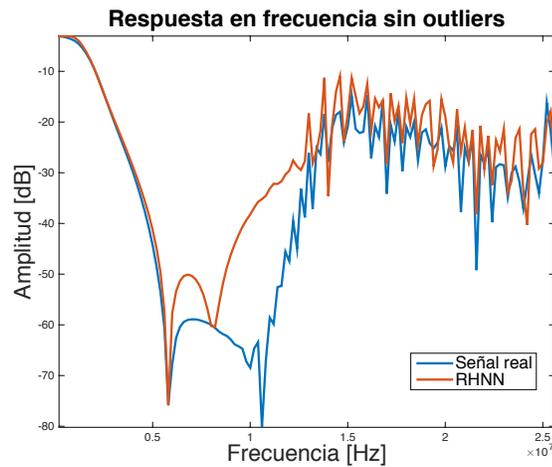
Figura 5.19: Respuesta temporal del sistema.

Los resultados también fueron comparados con metodologías propuestas por [45, 66, 68] (ver tabla 5.3). Cabe resaltar que las metodologías propuestas por [45, 66, 68] realizan una estimación sin haber incluido valores atípicos a sus datos.

Se puede ver en la tabla 5.3 que los resultados obtenidos por el modelo neuro-

(a) Modelo neuronal con estimador M de L_2

(b) Modelo neuronal NLARX



(c) Modelo neuronal con estimador M de Huber

Figura 5.20: Respuesta en la frecuencia de los modelos neuronales.

nal (RHNN) compite con las demás metodologías considerando que los índices de desempeño son relativamente cercanos y que el número de parámetros que utiliza es menor que el de todos los demás. A continuación, se han añadido valores atípicos a las señales con el fin de comprobar la aportación de la estimación robusta que se propone

En la figura 5.21 se muestran los resultados de la red robusta RHNN, la red neuronal con el estimador L_2 y de la caja de herramientas de MATLAB (NLARX). Se puede observar que el perfil o la forma de la señal del sistema real (señal de color

Tabla 5.3: Comparativa de modelos neuronales y otras metodologías.

Ref.	n_p	μ_t	s_t	e_{RMS_t}	e_{RMS_e}	$FIT_F\%$	$FIT_t\%$
RHNN	19	-0.00960	0.01065	0.0050645	0.005065	76.68	70.7
RUNN	19	0.001784	0.0282	0.02824	0.0282490	90.05	88.299
NLARX	684	0.0032	0.034	0.034	0.03389	90	85.94
Marcanto [45]	0.085	—	—	—	—	—	—
Wills [68]	22	-0.043	0.0808	0.096	—	—	—
Sjöberg [66]	797	—	—	0.043	0.042	—	—

azul) cambia cuando esta contiene outliers.

La comparativa entre los dos tipos de modelos neuronales se muestra en la tabla 5.4 utilizando los índices que anteriormente se usaron [63]. Otra forma de ver las contribuciones es observando la figura 5.22. En dicha figura se puede visualizar que los modelos obtenidos mediante la optimización de la norma L_2 es incapaz de predecir el comportamiento del outlier, mientras que el modelo con la función de Huber puede modelar tal comportamiento.

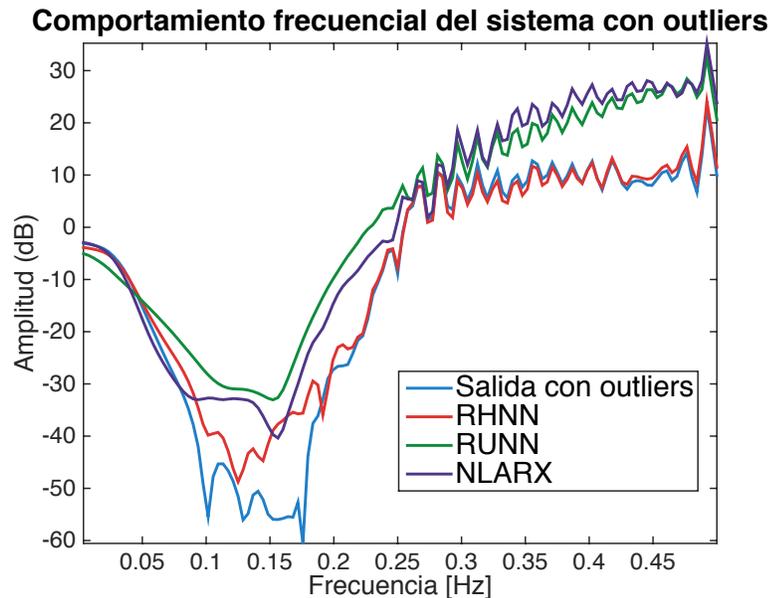


Figura 5.21: Respuesta frecuencial de los modelos neuronales en presencia de outliers.

Tabla 5.4: Comparativa de modelos neuronales.

Ref.	n_p	μ_t	s_t	e_{RMS_t}	e_{RMS_e}	$FIT_F \%$	$FIT_t \%$
RHNN	19	0.000573	0.208	0.208	0.207	89	75
RUNN	19	0.0008206	0.224	0.224	0.222	18.47	28.29
NLARX	684	-0.00116	0.216	0.216	0.213	31.04	14.65

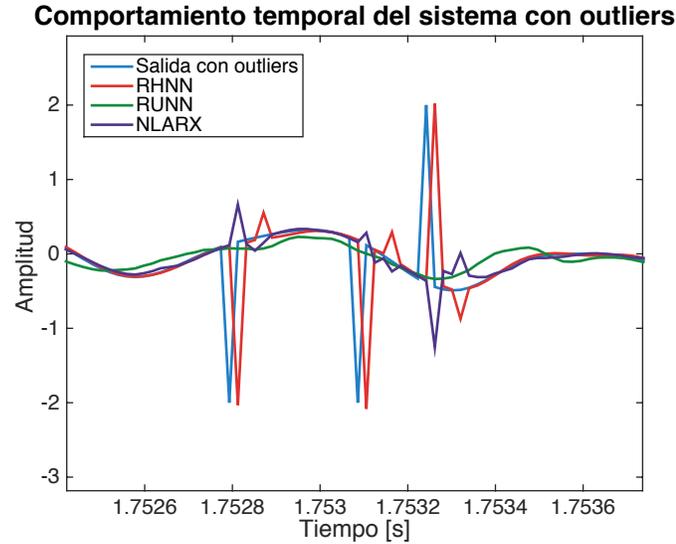


Figura 5.22: Respuesta temporal de los modelos neuronales en presencia de outliers.

5.2.6. Comentarios

La tabla 5.3 muestra que el modelo RHNN propuesto en este tema de tesis es capaz de competir con otras metodologías propuestas en la literatura con resultados satisfactorios en función de los índices de desempeño que establece el benchmark. De forma visual, en la figura 5.20(c) se observa que la respuesta de la RHNN tiene un comportamiento fiel al sistema real a bajas frecuencias. Por otro lado, realizando la estimación con valores atípicos añadidos, se puede observar que también se tiene una satisfactoria aproximación a bajas frecuencias y comparando los resultados mostrados en la tabla 5.4, se puede verificar que el desempeño de la RHNN es el mismo y en algunos casos asciende. La respuesta temporal (figura 5.22), se observa que la señal estimada producida por el RHNN es mucho más fiable que la señal que genera el NLARX y el RUNN. Los pesos sinápticos estimados se encuentran en el apéndice C.

5.3. Identificación de ducto acústico

Con el fin de seguir aprovechando el enfoque de reducción propuesto por Romero et al. [56] y de la función de Huber como función objetivo [20–23, 57], se ha identificado un sistema diferente.

5.3.1. El sistema

Este sistema es un dispositivo de ondas acústicas hecho de Plexiglas, es usado para diseñar controladores para eliminar el ruido (Figura 5.23). La salida del tubo se encuentra mayormente sellada herméticamente simulando un ducto acústico de longitud infinita, mientras que la entrada, se encuentra totalmente abierta. La entrada del sistema es una bocina excitada con una señal binaria pseudo-aleatoria con un largo de $L = 2^{10} - 1$ y un nivel de voltaje de $\pm 3V$. El periodo de muestreo es igual a $T_s = 500\mu s$. El número de datos total es de $N = 1024$ capturados con micrófono uno haciendo que la identificación de este sistema sea complicada. El retraso natural de la entrada es $\tau \approx 7T_s$ [5, 6]. En Corbier et al. [21] el sistema fue identificado utilizando una función de transferencia OE y añadiendo los siguientes valores atípicos. $y_{10} = 10$, $y_{50} = -11$, $y_{100} = 12$, $y_{200} = 13$, $y_{300} = -14$, $y_{400} = 15$, $y_{650} = 16$, $y_{800} = 18$ y $y_{1000} = 19$. La figura 5.24 muestra la salida original del sistema y la salida del sistema con los valores atípicos añadidos.

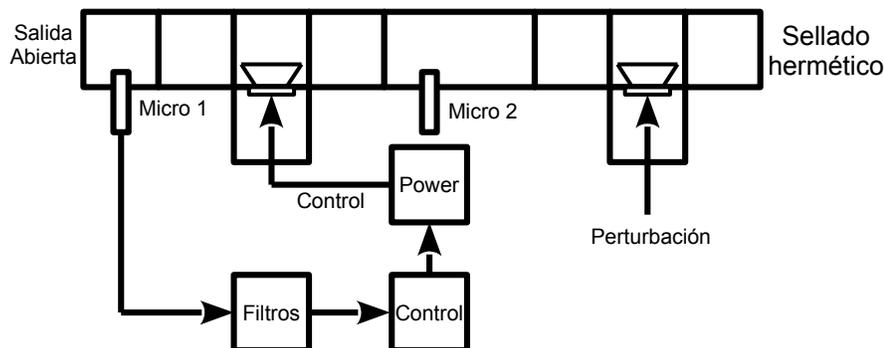


Figura 5.23: Respuesta temporal de los modelos neuronales en presencia de outliers.

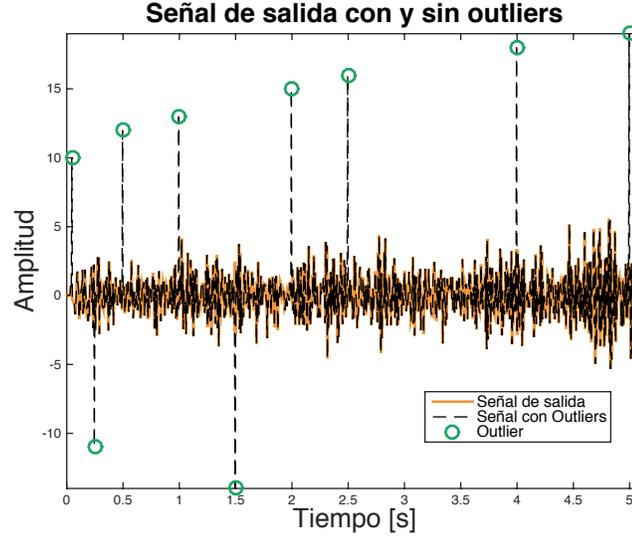


Figura 5.24: Señal de salida original y señal de salida con valores atípicos añadidos.

5.3.2. Experimentos

El sistema fue identificado considerando la red neuronal artificial mostrada en (5.9) con $\varphi_1(z) = \tanh(z)$ y $\varphi_2(z) = \varphi_3(z) = z$, $nn = 40$, $n_a = 15$, $n_b = 18$ y $n_k = 7$. En esta ocasión la magnitud de los ordenes no fueron obtenidos mediante la iteración de los mismos; se tomaron los valores encontrados en la literatura [5, 6, 21, 27, 55].

Tal como se propone en [56], los valores de los pesos sinápticos fueron inicializados iguales grupo a grupo y se ajustaron funciones de activación lineales de acuerdo con las dos suposiciones, por tanto se redujo el modelo neuronal resultando el modelo (5.10). Los 35 parámetros del modelo neuronal robusto que se propone fueron estimados utilizando el algoritmo de Levenberg-Marquadt utilizando a la función de Huber.

$$\begin{aligned}
 r_b &= \sum_{i=1}^{nn} V_{b_i} \varphi_1(J_u W_{b_i}) \\
 r_a &= \sum_{i=1}^{nn} V_{a_i} \varphi_1(J_y W_{a_i}) \\
 T &= r_b + r_a \\
 \hat{y}(k) &= X(T)
 \end{aligned} \tag{5.9}$$

$$\hat{y}(k) = \tilde{V}_B \tanh(J_u \tilde{W}_B) + \tilde{V}_A \tanh(J_y \tilde{W}_A) \quad (5.10)$$

5.3.3. Identificación del sistema

Con el fin de demostrar la eficiencia del estimador M de Huber aún y la señales no se encuentren contaminadas por valores atípicos, se realizó la estimación del sistema sin outliers. También se compara el desempeño del modelo neuronal con el modelo neuronal NLARX y el modelo neuronal RUNN. La apreciación visual se muestra en la figura 5.25 con el comportamiento temporal del sistema y en la figura 5.26 el comportamiento frecuencial de cada sistema. En la tabla 5.5 se muestran los resultados numéricos utilizando los índices de desempeño expresados en las ecuaciones 4.1-4.6 y agregando el FIT frecuencial.

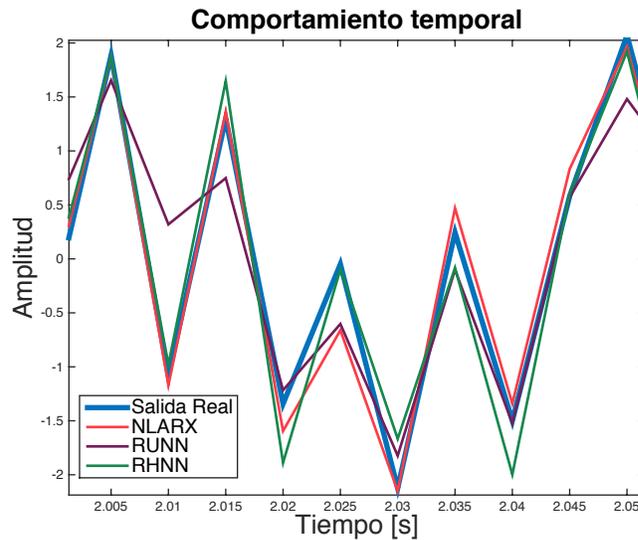
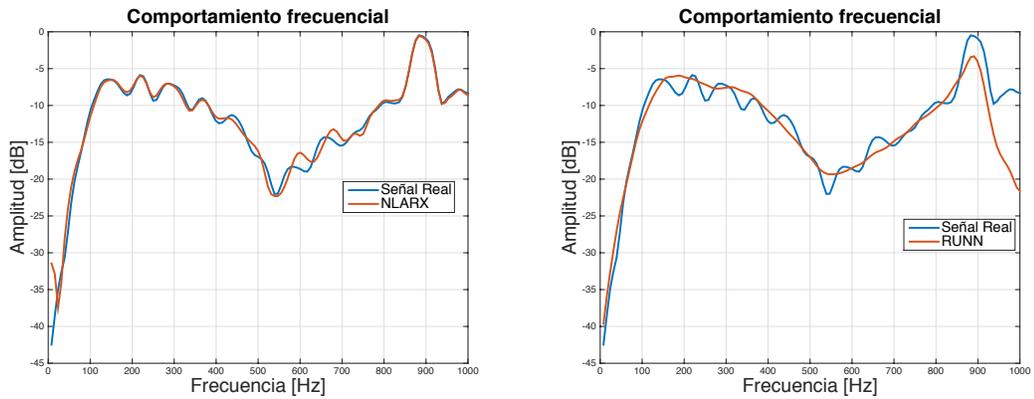


Figura 5.25: Respuesta temporal de los sistemas.

Tabla 5.5: Comparativa de modelos neuronales.

Ref.	n_p	MSE	\bar{e}	σ_N^e	IAE	ISE	FIT_F %	FIT_t %
RHNN	35	0.1923	-0.0081	0.4386	320.28	196.90	90.3557	72.56
RUNN	35	1.0668	-0.0010	1.0334	773.09	1092	64.8597	70
NLARX	684	0.0730	0.003	0.2702	205.2123	74.70	95.32	83



(a) Modelo neuronal NLARX

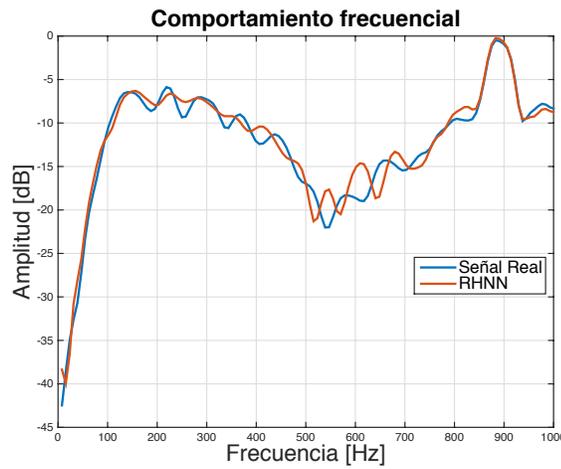
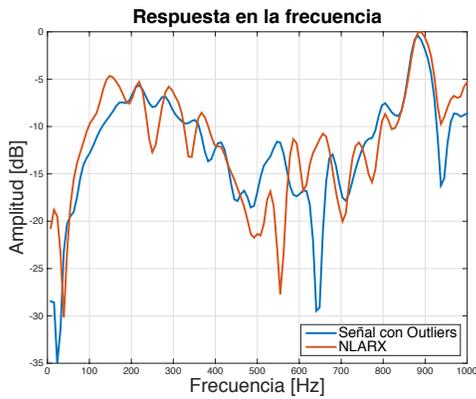
(b) Modelo neuronal con estimador $M L_2$ (c) Modelo neuronal con estimador M de Huber

Figura 5.26: Respuesta en la frecuencia de los modelos neuronales.

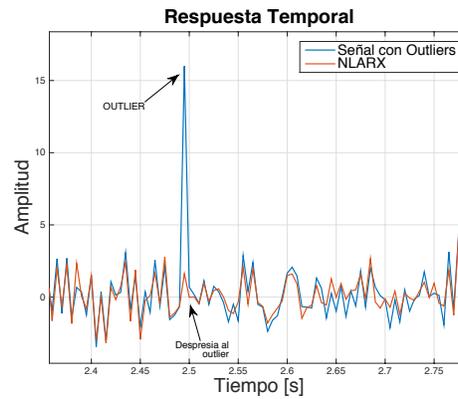
Se volvió a realizar el proceso de identificación pero introduciendo valores atípicos en la señal de salida original. Los resultados se encuentran en las figuras 5.27, 5.28 y 5.29.

Tabla 5.6: Comparativa de modelos neuronales.

Ref.	n_p	MSE	\bar{e}	σ_N^e	IAE	ISE	$FIT_F\%$	$FIT_t\%$
RHNN	35	0.9895	0.00899	0.9895	535.5	1002	78.6710	69
RUNN	35	2.1062	0.0748	2.189	1479	4542	61.5585	-0.7494
NLARX	684	1.8247	-0.0012	1.3515	622.92	1868	71.5880	35.3841

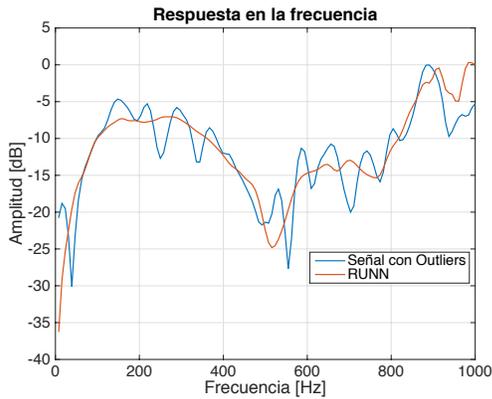


(a) Respuesta en la frecuencia de NLARX

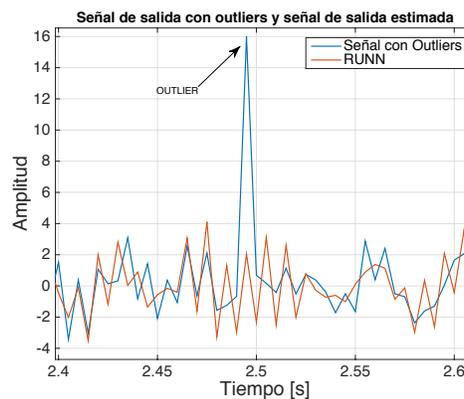


(b) Respuesta temporal de NLARX

Figura 5.27: Respuesta en la frecuencia y temporal del modelo neuronal NLARX.



(a) Respuesta en la frecuencia de RUNN



(b) Respuesta temporal de RUNN

Figura 5.28: Respuesta en la frecuencia y temporal del modelo neuronal RUNN.

En las figuras 5.27, 5.28 y 5.29 se observa que el perfil o la curva de la respuesta en la frecuencia del sistema real cambia por la introducción de outliers al sistema. En las figuras 5.27 y 5.28 se observa que disminuye el efecto del outlier, es decir, filtra al valor atípico, esto debido a que ambos modelos neuronales fueron optimizados mediante el estimador L_2 .

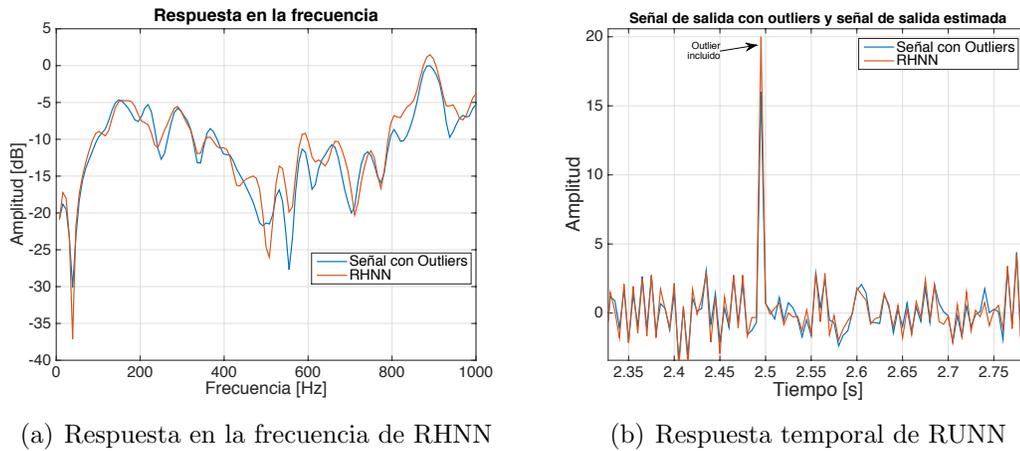


Figura 5.29: Respuesta en la frecuencia y temporal del modelo neuronal RHNN.

5.3.4. Comentarios

En la figura 5.29, se observa que el sistema RHNN tiene un desempeño aceptable en la respuesta frecuencial como en la respuesta temporal. Por otro lado, el sistema NLARX no es eficiente en ninguno de los dos casos. El modelo neuronal RUNN tampoco puede “observar” o modelar al outlier (ver figura 5.28b); esto debido a que el modelo neuronal RUNN al igual que el modelo NLARX fueron entrenados con el estimador L_2 solamente.

En el inciso (b) de las figuras 5.27, 5.28 y 5.29 se observa el comportamiento temporal de cada modelo neuronal con respecto a la salida con el valor atípico agregado. Es ahí donde se aprecia que la aportación de L_1 no es utilizada solamente en la optimización (ya que hace que la magnitud del error no crezca en gran medida a comparación con L_2), sino que también ayuda a introducir al valor atípico.

Comparando los resultados mostrados en la tabla 5.5 se observa que el modelo RHNN es capaz de competir con los estimadores que dependen solamente de L_2 . La ventaja es que tanto el modelo RHNN y RUNN contienen el mismo número de parámetros ya que son redes neuronales similares. El número de parámetros puede llegar a ser importante si se considerara el costo computacional; siempre y cuando se desarrolle un algoritmo de entrenamiento optimizado. En la tabla 5.6, se observa que tanto el modelo RHNN y el RUNN siguen teniendo el mismo número pequeño de parámetros en comparación del modelo NLARX; lo más importante de la tabla 5.6

es que tanto los índices de desempeño como el FIT temporal y frecuencial son adecuados en el modelo RHNN, es decir, el modelo RHNN que fue entrenado mediante el estimador M de Huber es mejor que los modelos neuronales entrenados mediante el estimador L_2 . Los pesos sinápticos estimados se encuentran en el apéndice C.

Capítulo 6

Conclusiones y trabajos futuros

6.1. Conclusiones

Las redes neuronales son una herramienta eficaz para la identificación de sistemas caja negra debido a su capacidad de aproximar funciones no lineales gracias a las funciones de activación, su capacidad de procesar un gran número de entradas y de salidas. Se implementaron las redes neuronales propuestas en [56] con el fin de aprovechar el enfoque de reducción que ofrecen. Se realizaron diversas pruebas con el fin de obtener una mejor estimación del sistema real.

Las redes neuronales tienen diversas aplicaciones ofreciendo resultados prometedores pero, la herramienta que les permite ser tan eficaces, es el algoritmo encargado de optimizar los pesos sinápticos que conforman a la red neuronal. A lo largo del desarrollo de esta tesis se probaron diversos algoritmos de optimización como el Gauss-Newton, el gradiente descendente y el Levenberg-Marquardt. Al probar cada uno de los algoritmos anteriores se optó por utilizar el algoritmo de Levenberg-Marquardt por su alta capacidad de encontrar rápidamente (en cuestión de épocas y no de tiempo real) los parámetros que se desean obtener. La problemática de usar este algoritmo radica en que su demostración se supone que la función que se desea minimizar es una ecuación cuadrática por lo tanto, se cree que tal vez el algoritmo de Levenberg-Marquardt es incapaz de realizar una estimación eficiente con un estimador de L_1 .

El estimador M de Huber es una herramienta eficaz para el desarrollo de la identificación de sistemas. Se demostró que la función de Huber no solo funciona para realizar la estimación de sistemas con valores atípicos sino que puede ser usada para la identificación de sistemas cuyo comportamiento es “normal”. La ventaja de utilizar la función de Huber es que se le da robustez a la estimación. Se implementó el estimador M de Huber, las redes neuronales, los algoritmos de optimización y se propuso una metodología para realizar la identificación de sistemas que contienen valores atípicos.

En el capítulo cuatro se propusieron métodos para determinar el orden de la red utilizando índices de desempeño e iterando el orden de la red neuronal. Primeramente se itera el valor de na en el intervalo que el usuario haya propuesto, al haber terminado las iteraciones de na , cambia el valor de nb hasta que nb haya igualado al límite superior del intervalo propuesto. Después de haber iterado nb en el intervalo propuesto, cambia una unidad (al igual que na y nb) el valor de nk . De esta manera se están estimando $(\overline{na} - \underline{na})(\overline{nb} - \underline{nb})(\overline{nk} - \underline{nk})$ modelos neuronales distintos. Después se realiza un análisis del valor de cada índice de desempeño calculado con base en los valores de na , nb y nk ; la selección de los ordenes utilizando índices de desempeño depende de las preferencias de cada usuario, es decir, que el modelo que se estima cumpla con una característica en especial como un comportamiento fidedigno a bajas frecuencias, altas frecuencias, temporal o buscando un balance entre todos. En el caso del estimador M de Huber, se varía el valor de κ de la misma manera que na , nb o nk . No se puede proponer un rango general o un valor en específico debido a que cada sistema es distinto al igual que sus distribuciones. En [36, 37, 73] se propone un valor igual a 1.456 para abarcar el 95% de la información siempre y cuando la distribución sea normal, este último valor se probó en uno de los experimentos (ver 5.3) y resultó eficiente pero, en el caso de los otros experimentos, no se tuvieron resultados satisfactorios. Por lo tanto, se cambió el valor de κ en un intervalo $\kappa \in (0, 2]$. Según el “tamaño de paso” será el número de modelos distintos que se estiman.

El parámetro μ en el algoritmo de Levenberg-Marquardt es indispensable debido a que es el encargado de hacer que el algoritmo conmute entre el algoritmo del gradiente

descendente (si $\mu \rightarrow \infty$) o el algoritmo de Gauss-Newton (si $\mu \rightarrow 0$). No se define el valor de μ constante, sino que varía a lo largo del número de épocas. El algoritmo que se implementó es el llamado “búsqueda y convergencia” [8]. Para el ajuste del parámetro μ_0 en el algoritmo de “búsqueda y convergencia” se consideró lo propuesto en [44], donde μ_0 se obtiene calculando la traza de la matriz Hessiana; para obtener la matriz Hessiana se realizó una aproximación con el producto de una matriz Jacobiana transpuesta por la misma matriz Jacobiana. El valor μ_0 es de suma importancia y no puede escogerse de manera aleatoria. Se realizaron diversas pruebas donde se define al valor de μ_0 de manera aleatoria, en algunos casos el algoritmo lograba converger los pesos sinápticos pero, en la mayoría de los casos, el algoritmo diverge. Por tal motivo se utilizó el método mediante el cálculo de la traza de la matriz Hessiana [44].

Se realizó una estancia en el laboratorio de tratamiento de señales e imágenes en la universidad Rennes 1 en Rennes, Francia. Durante la estancia se implementaron redes neuronales para realizar la clasificación de neonatos infectados con sepsis. La motivación de este trabajo se resume en el diagnóstico de sepsis mediante señales de respiración e intervalos RR (para mayor información ver el apéndice D).

6.2. Trabajos futuros

Durante el desarrollo de este tema de tesis se enfrentaron con diversos problemas en la implementación de programas que puedan ser un caso de estudio más. El problema principal de todo algoritmo de optimización es el valor inicial para realizar una estimación favorable.

El uso de las redes neuronales se ha vuelto fundamental en el área de control desde hace décadas. Las aplicaciones, como se mencionó con anterioridad, son extensas y diversas. Otro estudio que ha sido de interés para la comunidad científica en los últimos años es el cálculo fraccionario cuya definición precisa no existe. Se propone diseñar redes neuronales artificiales de orden fraccionario ya sea aplicado sobre la estructura de la red neuronal haciéndola dinámica o en el algoritmo de optimización con el fin abrir aún más la perspectiva del uso de las redes neuronales artificiales.

Otro caso de estudio es la selección de los ordenes na , nb y nk mediante la implementación de un algoritmo genético. En el capítulo cuatro se explicó la metodología que se implementó para seleccionar estos valores pero sería más eficiente si se ideara una forma más eficaz de realizar este proceso.

Por otro lado, existen estimadores M de orden fraccionario cuya principal ventaja es la de reducir el orden de la estimación [23] y con el fin de poder estimar sistemas de cualquier tipo de distribución.

Como se mostró en el capítulo tres, el estimador M de Huber cuenta un factor (κ) que es modificable, este valor se torna importante porque es el encargado de saber la cantidad de información que se procesará con el estimador de L_2 y de L_1 . En Huber [36], Huber et al. [37] y Zhang [73] se propone que el valor de κ debe ser igual a 1.345 para incluir dentro de la estimación por lo menos el 95 % de los datos. Se trató de utilizar este valor y funcionó en algunos casos pero en otros casos resultaba ser ineficiente el valor de 1.345. En el capítulo cuatro se propone una metodología para la selección de este parámetro que en algunas ocasiones resulta eficaz pero se propone la implementación de un algoritmo genético para estimar el valor óptimo para κ .

En este trabajo de tesis se realizó la estimación de sistemas SISO por lo que se propone en expandir las fronteras con la estimación de sistemas MIMO (para procesos químicos) y a la estimación de sistemas biológicos.

Apéndice A

Funciones de activación

Una función de activación es una regla de correspondencia que induce una modificación en la salida de la red neuronal según el argumento de la función.

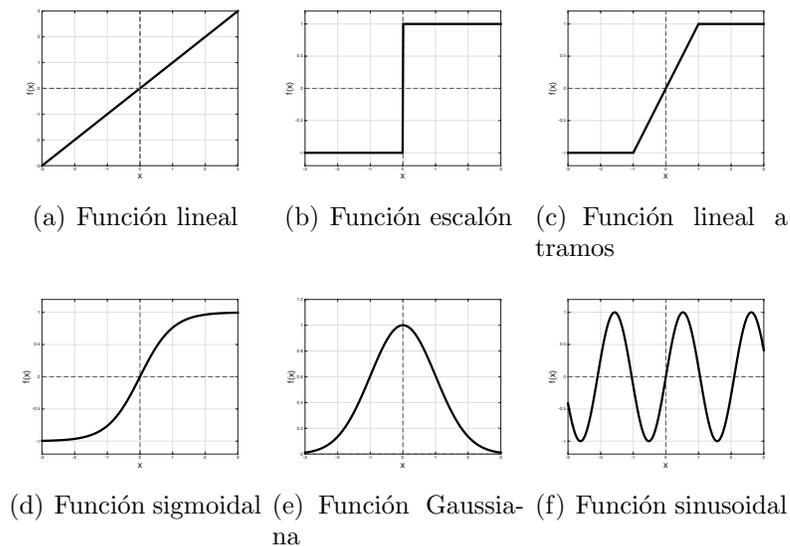


Figura A.1: Funciones de activación más comunes

El proceso de selección de las funciones de activación es de suma importancia para la metodología aplicada (enfoque de reducción) donde restringe la selección; la suposición uno establece que forzosamente las funciones de activación de por lo menos una capa de la red debe ser lineal (figura A.1(a)) para poder reducir el número de parámetros de la red neuronal $2nn - 2 - 1$.

Apéndice B

Modelo NLARX

El NLARX es un modelo ARX no lineal que se encarga de ajustar datos de estimación utilizando un número de ordenes (na , nb y nk). La estructura de la RNA puede verse en la figura B.1. El modelo de la red neuronal se muestra en (B.1).

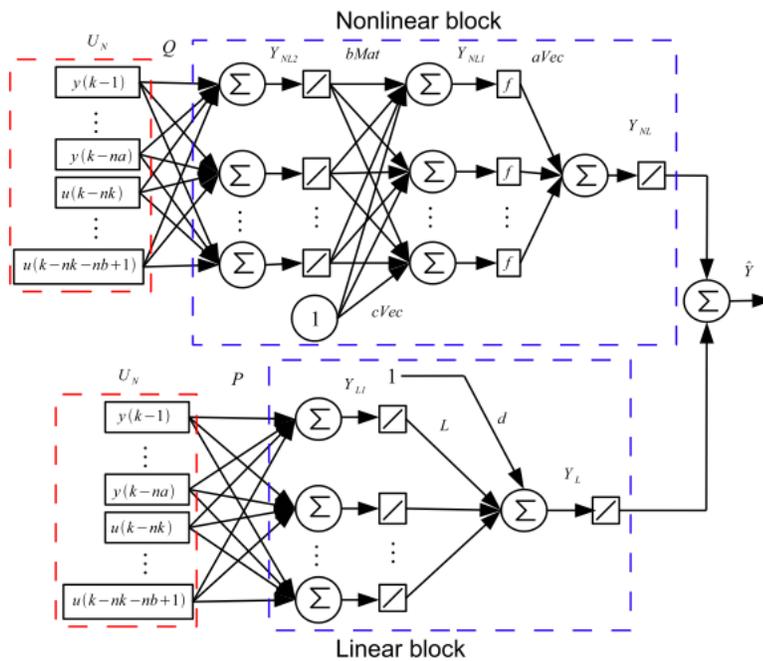


Figura B.1: Estructura de RNA NLARX.

$$\begin{aligned}
\hat{y} &= y_{NL} + y_L \\
y_{NL} &= \tanh(y_{NL1} aVec) \\
y_{NL1} &= y_{NL2} bMat + cVec \\
y_{NL2} &= U_N Q \\
y_L &= y_{L1} + d \\
y_{L1} &= U_N P
\end{aligned} \tag{B.1}$$

donde $U_N \in \mathfrak{R}^{N \times (na+nb)}$, $N \in \mathfrak{R}^n$, $Q \in \mathfrak{R}^{(na+nb)(na+nb)}$, $bMat \in \mathfrak{R}^{(na+nb)nn}$, $aVec \in \mathfrak{R}^{nn \times 1}$, $cVec \in \mathfrak{R}^{1 \times nn}$, $P \in \mathfrak{R}^{(na+nb)(na+nb)}$, $L \in \mathfrak{R}^{(na+nb) \times 1}$, $d \in \mathfrak{R}$.

Apéndice C

Pesos sinápticos para estimación y optimización de pesos sinápticos

Se realizó la estimación de tres sistemas diferentes (brazo robot, sistema Wiener-Hammerstein y ducto acústico). Cada sistema se estimó mediante estructuras neuronales diferentes por lo que varía el número de parámetros estimados (pesos sinápticos), la forma del entrenamiento, etc.

C.1. Brazo Robot

El sistema se estimó mediante la implementación de la red neuronal (C.1). Cabe mencionar que el modelo neuronal mostrado en (C.1) ha sido reducido utilizando la suposición uno y dos.

$$\begin{aligned}r_b &= \sum_{i=1}^{nn} V_{b_i} \tanh(J_u W_{b_i}) \\r_a &= \sum_{i=1}^{nn} V_{a_i} \tanh(J_y W_{a_i}) \\T &= Z_b(r_b) + Z_a(r_a) \\ \hat{y}(k) &= X \tanh(T)\end{aligned}\tag{C.1}$$

Al sustituir rb y ra en T , y T en $\hat{y}(k)$ se tiene el modelo neuronal mostrado en C.2.

$$\hat{y}(k) = X \tanh \left(\sum_{i=1}^{nn} (Z_b V_{b_i} \tanh(J_u W_{b_i}) + Z_a V_{a_i} \tanh(J_y W_{a_i})) \right) \quad (C.2)$$

donde:

$$J_u = \begin{bmatrix} u(k-2) & u(k-3) & \cdots & u(k-8) \end{bmatrix}$$

$$J_y = \begin{bmatrix} y(k-1) & y(k-2) & \cdots & y(k-8) \end{bmatrix}$$

C.1.1. Entrenamiento de la red neuronal

Optimización de X

$$X(k+1) = X(k) - [J(X)^T J(X) - \mu I] J(X)^T \quad (C.3)$$

$$J(X) = \frac{\partial E}{\partial e} \frac{\partial e}{\partial \hat{y}(k)} \frac{\partial \hat{y}(k)}{\partial X}$$

Utilizando el estimador M de Huber, la Jacobiana del parámetro X puede tener tres diferentes casos.

Caso 1: Si $ e(k) \geq \gamma$	Caso 2: Si $e(k) > \gamma$	Caso 3: Si $e(k) < -\gamma$
$J(X) = -e(k) \tanh(T)$	$J(X) = -\gamma \tanh(T)$	$J(X) = \gamma \tanh(T)$

Optimización de Z_a

$$Z_a(k+1) = Z_a(k) - [J(Z_a)^T J(Z_a) - \mu I] J(Z_a)^T \quad (C.4)$$

$$J(Z_a) = \frac{\partial E}{\partial e} \frac{\partial e}{\partial \hat{y}(k)} \frac{\partial \hat{y}(k)}{\partial T} \frac{\partial T}{\partial Z_a}$$

Caso 1: Si $|e(k)| \geq \gamma$, $J(Z_a) = -e(k) X \operatorname{sech}(T)^2 r_a$

Caso 2: Si $e(k) > \gamma$, $J(Z_a) = -\gamma X \operatorname{sech}(T)^2 r_a$

Caso 3: Si $e(k) < -\gamma$, $J(Z_a) = \gamma X \operatorname{sech}(T)^2 r_a$

Optimización de Z_b

$$Z_b(k+1) = Z_b(k) - [J(Z_b)^T J(Z_b) - \mu I] J(Z_b)^T$$

$$J(Z_b) = \frac{\partial E}{\partial e} \frac{\partial e}{\partial \hat{y}(k)} \frac{\partial \hat{y}(k)}{\partial T} \frac{\partial T}{\partial Z_b}$$
(C.5)

Caso 1: Si $|e(k)| \geq \gamma$, $J(Z_b) = -e(k)X \operatorname{sech}(T)^2 r_b$

Caso 2: Si $e(k) > \gamma$, $J(Z_b) = -\gamma X \operatorname{sech}(T)^2 r_b$

Caso 3: Si $e(k) < -\gamma$, $J(Z_b) = \gamma X \operatorname{sech}(T)^2 r_b$

Optimización de V_a

$$V_a(k+1) = V_a(k) - [J(V_a)^T J(V_a) - \mu I] J(V_a)^T$$

$$J(V_a) = \frac{\partial E}{\partial e} \frac{\partial e}{\partial \hat{y}(k)} \frac{\partial \hat{y}(k)}{\partial T} \frac{\partial T}{\partial r_a} \frac{\partial r_a}{\partial V_a}$$
(C.6)

Caso 1: Si $|e(k)| \geq \gamma$, $J(V_a) = -e(k)X \operatorname{sech}(T)^2 Z_a \sum_{i=1}^{nn} \tanh(J_y W_{a_i})$

Caso 2: Si $e(k) > \gamma$, $J(V_a) = -\gamma X \operatorname{sech}(T)^2 Z_a \sum_{i=1}^{nn} \tanh(J_y W_{a_i})$

Caso 3: Si $e(k) < -\gamma$, $J(V_a) = \gamma X \operatorname{sech}(T)^2 Z_a \sum_{i=1}^{nn} \tanh(J_y W_{a_i})$

Optimización de V_b

$$V_b(k+1) = V_b(k) - [J(V_b)^T J(V_b) - \mu I] J(V_b)^T$$

$$J(V_b) = \frac{\partial E}{\partial e} \frac{\partial e}{\partial \hat{y}(k)} \frac{\partial \hat{y}(k)}{\partial T} \frac{\partial T}{\partial r_b} \frac{\partial r_b}{\partial V_b}$$
(C.7)

Caso 1: Si $|e(k)| \geq \gamma$, $J(V_b) = -e(k)X \operatorname{sech}(T)^2 Z_b \sum_{i=1}^{nn} \tanh(J_u W_{b_i})$

Caso 2: Si $e(k) > \gamma$, $J(V_b) = -\gamma X \operatorname{sech}(T)^2 Z_b \sum_{i=1}^{nn} \tanh(J_u W_{b_i})$

Caso 3: Si $e(k) < -\gamma$, $J(V_b) = \gamma X \operatorname{sech}(T)^2 Z_b \sum_{i=1}^{nn} \tanh(J_u W_{b_i})$

Optimización de W_a

$$W_a(k+1) = W_a(k) - [J(W_a)^T J(W_a) - \mu I] J(W_a)^T \quad (C.8)$$

$$J(W_a) = \frac{\partial E}{\partial e} \frac{\partial e}{\partial \hat{y}(k)} \frac{\partial \hat{y}(k)}{\partial T} \frac{\partial T}{\partial r_a} \frac{\partial r_a}{\partial W_a}$$

Caso 1: Si $|e(k)| \geq \gamma$, $J(W_a) = -e(k)X \operatorname{sech}(T)^2 Z_a \sum_{i=1}^{nn} V_{a_i} \operatorname{sech}^2(J_y W_{a_i}) J_y^T$

Caso 2: Si $e(k) > \gamma$, $J(W_a) = -\gamma X \operatorname{sech}(T)^2 Z_a \sum_{i=1}^{nn} V_{a_i} \operatorname{sech}^2(J_y W_{a_i}) J_y^T$

Caso 3: Si $e(k) < -\gamma$, $J(W_a) = \gamma X \operatorname{sech}(T)^2 Z_a \sum_{i=1}^{nn} V_{a_i} \operatorname{sech}^2(J_y W_{a_i}) J_y^T$

Optimización de W_b

$$W_b(k+1) = W_b(k) - [J(W_b)^T J(W_b) - \mu I] J(W_b)^T \quad (C.9)$$

$$J(W_b) = \frac{\partial E}{\partial e} \frac{\partial e}{\partial \hat{y}(k)} \frac{\partial \hat{y}(k)}{\partial T} \frac{\partial T}{\partial r_b} \frac{\partial r_b}{\partial W_b}$$

Caso 1: Si $|e(k)| \geq \gamma$, $J(W_b) = -e(k)X \operatorname{sech}(T)^2 Z_b \sum_{i=1}^{nn} V_{b_i} \operatorname{sech}^2(J_u W_{b_i}) J_u^T$

Caso 2: Si $e(k) > \gamma$, $J(W_b) = -\gamma X \operatorname{sech}(T)^2 Z_b \sum_{i=1}^{nn} V_{b_i} \operatorname{sech}^2(J_u W_{b_i}) J_u^T$

Caso 3: Si $e(k) < -\gamma$, $J(W_b) = \gamma X \operatorname{sech}(T)^2 Z_b \sum_{i=1}^{nn} V_{b_i} \operatorname{sech}^2(J_u W_{b_i}) J_u^T$

C.1.2. Reducción del modelo y valores de pesos sinápticos

Si se aplican la suposición uno y dos se puede reducir el número de parámetros del modelo neuronal mostrado en (C.1 y C.2) tal como se muestra en C.10.

$$\hat{y}(k) = X \tanh \left(\tilde{Z}_b \tanh(J_u W_b) + \tilde{Z}_a \tanh(J_y W_a) \right) \quad (C.10)$$

Se debe recordar que se realizaron dos estimaciones; la primera se realizó sin outliers mientras que la segunda se realizó con valores atípicos.

También se debe considerar que debido a que solamente se seleccionó una función de activación lineal, la reducción de la red neuronal no se realizó completamente, es decir, $W_a \in \mathbb{R}^{na \times nn}$ y $W_b \in \mathbb{R}^{nb \times nn}$

- **Valor de pesos sinápticos con la estimación con valores atípicos.**

$$X = 1.0194$$

$$\tilde{Z}_a = nn \times V_a \times Z_a = 10.8611$$

$$\tilde{Z}_b = nn \times V_b \times Z_b = -7.5284$$

$$W_a = \begin{bmatrix} 0.1148 & \dots & 0.1148 \\ -0.0134 & \dots & -0.0134 \\ -0.0536 & \dots & -0.0536 \\ 0.0012 & \dots & 0.0012 \\ 0.0003 & \dots & 0.0003 \\ 0.0251 & \dots & 0.0251 \\ -0.0004 & \dots & -0.0004 \\ -0.0310 & \dots & -0.0310 \end{bmatrix}$$

$$W_b = \begin{bmatrix} -0.0327 & \dots & -0.0327 \\ 0.0388 & \dots & 0.0388 \\ 0.0030 & \dots & 0.0030 \\ -0.0218 & \dots & -0.0218 \\ 0.0086 & \dots & 0.0086 \\ 0.0326 & \dots & 0.0326 \\ -0.0331 & \dots & -0.0331 \end{bmatrix}$$

- **Valor de pesos sinápticos con la estimación sin valores atípicos.**

$$X = 5.7576$$

$$\tilde{Z}_a = nn \times V_a \times Z_a = 13.3308$$

$$\tilde{Z}_b = nn \times V_b \times Z_b = -0.0487$$

$$W_a = \begin{bmatrix} 0.0104 & \dots & 0.0104 \\ 0.0095 & \dots & 0.0095 \\ -0.0113 & \dots & -0.0113 \\ -0.0089 & \dots & -0.0089 \\ 0.0072 & \dots & 0.0072 \\ 0.0045 & \dots & 0.0045 \\ -0.0014 & \dots & -0.0014 \\ -0.0055 & \dots & -0.0055 \end{bmatrix}$$

$$W_b = \begin{bmatrix} 1.8606 & \dots & 1.8606 \\ -2.7369 & \dots & -2.7369 \\ 0.8408 & \dots & 0.8408 \\ 0.9364 & \dots & 0.9364 \\ -1.1876 & \dots & -1.1876 \\ -0.5440 & \dots & -0.5440 \\ 1.0827 & \dots & 1.0827 \end{bmatrix}$$

C.2. Sistema Wiener-Hammerstein

El sistema Wiener-Hammerstein se identificó mediante el modelo neuronal mostrado en 2.28. Debe considerarse que para este experimento se agregaron umbrales en cada neurona de capa de entrada, oculta y salida; el modelo neuronal reducido de 2.28 se muestra en 5.7. Por fines prácticos el modelo 5.7 se muestra a continuación:

$$\hat{y}(k) = \tilde{V}_B \tanh(J_u \tilde{W}_B + \tilde{W}_{BH}) + \tilde{V}_A \tanh(J_y \tilde{W}_A + \tilde{W}_{AH}) + \tilde{H}$$

donde:

$$J_u = \begin{bmatrix} u(k-4) & u(k-5) & \cdots & u(k-11) \end{bmatrix}$$

$$J_y = \begin{bmatrix} y(k-1) & y(k-2) & \cdots & y(k-4) \end{bmatrix}$$

C.2.1. Entrenamiento de la red neuronal

Optimización de W_a

$$W_a(k+1) = W_a(k) - [J(W_a)^T J(W_a) + \mu I]^{-1} J(W_a)^T \quad (\text{C.11})$$

$$J(W_a) = \frac{\partial E}{\partial e} \frac{\partial e}{\partial \hat{y}(k)} \frac{\partial \hat{y}(k)}{\partial T} \frac{\partial T}{\partial r_a} \frac{\partial r_a}{\partial W_a}$$

Caso 1: Si $|e(k)| \geq \gamma$, $J(W_a) = -e(k) X Z_a V_a \text{sech}^2(J_y W_a + W_{ah}) J_y^T$

Caso 2: Si $e(k) > \gamma$, $J(W_a) = -\gamma X Z_a V_a \text{sech}^2(J_y W_a + W_{ah}) J_y^T$

Caso 3: Si $e(k) < -\gamma$, $J(W_a) = \gamma X Z_a V_a \text{sech}^2(J_y W_a + W_{ah}) J_y^T$

Optimización de W_{ah}

$$W_{ah}(k+1) = W_{ah}(k) - [J(W_{ah})^T J(W_{ah}) + \mu I]^{-1} J(W_{ah})^T \quad (\text{C.12})$$

$$J(W_{ah}) = \frac{\partial E}{\partial e} \frac{\partial e}{\partial \hat{y}(k)} \frac{\partial \hat{y}(k)}{\partial T} \frac{\partial T}{\partial r_a} \frac{\partial r_a}{\partial W_{ah}}$$

Caso 1: Si $|e(k)| \geq \gamma$, $J(W_{ah}) = -e(k) X Z_a V_a \text{sech}^2(J_y W_a + W_{ah})$

Caso 2: Si $e(k) > \gamma$, $J(W_{ah}) = -\gamma X Z_a V_a \text{sech}^2(J_y W_a + W_{ah})$

Caso 3: Si $e(k) < -\gamma$, $J(W_{ah}) = \gamma X Z_a V_a \text{sech}^2(J_y W_a + W_{ah})$

Optimización de W_b

$$W_b(k+1) = W_b(k) - [J(W_b)^T J(W_b) + \mu I]^{-1} J(W_b)^T \quad (C.13)$$

$$J(W_b) = \frac{\partial E}{\partial e} \frac{\partial e}{\partial \hat{y}(k)} \frac{\partial \hat{y}(k)}{\partial T} \frac{\partial T}{\partial r_b} \frac{\partial r_b}{\partial W_b}$$

Caso 1: Si $|e(k)| \geq \gamma$, $J(W_b) = -e(k) X Z_b V_b \operatorname{sech}^2(J_u W_b) J_u^T$

Caso 2: Si $e(k) > \gamma$, $J(W_b) = -\gamma X Z_b V_b \operatorname{sech}^2(J_u W_b) J_u^T$

Caso 3: Si $e(k) < -\gamma$, $J(W_b) = \gamma X Z_b V_b \operatorname{sech}^2(J_u W_b) J_u^T$

Optimización de W_{bh}

$$W_{bh}(k+1) = W_{bh}(k) - [J(W_{bh})^T J(W_{bh}) + \mu I]^{-1} J(W_{bh})^T \quad (C.14)$$

$$J(W_{bh}) = \frac{\partial E}{\partial e} \frac{\partial e}{\partial \hat{y}(k)} \frac{\partial \hat{y}(k)}{\partial T} \frac{\partial T}{\partial r_b} \frac{\partial r_b}{\partial W_{bh}}$$

Caso 1: Si $|e(k)| \geq \gamma$, $J(W_{bh}) = -e(k) X Z_b V_b \operatorname{sech}^2(J_u W_b + W_{bh})$

Caso 2: Si $e(k) > \gamma$, $J(W_{bh}) = -\gamma X Z_b V_b \operatorname{sech}^2(J_u W_b + W_{bh})$

Caso 3: Si $e(k) < -\gamma$, $J(W_{bh}) = \gamma X Z_b V_b \operatorname{sech}^2(J_u W_b + W_{bh})$

Optimización de V_a

$$V_a(k+1) = V_a(k) - [J(V_a)^T J(V_a) + \mu I]^{-1} J(V_a)^T \quad (C.15)$$

$$J(V_a) = \frac{\partial E}{\partial e} \frac{\partial e}{\partial \hat{y}(k)} \frac{\partial \hat{y}(k)}{\partial T} \frac{\partial T}{\partial r_a} \frac{\partial r_a}{\partial V_a}$$

Caso 1: Si $|e(k)| \geq \gamma$, $J(V_a) = -e(k) X Z_a \tanh(J_y W_a + W_{ah})$

Caso 2: Si $e(k) > \gamma$, $J(V_a) = -\gamma X Z_a \tanh(J_y W_a + W_{ah})$

Caso 3: Si $e(k) < -\gamma$, $J(V_a) = \gamma X Z_a \tanh^2(J_y W_a + W_{ah})$

Optimización de V_{ah}

$$V_{ah}(k+1) = V_{ah}(k) - [J(V_{ah})^T J(V_{ah}) + \mu I]^{-1} J(V_{ah})^T$$

$$J(V_{ah}) = \frac{\partial E}{\partial e} \frac{\partial e}{\partial \hat{y}(k)} \frac{\partial \hat{y}(k)}{\partial T} \frac{\partial T}{\partial r_a} \frac{\partial r_a}{\partial V_{ah}}$$
(C.16)

Caso 1: Si $|e(k)| \geq \gamma$, $J(V_{ah}) = -e(k)XZ_a$

Caso 2: Si $e(k) > \gamma$, $J(V_{ah}) = -\gamma XZ_a$

Caso 3: Si $e(k) < -\gamma$, $J(V_{ah}) = \gamma XZ_a$

Optimización de V_b

$$V_b(k+1) = V_b(k) - [J(V_b)^T J(V_b) + \mu I]^{-1} J(V_b)^T$$

$$J(V_b) = \frac{\partial E}{\partial e} \frac{\partial e}{\partial \hat{y}(k)} \frac{\partial \hat{y}(k)}{\partial T} \frac{\partial T}{\partial r_b} \frac{\partial r_b}{\partial V_b}$$
(C.17)

Caso 1: Si $|e(k)| \geq \gamma$, $J(V_b) = -e(k)XZ_b \tanh(J_u W_b + W_{bh})$

Caso 2: Si $e(k) > \gamma$, $J(V_b) = -\gamma XZ_b \tanh(J_u W_b + W_{bh})$

Caso 3: Si $e(k) < -\gamma$, $J(V_b) = \gamma XZ_b \tanh^2(J_u W_b + W_{bh})$

Optimización de V_{bh}

$$V_{bh}(k+1) = V_{bh}(k) - [J(V_{bh})^T J(V_{bh}) + \mu I]^{-1} J(V_{bh})^T$$

$$J(V_{bh}) = \frac{\partial E}{\partial e} \frac{\partial e}{\partial \hat{y}(k)} \frac{\partial \hat{y}(k)}{\partial T} \frac{\partial T}{\partial r_b} \frac{\partial r_b}{\partial V_{bh}}$$
(C.18)

Caso 1: Si $|e(k)| \geq \gamma$, $J(V_{bh}) = -e(k)XZ_b$

Caso 2: Si $e(k) > \gamma$, $J(V_{bh}) = -\gamma XZ_b$

Caso 3: Si $e(k) < -\gamma$, $J(V_{bh}) = \gamma XZ_b$

Optimización de Z_a

$$Z_a(k+1) = Z_a(k) - [J(Z_a)^T J(Z_a) + \mu I]^{-1} J(Z_a)^T \quad (C.19)$$

$$J(Z_a) = \frac{\partial E}{\partial e} \frac{\partial e}{\partial \hat{y}(k)} \frac{\partial \hat{y}(k)}{\partial T} \frac{\partial T}{\partial Z_a}$$

Caso 1: Si $|e(k)| \geq \gamma$, $J(Z_b) = -e(k)Xr_a$

Caso 2: Si $e(k) > \gamma$, $J(Z_b) = -\gamma Xr_a$

Caso 3: Si $e(k) < -\gamma$, $J(Z_b) = \gamma Xr_a$

Optimización de Z_b

$$Z_b(k+1) = Z_b(k) - [J(Z_b)^T J(Z_b) + \mu I]^{-1} J(Z_b)^T \quad (C.20)$$

$$J(Z_b) = \frac{\partial E}{\partial e} \frac{\partial e}{\partial \hat{y}(k)} \frac{\partial \hat{y}(k)}{\partial T} \frac{\partial T}{\partial Z_b}$$

Caso 1: Si $|e(k)| \geq \gamma$, $J(Z_b) = -e(k)Xr_b$

Caso 2: Si $e(k) > \gamma$, $J(Z_b) = -\gamma Xr_b$

Caso 3: Si $e(k) < -\gamma$, $J(Z_b) = \gamma Xr_b$

Optimización de Z_{bh}

$$Z_{bh}(k+1) = Z_{bh}(k) - [J(Z_{bh})^T J(Z_{bh}) + \mu I]^{-1} J(Z_{bh})^T \quad (C.21)$$

$$J(Z_{bh}) = \frac{\partial E}{\partial e} \frac{\partial e}{\partial \hat{y}(k)} \frac{\partial \hat{y}(k)}{\partial T} \frac{\partial T}{\partial Z_{bh}}$$

Caso 1: Si $|e(k)| \geq \gamma$, $J(Z_{bh}) = -e(k)X$

Caso 2: Si $e(k) > \gamma$, $J(Z_{bh}) = -\gamma X$

Caso 3: Si $e(k) < -\gamma$, $J(Z_{bh}) = \gamma X$

Optimización de X

$$X(k+1) = X(k) - [J(X)^T J(X) + \mu I]^{-1} J(X)^T$$

$$J(X) = \frac{\partial E}{\partial e} \frac{\partial e}{\partial \hat{y}(k)} \frac{\partial \hat{y}(k)}{\partial X}$$
(C.22)

Caso 1: Si $|e(k)| \geq \gamma$, $J(X) = -e(k)T$

Caso 2: Si $e(k) > \gamma$, $J(X) = -\gamma T$

Caso 3: Si $e(k) < -\gamma$, $J(X) = \gamma T$

- Valor de pesos sinápticos de la estimación con valores atípicos.

$$\tilde{V}_A = -61.7852$$

$$\tilde{W}_{ah} = 0.0383$$

$$\tilde{V}_B = -0.4379$$

$$\tilde{W}_{bh} = 0.0507$$

$$\tilde{H} = 0.9645$$

$$\tilde{W}_A = \begin{bmatrix} 0.0171 & 0.0577 & -0.0202 & -0.0493 \end{bmatrix}^T$$

$$\tilde{W}_B = \begin{bmatrix} 0.0091 & 0.0415 & 0.0466 & 0.0379 & 0.0225 & 0.0330 & 0.0988 \end{bmatrix}^T$$

- Valor de pesos sinápticos con la estimación sin valores atípicos.

$$\tilde{V}_A = -25.0730$$

$$\tilde{W}_{ah} = 0.003$$

$$\tilde{V}_B = 0.0142$$

$$\tilde{W}_{bh} = 0.3344$$

$$\tilde{H} = -0.0787$$

$$\tilde{W}_A = \begin{bmatrix} 0.0859 & -0.0497 & -0.0048 & 0.0079 \end{bmatrix}^T$$

$$\tilde{W}_B = \begin{bmatrix} 0.4679 & -0.0754 & -0.3893 & 0.6033 & 0.4790 & -0.7535 & 0.5499 \end{bmatrix}^T$$

C.3. Ducto acústico

El modelo neuronal que se utilizó para la identificación del sistema se muestra en (C.23). El modelo (C.23) corresponde a la reducción bajo las suposiciones uno y dos del modelo neuronal mostrado en (5.9).

$$\hat{y}(k) = \tilde{V}_B \tanh(J_u \tilde{W}_B) + \tilde{V}_A \tanh(J_y \tilde{W}_A) \quad (\text{C.23})$$

donde:

$$J_u = \begin{bmatrix} u(k-7) & u(k-8) & \cdots & u(k-25) \end{bmatrix}$$

$$J_y = \begin{bmatrix} y(k-1) & y(k-2) & \cdots & y(k-15) \end{bmatrix}$$

C.3.1. Entrenamiento de la red neuronal

Optimización de W_a

$$W_a(k+1) = W_a(k) - [J(W_a)^T J(W_a) - \mu I] J(W_a)^T \quad (\text{C.24})$$

$$J(W_a) = \frac{\partial E}{\partial e} \frac{\partial e}{\partial \hat{y}(k)} \frac{\partial \hat{y}(k)}{\partial T} \frac{\partial T}{\partial r_a} \frac{\partial r_a}{\partial W_a}$$

Caso 1: Si $|e(k)| \geq \gamma$, $J(W_a) = -e(k) V_a \text{sech}^2(J_y W_a) J_y^T$

Caso 2: Si $e(k) > \gamma$, $J(W_a) = -\gamma V_a \text{sech}^2(J_y W_a) J_y^T$

Caso 3: Si $e(k) < -\gamma$, $J(W_a) = \gamma V_a \text{sech}^2(J_y W_a) J_y^T$

Optimización de W_b

$$W_b(k+1) = W_b(k) - [J(W_b)^T J(W_b) - \mu I] J(W_b)^T \quad (\text{C.25})$$

$$J(W_b) = \frac{\partial E}{\partial e} \frac{\partial e}{\partial \hat{y}(k)} \frac{\partial \hat{y}(k)}{\partial T} \frac{\partial T}{\partial r_b} \frac{\partial r_b}{\partial W_b}$$

Caso 1: Si $|e(k)| \geq \gamma$, $J(W_b) = -e(k) V_b \text{sech}^2(J_u W_b) J_u^T$

Caso 2: Si $e(k) > \gamma$, $J(W_b) = -\gamma V_b \text{sech}^2(J_u W_b) J_u^T$

Caso 3: Si $e(k) < -\gamma$, $J(W_b) = \gamma V_b \text{sech}^2(J_u W_b) J_u^T$

Optimización de V_a

$$V_a(k+1) = V_a(k) - [J(V_a)^T J(V_a) - \mu I] J(V_a)^T \quad (C.26)$$

$$J(V_a) = \frac{\partial E}{\partial e} \frac{\partial e}{\partial \hat{y}(k)} \frac{\partial \hat{y}(k)}{\partial T} \frac{\partial T}{\partial r_a} \frac{\partial r_a}{\partial V_a}$$

Caso 1: Si $|e(k)| \geq \gamma$, $J(V_a) = -e(k) \tanh(J_y W_a)$

Caso 2: Si $e(k) > \gamma$, $J(V_a) = -\gamma \tanh(J_y W_a)$

Caso 3: Si $e(k) < -\gamma$, $J(V_a) = \gamma \tanh(J_y W_a)$

Optimización de V_b

$$V_b(k+1) = V_b(k) - [J(V_b)^T J(V_b) - \mu I] J(V_b)^T \quad (C.27)$$

$$J(V_b) = \frac{\partial E}{\partial e} \frac{\partial e}{\partial \hat{y}(k)} \frac{\partial \hat{y}(k)}{\partial T} \frac{\partial T}{\partial r_b} \frac{\partial r_b}{\partial V_b}$$

Caso 1: Si $|e(k)| \geq \gamma$, $J(V_b) = -e(k) \tanh(J_y W_b)$

Caso 2: Si $e(k) > \gamma$, $J(V_b) = -\gamma \tanh(J_y W_b)$

Caso 3: Si $e(k) < -\gamma$, $J(V_b) = \gamma \tanh(J_y W_b)$

- Valor de pesos sinápticos de la estimación con valores atípicos.

$$\tilde{V}_A = 3.2548$$

$$\tilde{V}_B = 3.7876$$

$$\tilde{W}_A = \begin{bmatrix} -0.1608 \\ 0.1680 \\ -0.0346 \\ -0.1040 \\ 0.0025 \\ -0.0095 \\ 0.0067 \\ -0.0151 \\ 0.0078 \\ -0.0089 \\ -0.0030 \\ -0.0012 \\ -0.0031 \\ 0.0145 \\ -0.0016 \end{bmatrix} \quad \tilde{W}_B = \begin{bmatrix} -0.0288 \\ -0.0498 \\ 0.0288 \\ 0.0455 \\ 0.0234 \\ 0.0286 \\ -0.0162 \\ 0.0029 \\ 0.0020 \\ -0.0062 \\ -0.0064 \\ -0.0168 \\ 0.0058 \\ -0.0194 \\ -0.0039 \\ -0.0122 \\ 0.0050 \\ -0.0016 \end{bmatrix}$$

- Valor de pesos sinápticos de la estimación sin valores atípicos.

$$\tilde{V}_A = 6.4375$$

$$\tilde{V}_B = 3.3585$$

$$\tilde{W}_A = \begin{bmatrix} -0.0779 \\ 0.0742 \\ -0.0860 \\ -0.1231 \\ 0.0074 \\ -0.0091 \\ -0.0563 \\ -0.0478 \\ 0.0076 \\ -0.0175 \\ -0.0420 \\ -0.0130 \\ -0.0028 \\ -0.0150 \\ -0.0140 \end{bmatrix} \quad \tilde{W}_B = \begin{bmatrix} -0.0334 \\ -0.0569 \\ 0.0312 \\ 0.0390 \\ -0.0028 \\ 0.0299 \\ 0.0099 \\ 0.0199 \\ 0.0127 \\ -0.0008 \\ 0.0063 \\ -0.0163 \\ 0.0049 \\ -0.0177 \\ -0.0089 \\ -0.0105 \\ 0.0014 \\ -0.0075 \end{bmatrix}$$

Apéndice D

Estancia

Se realizó una estancia en la ciudad de Rennes, Francia con el fin de aumentar habilidades y comprobar que los algoritmos desarrollados hasta el momento funcionaran adecuadamente.

Durante la estancia se manejaron herramientas que para este tema de tesis podían ser de utilidad como estructuras de redes neuronales diferentes a las antes vistas, clasificación de patrones mediante el uso de redes neuronales, entre otros.



Centro Nacional de Investigación y
Desarrollo Tecnológico

Y

Université de Rennes 1

DETECCIÓN DE SEPSIS EN NEONATOS

Reporte de Estancia

Presenta:

Ing. Carlos Jesús Zúñiga Aguilar

Rennes, Francia - Cuernavaca, México. Septiembre 2015

Índice

- Índice** **2**
- 1. INTRODUCTION** **3**
- 2. RESAMPLE** **3**
 - 2.1. Electrocardiogram and R wave 3
 - 2.2. RR interval 4
- 3. ARX modelos** **5**
- 4. Extreme Learning Machine Classifier** **7**
- 5. CONCLUSION** **9**
- 6. REFERENCES** **10**

1. INTRODUCTION

Patern infants (born before 37 weeks of gestation) exhibit a very unstable breathing, typified by apneas or pauses in ventilation that may be accompanied by bradycardia, a decrease of the heart beat rythm. This phenomenon is called as apnea of prematurity (AOP) and is consequence for a underdeveloped brain and lungs. The AOP may appear spontaneously, but it can also be provoked or become more severe when other pathologies (specially spesis) are present.

The continuous monitoring of breathing and cardiac frequencies are of crucial importance to an early intervention and avoid or palliate the associated risks with recurrent apnea-bradycardia. We can see our contribution classifying and diagnosticating using the breathing signal and RR signal acquired in neonatal intensive care units (NICUs).

2. RESAMPLE

2.1. Electrocardiogram and R wave

The QRS complex is a name for the combination of three of the graphical deflections seen on a typical electrocardiogram (ECG). It is usually the central and most visually obvious part of the tracing. It corresponds to the depolarization of the right and left ventricles of the human heart. In adults, it normally lasts 0.06–0.10 s; in children and during physical activity, it may be shorter.

Typically an ECG has five deflections, arbitrarily named "P" to "T" waves. The Q, R, and S waves occur in rapid succession, do not all appear in all leads, and reflect a single event, and thus are usually considered together. A Q wave is any downward deflection after the P wave. An R wave follows as an upward deflection, and the S wave is any downward deflection after the R wave. The T wave follows the S wave, and in some cases an additional U wave follows the T wave. The R wave is the largest wave in the QRS complex (see Fig. 1).

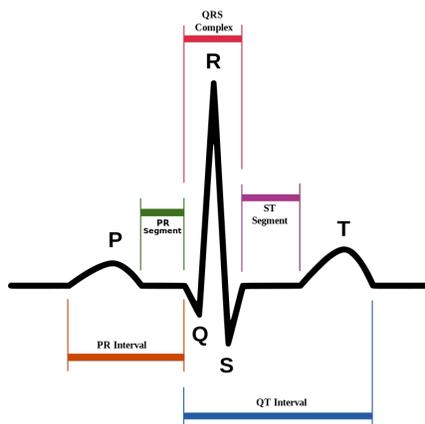


Fig. 1: Schematic representation of normal ECG

2.2. RR interval

The time intervals between consecutive heart beats are customarily measured in the electrocardiogram from the beginning of a QRS complex to the beginning of the next QRS complex, so these intervals might be called QQ intervals, but they are conventionally named RR intervals (see Fig. 2).

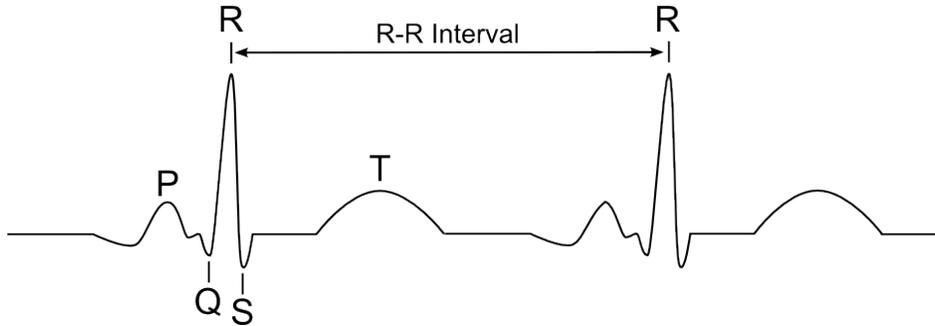


Fig. 2: Representation of the RR interval

As we said before, we used the rr and respiratory signal to make a classifier. First, we used an algorithm to detect the R wave (see Fig. 3) and then we made a difference between an actual R wave and the future R wave to create the RR interval (see Fig. 4). We did a resample because the sample frequencies was different. For the ECG signal was $1kHz$ and for the respiration signal was $62,5Hz$. To fix the above, we found a relation between the frequency of the ECG signal and the respiration signal and that is 16. After, I made the resample of the respiratory signal rising the sample frequency 16 times. Then, we took the instant of time of the R wave to make a sample with the amplitude of the respiratory signal. This allows us have the same lengths of data of the RR signal and the respiratory signal.

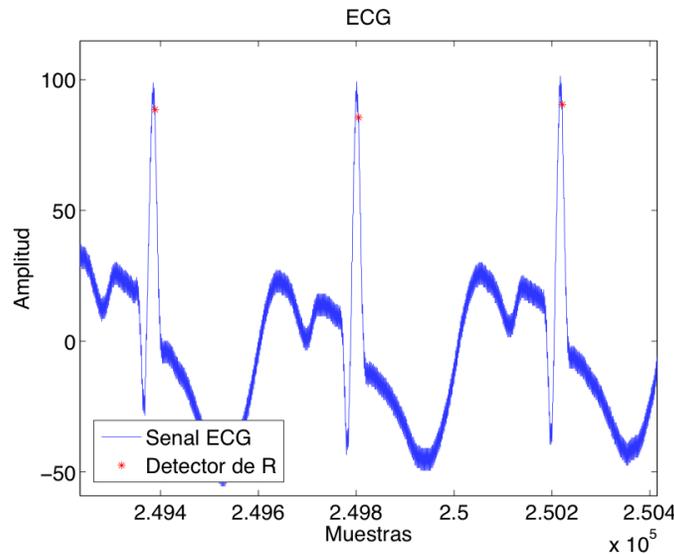


Fig. 3: R wave detector.

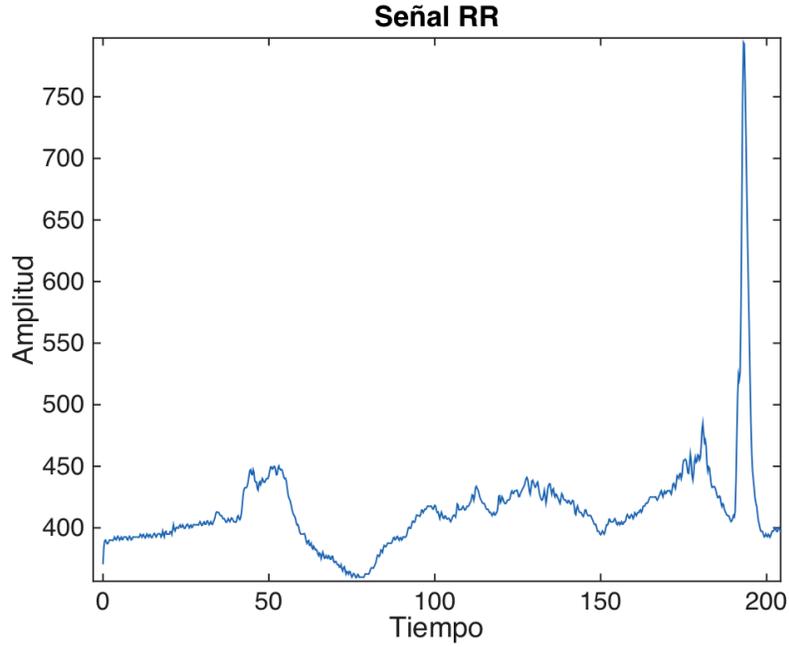


Fig. 4: RR interval.

3. ARX modeles

We can express an ARX model as:

$$y(t) + a_1 y(t-1) + \dots + a_{na} y(t-na) = u(t-n_k) + b_1 u(t-1-n_k) + \dots + b_{nb} u(t-n_b-n_k) + v(t) \quad (1)$$

where, a_n and b_n are a coefficients, $y(t)$ is the output signal, $u(t)$ is the input signal and $v(t)$ is the disturbance. Using a q^{-1} transformation we can represent the arx model in a compact form like (2)

$$y(t) = \frac{B(q^{-1})}{A(q^{-1})} u(t - n_k) + \frac{1}{A(q^{-1})} v(t) \quad (2)$$

to select the values of na, nb and nk, we set the value of nb and nk and we change the value of na 15 times. After the 15 times we change the value of nb and so on with nk. After that process we choose just one model of 2250 using the final prediction error criterion. We can see the results on the Fig. 5.

Because we did not have good results with the linear ARX model we proposed a Non-Linear ARX model. This type model uses and neural network architecture (see Fig. 6). We show in (3) the model for this architecure.

$$\begin{aligned} \hat{y} &= y_{NL} + y_L \\ y_{NL} &= \tanh(y_{NL1}) * aVec \\ y_{NL1} &= y_{NL2} * bMat + cVec \\ y_{NL2} &= U_N Q \\ y_L &= Y_{L1} * L + d \\ Y_{L1} &= U_N * P \end{aligned} \quad (3)$$

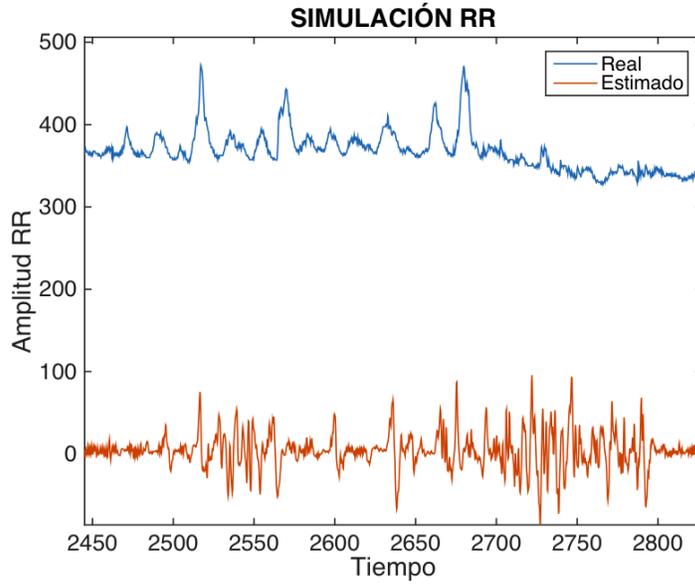


Fig. 5: Results of the Linear ARX Model.

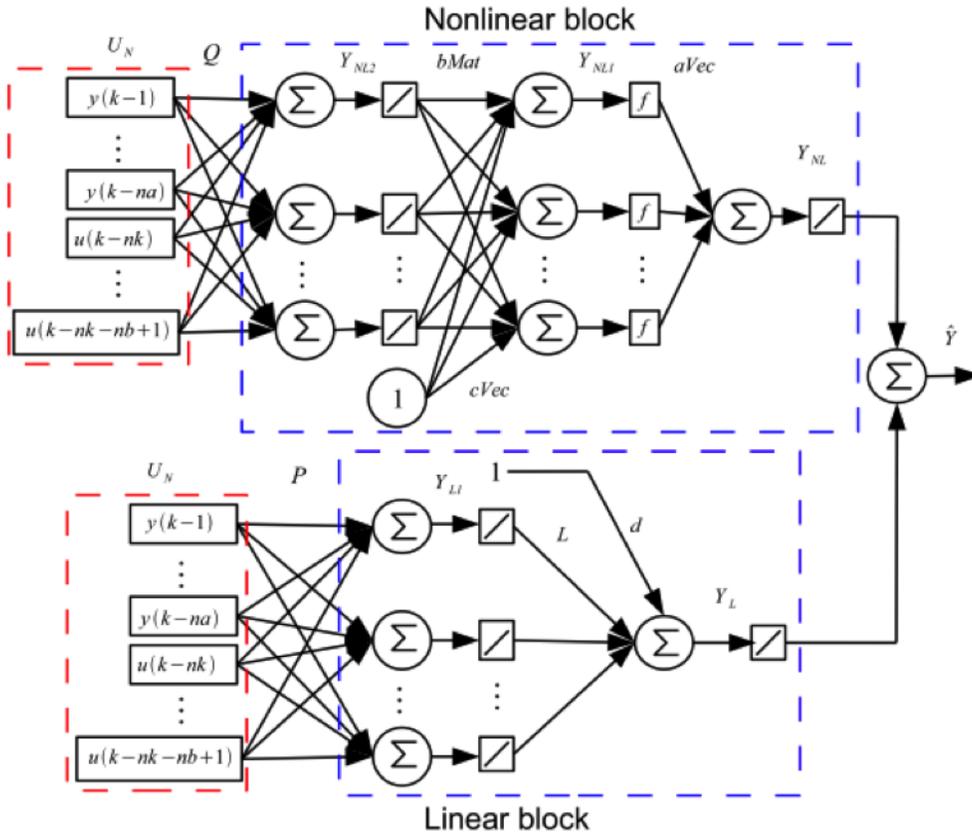


Fig. 6: Non-Linear ARX architecture

where $U_N \in \mathfrak{R}^{n_a+n_b}$, $Q \in \mathfrak{R}^{(n_a+n_b)(n_a+n_b)}$, $P \in \mathfrak{R}^{(n_a+n_b)(n_a+n_b)}$, $L \in \mathfrak{R}^{n_a+n_b}$, $aVec \in \mathfrak{R}^{nn}$, $bMat \in \mathfrak{R}^{(n_a+n_b)nn}$, $cVec \in \mathfrak{R}^{nn}$, $d \in \mathfrak{R}$ and nn is the number of neurons.

We did the same selection process of n_a , n_b and n_k . The results of this model are shown on Fig. 6.

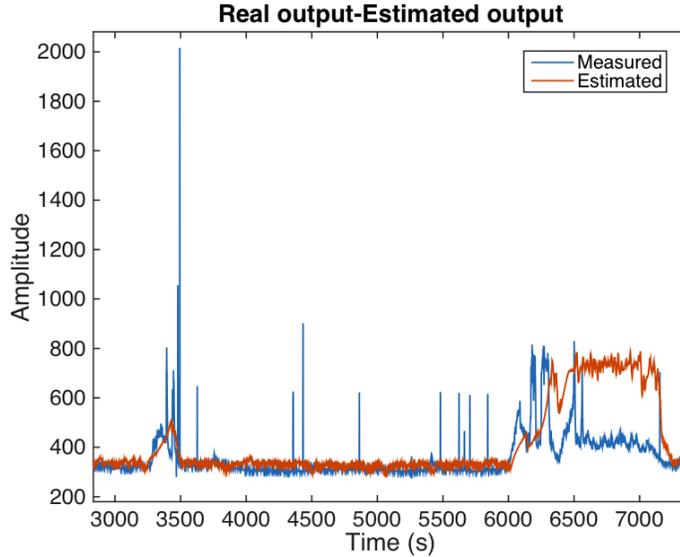


Fig. 7: Results of the non-linear ARX

The results are promising but the problem of this type of model is the number of parameters that we need to estimate. Because of that, we proposed another architecture of neural network.

4. Extreme Learning Machine Classifier

The learning speed of feedforward neural network is in general far slower than required and it has been a major bottleneck in their application. In this case we proposed the extreme learning machine algorithm for a single-hidden layer feedforward neural networks which randomly chooses hidden nodes and analytically determines the output weights. The Fig. 7 shows the architecture for the past neural network and in (4) the neuromodel of that neural network.

$$\hat{\mathbf{Y}} = \begin{bmatrix} \hat{y}_0 \\ \hat{y}_1 \end{bmatrix} = V * \varphi(W * X^T + \theta) \quad (4)$$

where $W \in \mathfrak{R}^{HN(N+M)}$, $V \in \mathfrak{R}^{2HN}$, $\theta \in \mathfrak{R}^{(HN)S}$ and $X \in \mathfrak{R}^{S(N+M)}$.

The input of the neural network was dynamical windows with five minutes of RR interval at 4Hz and the respiration signal. The output was one (if the baby has sepsis) and zero (if the baby has not sepsis). We can see the results of this methodology in the table 1 and table 2.

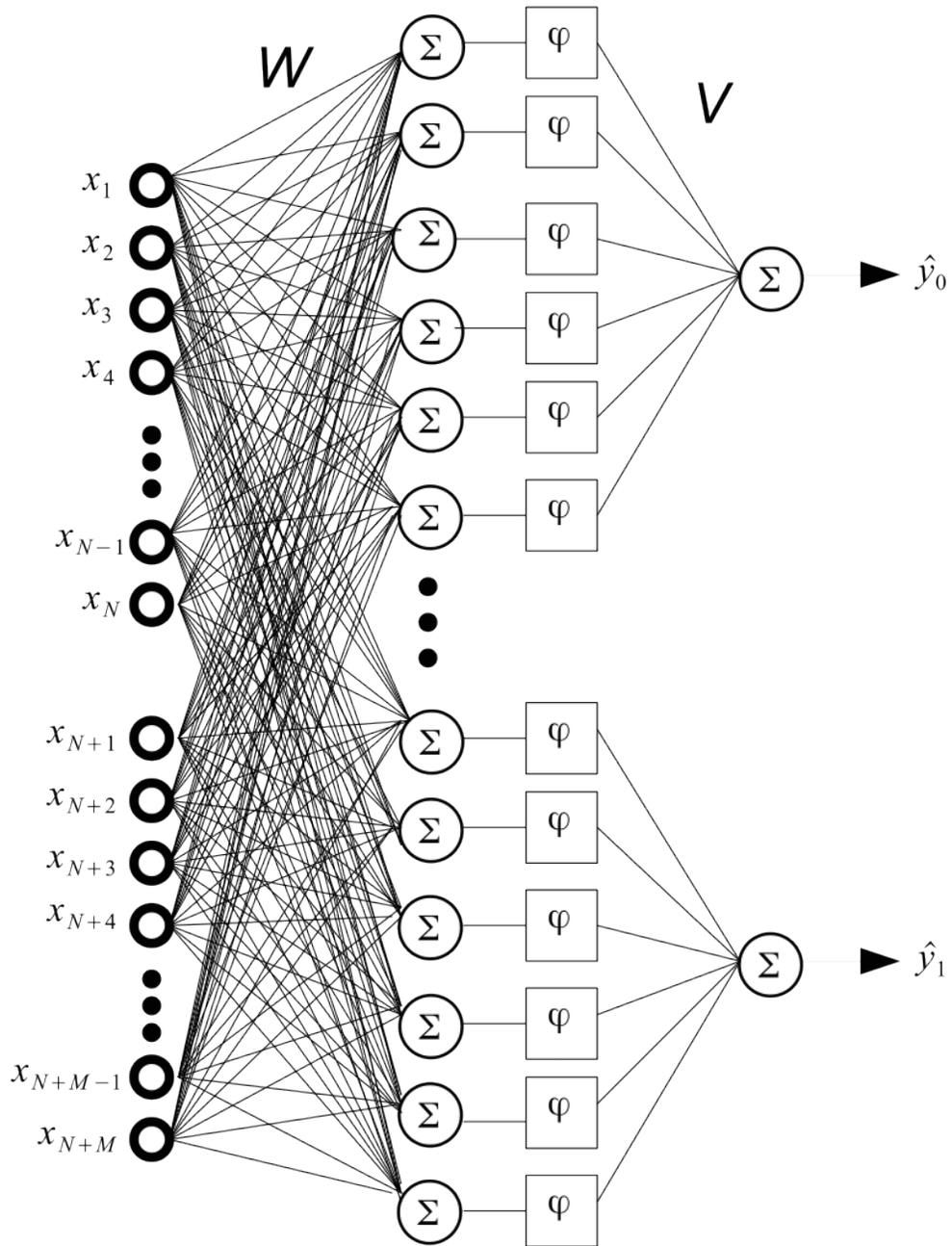


Fig. 8: Extreme Learning Machine Neural Network structure

Table 1: Classifier results for the patient one to nine.

PATIENT	1	2	3	4	5	6	7	8	9
SEPSIS	1	0	0	1	1	1	1	1	1
Classifier mean	0.7	0.1	0.001	0.4	0.97	0.95	0.69	0.89	0.75

Table 2: Classifier results for the patient ten to eighteen.

PATIENT	10	11	12	13	14	15	16	17	18
SEPSIS	0	1	1	1	1	1	0	1	0
Classifier mean	0.38	0.72	0.6	0.97	0.65	0.68	0.1	0.03	0.002

5. CONCLUSION

They wanted to get the mathematical model to describe the behavior of the RR-interval in order to know what the behavior of patients with sepsis and without sepsis. In order to diagnose patients with sepsis . The results of this methodology were not efficient so we made mobile windows in order to extend the information captured and have a better training in the ELM neural network. This methodology was efficient and the results can be seen in Table 1 and Table 2 where we compute the mean of the outpur for the neural network. If the mean of the output of the RNA exceeds the value of 0.5 meant that the patient has sepsis.

I am very grateful to the LTSI for the opportunity given to me as Dr. Hector Romero and Dr. Alfredo Hernandez for allowing me to work with them.

6. REFERENCES

[1] G. H. Å, Q. Zhu, and C. Siew, “Extreme learning machine : Theory and applications,” vol. 70, pp. 489–501, 2006.

[2] G. C. X. Navarro, F. Porée, A. Beuchée, “Denoising preterm EEG by signal decomposition and adaptive filtering : A comparative study Denoising preterm EEG by signal decomposition and adaptive filtering : A comparative study,”

[3] G. C. X. Navarro, F. Porée, A. Beuchée, “Artifact rejection and cycle detection in immature breathing : Application to the early detection of neonatal sepsis Artifact rejection and cycle detection in immature breathing : application to the early detection of neonatal sepsis,”

[4] V. Le Rolle, “Mathematical Modeling of Respiratory System Mechanics in the Newborn Lamb,”

Rennes, September, 12th 2015

To whom it may concern

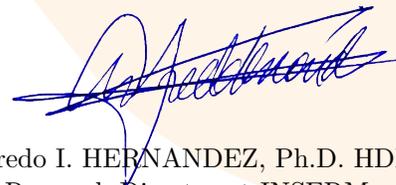
I hereby attest of the serious involvement of Mr. Carlos Jesús Zúñiga Aguilar during his internship at the Laboratoire Traitement du Signal et de l'Image of the University of Rennes 1 (LTSI), which is a research unit of the French National Institutes of Health and Medical Research (INSERM), located in Rennes, France.

Mr. Zúñiga spent three months (June 15th, 2015 - September 9th, 2015) in our laboratory, under my supervision and that of Dr. Hector Romero, to perform a research internship in the framework of a research program focused on signal processing for preterm newborn monitoring. His work was focused on cardiac and respiratory signal processing, with emphasis on the detection of adverse events (sepsis) using neural networks. He had the ability to complete the tasks that were appointed to him during this internship. At LTSI, he had the opportunity to learn how advanced signal processing and machine learning techniques may be applied to the biomedical engineering field.

Mr. Zúñiga adapted himself well, interacting with other members of the laboratory mainly in English.

I am at your disposal if you need further references.

Best regards,



Alfredo I. HERNANDEZ, Ph.D. HDR.
Research Director at INSERM.

French National Institutes of Health and Medical Research.

Bibliografía

- [1] Outlier. <https://en.wikipedia.org/wiki/Outlier>. Acceso: 10-05-2016.
- [2] Robot data. <ftp://ftp.esat.kuleuven.be/pub/SISTA/data/mechanical>. Acceso: 10-05-2016.
- [3] Hirotugu Akaike. Canonical correlation analysis of time series and the use of an information criterion. *Computational Methods for Modeling of Nonlinear Systems*, 126:27, 1977.
- [4] Hirotugu Akaike. On the likelihood of a time series model. *The Statistician*, págs. 217–235, 1978.
- [5] V. M. Alvarado y J. C. Carmona. Active noise control of a duct using robust control theory. *IEEE Transactions on Control Systems Technology*, 8(6):930–938, 2000. ISSN 10636536. doi:10.1109/87.880596. URL <http://www.ncbi.nlm.nih.gov/pubmed/21818836>.
- [6] V. M. Alvarado y J. C. Carmona. L1 prediction error approach in system identification. *Proceedings of the American Control Conference*, 4:3219–3223, 2002. ISSN 07431619. doi:10.1109/ACC.2002.1025286.
- [7] Panayiotis C Andreou, Chris Charalambous, y Spiros H Martzoukos. Robust Artificial Neural Networks for Pricing of European Options Citation for published item : Andreou , P . C . , Charalambous , C . , Martzoukos , H . S . (2006). Robust artificial View online & further information on publisher 's website : Robust Ar. págs. 329–351, 2006.

-
- [8] R. Unbehauen Andrzej Cichocki. *Neural Networks for Optimization and Signal Processing*. Wiley, 1993. ISBN 978-0-471-93010-5. URL <http://gen.lib.rus.ec/book/index.php?md5=66E46B13CCA0B4A2048674FF00EEFA65>.
- [9] Karl-Johan Åström y Torsten Bohlin. Numerical identification of linear dynamic systems from normal operating records. En *Theory of self-adaptive control systems*, págs. 96–111. Springer, 1966.
- [10] Karl Johan Åström y Björn Wittenmark. On self tuning regulators. *Automatica*, 9(2):185–199, 1973.
- [11] V Barnett y T Lewis. *Outliers in statistical data*. 1994.
- [12] Guilherme A. Barreto y Ana Luiza B P Barros. A Robust Extreme Learning Machine for pattern classification with outliers. *Neurocomputing*, págs. 1–11, 2014. ISSN 18728286. doi:10.1016/j.neucom.2014.10.095. URL <http://dx.doi.org/10.1016/j.neucom.2014.10.095>.
- [13] A. Bassam, R. A. Conde-Gutierrez, J. Castillo, G. Laredo, y J. A. Hernandez. Direct neural network modeling for separation of linear and branched paraffins by adsorption process for gasoline octane number improvement. *Fuel*, 124:158–167, 2014. ISSN 00162361. doi:10.1016/j.fuel.2014.01.080. URL <http://dx.doi.org/10.1016/j.fuel.2014.01.080>.
- [14] Mohamed Bouguessa. A practical outlier detection approach for mixed-attribute data. *Expert Systems with Applications*, 42(22):8637–8649, 2015. ISSN 09574174. doi:10.1016/j.eswa.2015.07.018. URL <http://linkinghub.elsevier.com/retrieve/pii/S0957417415004789>.
- [15] Colin Chen. Statistics and Data Analysis Paper 265-27 Robust Regression and Outlier Detection with the ROBUSTREG Procedure. *Statistics*, 2002. URL <http://scholar.google.com/scholar?hl=en{%&}btnG=Search{%&}q=intitle:Robust+Regression+and+Outlier+Detection+with+the+ROBUSTREG+Procedure{#}0>.
- [16] Long Cheng, Zeng-Guang Hou, Yingzi Lin, Min Tan, Wenjun Chris Zhang, y Fang-Xiang Wu. Recurrent neural network for non-smooth convex optimization

- problems with application to the identification of genetic regulatory networks. *IEEE Transactions on Neural Networks*, 22(5):714–726, 2011. ISSN 1941-0093. doi:10.1109/TNN.2011.2109735.
- [17] Chen Chia Chuang, Jin Tsong Jeng, y Pao Tsun Lin. Annealing robust radial basis function networks for function approximation with outliers. *Neurocomputing*, 56(1-4):123–139, 2004. ISSN 09252312. doi:10.1016/S0925-2312(03)00436-3.
- [18] A Cochocki y Rolf Unbehauen. *Neural networks for optimization and signal processing*. John Wiley & Sons, Inc., 1993.
- [19] Jt Connor, Rd Martin, y Le Atlas. Recurrent Neural Networks and Robust Time Series Prediction. *Neural Networks, IEEE . . .*, 5(2):240–254, 1994. ISSN 1045-9227. doi:10.1109/72.279188. URL http://ieeexplore.ieee.org/xpls/abs/_all.jsp?arnumber=279188.
- [20] C. Corbier, A. F. Boukari, J.-C. Carmona, V. A. Martinez, G. Moraru, y F. Malburet. On a robust modeling of piezo-systems. *Journal of Dynamic Systems, Measurement, and Control*, 134, 2012.
- [21] C. Corbier, J.-C. Carmona, y V. M. Alvarado. $L_1 - L_2$ Robust Estimation in Prediction Error System Identification. Toluca, Mexico, 2009.
- [22] C. Corbier, J.-C. Carmona, y V. M. Alvarado. System Identification with an Extended Threshold M-Estimator for Pseudo-Linear Models Structure. Brussels, Belgium, 2012.
- [23] Christophe Corbier y Jean-Claude Carmona. Mixed estimators variety for model order reduction in control oriented system identification. *Mathematical Problems in Engineering*, 2015, 2015.
- [24] Christophe Corbier y Hector Manuel Romero Ugalde. Low-order control-oriented modeling of piezoelectric actuator using huberian function with low threshold: pseudolinear and neural network models. *Nonlinear Dynamics*, págs. 1–18, 2016.

-
- [25] M. Costa, A. P. Braga, y B. R. De Menezes. Improved generalization learning with sliding mode control and the Levenberg-Marquadt algorithm. *Proceedings - Brazilian Symposium on Neural Networks, SBRN*, 2002-Janua:44–48, 2002. ISSN 15224899. doi:10.1109/SBRN.2002.1181433.
- [26] George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303–314, 1989.
- [27] U. Flores. *Identificación de sistemas no lineales mediante las estructuras NARX y Hammerstein-Wiener*. Proyecto Fin de Carrera, Centro Nacional de Investigación y Desarrollo Tecnológico, 2011.
- [28] Yu-Yi Fu, Chia-Ju Wu, Chia-Nan Ko, y Jin-Tsong Jeng. Radial basis function networks with hybrid learning for system identification with outliers. *Applied Soft Computing*, 11(3):3083–3092, 2011. ISSN 15684946. doi:10.1016/j.asoc.2010.12.010.
- [29] Frank E Grubbs. Procedures for detecting outlying observations in samples. *Technometrics*, 11(1):1–21, 1969.
- [30] Jihyun Ha, Seulgi Seok, y Jong Seok Lee. A precise ranking method for outlier detection. *Information Sciences*, 324:88–107, 2015. ISSN 00200255. doi:10.1016/j.ins.2015.06.030. URL <http://dx.doi.org/10.1016/j.ins.2015.06.030>.
- [31] Xuan Han, Wen-Fang Xie, Zhijun Fu, Weidong Luo, y J Zhang. Nonlinear systems identification using dynamic multi-time scale neural networks. *Neurocomputing*, 74(17):3428–3439, 2011. ISSN 09252312. doi:10.1016/j.neucom.2011.06.007. URL <http://dx.doi.org/10.1016/j.neucom.2011.06.007>.
- [32] BL Ho y RE Kalman. Effective construction of linear state variable models from input output data. En *Proc. 3rd Allerton Conf*, págs. 545–548. 1965.
- [33] Guang-Bin Huang, Hongming Zhou, Xiaojian Ding, y Rui Zhang. Extreme learning machine for regression and multiclass classification. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 42(2):513–529, 2012.

- [34] Guang-Bin Huang, Qin-Yu Zhu, y Chee-Kheong Siew. Extreme learning machine: a new learning scheme of feedforward neural networks. En *Neural Networks, 2004. Proceedings. 2004 IEEE International Joint Conference on*, tomo 2, págs. 985–990. IEEE, 2004.
- [35] Guang-Bin Huang, Qin-Yu Zhu, y Chee-Kheong Siew. Extreme learning machine: theory and applications. *Neurocomputing*, 70(1):489–501, 2006.
- [36] Peter J Huber. *Robust statistics*. Springer, 2011.
- [37] Peter J Huber et al. Robust estimation of a location parameter. *The Annals of Mathematical Statistics*, 35(1):73–101, 1964.
- [38] Javed Iqbal, Asif Iqbal, y Muhammad Arif. Levenberg–Marquardt method for solving systems of absolute value equations. *Journal of Computational and Applied Mathematics*, 282:134–138, 2015. ISSN 03770427. doi:10.1016/j.cam.2014.11.062. URL <http://www.sciencedirect.com/science/article/pii/S0377042714005433>.
- [39] RE Kalman. Design of a self-organising control system. *Trans. ASME D*, 86:51–60, 1958.
- [40] Chia-Nan Ko. Identification of nonlinear systems with outliers using wavelet neural networks based on annealing dynamical learning algorithm. *Engineering Applications of Artificial Intelligence*, 25(3):533–543, 2012. ISSN 09521976. doi:10.1016/j.engappai.2011.09.019. URL <http://dx.doi.org/10.1016/j.engappai.2011.09.019>.
- [41] Wallace E Larimore. System identification, reduced-order filtering and modeling via canonical variate analysis. En *American Control Conference, 1983*, págs. 445–451. IEEE, 1983.
- [42] Bo Liu, Huiguang Li, y Tihua Wu. Neural Network Identification Method Applied to the Nonlinear System. *2009 WRI Global Congress on Intelligent Systems*, págs. 120–124, 2009. doi:10.1109/GCIS.2009.446. URL <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5209331>.

- [43] L. Ljung. *System Identification Theory For The User*. PTR Prentice Hall, 1999.
- [44] Kaj Madsen, Hans Bruun Nielsen, y Ole Tingleff. *Methods for non-linear least squares problems*. 2004.
- [45] Anna Marconato, Jonas Sjöberg, y Johan Schoukens. Initialization of nonlinear state-space models applied to the wiener–hammerstein benchmark. *Control Engineering Practice*, 20(11):1126–1132, 2012.
- [46] Donald W Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *Journal of the society for Industrial and Applied Mathematics*, 11(2):431–441, 1963.
- [47] Bonifacio Martín y Alfredo Sanz. *Redes neuronales y sistemas borrosos*. 2006.
- [48] Warren S McCulloch y Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.
- [49] Franklin Riabani Mercado. *Extreme learning machine*.
- [50] JO Moody y PJ Antsaklis. The Dependence Identification Neural Network Construction Algorithm. *Neural Networks, IEEE Transactions . . .*, I(1), 1996. URL http://ieeexplore.ieee.org/xpls/abs/_all.jsp?arnumber=478388.
- [51] Roaldje Nadjiasngar y Michael Inggs. Gauss-Newton filtering incorporating Levenberg-Marquardt methods for tracking. *Digital Signal Processing: A Review Journal*, 23(5):1662–1667, 2013. ISSN 10512004. doi:10.1016/j.dsp.2012.12.005. URL <http://dx.doi.org/10.1016/j.dsp.2012.12.005>.
- [52] Kumpati S Narendra y Kannan Parthasarathy. Identification and control of dynamical systems using neural networks. *Neural Networks, IEEE Transactions on*, 1(1):4–27, 1990.
- [53] William Rey. *Introduction to robust and quasi-robust statistical methods*. Springer Science & Business Media, 2012.
- [54] H. Romero. *Identificación de Sistemas Utilizando Redes Neuronales*. Proyecto Fin de Carrera, Centro Nacional de Investigación y Desarrollo Tecnológico, 2008.

- [55] H. M. Romero. *Identificación de Sistemas Utilizando Redes Neuronales: Un enfoque basado en balance entre sencillez, precisión y costo computacional*. Tesis Doctoral, Centro Nacional de Investigación y Desarrollo Tecnológico, 2013.
- [56] H. M. Ugalde Romero, J.-C. Carmona, V. M. Alvarado, y J. Reyes-Reyes. Neural network design model reduction approach for black box nonlinear system identification with reduced number of parameters. *Neurocomputing*, págs. 170–180.
- [57] H. M. Ugalde Romero, Jean Claude Carmona, Juan Reyes-Reyes, Victor M. Alvarado, y Juan Mantilla. Computational cost improvement of neural network models in black box nonlinear system identification. *Neurocomputing*, 166(November):96–108, 2015. ISSN 18728286. doi:10.1016/j.neucom.2015.04.022.
- [58] H. M. Ugalde Romero y Christophe Corbier. Robust estimation of balanced simplicity-accuracy neural networks-based models. *Journal of Dynamic Systems, Measurement, and Control*, 138(5):051001, 2016.
- [59] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.
- [60] Frank Rosenblatt. Principles of neurodynamics. Perceptrons and the theory of brain mechanisms. Inf. téc., DTIC Document, 1961.
- [61] Peter J Rousseeuw y Annick M Leroy. *Robust regression and outlier detection*, tomo 589. John Wiley & Sons, 2005.
- [62] Mohamed Aymen Sahnoun, Hector M. Romero Ugalde, Jean-Claude Carmona, y Julien Gomand. Maximum Power point Tracking Using P&O Control Optimized by a Neural Network Approach: A Good Compromise between Accuracy and Complexity. *Energy Procedia*, 42:650–659, 2013. ISSN 18766102. doi:10.1016/j.egypro.2013.11.067. URL <http://www.sciencedirect.com/science/article/pii/S1876610213017694>.

- [63] J. Schoukens, J. Suykens, y L. Ljung. Wiener-Hammerstein Benchmark. *Proceedings of the 15th IFAC Symposium on System Identification (SYSID 2009)*, pág. 182, 2009.
- [64] Horng Lin Shieh, Ying Kuei Yang, Po Lun Chang, y Jin Tsong Jeng. Robust neural-fuzzy method for function approximation. *Expert Systems with Applications*, 36(3 PART 2):6903–6913, 2009. ISSN 09574174. doi:10.1016/j.eswa.2008.08.072.
- [65] J. Sjöberg, L. Lauwers, y J. Schoukens. Identification of Wiener-Hammerstein models: Two algorithms based on the best split of a linear model applied to the SYSID'09 benchmark problem. *Control Engineering Practice*, 20(11):1119–1125, 2012. ISSN 09670661. doi:10.1016/j.conengprac.2012.07.001.
- [66] Jonas Sjöberg, L Lauwers, y Johan Schoukens. Identification of wiener-hammerstein models: Two algorithms based on the best split of a linear model applied to the sysid'09 benchmark problem. *Control Engineering Practice*, 20(11):1119–1125, 2012.
- [67] Peter Van Overschee y BL De Moor. *Subspace identification for linear systems: Theory—Implementation—Applications*. Springer Science & Business Media, 2012.
- [68] Adrian Wills y Brett Ninness. Generalised hammerstein-wiener system estimation and a benchmark application. *Control Engineering Practice*, 20(11):1097–1108, 2012.
- [69] Zhang Xing-gui y Wang Man-qiang. Adaptive PID control based on RBF neural network identification. *17th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'05)*, págs. 3 pp.–683, 2005. doi:10.1109/ICTAI.2005.26. URL <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1563016>.
- [70] Wen Yu. Nonlinear system identification using discrete-time recurrent neural networks with stable learning algorithms. *Information sciences*, 158:131–147, 2004.

- [71] Wen Yu, Alexander S Poznyak, y Edgar N Sanchez. Dynamic multilayer neural networks for nonlinear system on-line identification. En *Intelligent Control, 2000. Proceedings of the 2000 IEEE International Symposium on*, págs. 25–30. IEEE, 2000.
- [72] Lofti A Zadeh. On the identification problem. *Circuit Theory, IRE Transactions on*, 3(4):277–281, 1956.
- [73] Zhengyou Zhang. Parameter estimation techniques: A tutorial with application to conic fitting. *Image and vision Computing*, 15(1):59–76, 1997.